6.8.1 二项队列结构

二项队列(binomial queue)与我们已经看到的所有优先队列的实现的区别在于，一个二项队列不是一棵堆序的树，而是堆序的树的集合，称为**森林**(forest)。每一棵堆序树都是有约束的形式，叫做**二项树**(binomial tree，后面将看到该名称的由来是显然的)。每一个高度上至多存在一棵二项树。高度为0的二项树是一棵单节点树；高度为 k 的二项树 B_k 通过将一棵二项树 B_{k-1} 附接到另一棵二项树 B_{k-1} 的根上而构成。图6-34显示二项树 B_0、B_1、B_2、B_3 以及 B_4。

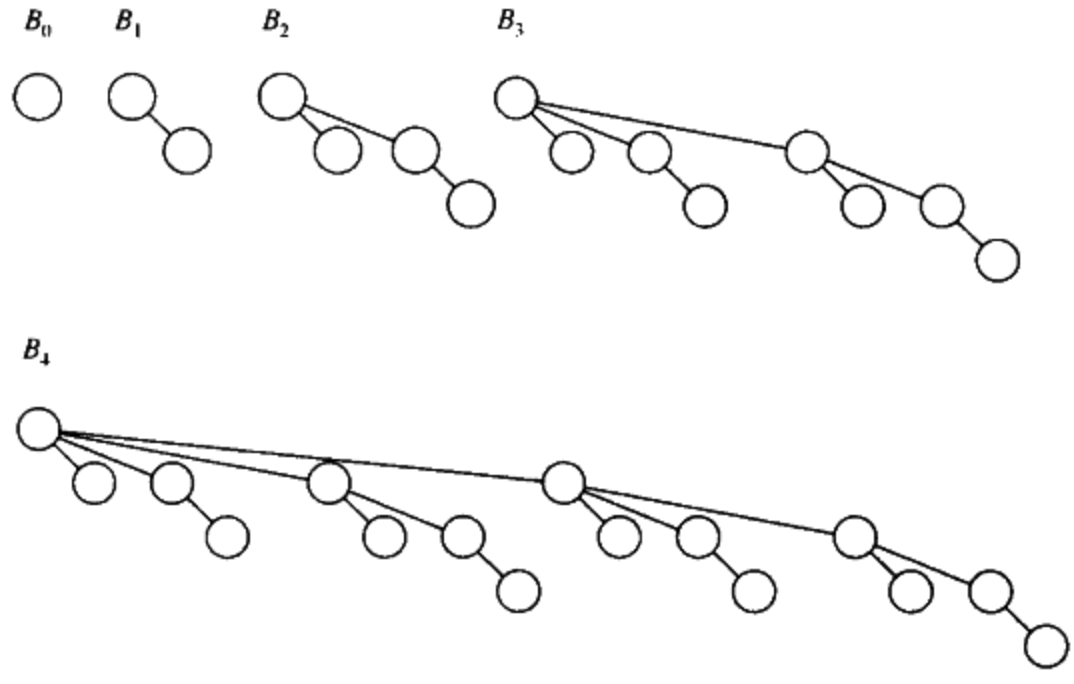

图6-34 二项树 B_0、B_1、B_2、B_3 以及 B_4

从图中看到，二项树 B_k 由一个带有儿子 $B_0, B_1, \ldots, B_{k-1}$ 的根组成。高度为 k 的二项树恰好有 2^k 个节点，而在深度 d 处的节点数是二项系数 $\binom{k}{d}$。如果我们把堆序施加到二项树上并允许任意高度上最多一棵二项树，那么就能够用二项树的集合表示任意大小的优先队列。例如，大小为13的优先队列可以用森林 B_3，B_2，B_0 表示。我们可以把这种表示写成1101，它不仅以二进制表示了13，而且也表示这样的事实：在上述表示中，B_3，B_2，B_0 出现，而 B_1 则没有。

作为一个例子，6个元素的优先队列可以表示为图6-35中的形状。

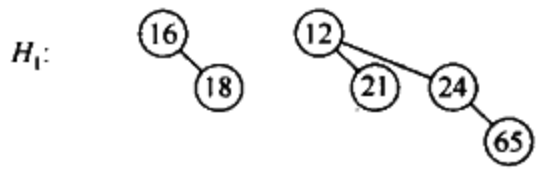

图6-35 具有6个元素的二项队列 H_1

6.8.2 二项队列操作

此时，最小元可以通过搜索所有的树的根来找出。由于最多有 $\log N$ 棵不同的树，因此找到最小元的时间可以为 $O(\log N)$。另外，如果我们记住当最小元在其他操作期间变化时更新它，那么也可保留最小元的信息并以 $O(1)$ 时间执行这种操作。

合并两个二项队列在概念上是一个容易的操作，我们将通过例子描述它。考虑两个二项队列 H_1 和 H_2，它们分别具有6个和7个元素，见图6-36。

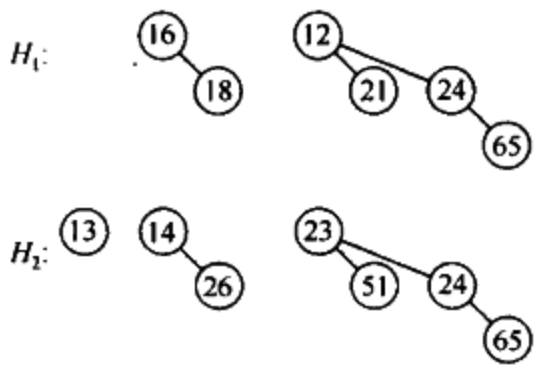

图6-36 两个二项队列 H_1 和 H_2