第三章 树

在各种各样的图中，有一类简单的，然而又是重要的图，就是所谓的“树”。树之所以重要，不仅在于它在许多不同领域中有着广泛的应用，而且还在于它在图问题的研究中所扮演的特殊角色。在图论中，解决一些困难的问题往往首先从树入手，打开缺口；还有一些问题对一般的图未能解决或者没有简便方法，而对于树则已圆满解决，且方法较为简便。

树首先是作为无向图被讨论，以下除非特别说明，限于讨论无向图。

§3.1 树的定义及其性质

定义 1.1 不含任何回路的连通图称为树。常用T表示树。T中的边称为树枝。度为1的结点称为树叶，度大于1的结点称为分支点。

注 1: 只含一个结点的树称为平凡树。
注 2；若图G的每个连通分支都是树，则称G为森林。森林是二部图。
注 3：树的每条边都不属于任何回路，这样的边称为割边。

定义 1.2 设e是G的一条边，若$G-e$的连通分支数比G多，则称e是G的一条割边，也称为桥。

注：事实上，若e是G的割边，则$\kappa(G-e)=\kappa(G)+1$。

设割边$e=(u,v)$，从G中删去e，则将结点u,v所在的连通分支分割为两个连通分支，u和v各在其一。

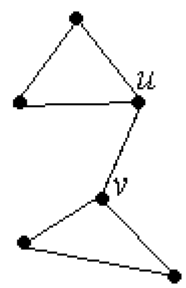

图 3.1

定理 1.1 $e=(u,v)$是G的割边，当且仅当e不属于G的任何回路。

证明：必要性：若$e=(u,v)$属于G的某个回路C，则$C-e$是$G-e$中u到v的一条道路，u和v仍在同一连通分支，e不是G的割边。

充分性：若e不是G的割边，则$\kappa(G-e)=\kappa(G)$，u和v在$G-e$中仍连通，即$G-e$中存在u到v的一条道路$P(u,v)$，$P(u,v)+e$形成G中的一条回路。

为更充分地了解数地特性，下面给出树的一些等价刻画。

定理 1.2 设T是阶不小于2的无向图，则下列说法等价。
（1）T连通且无回路；
（2）T连通且每条边都是割边；
（3）T连通且有$n-1$条边；
（4）T有$n-1$条边且无回路；
（5）T的任意两结点间有唯一道路；
（6）T无回路，但任意加上一条边后恰有一个回路。

证明：（1）$\Leftrightarrow$（2）：由定理 1.1。
（2）$\Rightarrow$（3）：对T的结点数n施行归纳。当$n=2$时命题显然成立。假设当$n \le k$（$k \ge 2$）时命题成立，则当$n=k+1$时，由于T的每条边都是割边，任取T的一条边e，则$T-e$有两个连通分支T_1和T_2，设T_1和T_2的结点数，边数分别为n_1，m_1，n_2，m_2，由于T_1和T_2都满足（2）且$n_1 \le k$，$n_2 \le k$，对T_1和T_2分别应用归纳假设得，$m_1 = n_1 - 1$，$m_2 = n_2 - 1$，从而$m = m_1 + m_2 + 1 = n_1 - 1 + n_2 - 1 + 1 = n_1 + n_2 - 1 = n - 1$。
（3）$\Rightarrow$（4）：如果T有回路C，则任取C上任意一条边e，$T-e$仍连通，这与T的最小连通性矛盾。
（4）$\Rightarrow$（5）：首先证T是连通的。设T的连通分支数为k，这些连通分支的结点数和边数分别为n_i，m_i（$1 \le i \le k$），由（1）$\Rightarrow$（3）知，$m_i = n_i - 1$，所以$m = \sum_{i=1}^{k} m_i$ $= \sum_{i=1}^{k}(n_i - 1) = \sum_{i=1}^{k} n_i - k = n - k$，从而得知，$k=1$。这说明$T$得任意两结点$u$，$v$之间存在道路$P(u,v)$，若$u$，$v$之间还有一条不同的道路$P'(u,v)$，则$P(u,v) \oplus P'(u,v)$至少含有一个回路，与$T$无回路矛盾。
（5）$\Rightarrow$（6)，（6）$\Rightarrow$（1）均显然。

推论 1.1 设G是结点数为n连通分支数为k的森林，则G的边数为$n-k$。

推论 1.2 非平凡树T至少存在两片树叶。

证明：应用握手定理得，$\sum_{i=1}^{n} d(v_i) = 2(n-1)$，即$\sum_{i=1}^{n}(d(v_i) - 1) = n - 2$，由于$d(v_i) - 1$非负，所以至少存在两个结点$v$，使得$d(v) - 1 = 0$，即$v$是树叶。

定义 1.3 如果树T是图G的生成子图，则称T是G的一棵生成树，简称为G的树。

注 1：当且仅当G是连通图时，G有生成树。

注 2：给定图G的一棵树T，我们称$G-T$（从G中删去T中各边后得到的图）为T的余树，用$\overline{T}$表示，$\overline{T}$的边称为T的弦。

图 3.2 中，$T=\{e_1,e_2,e_5,e_6,e_8\}$，

$\overline{T}=\{e_3,e_4,e_7,e_9\}$。

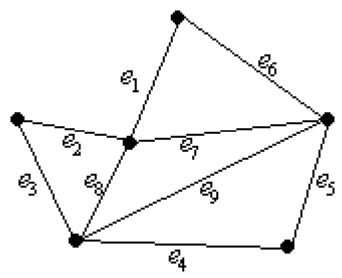

图 3.2

作业

1．非平凡树中最长道路的端点一定是树叶。

2．设G为n（$n\geq 5$）阶无向简单图，证明G或$\overline{G}$必含回路。

3．设$d_1,\cdots,d_n$是n（$n\geq 2$）个正整数，证明：若$\sum_{i=1}^{n}d_i=2(n-1)$，则存在一棵结点度分别为$d_1,\cdots,d_n$的n阶树T。

§3.2 回路与割集

3.2.1 生成树与基本回路系统

定义 2.1 设T是连通图G的生成树，$e_1,e_2,\cdots,e_{m-n+1}$是T的弦，记C_r（$1\leq r\leq m-n+1$）是T加弦e_r产生的回路，称C_r为对应于弦e_r的基本回路，称$\{C_1,\cdots,C_r\}$为生成树T对应的基本回路系统。

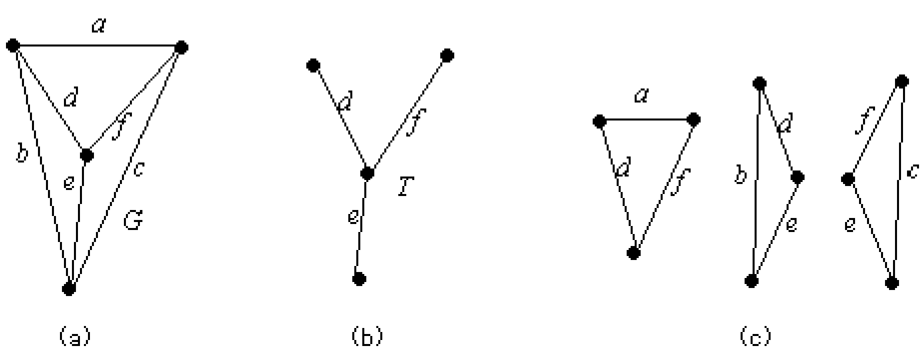

图 3.3

注：连通图G的生成树不唯一，从而基本回路系统不唯一，但基本回路的个数是相同的，都是$m-n+1$。

给定连通图G的一棵生成树T，就确定G中的一个基本回路系统，那么G中任意一个回路与基本回路有什么关系呢？

首先来说明这$m-n+1$个基本回路是互相独立的。

定理 2.1 设T是连通图G的一棵生成树，C_k是对应于弦e_k（$1 \le k \le m-n+1$）的基本回路，对于任意r，$1 \le r \le m-n+1$，及任意序列$i_1, \cdots, i_r$，$1 \le i_1 < \cdots < i_r \le m-n+1$，有：$e_{i_1}, \cdots, e_{i_r}$是$C_{i_1} \oplus C_{i_2} \oplus \cdots \oplus C_{i_r}$里的所有弦。

证明：只需注意到弦e_{i_t}只在C_{i_t}中出现（$1 \le t \le r$）。

下面来说明G中任意回路都可由基本回路生成。

引理 2.1 设图G_1，G_2的结点都是偶结点，则$G_1 \oplus G_2$的结点也都是偶结点。

证明：取$G_1 \oplus G_2$的任意结点v，设$e_{1,1}, \cdots, e_{1,r}$是G_1中与v关联的边，$e_{2,1}, \cdots, e_{2,s}$是G_2中与v关联的边，其中，r, s都是偶数，$|\{e_{1,1}, \cdots, e_{1,r}\} \cap \{e_{2,1}, \cdots, e_{2,s}\}| = t$，则$G_1 \oplus G_2$中与$v$关联的边有$r+s-2t$条，即$v$是$G_1 \oplus G_2$的偶结点。

定理 2.2 连通图G的任意回路C均可表示为生成树T的若干基本回路的环和。

证明：设C中含T的r（$1 \le r \le m-n+1$）条弦：$e_{i_1}, \cdots, e_{i_r}$，令$C' = C_{i_1} \oplus \cdots \oplus C_{i_r}$，断言：$C' = C$。如果$C' \ne C$，则$C \oplus C'$非空。由定理 2.1 知，$e_{i_1}, \cdots, e_{i_r}$都在$C'$中，所以$C \oplus C'$只包含$T$上的边，但由引理 2.1 知，$C \oplus C'$的结点都是偶结点，必包含回路，矛盾。

推论 2.1 设连通图G的回路个数为c，则$m-n+1 \le c \le 2^{m-n+1}-1$。

注：所以称$m-n+1$为连通图G的回路秩。

3.2.1 生成树与基本割集系统

定义 2.2 设S是连通图$G=(V,E)$的边子集，如果

（1）$G'=(V,E-S)$是分离图；

（2）对任意$S'\subset S$，$G''=(V,E-S')$仍连通；

则称S是G的一个割集。

注：割集S是这样一个边集：若在G中去掉S的所有边后，则G变成具有两个连通分支的分离图，但是若在G中只去掉S的部分边后，G仍然连通。由此可知，割集由$V(G)$的某一划分V_1与V_2之间的所有边组成，该划分使得$G[V_1]$和$G[V_2]$都是连通图。

如图 3.4 所示的图中，边子集$\{e_1,e_2,e_6\}$，$\{e_5,e_6,e_7,e_8\}$，$\{e_3,e_4,e_8\}$，$\{e_2,e_4,e_5,e_6\}$，$\{e_9\}$等都是割集。

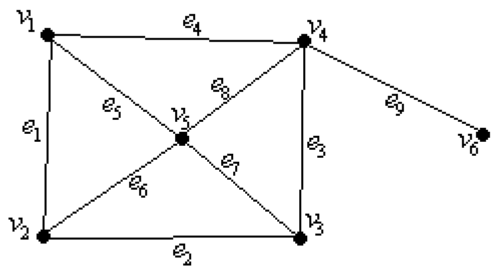

图 3.4

定理 2.3 设T是连通图G的生成树，则T的每一树枝都对应G中的一个割集。

证明：设e是T的任一树枝，从T中删去e得到二连通分支T_1，T_2，设它们的结点集分别为V_1与V_2，G中所有端点分别在V_1与V_2的边组成的集合记为S_e，则S_e是仅包含T中一条边e的割集。

定义 2.3 设T是连通图G的生成树，$n-1$个树枝对应的割集$S_1,\cdots,S_{n-1}$称为对应T的基本割集，称$\{S_1,\cdots,S_{n-1}\}$为对应T的基本割集系统。

例 2.1 图 3.4 中，设T是边集$\{e_5,e_6,e_7,e_8,e_9\}$确定的生成树，则e_5,e_6,e_7,e_8,e_9对应的割集分别是$\{e_5,e_1,e_4\}$，$\{e_6,e_1,e_2\}$，$\{e_7,e_2,e_3\}$，$\{e_8,e_3,e_4\}$，$\{e_9\}$。

基本割集系统与基本回路系统有类似的性质。

定理 2.4 连通图G中的每个割集必包含G中每棵生成树的至少一个树枝。

定理 2.5 设$\{S_1,\cdots,S_{n-1}\}$为连通图G的对应于生成树T的基本割集系统，则$e_{i_1},\cdots,e_{i_k}$是$S_{i_1}\oplus S_{i_2}\oplus\cdots\oplus S_{i_k}$中的所有树枝，其中，$1\le k\le n-1$，$1\le i_1<\cdots<i_k\le n-1$。

本定理的证明类似于定理 2.1。

定义 2.3 设$G=(V,E)$是一个无向图，V_1是V的非空真子集，所有两端点分别在V_1和$\overline{V_1}$的边组成的集合记作$E(V_1\times\overline{V_1})$，称为$G$的一个断集。

注：割集显然是断集，断集不一定是割集。

下面来说明G中任意割集都可由基本割集生成。

引理 2.2 设S_1，S_2为无向图G的两个不同断集，则$S_1\oplus S_2$为G中的断集。

证明：设$S_1=E(V_1\times\overline{V_1})$，$S_2=E(V_2\times\overline{V_2})$，则

$$S_1=E(V_1\cap V_2\times\overline{V_1}\cap V_2)\cup E(V_1\cap\overline{V_2}\times\overline{V_1}\cap V_2)\cup E(V_1\cap V_2\times\overline{V_1}\cap\overline{V_2})\cup E(V_1\cap\overline{V_2}\times\overline{V_1}\cap\overline{V_2})$$

$$S_2=E(V_2\cap V_1\times\overline{V_2}\cap V_1)\cup E(V_2\cap\overline{V_1}\times\overline{V_2}\cap V_1)\cup E(V_2\cap V_1\times\overline{V_2}\cap\overline{V_1})\cup E(V_2\cap\overline{V_1}\times\overline{V_2}\cap\overline{V_1})$$

$$S_1\oplus S_2=E(V_1\cap V_2\times\overline{V_1}\cap V_2)\cup E(V_1\cap\overline{V_2}\times\overline{V_1}\cap\overline{V_2})\cup E(V_2\cap V_1\times\overline{V_2}\cap V_1)\cup E(V_2\cap\overline{V_1}\times\overline{V_2}\cap\overline{V_1})$$

记$V_0=(V_1\cap V_2)\cup(\overline{V_1}\cap\overline{V_2})$，

则$\overline{V_0}=(\overline{V_1}\cup\overline{V_2})\cap(V_1\cup V_2)$

$=(V_1\cap\overline{V_2})\cup(\overline{V_1}\cap V_2)$，

$S_1\oplus S_2=E(V_0\times\overline{V_0})$，注意到$V_0$和$\overline{V_0}$都非空，所以$S_1\oplus S_2$是断集。

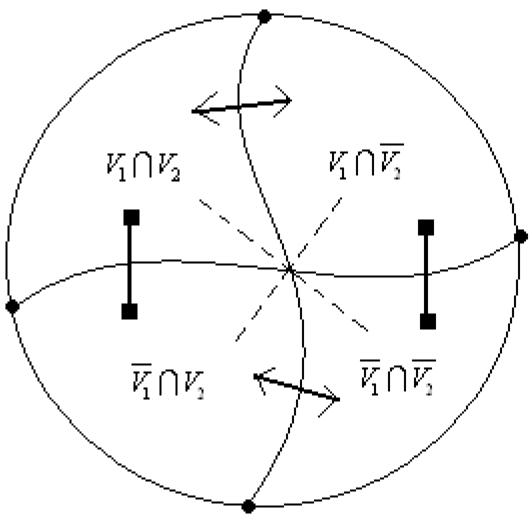

图 3.5

定理 2.6 连通图G的任一割集可表示为生成树T对应的若干基本割集的环和。

证明：任取G的一个割集S，设S包含生成树T的k条边$e_{i_1},\cdots,e_{i_k}$，则可以断言：$S = S_{i_1} \oplus S_{i_2} \oplus \cdots \oplus S_{i_k}$。否则$S \oplus S_{i_1} \oplus S_{i_2} \oplus \cdots \oplus S_{i_k}$是一个断集，该断集不含$T$的任何边，这是不可能的。

注：所以称$n-1$为连通图G的割集秩。

推论 2.2 设连通图G的割集个数为s，则$n-1 \le s \le 2^{n-1}-1$。

3.2.3 基本回路与基本割集

定理 2.7 任何一个回路和任何一个割集都有偶数条公共边。

证明：对任意割集S和任意回路C。设S将G分割成两个各自连通的结点集V_1和V_2。若C上的结点都在V_1（或V_2）中，则$S \cap C = \varnothing$。否则，C上既有V_1也有V_2中的结点，此时不妨设$v_0 \in V_1$是C的起点，C从v_0出发，只有经过偶数条S中的边才能重新回到V_1中，于是C中含有偶数条S中的边。

推论 3.3 设T是连通图G的生成树，则决定基本回路C_i的弦e_i只出现在所有C_i中的树枝对应的基本割集中。

推论 3.4 设T是连通图G的生成树，则决定基本割集S_i的树枝e_i只出现在所有S_i中的弦对应的基本回路中。

这两个推论的证明留作作业。

作业：

1. 完全图K_n($n \ge 3$)的所有边都赋以整数权，每一个回路的权定义为其所有边权之和。证明：K_n的每个回路的权都是偶数当且仅当K_n的每个三角形回路的权都是偶数。

2．证明推论 3.3 和推论 3.4。

3．设T_1，T_2是无向连通图G的两棵生成树，$e_1 \in E(T_1)-E(T_2)$，证明：存在

$e_2 \in E(T_2) - E(T_1)$，使得$T_1 - e_1 + e_2$和$T_2 - e_2 + e_1$都是G的生成树。

§3.3 最小生成树

在赋权连通图中，有时需要计算总长最小或最大的生成树，这可归结为最小生成树问题。例如要在若干加油站之间铺设输油管道，已知任意两个加油站之间输油管道的铺设费用，如果要让每个站都能保障油的供应，那么最少的铺设费用就是一个赋权图的最小生成树的权。

计算最小树的算法很多，我们介绍两种常用的算法：Kruskal 算法和 Prim 算法。

3.3.1 基本树变换

在介绍两个算法之前，首先介绍基本树变换的概念。

设T是连通图G的一棵生成树，e是T的一弦，C_e是由e决定的基本回路。若C_e不是自环，则必存在边$e' \in C_e - e$。于是，$T' = T \oplus \{e, e'\}$为G的另一棵生成树，且T与T'只有一条边不同。

定义 3.1 设T_1与T_2都是G的生成树，若T_1与T_2恰有k条边不同，则称T_1与T_2的距离为$d(T_1, T_2) = k$。

定义 3.2 设T_1，T_2是连通图G的距离为1的树，$T_1 - T_2 = e_1$，$T_2 - T_1 = e_2$，则$T_2 = T_1 \oplus \{e_1, e_2\}$称为$T_1$到$T_2$的基本树变换。

注：e_2作为T_1的一条弦，决定一个基本回路C_{e_2}，必有$e_1 \in C_{e_2}$，否则$C_{e_2} \subseteq T_2$，矛盾。

定理 3.1 设T_0是连通图G的一棵生成树，则G的任意其它生成树都可由T_0通过若干次基本树变换得到。

证明：任取G的生成树T，设$d(T, T_0) = k$。任取$e \in T - T_0$，则$T_0 \oplus e$包含回路C_e。C_e上必有属于T_0而不属于T的边e'，作基本树变换$T_1 = T_0 \oplus \{e, e'\}$，则$d(T, T_1) = k - 1$。由归纳可知，经过$k$次基本树变换，可由$T_0$变到$T$。

3.3.2 Kruskal 算法

Kruskal 算法可描述如下：

1.（初始化）$T \leftarrow \varnothing$；

2. 当$|T|<n-1$且$E \neq \varnothing$时，

Begin

a. $e \leftarrow E$中最短边，

b. $E \leftarrow E-e$，

c. 若$T+e$无回路，则$T \leftarrow T+e$。

End

3. 若$|T|<n-1$，打印"非连通"；否则输出最小树T。

例 3.1 如图 3.6，执行 Kruskal 算法的过程是

$T \leftarrow \varnothing$，$T \leftarrow T+(v_3,v_4)$，

$T \leftarrow T+(v_1,v_2)$，$T \leftarrow T+(v_4,v_5)$

(v_3,v_5)跳过，$T \leftarrow T+(v_1,v_3)$。

最后得到最小树

$\{(v_3,v_4),(v_1,v_2),(v_4,v_5),(v_1,v_3)\}$.

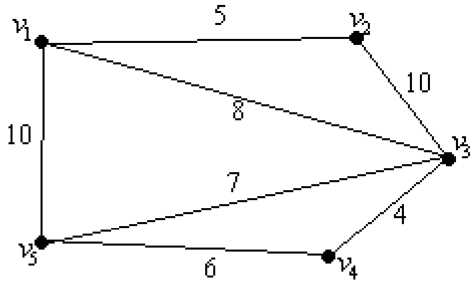

图 3.6

该算法的基本思路是逐一往T中加入尽可能短的边，直至T称为一棵生成树。

下面来说明由 Kruskal 算法得到的树真是最小树。

定理 3.2 Kruskal 算法得到的树是最小树

证明： Kruskal 算法得到的树记为T^*，算法执行过程中，$n-1$条边加入T^*的顺序为：$e_1,\cdots,e_{n-1}$，满足：e_k（$1 \le k \le n-1$）是$E(G)$中与$k-1$条边$\{e_1,\cdots,e_{k-1}\}$一起形不成回路的最短边。设T是不同于T^*的任何生成树，记$i(T)=\min\{i: e_i \notin T\}$，即$e_1$，$\cdots$，$e_{i(T)-1}$都在$T$中，但$e_{i(T)}$不在$T$中。如果$T^*$不是最小树，则取$T_0 = \arg\max\limits_{T\text{是最小树}} i(T)$。为简化记号，记$t=i(T_0)$。因为$e_t$是$T_0$的弦，可取$e \in C_{e_t}-T^*$，作基本树变换

$T' = T \oplus \{e, e_t\}$，则 $w(T') = W(T_0) - W(e) + W(e_t)$。因为 $\{e_1, \cdots, e_{t-1}, e\}$ 不包含回路，所以 $w(e) \ge w(e_t)$，于是 $w(T') \le w(T_0)$，即 T' 也是最小树。但 $e_1, \cdots, e_t \in T'$，$i(T') > i(T_0)$，与 T_0 的选取矛盾。所以 T^* 是最小树。

3.3.3 Prim 算法

Prim 算法的基本思想是：首先任选一结点 v_0 作为一棵“小树”，不断往里添加树上点与树外点之间的最短边，直到形成一棵生成树为止。

Prim 算法描述如下：

1. （初始化）$U \leftarrow \{v_0\}$，$T \leftarrow \varnothing$；$\forall v \in V - U$，$t(v) \leftarrow v_0$
2．While $U \neq V$ do
   Begin
   a. $u \leftarrow \arg\min\limits_{u \in V-U} w(t(u), u)$,
   b. $U \leftarrow U + u$,$T \leftarrow T + (t(u), u)$,
   c. $\forall v \in V - U$, if $w(u, v) < w(t(v), v)$, then $t(v) \leftarrow u$
   End
3. 输出最小树 T。

我们再来说明 Prim 算法的正确性。

定理 3.3　Prim 算法的结果得到了赋权连通图 G 的一棵最小生成树。

证明：显然 Prim 算法得到的是一棵生成树，记为 T_0。设 G 中的边加入 T_0 的顺序为 $e_1, \cdots$，e_{n-1}，加入边 e_r（$1 \le r \le n-1$）前 T_0 的结点集为 V_r，则 e_r 是 V_r 与 $V - V_r$ 之间的最短边。对于 G 中任意一棵生成树 T，记 $i(T) = \min\{i : e_i \notin T\}$，约定 $i(T_0) = n$。若 $T \neq T_0$，则 $i(T) \le n-1$。记 $t = i(T)$，因为 e_t 是 T 的一条弦，必有 $e \in C_{e_t} - e_t$，e 的两端点也分别在 V_t 与 $V - V_t$ 之间，于是 $w(e) \ge w(e_t)$。记 $T' = T \oplus \{e, e_t\}$，则 $w(T') = w(T) + w(e_t) - w(e) \le w(T)$，且 $i(T') \ge t + 1 > i(T)$。这说明只要 $T \neq T_0$，

必有T'，$i(T')>i(T)$，使得$w(T')\le w(T)$，从而得知$w(T_0)\le w(T)$。

作业

1．求下面赋权图的最小树

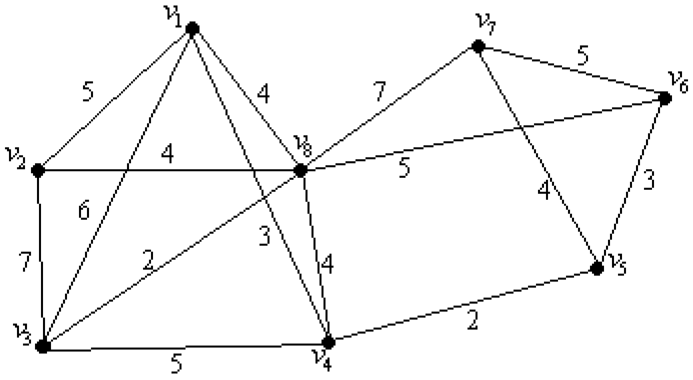

图 3. 7

2. 设T是赋权连通图G的生成树，令$\max(T)$表示T的最大边权，minmax 表示所有生成树中最小的$\max(T)$。证明：若T是G的最小生成树，则$\max(T)$=minmax 。并举例说明反之不然。

3. 设T，T'是赋权连通图G的两个不同的最小生成树，证明：经过一系列基本树变换，可将T变为T'，且这些基本树变换产生的树都是最小生成树。

4．设G是一个赋权连通图，$T_1=\{e_{i_1},e_{i_2},\cdots,e_{i_{n-1}}\}$，$T_2=\{e_{j_1},e_{j_2},\cdots,e_{j_{n-1}}\}$是$G$的两个最小生成树，满足：

$$w(e_{i_1})\le w(e_{i_2})\le\cdots\le w(e_{i_{n-1}}),\quad w(e_{j_1})\le w(e_{j_2})\le\cdots\le w(e_{j_{n-1}}),$$

其中w是权函数。证明：$w(e_{i_1})=w(e_{j_1})$，$w(e_{i_2})=w(e_{j_2})$，$\cdots$，$w(e_{i_{n-1}})=w(e_{j_{n-1}})$。

5. 设C是赋权连通图G的一个回路，e是C上权最大的边，证明：存在G的一个不含e的最小生成树。由此证明破圈算法可产生一个最小生成树。

§3.4 有向树

定义 4.1 若有向图D的基图是树，则称D为有向树。

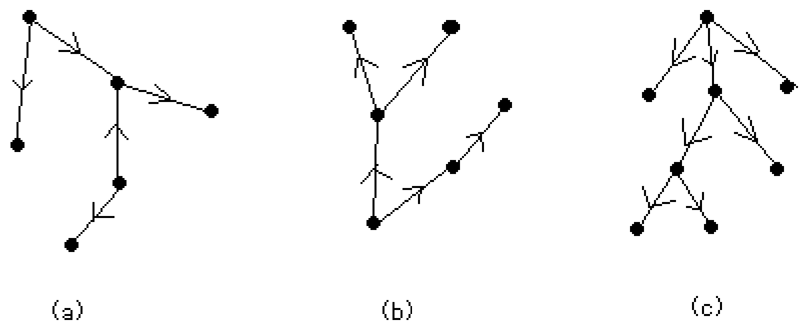

图 3.8

图 3.8 中，(a), (b), (c)所示均为有向树，但我们更关心象(b), (c)一类的有向树，它们被称为根树。

定义 4.2 设T是非平凡的有向树，如果它有一个结点的入度为0，其余结点的入度均为1，则称T为根树。入度为0的结点称为树根，出度为0的结点称为树叶（或外点），出度非0的结点称为分支点（或内点）。
注：在根树中，从根到其余每个结点都有唯一的有向路。

根树中还有一些专门术语，介绍如下：

定义 4.3 设u是根树的分支点，若可从u邻接到v，则称v为u的儿子，u为v的父亲；同一个分支点的所有儿子互称为兄弟；若从u到w有一条有向道路，则称w是u的子孙，u是w的祖先。从根到结点v的有向路的长度称为v的层数；从根到树叶的最大层数称为根树的高度（或深度）。

图 3.9 中，结点 1 是树根，结点 1，2，3，4，5，7 是分支点，结点 6，8，9，10，11，12 是树叶，结点 1 的层数是 0，结点 2，3 的层数是 1，结点 4，5，6，7，8 的层数是 2，结点 9，10，11，12 的层数是 3。根树的高度为 3。

注意，在画根树时，层数小的结点画在上方，这样有向边的方向都指向下方，因而可以省略掉。

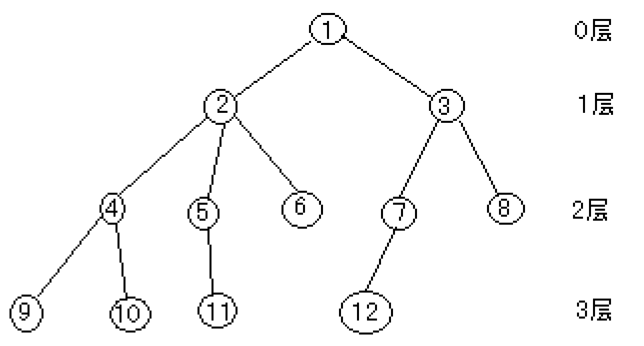

图 3.9

根树的概念非常重要，因为它描述了一个离散结构的层次关系，而层次结构是一种重要的数据结构，所以根树结构可应用于相当广泛的领域中。

有时候只需要考虑局部层次关系，为此引入子树的概念。

定义 4.4 设T是一棵根树，u是T的一个分支点， u及其子孙导出的子图T_u称为T的子树。

易知，T_u是以u为根的根树。

如果要考虑分支点的儿子们的顺序（在计算机科学的许多具体问题，如编码理论，程序语言中，一定要考虑这种顺序），就形成了有序树的概念。

定义 4.5 如果对根树的每个内点的儿子们规定了顺序，则称此根树为有序树。

图 3.10 中，(a), (b)所表示的有序树是不同构的。

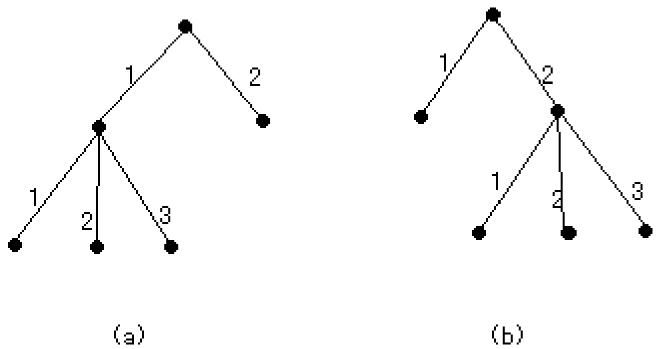

图 3.10

定义 4.6 设T是一棵根树，$m \geq 2$，

（1） 若T的每个分支点至多有m个儿子，则称T为m元树；
（2） 若T的每个分支点都恰有m个儿子，则称T为正则m元树；
（3） 若m元树T是有序的，则称T为有序m元树；
（4）若正则m元树T是有序的，则称T为有序正则m元树；
（5）若正则m元树的所有树叶均在同一层，则称T为完全m元树。

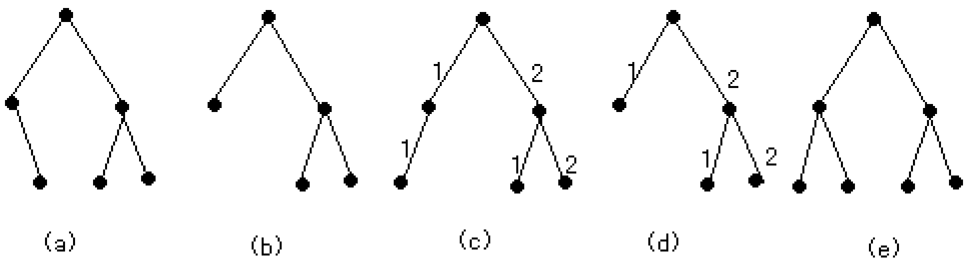

图 3.11

图 3.11 中，(a)是二元树；(b)是正则二元树；(c)是有序二元树；(d)是有序正则二元树；(e)是完全二元树。

有序正则二元树尤其重要，它的每个分支点的两个儿子分别称为左儿子和右儿子。

例 4.1 算术表达式$a-(b+(\frac{c}{d}+\frac{e}{f}))$可以用图 3.12 所示的有序正则二元树来表示。

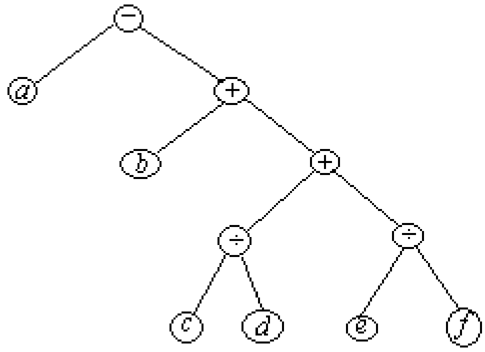

图 3.12

定理 4.1 设T是一棵正则二元树，它的分支点数i和树叶数t满足：$i=t-1$。

证明：一方面，因为分支点数为i，所以儿子数为$2i$；另一方面，边数为$i+t-1$，每一边对应一个儿子，所以儿子数为$i+t-1$；于是$2i=i+t-1$，即$i=t-1$。

定理 4.2 设T是一棵正则二元树，I表示各分支点的层数之和，E表示各树叶的层数之和，则，$E=I+2i$。其中i是T的分支点数。

证明：对分支点数i施行归纳。 $i=1$时，$E=2$，$I=0$，$E=I+2i$成立。设$i=k$（$k\geq 1$）时等式成立，来考察$i=k+1$时的树T。设T的高度为h，因而存在兄弟树叶v_1，v_2，它们的层数是h，记其父结点为v。删除树叶v_1，v_2得到正则二元树T'，v为T'的一片树叶。设T'的分支点数为i'，分支点层数之和为I'，树叶层数之和为E'，则易知$i'=i-1$，$I'=I-(h-1)$，$E'=E-2h+(h-1)$。由归纳假设，在T'中，等式$E'=I'+2i'$成立，即有$E-h-1=I-h+1+2(i-1)$，整理得$E=I+2i$。

推论 4.1 对正则m元树T，有：$(m-1)i=t-1$，$E=(m-1)I+mi$。其中，i是T的分支点数，t是树叶数，E是各树叶的层数和，I是各分支点的层数和。

作业

1．证明推论 4.1。

§3.5 最优树

本节讨论树叶带权的赋权二元树

定义 5.1 设二元树T有t片树叶$v_1,\cdots,v_t$，分别带权$w_1,\cdots,w_t$（w_i为正实数），则称T为赋权二元树，称$w(T)=\sum_{i=1}^{t} w_i \cdot l(v_i)$为二元树$T$的权，其中$l(v_i)$是树叶$v_i$的层数。

定义 5.2 在所有的含t片树叶且分别带权$w_1,\cdots,w_t$的二元树中，权最小的二元树称为最优二元树。

给定t个正实数$w_1,\cdots,w_t$，如何构造最优二元树呢？哈夫曼（Huffman）给出一个算法。在介绍 Huffman 算法之前，先考查几个事实，作为理解算法的依据。

首先说明，给定t（$t \geq 2$）个权$w_1,\cdots,w_t$的情况下，最优二元树总是存在的。

1. 如果带权二元树T不是正则二元树，则总可以通过收缩T的只有一个儿子的分支点得到一正则二元树T'，使$w(T')<w(T)$。

2. 带权$w_1,\cdots,w_t$的正则二元树T的数目$\leq n^{n-2}$（其中$n=2t-1$是T的结点数）。标定完全图K_n的生成树的数目为n^{n-2}（凯莱定理）。将T的根结点标记为v_0，带权$w_1,\cdots,w_t$的叶结点分别标记为$v_1,\cdots,v_t$，其余内结点按层数由小到大的顺序分别标以$v_{t+1},\cdots,v_{2t-2}$（同一层的内点标号大小由其最小标号的子孙而定），这样将带权正则二元树T对应到K_n的某个生成树，且对应是单射。从而带权正则二元树的数目不超过K_n的生成树的数目。

3. 既然带权$w_1,\cdots,w_t$的正则二元树T的数目有限，则必存在最优二元树。

定理 5.1 存在带权为$w_1 \leq w_2 \leq \cdots \leq w_t$的最优二元树$T$，使得带权为$w_1$，$w_2$的两树叶为最深层的兄弟。

证明：设T是带权为$w_1,\cdots,w_t$的最优二元树，设带权w_i（$1 \leq i \leq t$）的树叶结点为v_i，v_i的层数为l_i，且设T的高度为h。因T是正则二元树，所以可取T的两个层数为h的兄弟树叶v_i，v_j（$i<j$），将它们的权分别与树叶v_1，v_2的权互换，则T的权变化量为

$$(w_1h+w_2h+w_il_1+w_jl_2)-(w_1l_1+w_2l_2+w_ih+w_jh)$$

$$=(w_1-w_i)(h-l_1)+(w_2-w_j)(h-l_2)\le 0,$$

由T的最优性，上式应取等号，即调整权值后的T仍是最优树。

定理 5.2 设有一棵带权w_1+w_2，w_3，$\cdots$，w_t的最优二元树T'，其中$w_1\le w_2\le\cdots\le w_t$，在$T'$中，让带权$w_1+w_2$的树叶产生两个儿子，分别带权$w_1$和$w_2$，则得到的二元树$T^*$是带权$w_1,\cdots,w_t$的最优二元树。

证明：由定理中的条件可知，

$$w(T^*)=w(T')+w_1+w_2。\qquad（1）$$

设T是带权$w_1,\cdots,w_t$的最优二元树，而且在T中，分别带权w_1和w_2的二树叶v_1和v_2是层数最高的二兄弟。在T中，删除树叶v_1，v_2，使它们的父结点带权w_1+w_2，则得带权w_1+w_2，w_3，$\cdots$，w_t的二元树$\tilde{T}$，显然

$$w(T)=w(\tilde{T})+w_1+w_2。\qquad（2）$$

比较（1），（2）式且考虑到T'的最优性知

$$w(T^*)\le w(T)。\qquad（3）$$

再由T的最优性知上式等号成立，即T^*是最优二元树。

给定正实数$w_1,\cdots,w_t$（$t\ge 2$），构造一棵带权$w_1,\cdots,w_t$的最优二元树的哈夫曼算法可描述如下：

1．（初始化）令$S=\{w_1,w_2,\cdots,w_t\}$，$V=\{v_1,v_2,\cdots,v_t\}$，$E=\varnothing$，v_i的权为w_i（$1\le i\le t$）；

2．从S中取出两个最小的权w_x，w_y，令$w'=w_x+w_y$；

3．$S\leftarrow(S-\{w_x,w_y\})\bigcup\{w'\}$，$V\leftarrow V\bigcup\{v'\}$，$w'$作为$v'$的权，

$E\leftarrow E\bigcup\{(v',v_x),(v',v_y)\}$；

4．判断S是否只含一个元素？若是，则停止；否则转 2。

算法结束时，$T=(V,E)$是带权$w_1,\cdots,w_t$的最优二元树，注意

$$w(T)=\sum_{v\in V, v\text{不是根}} w(v)\text{。}$$

这是从（1）式来的。

例 5.1 构造一棵带权$2,3,5,7,9,11$的最优二元树，算法过程如图 3.13 所示。

由定义，$w(T)=2\times4+3\times4+5\times3+11\times2+7\times2+9\times2=89$，

另外，$w(T)=\sum_{v\in V, v\text{是内点}} w(v)$＝5＋10＋21＋16＋37＝89。

第二种方法更简单，因为内点数比树叶数少，且只需加法。学过实变函数课程的同学，也许容易看出两种算式的内在关系，简单说，好比是 Lebesgue 积分与 Riemann 积分的关系。

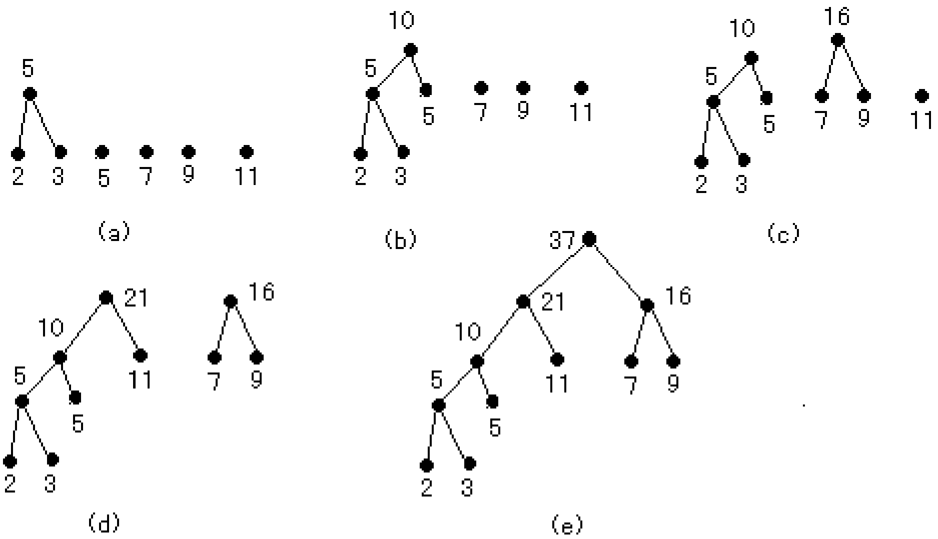

图 3.13

我们还可以推广求最优二元树的 Huffmann 算法，给出求最优m元树的算法。

对于正则m元树来说，满足下面关系：$i=\dfrac{t-1}{m-1}$，其中i为分支点数，t为树叶数。

给定t个正实数$w_1,\cdots,w_t$，且$w_1\le w_2\le\cdots\le w_t$，求一棵最优$m$（$m\ge3$）元树的算法分下面两种情况

（1） 若$\dfrac{t-1}{m-1}$为整数，说明所求的树T为正则m元树，此时可仿照求最优二元树的方法求最优m元树；

（2） 否则，设$\dfrac{t-1}{m-1}$的余数为r，$1 \le r \le m-1$，先将$r+1$个较小的权$w_1, \cdots, w_{r+1}$对应的树叶作为兄弟得到一父结点，然后将父结点看作赋权为$w_1 + \cdots + w_{r+1}$的树叶，对权$w_1 + \cdots + w_{r+1}$，w_{r+2}，$\cdots$，w_t，仿照求最优二元树的方法求最优m元树。

下面介绍最优二元树的一个重要应用问题：编码。

在计算机存储和远程通讯中，常用二进制编码来表示字符，存储（或传输）的信息由有限个字符所组成。如果给定了这些字符的种类和出现的频率，如何对它们进行二进制编码，而使所占用的比特位最少？这就是一个求最优二元树的问题。

我们首先来看编码必须满足的一些约束。

定义 5.3 设$\alpha_1 \cdots \alpha_n$是长度为n的符号串，称其子串α_1，$\alpha_1\alpha_2$，$\cdots$，$\alpha_1\alpha_2\cdots\alpha_{n-1}$分别是$\alpha_1 \cdots \alpha_n$的长度为$1$，$2$，$\cdots$，$n-1$的前缀。

定义 5.4 设$A = \{\beta_1, \cdots, \beta_m\}$是一个符号串集合，若对任意的$\beta_i, \beta_j \in A$，$i \ne j$，都有$\beta_i$不是$\beta_j$的前缀，则称$A$为前缀码，$A$中的符号串$\beta_i$（$1 \le i \le m$）称为码字。若$\beta_i$（$1 \le i \le m$）是二进制序列，则称$A$为二元前缀码。

例 5.2 $\{1, 01, 001, 000\}$是前缀码，$\{1, 11, 001, 0011\}$不是前缀码。

对给定的字符种类做的编码必须是前缀码，才能用这些码字唯一地表示信息，不致引起混淆。

例 5.3 做下列编码：$A:1$，$B:01$，$C:001$，$D:000$，那么，$1110010100000 1101$只可能表示$AAACBDCAB$。但如果做下列编码：$A:1$，$B:111$，$C:001$，$D:0001$，那么，1110010001既可能是$AAACD$，也可能是BCD。

下面来说明二元前缀码与二元树的关系。

定理 5.3 一个二元前缀码唯一对应着一棵有序二元树。

证明：给定一棵有序二元树，从根开始，对每个分支点，指向左儿子的边上标记0，指向右儿子的边上标记1。从根到每个树叶的路上边的标号组成的二元序列标记在该树叶上，则所有这些二元序列组成一个前缀码。反过来，给定一个二元前缀码，设其中码字的最大长度为

h。画一棵高度为h的有序完全二元树，每个分支点的指向左右儿子的边上分别标记0和1，从根到每个结点的路上边的标号组成的二元序列标记在该结点上。保留前缀码中码字对应的结点及到根的路上所有的结点和边，删去其余的结点和边，便得到一棵有序二元树，它的树叶对应的二元序列全体正是给定的二元前缀码。

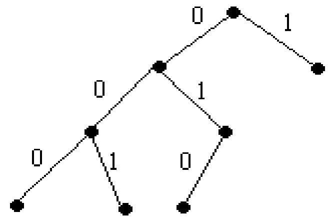

图 3.14

例 5.4 二元前缀码$\{1,000,001,010\}$对应的有序二元树如图 3.14 所示

由二元前缀码对应的有序二元树，我们就可以破译任何二进序列表示的信息。从二元树的根开始，按二元序列的排列，当遇到0，就沿着标记为0的边走，当遇到1，就沿着标记为1的边走，一直走到树叶，得到一个码字代表的字符。然后回到根，继续上述操作，知道查完二元序列。

现在比较容易看清最优编码和最优二元树的关系了。

给定字符集$\{A_1,\cdots,A_t\}$，及一个表示这些字符的二元前缀码$\{\beta_1,\cdots,\beta_t\}$，字符$A_1,\cdots,A_t$在所表示的信息中出现的频率分别为$w_1,\cdots,w_t$，码字$\beta_1,\cdots,\beta_t$的长度分别是$l_1,\cdots,l_t$，则用此二元前缀码表示信息花费的比特量为：$\sum_{i=1}^{t} l_i w_i$。这正是二元前缀码对应的带权为$w_1,\cdots,w_t$的二元树的权。所以，给定字符集$\{A_1,\cdots,A_t\}$和出现频率$w_1,\cdots,w_t$，求最优二元前缀码的问题就转化为求带权$w_1,\cdots,w_t$的最优二元树。

作业

1. 给出字符串 state act as a seat
   (a). 最优二进制编码；
   (b). 如果二进制字符串不允许带空格，求该字符串的最优二进制编码。
2. 设$t \ge 2$，$l_1,\cdots,l_t$是t个正整数，存在二元前缀码$\{\beta_1,\cdots,\beta_t\}$，使得码字$\beta_1,\cdots,\beta_t$的长度分别是$l_1,\cdots,l_t$的充要条件是

$$\sum_{i=1}^{t} 2^{-l_i} \le 1 \quad \text{（Kraft 不等式）。}$$

3. 给定一个点集$P=\{p_1,\cdots,p_n\}$和其上的距离函数d，d满足：

（1）$d(p_i,p_j)=d(p_j,p_i)>0$，$i\neq j$；

（2）$d(p_i,p_i)=0$。

我们可以如下构造P上的分层距离τ： 构作一个含n片叶子的根树T，对T的每个结点v赋以高度h_v， h_v满足

（1）如果v是叶子，则$h_v=0$；

（2）如果u是v的祖先，则$h_u\geq h_v$。

将P中的点分别放在不同的叶子上，对任意两点p_i和p_j，定义$\tau(p_i,p_j)=h_v$，其中v是p_i和p_j最近的共同祖先。我们说分层距离τ与d是相容的，如果对任意两点p_i和p_j，$\tau(p_i,p_j)\leq d(p_i,p_j)$。请设计多项式时间复杂度的算法，以$P=\{p_1,\cdots,p_n\}$和其上的距离函数$d$为输入，以满足下面性质的分层距离$\tau$为输出：

（1）τ与d是相容的；

（2）若τ'是任意与d相容的分层距离，则对任意两点p_i和p_j，$\tau'(p_i,p_j)\leq\tau(p_i,p_j)$。