树的非叶结点之数量。这个切分阶段的树是一棵满二叉树，叶结点是 n 个，那么非叶结点的数量就是 $n-1$ 个，所以切分的次数也就是 $n-1$ 次，如图 5-2 所示。如果切分数据的时候并不重新生成新的数据，只是生成切分边界的下标，那么时间复杂度就是 $O(n-1)$。

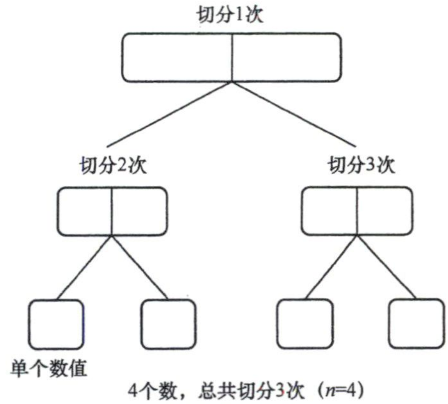

图 5-2 n 个数值的二分次数

在数据归并阶段，情况稍微复杂一些。和切分不用，不同的合并步骤意味着不同的数组长度。这个时候，我们可以看到二叉树的高度为 $\log_2 n$，如图 5-3 所示。另外，无论在树的哪一层，每次归并都需要扫描整个长度为 n 的数组，因此归并阶段的时间复杂度为 $O(n\times\log_2 n)$。两个阶段加起来的时间复杂度为 $O(n-1)+O(n\times\log_2 n)$，最终简化为 $O(n\log_2 n)$，非常直观。

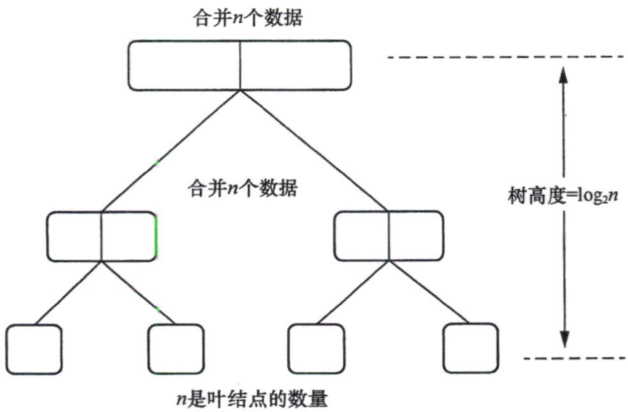

图 5-3 n 个数据的归并

当然，除了图论，很多简单的图表也能帮助我们做分析。例如，在使用动态规划的时候，我们经常要画出状态转移的表格。看到这类表格，可以很容易地得出该算法的时间复杂度和空间复杂度。以编辑距离为例，参看表 5-1，我们可以发现每个单元格都对应了 3 次计算，以及一个存储单元，而总共的单元格数量为 $m\times n$，其中 m 为第一个字符串的长度，n 为第二个字符串的长度。所以，很快就能得出这种算法的时间复杂度为 $O(3\times m\times n)$，简化为 $O(m\times n)$，空间复杂度为 $O(m\times n)$。

表 5-1 动态规划的状态转移表格

	空 *B*	m	o	u	s	e
空 *A*	0	1	2	3	4	5
m	1	min(2,2,0)=0	min(3,1,2)=1	min(4,2,3)=2	min(5,3,4)=3	min(6,4,5)=4