

算法设计与分析



蒋婷婷

上节课回顾

- 函数渐进的界
- 数学基础
 - 对数函数
 - 阶乘
 - 取整函数
 - 求和
 - 估计和式上界

递推方程的求解

设序列 $a_0, a_1, \dots, a_n, \dots$, 简记为 $\{a_n\}$, 一个把 a_n 与某些个 $a_i (i < n)$ 联系起来的等式叫做关于序列 $\{a_n\}$ 的**递推方程**

求解方法:

迭代法

直接迭代: 插入排序最坏情况下时间分析

换元迭代: 二分归并排序最坏情况下时间分析

差消迭代: 快速排序平均情况下的时间分析

迭代模型: 递归树

尝试法: 快速排序平均情况下的时间分析

主定理: 递归算法的分析

迭代：Hanoi塔

$$T(n) = 2 T(n-1) + 1, \quad T(1) = 1,$$

迭代

$$\begin{aligned} T(n) &= 2T(n-1) + 1 \\ &= 2[2T(n-2) + 1] + 1 = 2^2T(n-2) + 2 + 1 \\ &= 2^3T(n-3) + 2^2 + 2 + 1 = \dots \\ &= 2^{n-1}T(1) + 2^{n-2} + \dots + 2 + 1 \\ &= 2^n - 1 \end{aligned}$$

插入排序

算法1.4 InsertSort(A, n) // A 为 n 个数的数组

1. for $j \leftarrow 2$ to n
 2. $x \leftarrow A[j]$
 3. $i \leftarrow j-1$ // 行3到行7把 $A[j]$ 插入 $A[1..j-1]$ 之中
 4. while $i > 0$ and $x < A[i]$ do
 5. $A[i+1] \leftarrow A[i]$
 6. $i \leftarrow i-1$
 7. $A[i+1] \leftarrow x$
- $$\begin{cases} W(n) = W(n-1) + n - 1 \\ W(1) = 0 \end{cases}$$

$$\begin{aligned} W(n) &= W(n-1) + n - 1 \\ &= [W(n-2) + n - 2] + n - 1 = W(n-2) + (n-2) + (n-1) \\ &= [W(n-3) + n - 3] + (n-2) + (n-1) = \dots \\ &= W(1) + 1 + 2 + \dots + (n-2) + (n-1) \\ &= 1 + 2 + \dots + (n-2) + (n-1) = n(n-1)/2 \end{aligned}$$

二分归并排序

算法1.5 MergeSort(A, p, r) // 归并排序数组 $A[p..r]$

1. if $p < r$
2. then $q \leftarrow \lfloor (p+r)/2 \rfloor$
3. MergeSort(A, p, q)
4. MergeSort($A, q+1, r$)
5. Merge(A, p, q, r)

$$\begin{cases} W(n) = 2W(n/2) + n - 1 \\ W(1) = 0 \end{cases}$$

归并过程

算法1.6 Merge(A, p, q, r) // 将排序数组 $A[p..q]$ 与 $A[q+1, r]$ 合并

1. $x \leftarrow q - p + 1, y \leftarrow r - q$ // x, y 分别为两个子数组的元素数

2. 将 $A[p..q]$ 复制到 $B[1..x]$, 将 $A[q+1..r]$ 复制到 $C[1..y]$

3. $i \leftarrow 1, j \leftarrow 1, k \leftarrow p$

4. While $i \leq x$ and $j \leq y$ do

5. if $B[i] \leq C[j]$ // B 的首元素不大于 C 的首元素

6. $A[k] \leftarrow B[i]$ // 将 B 的首元素放到 A 中

7. $i \leftarrow i + 1$

8. else

9. $A[k] \leftarrow C[j]$

10. $j \leftarrow j + 1$

11. $k \leftarrow k + 1$

12. if $i > x$ then 将 $C[j..y]$ 复制到 $A[k..r]$ // B 已经是空数组

13. else 将 $B[i..x]$ 复制到 $A[k..r]$ // C 已经是空数组

换元迭代

$$\begin{cases} W(n) = 2W(n/2) + n - 1, & n = 2^k \\ W(1) = 0 \end{cases}$$

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

使用迭代法，对解可以通过数学归纳法验证

差消化简后迭代

$$\begin{cases} T(n) = \frac{2}{n} \sum_{i=1}^{n-1} T(i) + O(n), & n \geq 2 \\ T(1) = 0 \end{cases} \quad \text{快速排序平均时间分析}$$

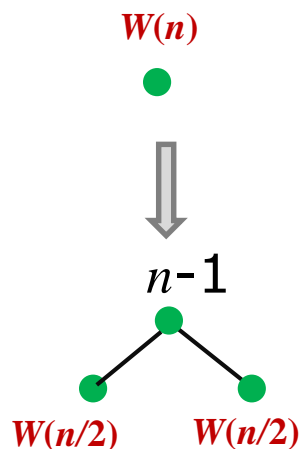
$$nT(n) = 2 \sum_{i=1}^{n-1} T(i) + cn^2, \quad c \text{ 为某个常数}$$

$$(n-1)T(n-1) = 2 \sum_{i=1}^{n-2} T(i) + c(n-1)^2$$

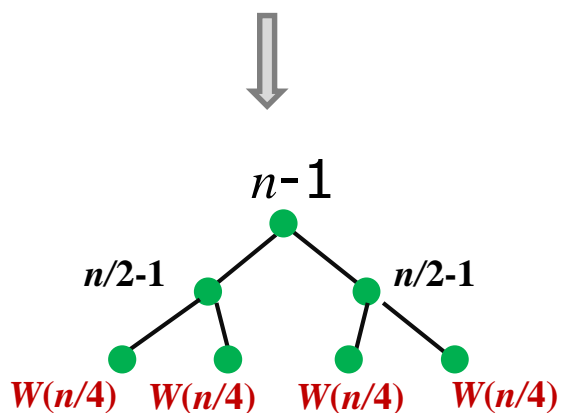
$$nT(n) = (n+1)T(n-1) + O(n)$$

$$\begin{aligned} \frac{T(n)}{n+1} &= \frac{T(n-1)}{n} + \frac{c_1}{n+1} = \dots = c_1 \left[\frac{1}{n+1} + \frac{1}{n} + \dots + \frac{1}{3} \right] + \frac{T(1)}{2} \\ &= c_1 \left[\frac{1}{n+1} + \frac{1}{n} + \dots + \frac{1}{3} \right] = \Theta(\log n), \quad T(n) = \Theta(n \log n) \end{aligned}$$

迭代模型：递归树



$$W(n) = 2W(n/2) + n - 1$$

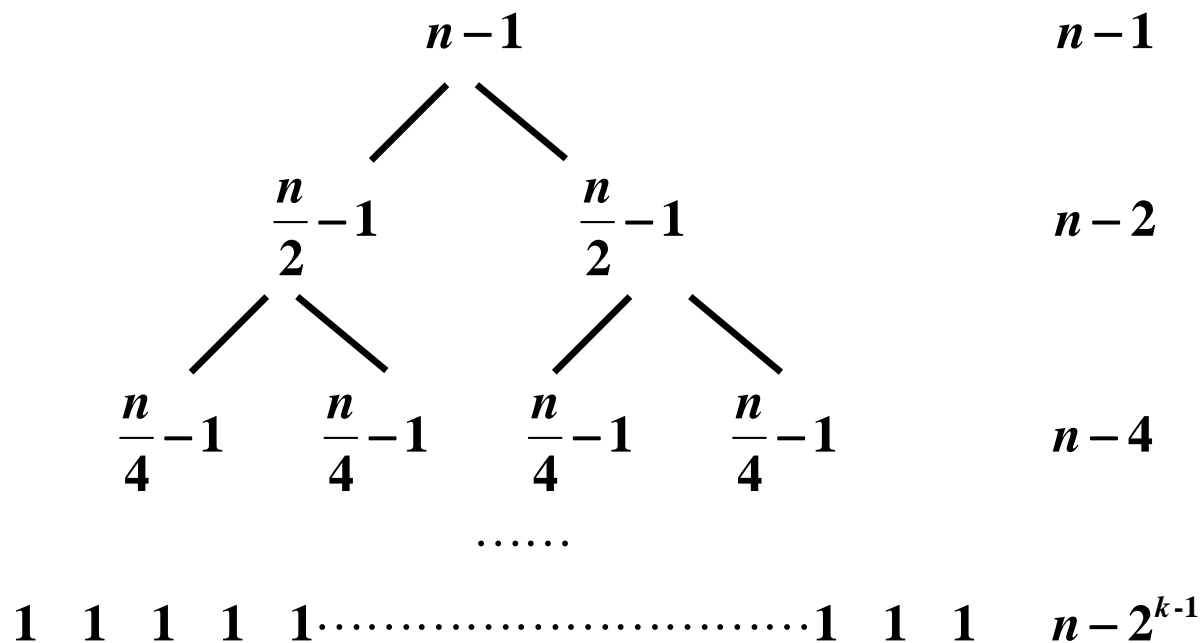


$$W(n/2) = 2W(n/4) + n/2 - 1$$

递归树的迭代生成

递归树

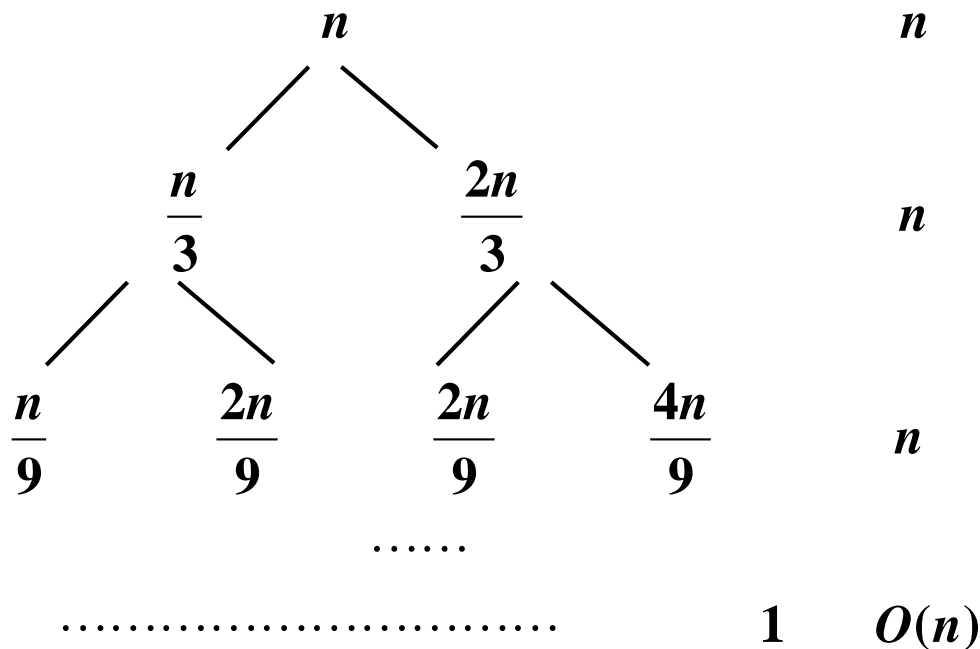
$$W(n) = 2W(n/2) + n - 1, \quad n = 2^k, \quad W(1) = 0$$



$$\begin{aligned}
 W(n) &= n-1 + n-2 + \dots + n-2^{k-1} \\
 &= kn - (2^k - 1) = n \log n - n + 1
 \end{aligned}$$

递归树的应用实例

$$T(n) = T(n/3) + T(2n/3) + n$$



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

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

尝试法：快速排序

(1) $T(n)=C$ 为常函数, 左边= $O(1)$

$$\text{右边} = \frac{2}{n} C(n-1) + O(n)$$

$$= 2C - \frac{2C}{n} + O(n)$$

$$T(n) = \frac{2}{n} \sum_{i=1}^{n-1} T(i) + O(n)$$

(2) $T(n)=cn$, 左边= cn

$$\text{右边} = \frac{2}{n} \sum_{i=1}^{n-1} ci + O(n)$$

$$= \frac{2c}{n} \frac{(1+n-1)(n-1)}{2} + O(n)$$

$$= cn - c + O(n)$$

尝试法：快速排序

(3) $T(n)=cn^2$, 左边= cn^2

$$\text{右边} = \frac{2}{n} \sum_{i=1}^{n-1} ci^2 + O(n)$$

$$= \frac{2}{n} \left[\frac{cn^3}{3} + O(n^2) \right] + O(n) = \frac{2c}{3} n^2 + O(n)$$

$$\begin{cases} T(n) = \frac{2}{n} \sum_{i=1}^{n-1} T(i) + O(n), & n \geq 2 \\ T(1) = 0 \end{cases}$$

(4) $T(n)=cn \log n$, 左边= $cn \log n$

$$\text{右边} = \frac{2c}{n} \sum_{i=1}^{n-1} i \log i + O(n)$$

$$= \frac{2c}{n} \left[\frac{n^2}{2} \log n - \frac{n^2}{4 \ln 2} + O(1) \right] + O(n)$$

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

以积分作为求和的近似

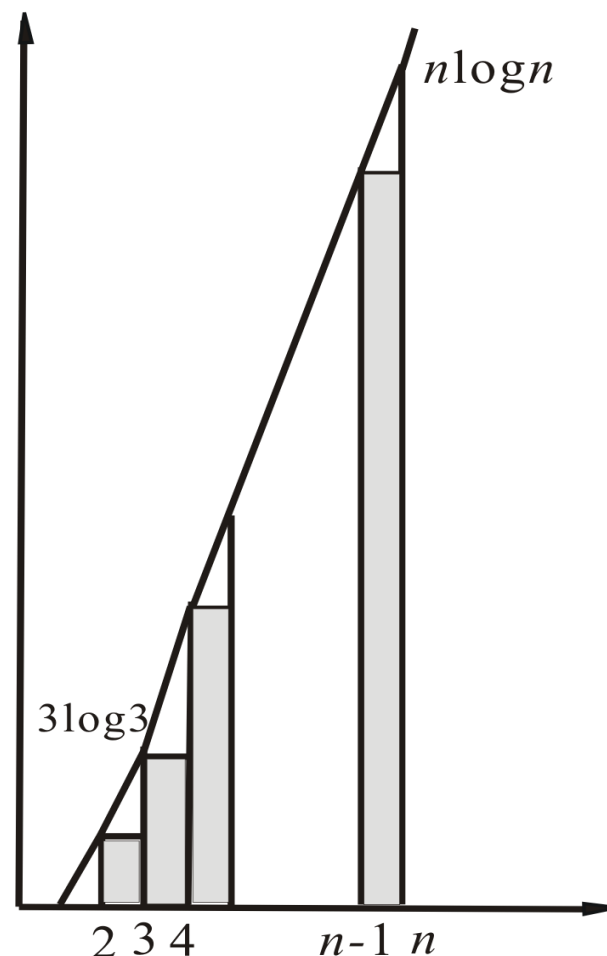
$$\int_2^n x \log x dx = \int_2^n \frac{x}{\ln 2} \ln x dx$$

$$= \frac{1}{\ln 2} \left[\frac{x^2}{2} \ln x - \frac{x^2}{4} \right] \Big|_2^n$$

$$= \frac{1}{\ln 2} \left(\frac{n^2}{2} \ln n - \frac{n^2}{4} \right) - \frac{1}{\ln 2} \left(\frac{4}{2} \ln 2 - \frac{4}{4} \right)$$

$$\sum_{i=1}^{n-1} i \log i$$

$$= \frac{n^2}{2} \log n - \frac{n^2}{4 \ln 2} + O(1)$$



主定理

主定理： 设 $a \geq 1, b > 1$ 为常数， $f(n)$ 为函数， $T(n)$ 为非负整数，且

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

则有以下结果：

1. 若 $f(n) = O(n^{\log_b a - \varepsilon}), \varepsilon > 0$ ，那么 $T(n) = \Theta(n^{\log_b a})$
2. 若 $f(n) = \Theta(n^{\log_b a})$ ，那么 $T(n) = \Theta(n^{\log_b a} \log n)$
3. 若 $f(n) = \Omega(n^{\log_b a + \varepsilon}), \varepsilon > 0$ ，且对于某个常数 $c < 1$ 和充分大的 n 有 $a f(n/b) \leq c f(n)$ ，那么 $T(n) = \Theta(f(n))$

迭代

设 $n=b^k$

$$T(n) = aT\left(\frac{n}{b}\right) + f(n)$$

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

$= \dots$

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

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

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

情况1

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

$$\begin{aligned} T(n) &= c_1 n^{\log_b a} + \sum_{j=0}^{k-1} a^j f\left(\frac{n}{b^j}\right) \\ &= c_1 n^{\log_b a} + O\left(\sum_{j=0}^{\log_b n - 1} a^j \left(\frac{n}{b^j}\right)^{\log_b a - \varepsilon}\right) \\ &= c_1 n^{\log_b a} + O\left(n^{\log_b a - \varepsilon} \sum_{j=0}^{\log_b n - 1} \frac{a^j}{(b^{\log_b a - \varepsilon})^j}\right) \\ &= c_1 n^{\log_b a} + O\left(n^{\log_b a - \varepsilon} \sum_{j=0}^{\log_b n - 1} (b^\varepsilon)^j\right) \\ &= c_1 n^{\log_b a} + O\left(n^{\log_b a - \varepsilon} \frac{b^{\varepsilon \log_b n} - 1}{b^\varepsilon - 1}\right) \\ &= c_1 n^{\log_b a} + O(n^{\log_b a - \varepsilon} n^\varepsilon) = O(n^{\log_b a}) \end{aligned}$$

情况2

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

$$\begin{aligned} T(n) &= c_1 n^{\log_b a} + \sum_{j=0}^{k-1} a^j f\left(\frac{n}{b^j}\right) \\ &= c_1 n^{\log_b a} + \Theta\left(\sum_{j=0}^{\log_b n - 1} a^j \left(\frac{n}{b^j}\right)^{\log_b a}\right) \\ &= c_1 n^{\log_b a} + \Theta\left(n^{\log_b a} \sum_{j=0}^{\log_b n - 1} \frac{a^j}{a^j}\right) \\ &= c_1 n^{\log_b a} + \Theta(n^{\log_b a} \log n) \\ &= \Theta(n^{\log_b a} \log n) \end{aligned}$$

情况3

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

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

$$\leq c_1 n^{\log_b a} + \sum_{j=0}^{\log_b n - 1} c^j f(n) \quad \left(af\left(\frac{n}{b}\right) \leq cf(n)\right)$$

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

$$= c_1 n^{\log_b a} + \Theta(f(n)) \quad (c < 1)$$

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

主定理的应用

例 求解递推方程 $T(n) = 9T(n/3) + n$

解 上述递推方程中的 $a = 9, b = 3, f(n) = n$, 那么

$$n^{\log_3 9} = n^2, \quad f(n) = O(n^{\log_3 9 - 1}),$$

相当于主定理的第一种情况, 其中 $\varepsilon = 1$. 根据定理得到

$$T(n) = \Theta(n^2)$$

例 求解递推方程 $T(n) = T(2n/3) + 1$

解 上述递推方程中的 $a = 1, b = 3/2, f(n) = 1$, 那么

$$n^{\log_{3/2} 1} = n^0 = 1, \quad f(n) = 1$$

相当于主定理的第二种情况. 根据定理得到 .

$$T(n) = \Theta(n^0 \log n) = \Theta(\log n)$$

实例

例 求解递推方程 $T(n) = 3T(n/4) + n \log n$

上述递推方程中的 $a=3, b=4, f(n)=n \log n$, 那么

$$n \log n = \Omega(n^{\log_4 3 + \varepsilon}) = \Omega(n^{0.793 + \varepsilon}) \quad \varepsilon \approx 0.2$$

此外, 要使 $a f(n/b) \leq c f(n)$ 成立, 代入 $f(n)=n \log n$, 得到

$$\frac{3n}{4} \log \frac{n}{4} \leq c n \log n$$

显然只要 $c \geq 3/4$, 上述不等式就可以对充分大的 n 成立.
相当于主定理的第三种情况. 因此有

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

不能使用主定理的例子

$$T(n) = 2T(n/2) + n \log n$$

$$T(1) = 1$$

□ 不能使用主定理

$$a = 2, b = 2, n^{\log_b a} = n, f(n) = n \log n$$

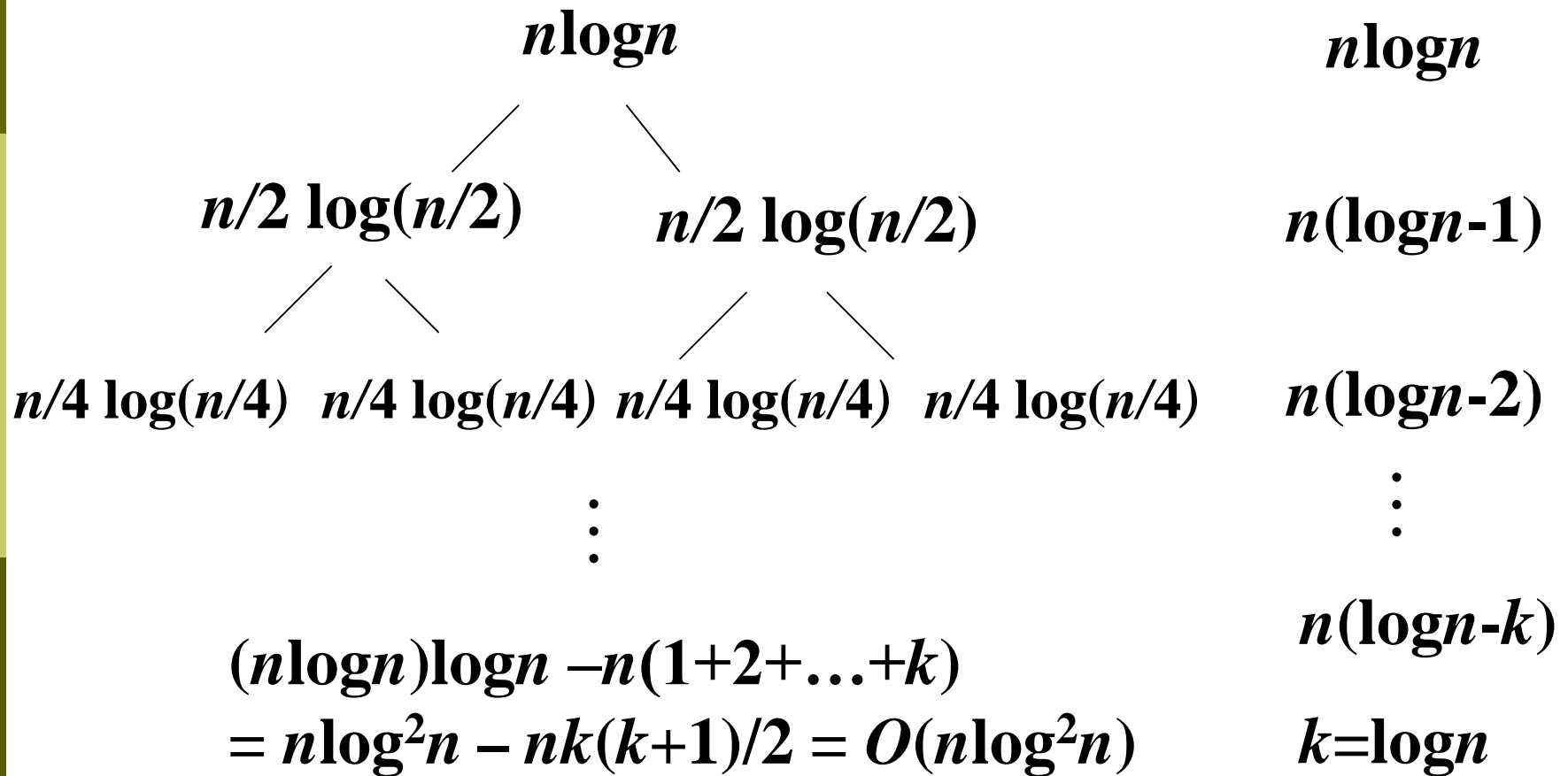
不存在 $c < 1$ 使得 $2(n/2) \log(n/2) \leq c n \log n$

不存在 $\varepsilon > 0$ 使得 $f(n) = n \log n = \Omega(n^{1+\varepsilon})$

□ 只能使用递归树，结果等于

$$T(n) = n \log^2 n$$

递归树求解: $T(n)=2T(n/2)+O(n\log n)$



递推方程中 $\lfloor x \rfloor$ 和 $\lceil x \rceil$ 的处理

先猜想解，然后用数学归纳法证明

例16 估计以下递推关系的阶

$$T(n) = 2T\left(\left\lfloor \frac{n}{2} \right\rfloor\right) + n$$

$$T(1) = 1$$

根据 $T(n) = 2T\left(\frac{n}{2}\right) + n$

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

猜想原递推方程的解的阶是 $O(n \log n)$

证明： $T(n) \leq cn \log n$, 用数学归纳法

证明

归纳基础

对于 $T(1)=1$ ，显然没有 $T(1) \leq c \cdot 1 \log 1$.

考虑 $T(2)$

$$T(2) = 2T(\lfloor 1 \rfloor) + 2 = 4 \leq 2 \times 2 \log 2, \quad c=2$$

归纳步骤

假设对于小于 n 的正整数命题为真，那么

$$\begin{aligned} T(n) &= 2T\left(\left\lfloor \frac{n}{2} \right\rfloor\right) + n \leq 2c \left\lfloor \frac{n}{2} \right\rfloor \log\left(\left\lfloor \frac{n}{2} \right\rfloor\right) + n \\ &\leq 2c \frac{n}{2} (\log n - \log 2) + n = c n \log n - c n + n \\ &\leq c n \log n, \quad c = 2 \end{aligned}$$