数据结构与算法

计算机学院

朱晨阳 副教授

zhuchenyang07@nudt.edu.cn

整数分划问题

- 把一个正整数 $\boldsymbol{n}$ 表示成形如 $\boldsymbol{n} = \boldsymbol{n1} + \boldsymbol{n2} + \cdots + \boldsymbol{nk}$ 的一系列正整数的和，称为 $\boldsymbol{n}$ 的一个分划。其中，$\boldsymbol{ni} \geq \mathbf{1}, \boldsymbol{i} = \mathbf{1}, \mathbf{2}, \ldots, \boldsymbol{k}, \boldsymbol{k} \geq \mathbf{1}$，且满足条件 $\boldsymbol{n1} \geq \boldsymbol{n2} \geq \cdots \geq \boldsymbol{nk} \geq \mathbf{1}$
- 形如 $\boldsymbol{n} = \boldsymbol{n1} + \boldsymbol{n2} + \cdots + \boldsymbol{nk}$ 的不同表达式的个数称为整数 $\boldsymbol{n}$ 的分划数。记为 $\boldsymbol{Pn}$ 或 $\boldsymbol{P}(\boldsymbol{n})$
- 问题：给定任意正整数 $\boldsymbol{n}$，求其分划数 $\boldsymbol{P}(\boldsymbol{n})$

整数分划问题

- 整数 6 的分划

6；

5+1；

4+2， 4+1+1；

3+3， 3+2+1， 3+1+1+1 ；

2+2+2， 2+2+1+1， 2+1+1+1+1 ；

1+1+1+1+1+1

→ $P(6)=11$

整数分划问题

- 引入辅助变量 $\boldsymbol{m}$
- $\boldsymbol{Q(n, m)}$ 定义为
  - $\boldsymbol{n}$ 的分划数，且满足最大加数不大于 $\boldsymbol{m}$
- $\boldsymbol{P(n) = Q(n, n)}$

整数分划问题

$$Q(n,m)=\begin{cases} 1, & m=1 \text{ or } n=1 \\ Q(n,n), & n<m \\ 1+Q(n,n-1), & n=m \\ Q(n,m-1)+Q(n-m,m), & n>m>1 \end{cases}$$

整数分划问题

- 算法 Q(n, m)
  - 输入：正整数 n
  - 输出： n 的最大加数不大于 m 的分划数

1. if n<1 or m<1
2. return 0
3. if n==1 or m==1
4. return 1
5. if n<m
6. return Q(n, n)
7. if n==m
8. return Q(n, m-1)+1
9. return Q(n, m-1)+Q(n-m, m)

递归效率分析

- 递归方程求解
  - 迭代法
  - 生成函数求解
  - 特征方程求解
  - 递归树方法

迭代法

- **算法** $Hanoi(n, A, B, C)$
  - $T(n) = 2T(n-1) + 1, \; T(1) = 1$

$$
\begin{aligned}
T(n) &= 2\underline{T(n-1)}+1 \\
&= 2(\underline{2T(n-2)+1})+1 \\
&\cdots \\
&= 2^{n-1}+2^{n-2}+\ldots+1 \\
&= 2^{n}-1
\end{aligned}
$$

代入初值求和

迭代法

- 不断用递推方程的右部替代左部
- 每次替换，随着 n 的降低在和式中多出一项
- 直到出现初值停止迭代
- 将初值代入并对和式求和
- 可用数学归纳法验证解的正确性

迭代法

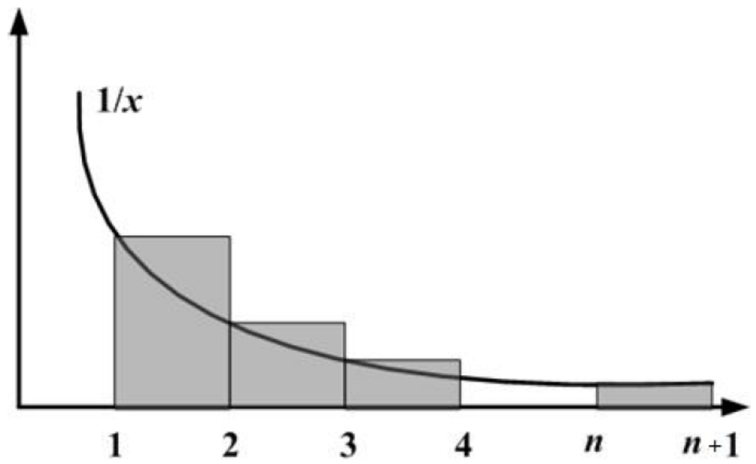
- 等差、等比数列与调和级数

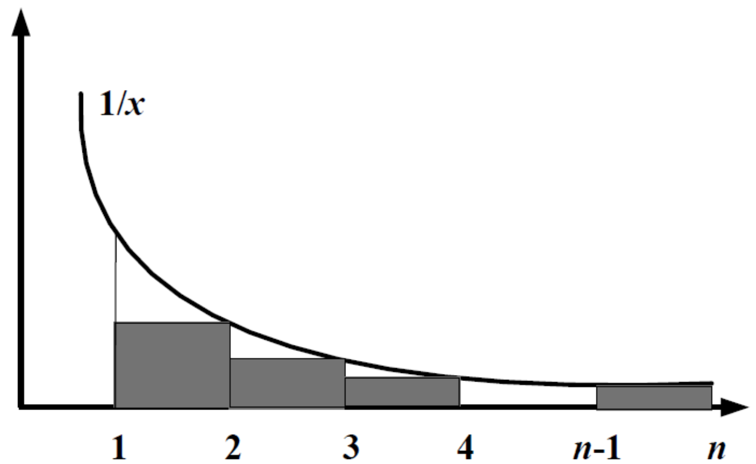
$$\sum_{k=1}^{n} a_k = \frac{n(a_1 + a_n)}{2}$$

$$\sum_{k=0}^{n} aq^k = \frac{a(a_1 - q^{n+1})}{1-q}, \sum_{k=0}^{\infty} aq^k = \frac{a}{1-q}(q<1)$$

$$\sum_{k=1}^{n} \frac{1}{k} = \ln n + O(1)$$

调和级数求和





$$\sum_{k=1}^{n}\frac{1}{k}\geq\int_{1}^{n+1}\frac{\mathrm{d}x}{x}=\ln(n+1)$$

$$\sum_{k=2}^{n}\frac{1}{k}\leq\int_{1}^{n}\frac{\mathrm{d}x}{x}=\ln n$$

$$\sum_{k=1}^{n}\frac{1}{k}=1+\sum_{k=2}^{n}\frac{1}{k}\leq\ln n+1$$

$$\sum_{k=1}^{n}\frac{1}{k}=\ln n+O(1)$$

换元迭代

- 算法 $MergeSort(A,p,r)$ 最坏情况时间复杂度递推公式为
- $W(n)=2W(n/2)+n-1, W(1)=0$

换元迭代

- $\boldsymbol{W(n) = 2W(n/2) + n - 1, W(1) = 0}$
- 令$n = 2^k$，换元迭代求解

$$
\begin{aligned}
W(n) &= 2W(2^{k-1}) + 2^k - 1 \\
&= 2[2W(2^{k-2}) + 2^{k-1} - 1] + 2^k - 1 \\
&= 2^2[2W(2^{k-3}) + 2^{k-2} - 1] + 2^k - 2 + 2^k - 1 \\
&\circ\ \circ\ \circ \\
&= 2^k W(1) + k2^k - (2^{k-1} + 2^{k-2} + \ldots + 2 + 1)
\end{aligned}
$$

换元迭代

$$= 2^k W(1) + k2^k - (2^{k-1} + 2^{k-2} + \ldots + 2 + 1)$$

$$= k2^k - \underline{(2^k - 1)}$$

换回 $k = \log n$

$$= n\log n - n + 1$$

$$= O(n\log n)$$

换元迭代

- 将对 n 的递推式换成对其他变元k的递推式
  - 对 k 直接迭代
  - 将解（关于 k 的函数）转换成关于 n 的函数
- 常用替换 $n = 2^k$ （二分法）
  - 主要关心算法的渐进时间复杂度，因此当 n 不是 2 的整幂时，可采用“规模”整幂化或近似等分法。

递归效率分析

- 设序列 $a0, a1, \ldots, an, \ldots$, 简记为 $\{an\}$, 一个把 an 与某些个 $ai(i < n)$ 联系起来的等式叫做关于序列的递推方程
- 递推方程的求解：
  - 给定关于序列 $\{an\}$ 的递推方程和若干初值，计算an

特征方程法

- k阶常系数线性齐次递归方程可定义为
- $\begin{cases} f(n) = a_1 f(n-1) + a_2 f(n-2) + \cdots + a_k f(n-k) \\ f(i) = b_i, 0 \le i < k \end{cases}$
- 用x^n代替$f(n)$，得到上述递归方程的特征方程
- $x^n = a_1 x^{n-1} + a_2 x^{n-2} + \cdots + a_k x^{n-k}$

特征方程法

- 当特征方程的k个根互不相同时，令$q_1, q_2, \ldots, q_k$是特征方程的k个不同的根，递归方程的通解为
- $f(n) = c_1 q_1^n + c_2 q_2^n + \cdots + c_k q_k^n$

特征方程法

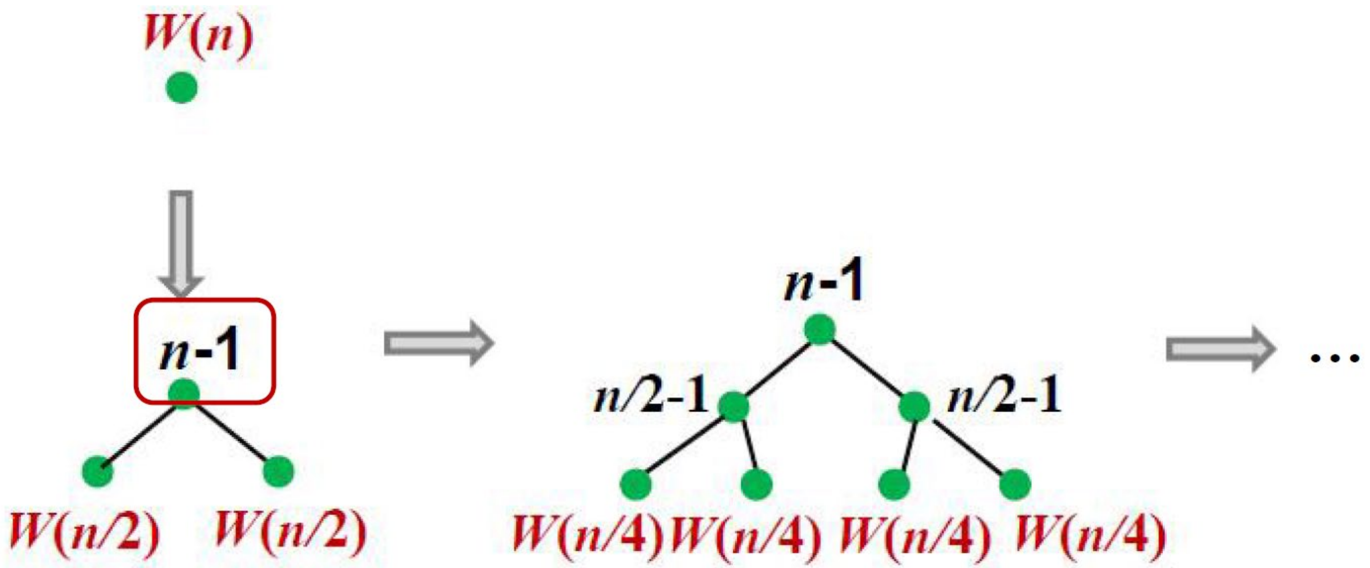
- 当特征方程的k个根有r个重根时，$q_{\mathrm{i}}, q_{i+1}, \cdots, q_{i+r-1}$
- $f(n) = c_1 q_1^n + c_2 q_2^n + \cdots + (c_i + c_{i+1} n + \cdots + c_{i+r-1} n^{r-1}) q_i^n + \cdots + c_k q_k^n$

特征方程法

- 递推方程：$\mathrm{T}_n = 4(T_{\mathrm{n}-1} - T_{\mathrm{n}-2})$
- 初值： $\mathrm{T}_1 = 0, \mathrm{T}_2 = 4$
- 如何求解通项公式?

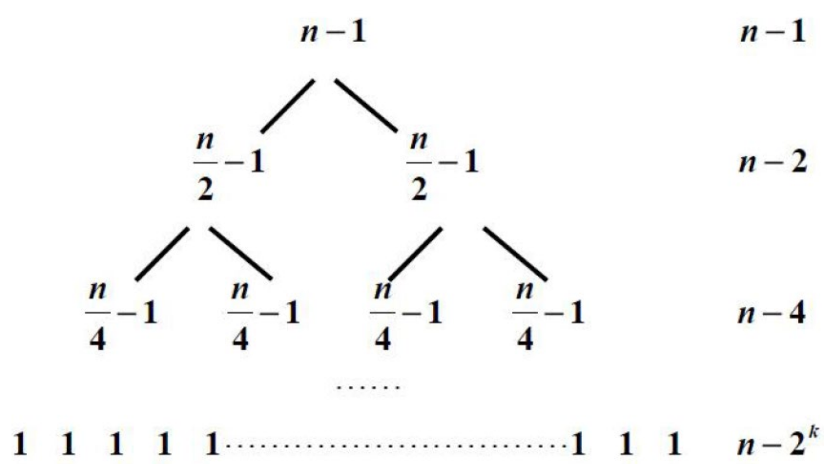
递归树

- 二分归并排序
- $W(n) = 2W(n/2) + \boxed{n-1}, W(1) = 0$

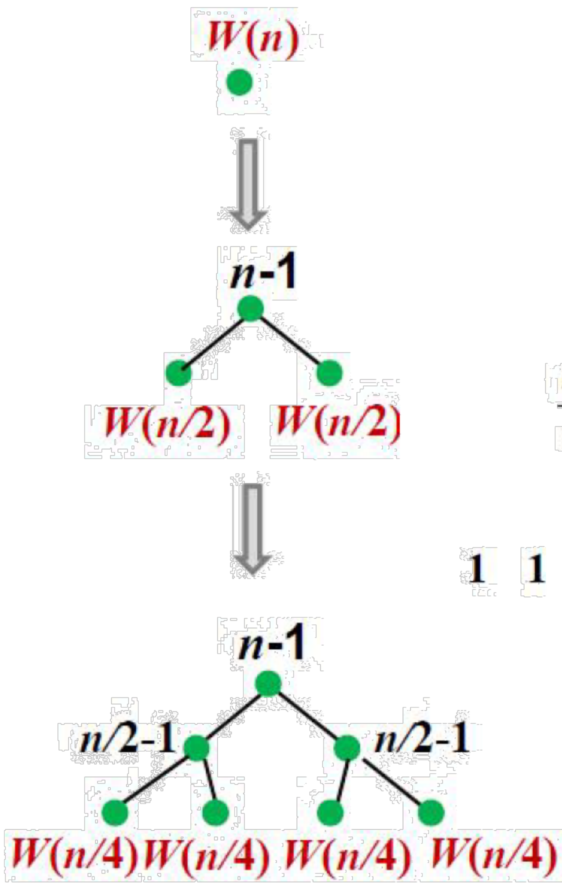


递归树

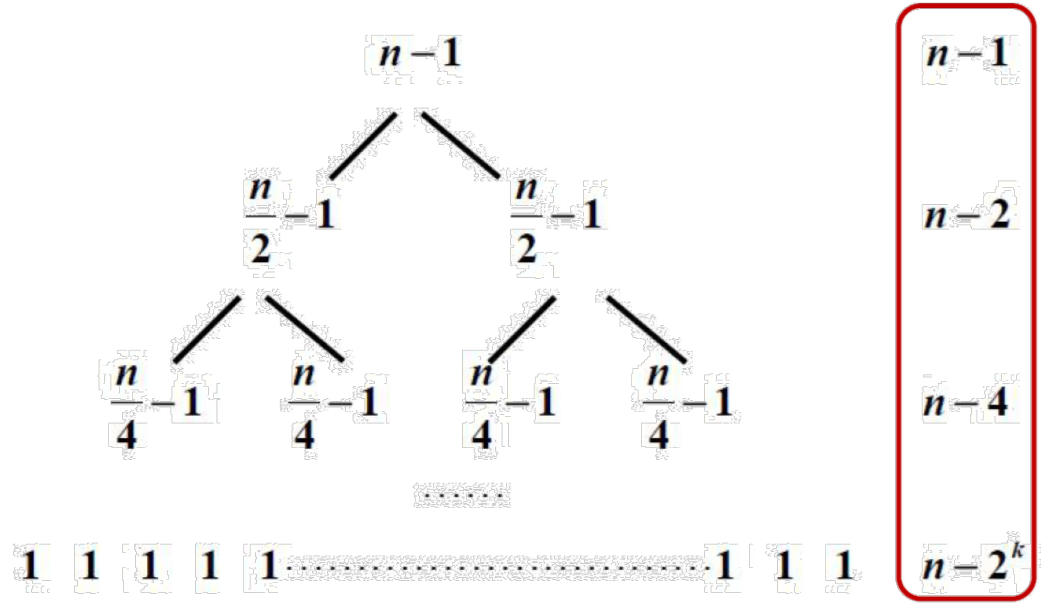
- 递归树是迭代计算的模型
- 递归树的生成过程与迭代过程一致
- 递归树上所有项恰好是迭代之后产生和式中的项
- 对递归树上的项求和就是迭代后方程的解



递归树



生成过程



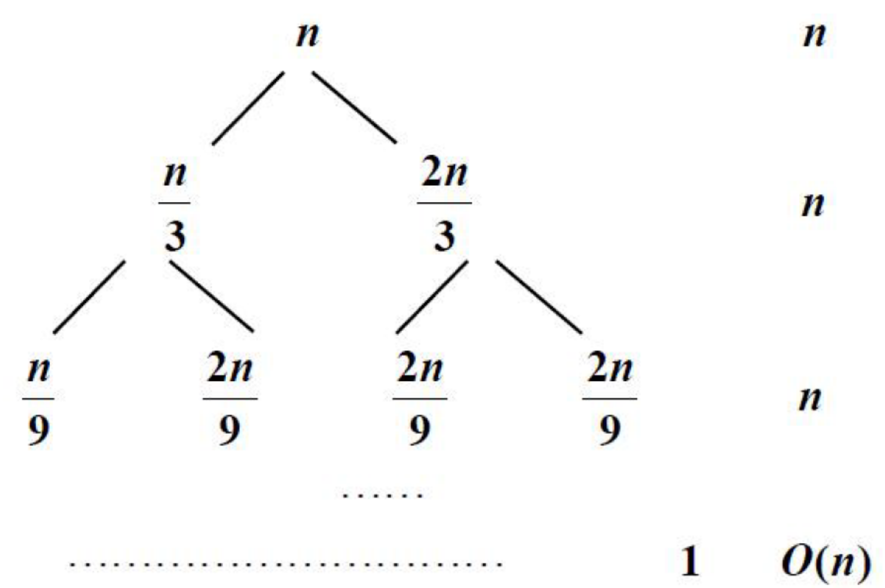
$$W(n)=2W(n/2)+n-1,\ n=2^k,\ W(1)=0$$

$$W(n)=(n-1)+(n-2)+\dots+(n-2^{k-1})$$

$$=kn-(2^k-1)=n\log n-n+1$$

递归树的应用实例

- 递推式：$T(n) = T(n/3) + T(2n/3) + n, T(1) = 1$



层数 k：$n(2/3)^k = 1 \Rightarrow (3/2)^k = n \Rightarrow k = O(\log_{3/2} n)$

$\Rightarrow \quad T(n) = O(n\log n)$

主定理

- 二分检索： $W(n) = W(n/2) + 1$
- 二分归并排序： $W(n) = 2W\left(\frac{n}{2}\right) + n - 1$
- 求解递推方程
- $T(n) = aT(n/b) + f(n)$
  - a: 归约后的子问题个数
  - n/b: 归约后子问题的规模
  - $f(n)$: 归约过程及组合子问题解的工作量

多项式大于小于

- $f(x)$多项式大于$g(x)$:
  - 存在实数$\varepsilon > 0$当,使得$f(x) > g(x) * x^{\varepsilon}$
- $f(x)$多项式小于$g(x)$:
  - 存在实数$\varepsilon > 0$,使得$f(x) < g(x) * x^{\varepsilon}$

主定理

- $T(n) = aT(n/b) + f(n)$

1. $f(n)$多项式小于$n^{log_b^a}$，复杂度为$\Theta(n^{log_b^a})$
2. $f(n) = \Theta(n^{log_b^a})$，复杂度为$\Theta(n^{log_b^a}\mathrm{logn})$
3. $f(n)$多项式大于$n^{log_b^a}$

   且$\exists \mathrm{c} < 1,$与所有足够大的n，有$\mathrm{a}f\left(\frac{n}{b}\right) \le cf(n)$

   复杂度为$\Theta(f(n))$

主定理

- 二分检索 $W(n) = W(n/2) + 1, W(1) = 1$
  - $a = 1, b = 2, n^{log_2^1} = 1, f(n) = 1,$
  - 属于情况2，因此
  - $W(n) = \Theta\left(n^{log 1} \log n\right) = \Theta(\log n)$
- 二分归并排序 $W(n) = 2W(n/2) + n - 1, W(1) = 0$
  - $a = 2, b = 2, n^{log_2^2} = n, f(n) = n - 1,$
  - 属于情况2，因此
  - $W(n) = \Theta(n^{log 2} \log n) = \Theta(nlog\, n)$

主定理

- 递推方程 $T(n) = 9T(n/3) + n$
  - 递推式中 $a = 9, b = 3, f(n) = n,$ 有
  - $n^{log_3^9} = n^2, f(n) = O(n^{log_3^9 - 1})$
- 属于主定理的第一种情况，其中 $\varepsilon = 1$，根据主定理得
- $T(n) = \Theta(n^2)$

主定理

设 $n=b^k$，即 $k=\log_b n$

$$T(n)=a\underline{T(\frac{n}{b})}+f(n)$$

$$=a\underline{\left[aT(\frac{n}{b^2})+f(\frac{n}{b})\right]}+f(n)$$

$$=a^2T(\frac{n}{b^2})+af(\frac{n}{b})+f(n)$$

$$=\ldots$$

主定理

$$=a^kT(\frac{n}{b^k})+a^{k-1}f(\frac{n}{b^{k-1}})+\ldots+af(\frac{n}{b})+f(n)$$

$$=\boxed{a^kT(1)}+\sum_{j=0}^{k-1}a^jf(\frac{n}{b^j})$$

$$k=\log_b n$$
$$a^{\log_b n}=n^{\log_b a}$$

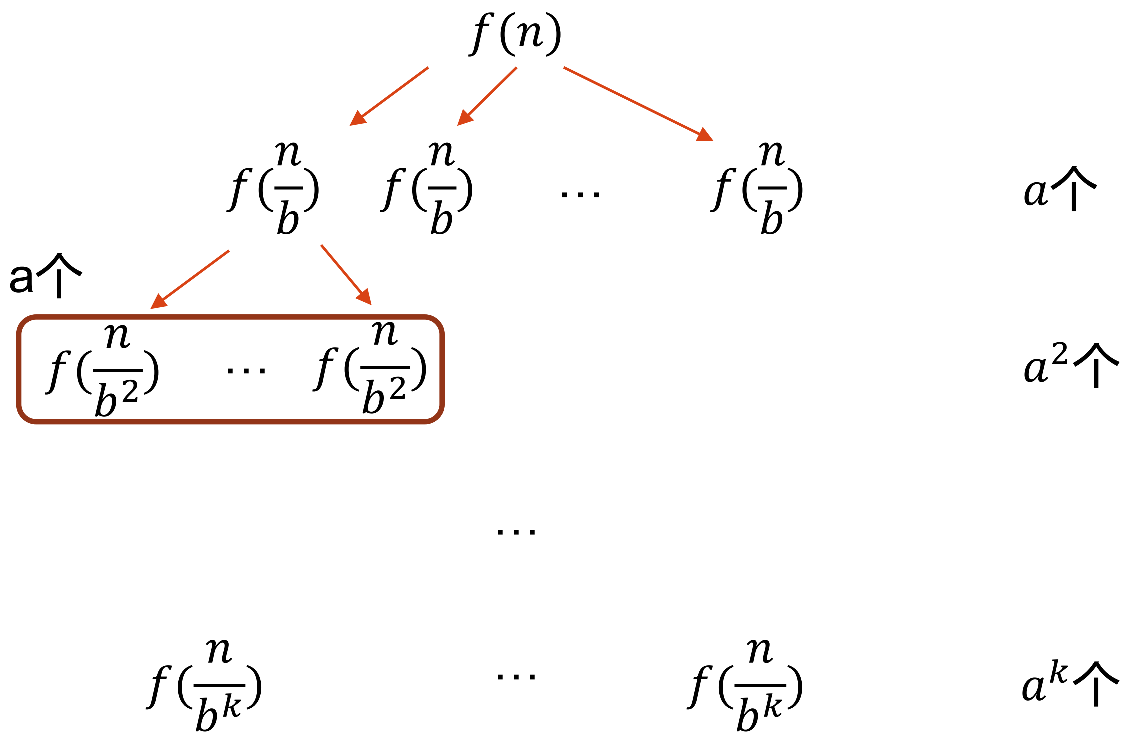
$$=c_1\boxed{n^{\log_b a}}+\sum_{j=0}^{k-1}a^jf(\frac{n}{b^j})$$

$$T(1)=c_1$$

第一项为所有最小子问题的计算工作量

第二项为迭代过程归约到子问题及综合解的工作量

主定理


$f(n)$
$f(\frac{n}{b})$ $f(\frac{n}{b})$ $\cdots$ $f(\frac{n}{b})$
a个
a个
$f(\frac{n}{b^2})$ $\cdots$ $f(\frac{n}{b^2})$
a^2个
$\cdots$
$f(\frac{n}{b^k})$ $\cdots$ $f(\frac{n}{b^k})$
a^k个

主定理-情况1

$$f(n) = O(n^{\log_b a - \varepsilon})$$

$$T(n) = c_1 n^{\log_b a} + \sum_{j=0}^{k-1} a^j \boxed{f(\frac{n}{b^j})}$$

代入条件

$$= c_1 n^{\log_b a} + O\left(\sum_{j=0}^{\log_b n - 1} a^j \boxed{(\frac{n}{b^j})^{\log_b a - \varepsilon}} \right)$$

$$= c_1 n^{\log_b a} + O\left(n^{\log_b a - \varepsilon} \sum_{j=0}^{\log_b n - 1} \frac{a^j}{(b^{\log_b a - \varepsilon})^j} \right)$$

与求和无关的项外提

主定理-情况1

$$=c_1 n^{\log_b a}+O\left(n^{\log_b a-\varepsilon}\sum_{j=0}^{\log_b n-1}\boxed{\frac{a^j}{(b^{\log_b a-\varepsilon})^j}}\right)$$

化简

$$\boxed{\frac{1}{(b^{\log_b a-\varepsilon})^j}=\frac{b^{\varepsilon j}}{(b^{\log_b a})^j}=\frac{b^{\varepsilon j}}{a^j}}$$

n^{ε}

$$=c_1 n^{\log_b a}+O\left(n^{\log_b a-\varepsilon}n^{\varepsilon}\right)=\Theta(n^{\log_b a})$$

主定理-情况2

$$T(n) = c_1 n^{\log_b a} + \sum_{j=0}^{k-1} a^j \boxed{f\left(\frac{n}{b^j}\right)}$$

$$\boxed{f(n) = \Theta(n^{\log_b a})}$$

代入条件

$$= c_1 n^{\log_b a} + \Theta\left(\sum_{j=0}^{\log_b n-1} a^j \boxed{\left(\frac{n}{b^j}\right)^{\log_b a}}\right)$$

$$= c_1 n^{\log_b a} + \Theta\left(n^{\log_b a} \boxed{\sum_{j=0}^{\log_b n-1}} \frac{a^j}{a^j}\right)$$

与求和无关的项外提

$$= c_1 n^{\log_b a} + \Theta\left(n^{\log_b a} \log n\right)$$

$$= \boxed{\Theta\left(n^{\log_b a} \log n\right)}$$

主定理-情况3

$$f(n) = \Omega(n^{\log_b a+\varepsilon}) \quad (1)$$

$$af(\frac{n}{b}) \le cf(n) \quad (2)$$

$$T(n) = c_1 n^{\log_b a} + \sum_{j=0}^{k-1} a^j f(\frac{n}{b^j})$$

$$\le c_1 n^{\log_b a} + \sum_{j=0}^{\log_b n-1} c^j f(n)$$

$$a^j f(\frac{n}{b^j}) \le a^{j-1} cf(\frac{n}{b^{j-1}})$$

$$= c^1 a^{j-1} f(\frac{n}{b^{j-1}}) \le \ldots \le c^j a^0 f(n)$$

主定理-情况3

$$f(n) = \Omega(n^{\log_b a+\varepsilon}) \quad (1)$$

$$af(\frac{n}{b}) \le cf(n) \quad (2)$$

$$T(n) \le c_1 n^{\log_b a} + \boxed{\sum_{j=0}^{\log_b n-1} c^j} f(n)$$

等比数列
$c < 1$

$$= c_1 n^{\log_b a} + f(n) \boxed{\frac{c^{\log_b n} - 1}{c-1}}$$

$$= \underline{c_1 n^{\log_b a}} + \underline{\Theta(f(n))}$$

$$= \Theta(f(n))$$

主定理-练习

- $T(n) = 3T\left(\frac{n}{2}\right) + n^2$
- $T(n) = 4T\left(\frac{n}{2}\right) + n^2$
- $T(n) = T\left(\frac{n}{2}\right) + 2^n$
- $T(n) = 2^n T\left(\frac{n}{2}\right) + n^n$
- $T(n) = 16T\left(\frac{n}{4}\right) + n$