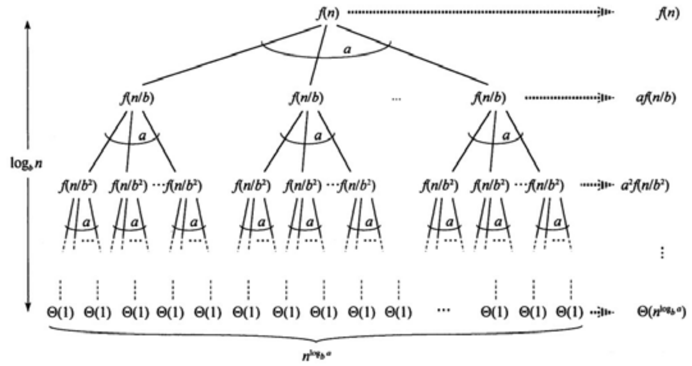
主方法

主方法是对分治法进行时间复杂度计算的一种方法，可将其看作一种“黑盒工具”。

主方法定义：若算法的运算时间 $\mathrm{T}(n)=a\mathrm{T}\left(\frac{n}{b}\right)+cn^d$，其中 a≥1 为子问题个数，b≥1 为问题规模减小的倍数，c≥0 为递归过程外的时间复杂度系数，d≥0 为递归过程之外的时间复杂度指数，则：

$$\mathrm{T}(n)=\begin{cases}O(n^d) & a<b^d\\ O(n^d logn) & a=b^d\\ O(n^{log_b a}) & a>b^d\end{cases}$$

证明：对于以下递归树：



1.递归树共有 $log_b n$ 层，其中第 j 层，共有 a^j 个子问题

2.每个子问题的规模为 $\frac{n}{b^j}$

3.对第 j 层的一个子问题，其除递归之外的工作量（这里不说时间复杂度，因为还没有确定 a 和 b^d 的关系）不会超过 $c\cdot\left(\frac{n}{b^j}\right)^d$

4.第 j 层的工作量上界为 $a^j\cdot c\cdot\left(\frac{n}{b^j}\right)^d=c\cdot n^d\cdot\left(\frac{a}{b^d}\right)^j$ （1）

5.所有层总的工作量上界为 $cn^d\sum_{j=0}^{\log_b n}\left(\frac{a}{b^d}\right)^j$ （2）

对(1)式的解释：a 为子问题的增长率（rate of subproblem proliferation, RSP），b^d 为子问题工作量的缩减率（rate of work shrinkage, RWS）

1.当 $a=b^d$，即 RSP=RWS 时，每一层的工作量不变，$(2)=cn^d(\log_b n+1)=O(cn^d\log n)$

2.当 $a<b^d$，即 RSP<RWS 时，令 $\frac{a}{b^d}=x$，由等差数列求和公式可知，$\frac{(2)}{cn^d}=\frac{1-x^{\log_b n+1}}{1-x}<\frac{1}{1-x}$，因此(2)式求和部分上界为常量，$(2)=O(n^d)$

3.当 $a>b^d$，即 RSP>RWS 时，$\frac{(2)}{cn^d}=\frac{x^{\log_b n+1}-1}{x-1}=O\left(x^{\log_b n}\right)=O\left(\left(\frac{a}{b^d}\right)^{\log_b n}\right)=O(\frac{n^{\log_b a}}{n^d})$，因此，$(2)=O(n^{\log_b a})$

至此，证明结束。