第4章　树

对于大量的输入数据，链表的线性访问时间太慢，不宜使用。本章讨论一种简单的数据结构，其大部分操作的运行时间平均为 $O(\log N)$。我们还要简述对这种数据结构在概念上的简单的修改，它保证了在最坏情形下上述的时间界。此外，还讨论了第二种修改，对于长的指令序列它基本上给出每种操作的 $O(\log N)$运行时间。

我们涉及到的这种数据结构叫做**二叉查找树**(binary search tree)。二叉查找树是两种库集合类 TreeSet 和 TreeMap 实现的基础，它们用于许多应用之中。在计算机科学中树(tree)是非常有用的抽象概念，因此，我们将讨论树在其他更一般的应用中的使用。在这一章，我们将

- 看到树是如何用于实现几个流行的操作系统中的文件系统的。
- 看到树如何能够用来计算算术表达式的值。
- 指出如何利用树支持以 $O(\log N)$平均时间进行的各种搜索操作，以及如何细化以得到最坏情况时间界 $O(\log N)$。我们还将讨论当数据被存放在磁盘上时如何来实现这些操作。
- 讨论并使用 TreeSet 类和 TreeMap 类。

4.1　预备知识

树(tree)可以用几种方式定义。定义树的一种自然的方式是递归的方式。一棵树是一些节点的集合。这个集合可以是空集；若不是空集，则树由称做**根**(root)的节点 r 以及 0 个或多个非空的(子)树 T_1，T_2，…，T_k 组成，这些子树中每一棵的根都被来自根 r 的一条有向的**边**(edge)所连结。

每一棵子树的根叫做根 r 的儿子(child)，而 r 是每一棵子树的根的**父亲**(parent)。图 4-1 显示用递归定义的典型的树。

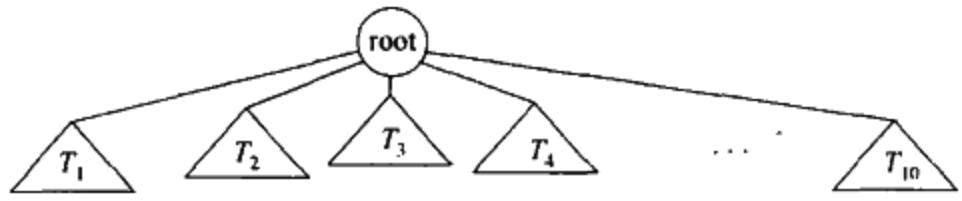

图 4-1　一般的树

从递归定义中我们发现，一棵树是 N 个节点和 $N-1$ 条边的集合，其中的一个节点叫做根。存在 $N-1$ 条边的结论是由下面的事实得出的：每条边都将某个节点连接到它的父亲，而除去根节点外每一个节点都有一个父亲(见图 4-2)。

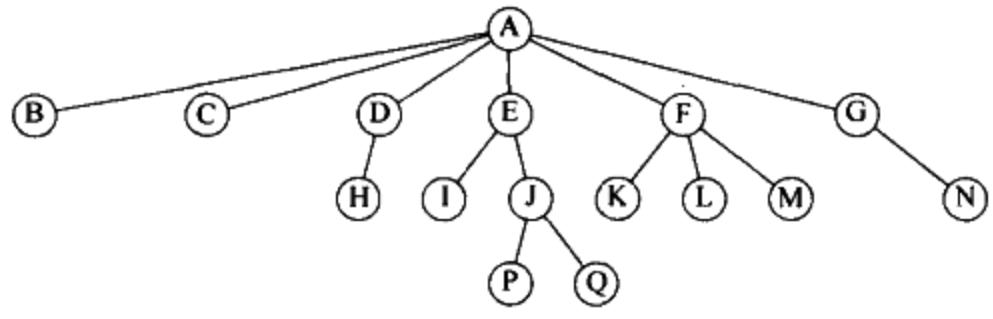

图 4-2　一棵树