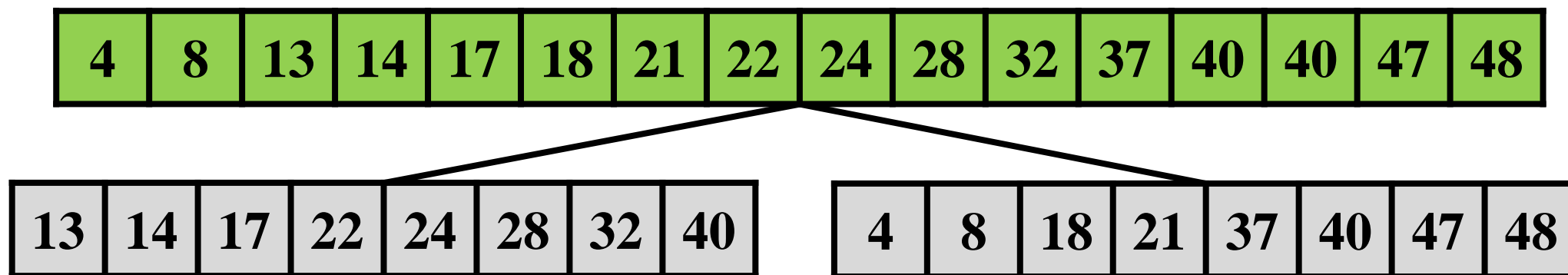


Design and Analysis of Algorithms

Lecture 3: Solving Recurrences

童咏昕

北京航空航天大学
计算机学院



● 算法流程

- 将数组 $A[1, n]$ 排序问题**分解**为 $A[1, \lfloor \frac{n}{2} \rfloor]$ 和 $A[\lfloor \frac{n}{2} \rfloor + 1, n]$ 排序问题
- **递归解决**子问题得到两个有序的子数组
- 将两个有序子数组**合并**为一个有序数组

分解原问题

解决子问题

合并问题解

归并排序：分解数组，递归求解，合并排序

归并排序：复杂度分析

- $T(n)$: 完成MergeSort($A, 1, n$) 的运行时间
 - 为便于分析, 假设 n 是2的幂
- MergeSort($A, left, right$)

输入: 数组 $A[1..n]$, 数组下标 $left, right$

输出: 递增数组 $A[left..right]$

if $left \geq right$ then

 | return $A[left..right]$

end

$mid \leftarrow \lfloor \frac{left+right}{2} \rfloor$

MergeSort($A, left, mid$)

MergeSort($A, mid + 1, right$)

Merge($A, left, mid, right$)

return $A[left..right]$

} $O(1)$

→ $T(n/2)$

→ $T(n/2)$

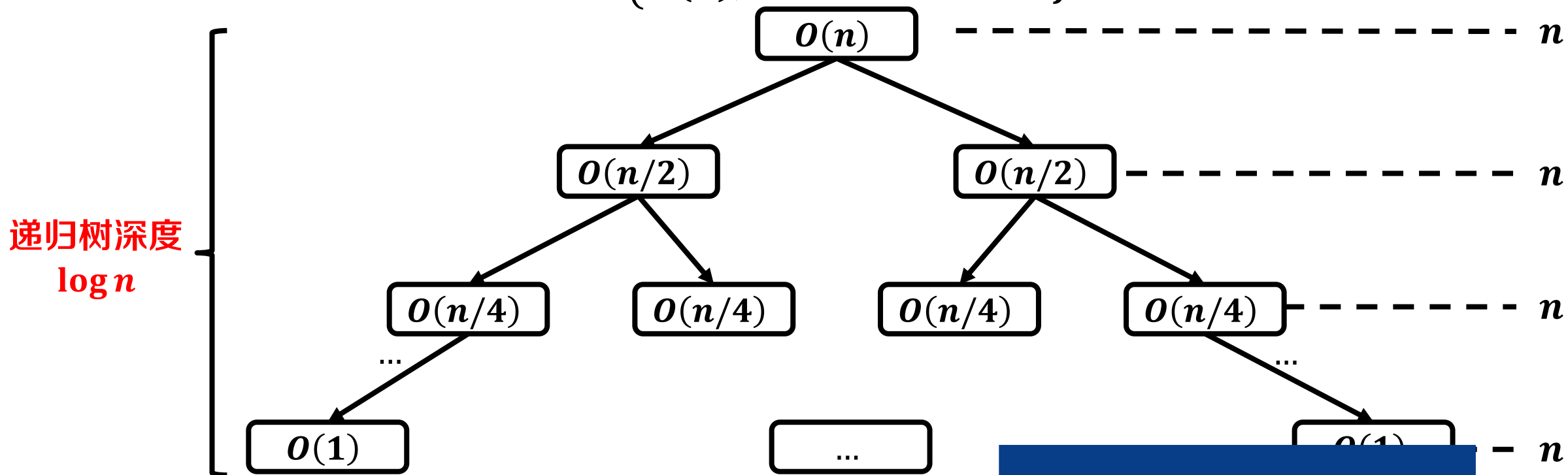
→ $O(n)$

$$T(n) = \begin{cases} 2T(n/2) + O(n), & \text{if } n > 1 \\ O(1), & \text{if } n = 1 \end{cases} \quad ?$$

归并排序：复杂度分析

- 递归树法：用树的形式表示抽象递归

$$T(n) = \begin{cases} 2T(n/2) + O(n), & \text{if } n > 1 \\ O(1), & \text{if } n = 1 \end{cases}$$



二者相乘为总代价

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

递归树法

代入法

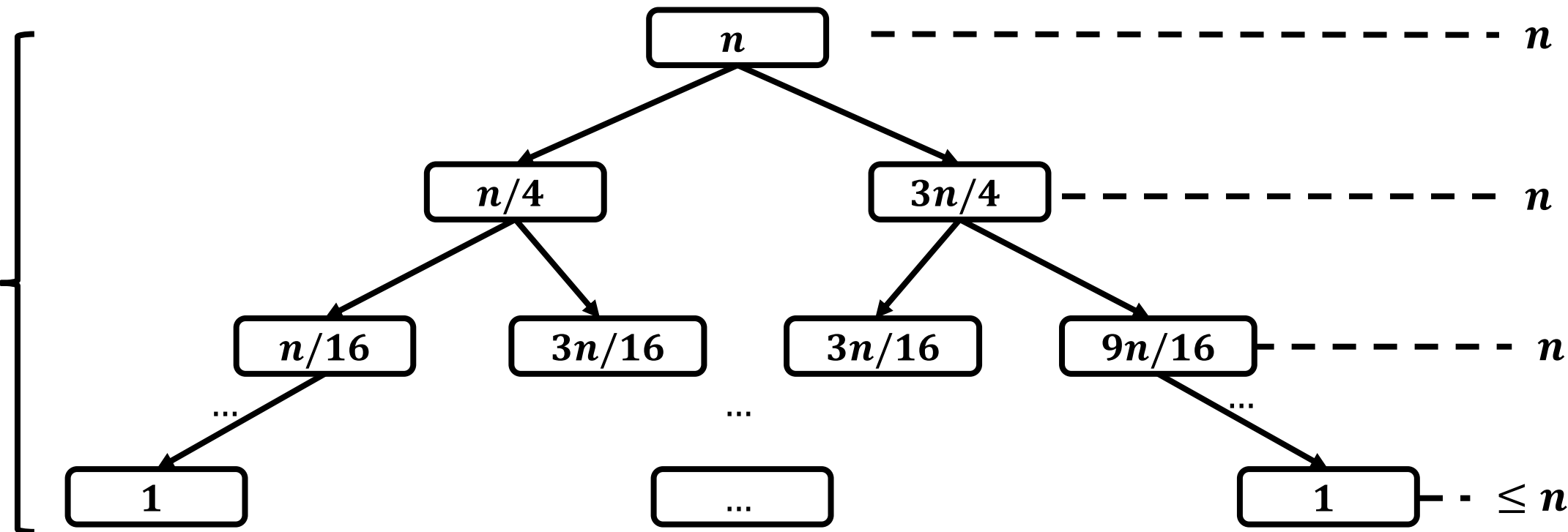
主定理法

递归树法：实例



$$T(n) = \begin{cases} T(n/4) + T(3n/4) + n, & \text{if } n \geq 4 \\ 1, & \text{if } n < 4 \end{cases}$$

深度最多
 $\log_4 n$
 $\frac{1}{3}$



$$T(n) = O(n \log_4 n) = O\left(n \frac{\log_2 n}{\log_2 4}\right) = O(n \log n)$$

对数换底公式

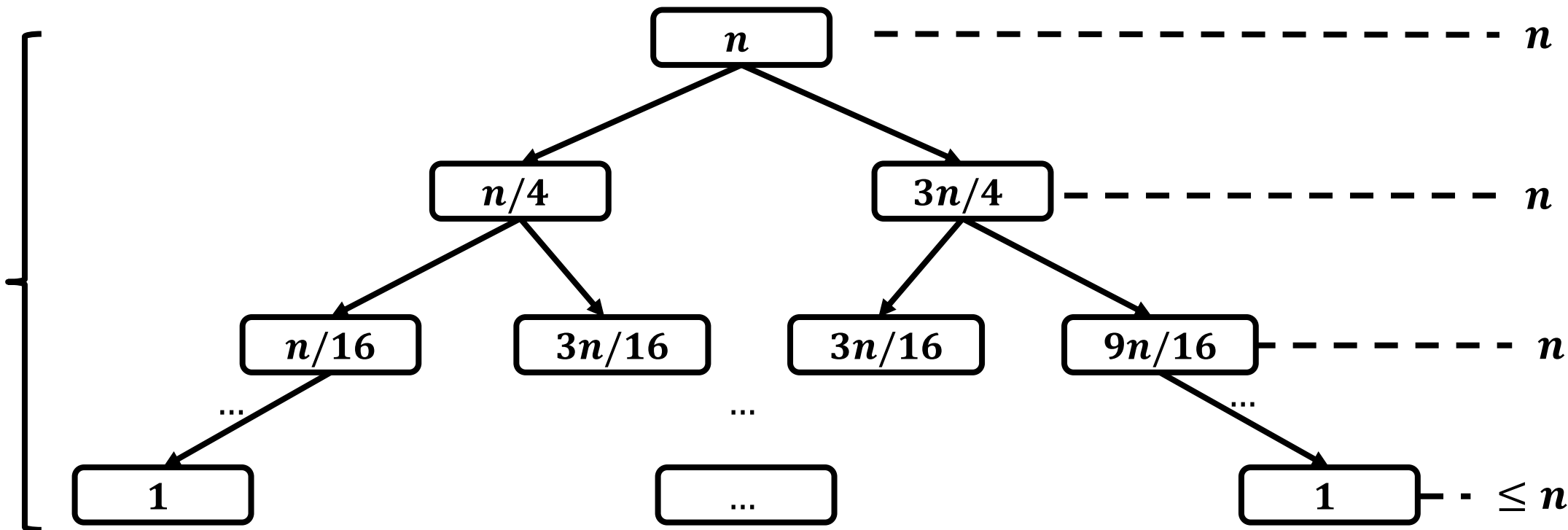
$$\log_x N = \frac{\log_y N}{\log_y x}$$

递归树法：实例



$$T(n) = \begin{cases} T(n/4) + T(3n/4) + n, & \text{if } n \geq 4 \\ 1, & \text{if } n < 4 \end{cases}$$

深度最多
 $\log_4 n$



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

问题：该界是否为渐进紧确界？

递归树法

代入法

主定理法

代入法：实例

- $T(n) = \begin{cases} T(n/4) + T(3n/4) + n, & n \geq 4 \\ 1, & n < 4 \end{cases}$
- 猜测： $T(n) = \Theta(n \log n)$
 - 即证明 $\exists c_1, c_2, n_0 > 0$, 使得 $\forall n > n_0, c_1 \cdot n \log n \leq T(n) \leq c_2 \cdot n \log n$

Θ 记号

定义：

- 对于给定的函数 $g(n)$, $\Theta(g(n))$ 表示以下函数的集合：
 $\Theta(g(n)) = \{T(n) : \exists c_1, c_2, n_0 > 0, \text{使得} \forall n \geq n_0, c_1 g(n) \leq T(n) \leq c_2 g(n)\}$

- $T(n) = \begin{cases} T(n/4) + T(3n/4) + n, & n \geq 4 \\ 1, & n < 4 \end{cases}$
- 猜测： $T(n) = \Theta(n \log n)$
 - 即证明 $\exists c_1, c_2, n_0 > 0$, 使得 $\forall n > n_0, c_1 \cdot n \log n \leq T(n) \leq c_2 \cdot n \log n$
- 数学归纳法
 - $n = 3$ 时：使 $c_1 \cdot 3 \log 3 \leq 1 \leq c_2 \cdot 3 \log 3$, 需取 $0 < c_1 \leq \frac{1}{3 \log 3}, c_2 \geq \frac{1}{3 \log 3}$
 - 小于 n 时：假设命题成立
 - 等于 n 时：代入可得
 - $T(n) = T(n/4) + T(3n/4) + n \leq c_2 \cdot \frac{n}{4} \cdot \log \frac{n}{4} + c_2 \cdot \frac{3n}{4} \cdot \log \frac{3n}{4} + n$

- 代入并整理表达式

$$T(n) = T(n/4) + T(3n/4) + n$$

$$\leq c_2 \cdot \frac{n}{4} \cdot \log \frac{n}{4} + c_2 \cdot \frac{3n}{4} \cdot \log \frac{3n}{4} + n$$

$$= \left(c_2 \cdot \frac{n}{4} \cdot (\log n - \log 4) \right) + \left(c_2 \cdot \frac{3n}{4} \cdot \left(\log n - \log \frac{4}{3} \right) \right) + n$$

$$= c_2 n \log n - \left(c_2 n \left(\frac{1}{4} \log 4 + \frac{3}{4} \log 4 - \frac{3}{4} \log 3 \right) \right) + n$$

$$= c_2 n \log n - \left(c_2 \left(\log 4 - \frac{3}{4} \log 3 \right) - 1 \right) n$$

只需此部分 ≥ 0

- 令 $\left(c_2 \left(\log 4 - \frac{3}{4} \log 3 \right) - 1 \right) n \geq 0$, 解得 $c_2 \geq \frac{1}{\log 4 - \frac{3}{4} \log 3} > 0$

代入法：实例



- $T(n) = \begin{cases} T(n/4) + T(3n/4) + n, & n \geq 4 \\ 1, & n < 4 \end{cases}$
- 猜测： $T(n) = \Theta(n \log n)$
 - 即证明 $\exists c_1, c_2, n_0 > 0$, 使得 $\forall n > n_0, c_1 \cdot n \log n \leq T(n) \leq c_2 \cdot n \log n$
- 数学归纳法
 - $n = 3$ 时：使 $c_1 \cdot 3 \log 3 \leq 1 \leq c_2 \cdot 3 \log 3$, 需取 $0 < c_1 \leq \frac{1}{3 \log 3}, c_2 \geq \frac{1}{3 \log 3}$
 - 小于 n 时：假设命题成立
 - 等于 n 时：代入可得
 - $T(n) = T(n/4) + T(3n/4) + n \leq c_2 \cdot \frac{n}{4} \cdot \log \frac{n}{4} + c_2 \cdot \frac{3n}{4} \cdot \log \frac{3n}{4} + n$
 - 若想 $T(n) \leq c_2 \cdot n \log n$, 需取 $c_2 \geq \frac{1}{\log 4 - \frac{3}{4} \log 3}$

两条件需同时满足

- $T(n) = \begin{cases} T(n/4) + T(3n/4) + n, & n \geq 4 \\ 1, & n < 4 \end{cases}$
- 猜测： $T(n) = \Theta(n \log n)$
 - 即证明 $\exists c_1, c_2, n_0 > 0$, 使得 $\forall n > n_0, c_1 \cdot n \log n \leq T(n) \leq c_2 \cdot n \log n$
- 数学归纳法
 - $n = 3$ 时：使 $c_1 \cdot 3 \log 3 \leq 1 \leq c_2 \cdot 3 \log 3$, 需取 $0 < c_1 \leq \frac{1}{3 \log 3}, c_2 \geq \frac{1}{3 \log 3}$
 - 小于 n 时：假设命题成立
 - 等于 n 时：代入可得
 - $T(n) = T(n/4) + T(3n/4) + n \leq c_2 \cdot \frac{n}{4} \cdot \log \frac{n}{4} + c_2 \cdot \frac{3n}{4} \cdot \log \frac{3n}{4} + n$
 - 若想 $T(n) \leq c_2 \cdot n \log n$, 需取 $c_2 \geq \frac{1}{\log 4 - \frac{3}{4} \log 3}$
 - $c_2 \geq \max \left\{ \frac{1}{\log 4 - \frac{3}{4} \log 3}, \frac{1}{3 \log 3} \right\}$

- $T(n) = \begin{cases} T(n/4) + T(3n/4) + n, & n \geq 4 \\ 1, & n < 4 \end{cases}$
- 猜测： $T(n) = \Theta(n \log n)$
 - 即证明 $\exists c_1, c_2, n_0 > 0$ ，使得 $\forall n > n_0, c_1 \cdot n \log n \leq T(n) \leq c_2 \cdot n \log n$
- 数学归纳法
 - $n = 3$ 时：使 $c_1 \cdot 3 \log 3 \leq 1 \leq c_2 \cdot 3 \log 3$ ，需取 $0 < c_1 \leq \frac{1}{3 \log 3}, c_2 \geq \frac{1}{3 \log 3}$
 - 小于 n 时：假设命题成立
 - 等于 n 时：代入可得
 - 取 $c_2 \geq \max \left\{ \frac{1}{\log 4 - \frac{3}{4} \log 3}, \frac{1}{3 \log 3} \right\}$ ，可得 $T(n) \leq c_2 \cdot n \log n$
 - 取 $0 < c_1 \leq \min \left\{ \frac{1}{\log 4 - \frac{3}{4} \log 3}, \frac{1}{3 \log 3} \right\}$ ，可得 $T(n) \geq c_1 \cdot n \log n$
- 得证 $T(n) = \Theta(n \log n)$

问题：猜测解不易得时如何求解递归式？

递归树法

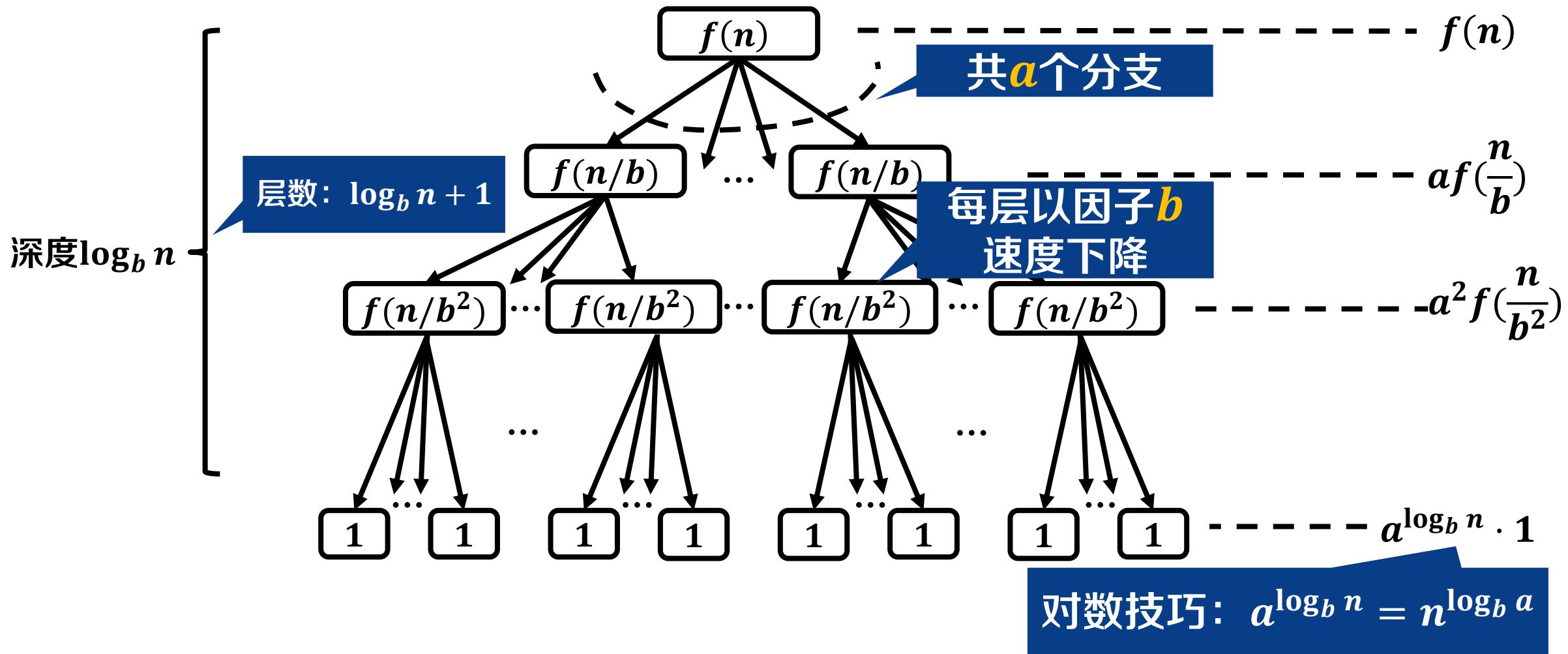
代入法

主定理法

递归式分析：主定理法



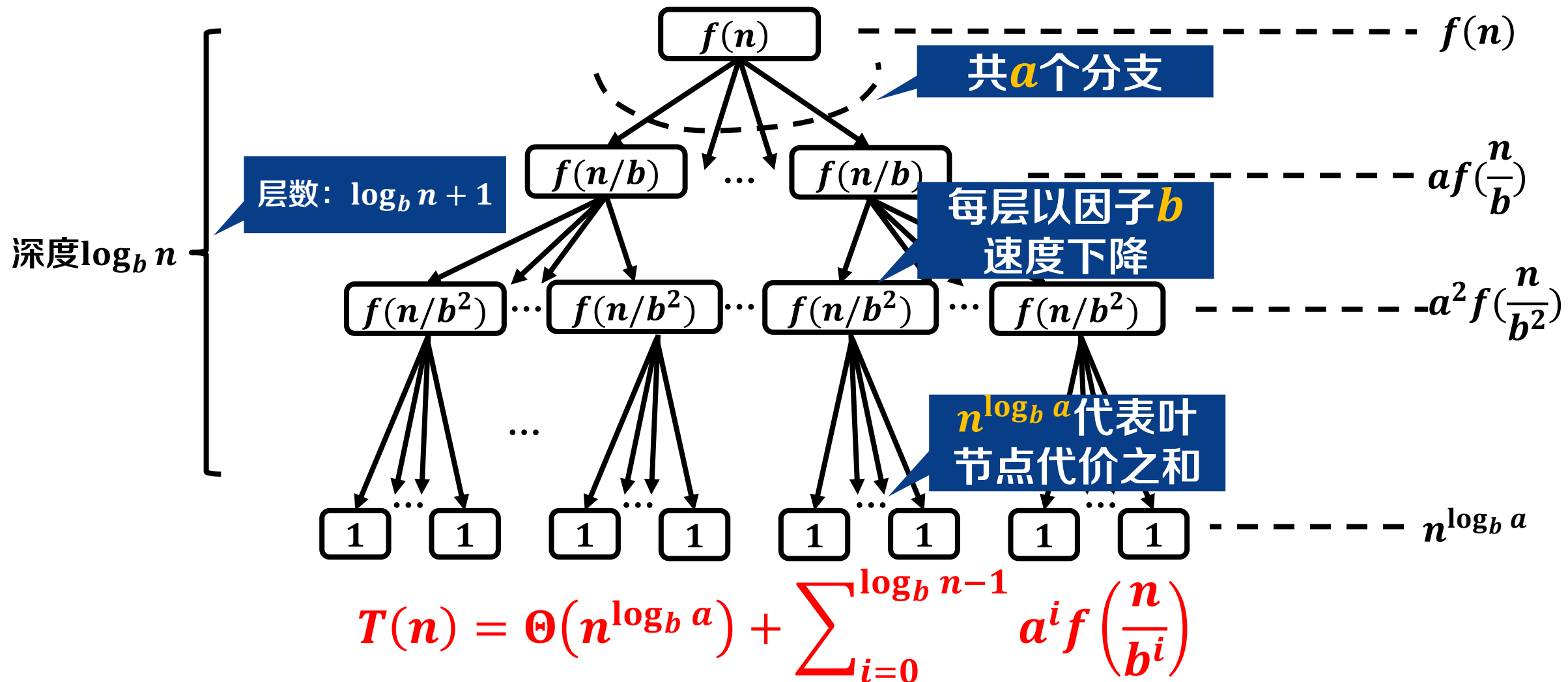
- 对形如 $T(n) = aT\left(\frac{n}{b}\right) + f(n)$ 的递归式



递归式分析：主定理法



- 对形如 $T(n) = aT\left(\frac{n}{b}\right) + f(n)$ 的递归式



递归式分析：主定理法



- 主定理：对形如 $T(n) = aT\left(\frac{n}{b}\right) + f(n)$ 的递归式

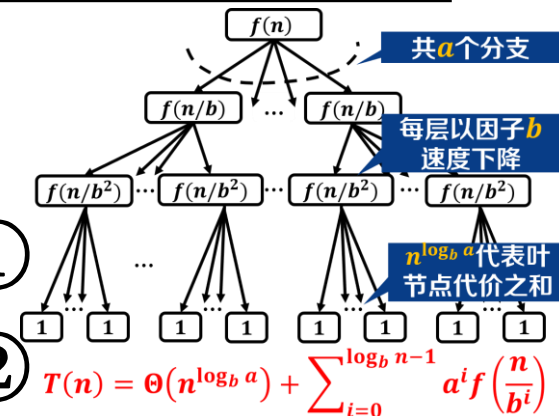
$$T(n) = \begin{cases} \Theta(f(n)) \\ \Theta(n^{\log_b a} \log n) \\ \Theta(n^{\log_b a}) \end{cases}$$

$$\text{if } f(n) = \Omega(n^{\log_b a + \epsilon}) \quad \textcircled{1}$$

$$\text{if } f(n) = \Theta(n^{\log_b a}) \quad \textcircled{2}$$

$$\text{if } f(n) = O(n^{\log_b a - \epsilon}) \quad \textcircled{3}$$

比较根节点代价 $f(n)$ 与
叶节点代价之和 $n^{\log_b a}$

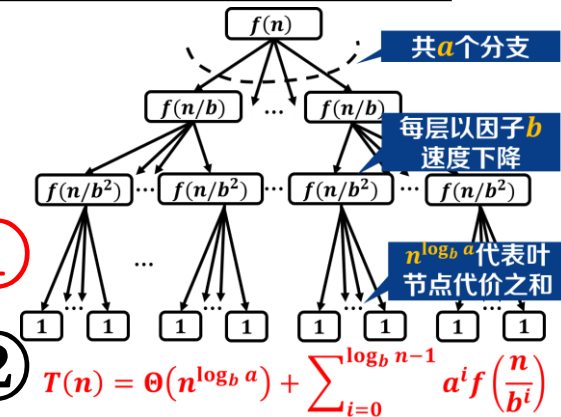


递归式分析：主定理法



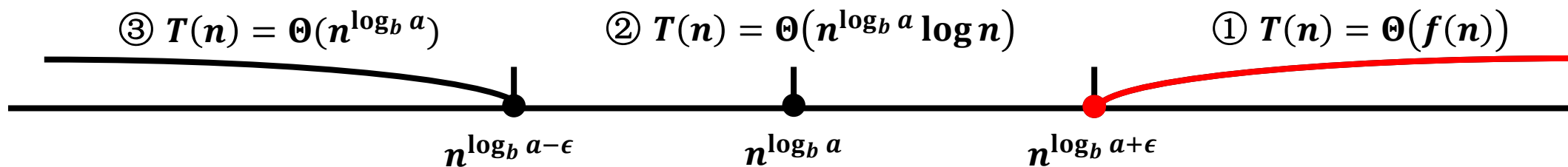
- 主定理：对形如 $T(n) = aT\left(\frac{n}{b}\right) + f(n)$ 的递归式

$$T(n) = \begin{cases} \Theta(f(n)) & \text{if } f(n) = \Omega(n^{\log_b a + \epsilon}) \quad \textcircled{1} \\ \Theta(n^{\log_b a} \log n) & \text{if } f(n) = \Theta(n^{\log_b a}) \quad \textcircled{2} \\ \Theta(n^{\log_b a}) & \text{if } f(n) = O(n^{\log_b a - \epsilon}) \quad \textcircled{3} \end{cases}$$



- 若存在常数 $\epsilon > 0$ 使 $f(n) = \Omega(n^{\log_b a + \epsilon})$ ，且存在常数 $c < 1$ 和足够大的 n 使得 $af\left(\frac{n}{b}\right) \leq cf(n)$

$f(n)$ 多项式意义大于 $n^{\log_b a}$ ：
不止渐进大于且相差因子 n^ϵ



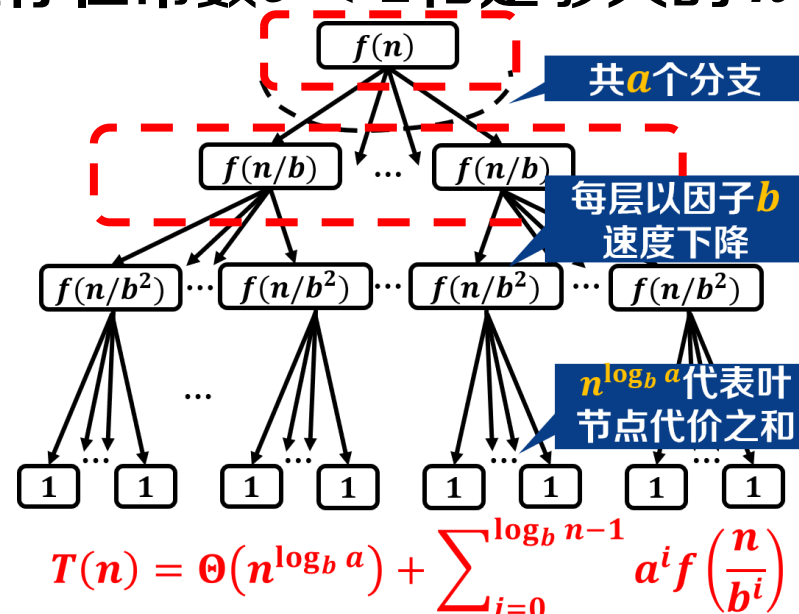
递归式分析：主定理法

- 主定理：对形如 $T(n) = aT\left(\frac{n}{b}\right) + f(n)$ 的递归式

$$T(n) = \begin{cases} \Theta(f(n)) & \text{if } f(n) = \Omega(n^{\log_b a + \epsilon}) \quad \textcircled{1} \\ \Theta(n^{\log_b a} \log n) & \text{if } f(n) = \Theta(n^{\log_b a}) \quad \textcircled{2} \\ \Theta(n^{\log_b a}) & \text{if } f(n) = O(n^{\log_b a - \epsilon}) \quad \textcircled{3} \end{cases}$$

- 若存在常数 $\epsilon > 0$ 使 $f(n) = \Omega(n^{\log_b a + \epsilon})$ ，且存在常数 $c < 1$ 和足够大的 n 使得 $af\left(\frac{n}{b}\right) \leq cf(n)$ ，则 $T(n) = \Theta(f(n))$

保证了根节点代价
大于下一层代价之和

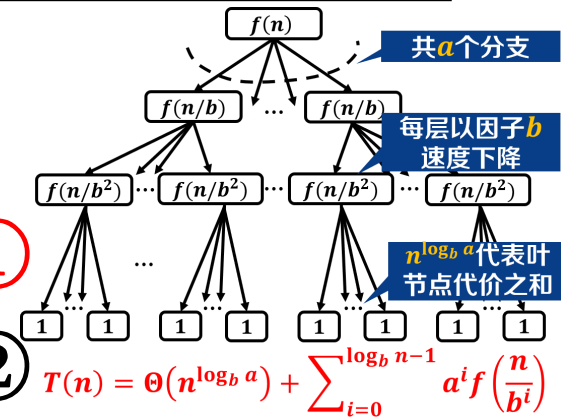


递归式分析：主定理法



- 主定理：对形如 $T(n) = aT\left(\frac{n}{b}\right) + f(n)$ 的递归式

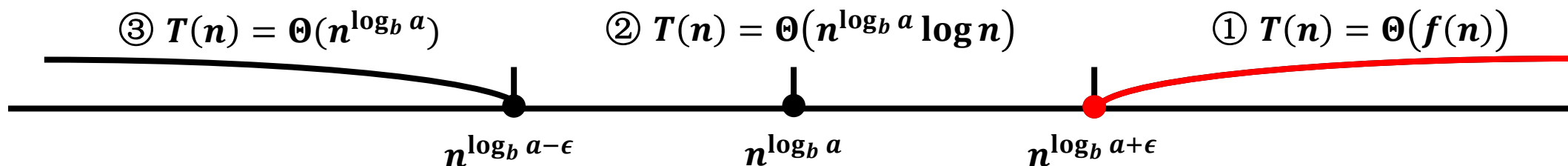
$$T(n) = \begin{cases} \Theta(f(n)) & \text{if } f(n) = \Omega(n^{\log_b a + \epsilon}) \quad \textcircled{1} \\ \Theta(n^{\log_b a} \log n) & \text{if } f(n) = \Theta(n^{\log_b a}) \quad \textcircled{2} \\ \Theta(n^{\log_b a}) & \text{if } f(n) = O(n^{\log_b a - \epsilon}) \quad \textcircled{3} \end{cases}$$



- 若存在常数 $\epsilon > 0$ 使 $f(n) = \Omega(n^{\log_b a + \epsilon})$ ，且存在常数 $c < 1$ 和足够大的 n 使得 $af\left(\frac{n}{b}\right) \leq cf(n)$ ，则 $T(n) = \Theta(f(n))$

称为“正则”条件

保证了根节点代价
大于下一层代价之和

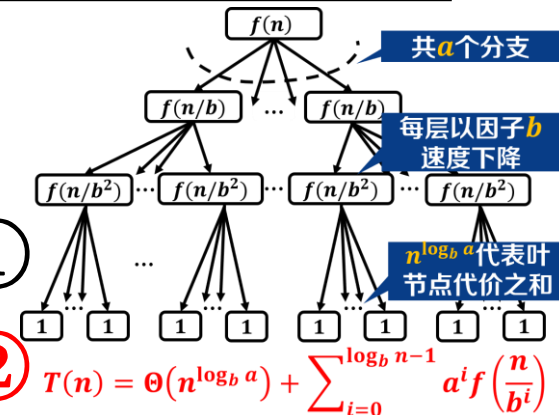


递归式分析：主定理法

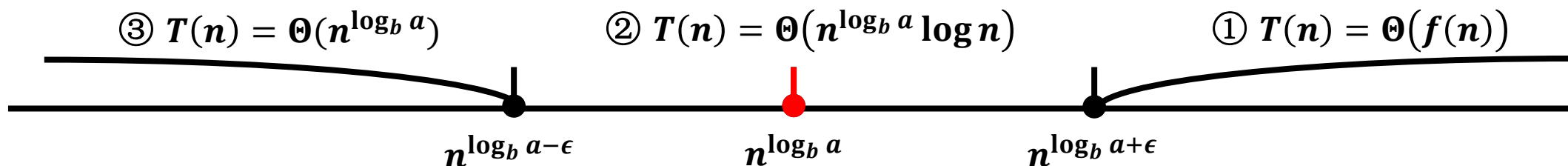


- 主定理：对形如 $T(n) = aT\left(\frac{n}{b}\right) + f(n)$ 的递归式

$$T(n) = \begin{cases} \Theta(f(n)) & \text{if } f(n) = \Omega(n^{\log_b a + \epsilon}) \quad \textcircled{1} \\ \Theta(n^{\log_b a} \log n) & \text{if } f(n) = \Theta(n^{\log_b a}) \quad \textcircled{2} \\ \Theta(n^{\log_b a}) & \text{if } f(n) = O(n^{\log_b a - \epsilon}) \quad \textcircled{3} \end{cases}$$



- 若 $f(n) = \Theta(n^{\log_b a})$, 则 $T(n) = \Theta(n^{\log_b a} \log n)$

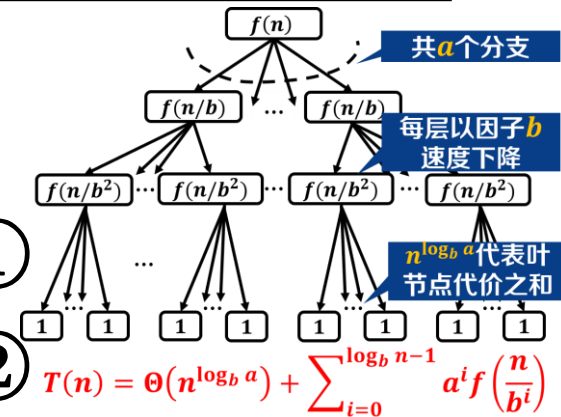


递归式分析：主定理法



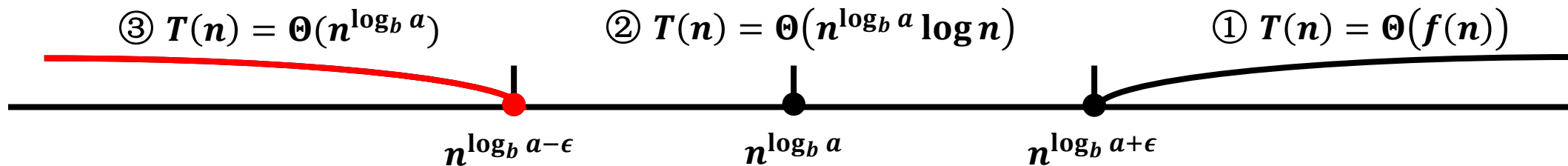
- 主定理：对形如 $T(n) = aT\left(\frac{n}{b}\right) + f(n)$ 的递归式

$$T(n) = \begin{cases} \Theta(f(n)) & \text{if } f(n) = \Omega(n^{\log_b a + \epsilon}) \quad \textcircled{1} \\ \Theta(n^{\log_b a} \log n) & \text{if } f(n) = \Theta(n^{\log_b a}) \quad \textcircled{2} \\ \Theta(n^{\log_b a}) & \text{if } f(n) = O(n^{\log_b a - \epsilon}) \quad \textcircled{3} \end{cases}$$



- 若存在常数 $\epsilon > 0$ 使 $f(n) = O(n^{\log_b a - \epsilon})$ ，则 $T(n) = \Theta(n^{\log_b a})$

$f(n)$ 多项式意义小于 $n^{\log_b a}$ ：
不止渐进小于且相差因子 n^ϵ

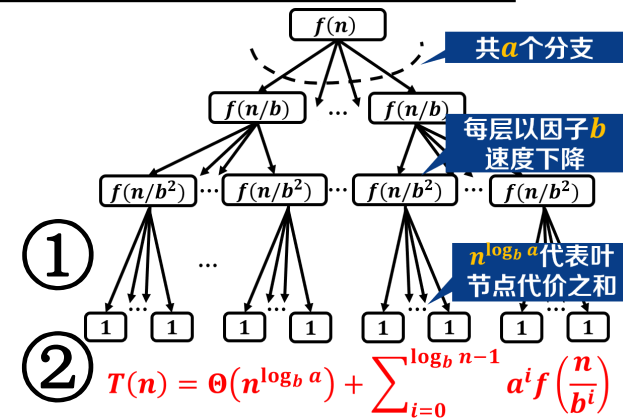


递归式分析：主定理法



- 主定理：对形如 $T(n) = aT\left(\frac{n}{b}\right) + f(n)$ 的递归式

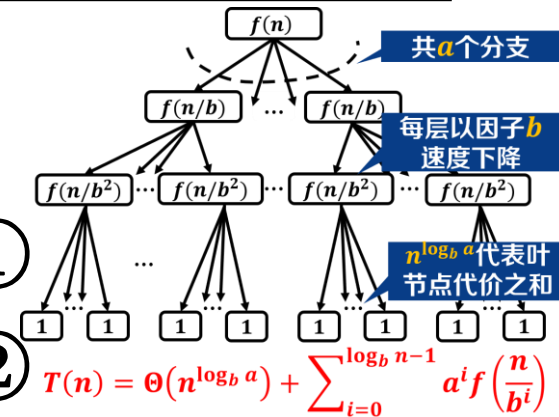
$$T(n) = \begin{cases} \Theta(n^{a+\epsilon}) & \text{当 } f(n) \text{ 形式为 } n^k \text{ 时, 可简化主定理公式} \\ \Theta(n^{\log_b a} \log n) & \text{if } f(n) = \Theta(n^{\log_b a}) \\ \Theta(n^{\log_b a}) & \text{if } f(n) = O(n^{\log_b a - \epsilon}) \end{cases}$$



递归式分析：主定理法

- 主定理：对形如 $T(n) = aT\left(\frac{n}{b}\right) + f(n)$ 的递归式

$$T(n) = \begin{cases} \Theta(f(n)) & \text{if } f(n) = \Omega(n^{\log_b a + \epsilon}) \quad ① \\ \Theta(n^{\log_b a} \log n) & \text{if } f(n) = \Theta(n^{\log_b a}) \quad ② \\ \Theta(n^{\log_b a}) & \text{if } f(n) = O(n^{\log_b a - \epsilon}) \quad ③ \end{cases}$$



- 主定理(简化形式)：对形如 $T(n) = aT\left(\frac{n}{b}\right) + n^k$ 的递归式

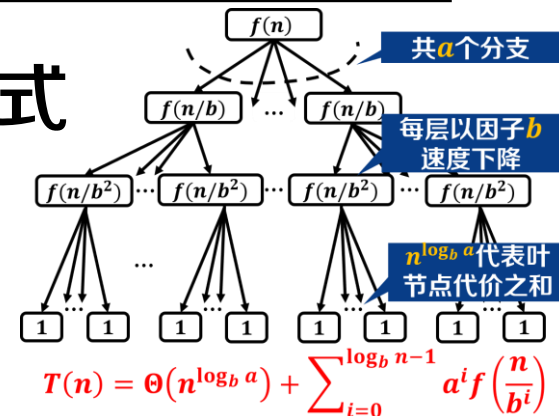
$$T(n) = \begin{cases} \Theta(n^k) & \text{if } k > \log_b a \quad ① \\ \Theta(n^k \log n) & \text{if } k = \log_b a \quad ② \\ \Theta(n^{\log_b a}) & \text{if } k < \log_b a \quad ③ \end{cases}$$

主定理法：实例一



- 主定理(简化形式): 对形如 $T(n) = aT\left(\frac{n}{b}\right) + n^k$ 的递归式

$$T(n) = \begin{cases} \Theta(n^k) & \text{if } k > \log_b a \quad \textcircled{1} \\ \Theta(n^k \log n) & \text{if } k = \log_b a \quad \textcircled{2} \\ \Theta(n^{\log_b a}) & \text{if } k < \log_b a \quad \textcircled{3} \end{cases}$$



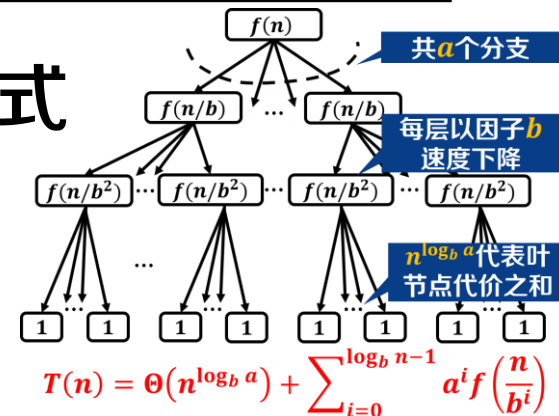
- 例一: $T(n) = 2T\left(\frac{n}{2}\right) + n$
 - $k = 1$
 - $a = 2, b = 2, \log_b a = 1$
 - $k = \log_b a$, 属于情况②
 - $T(n) = \Theta(n^k \log n) = \Theta(n \log n)$

主定理法：实例二



- 主定理(简化形式): 对形如 $T(n) = aT\left(\frac{n}{b}\right) + n^k$ 的递归式

$$T(n) = \begin{cases} \Theta(n^k) & \text{if } k > \log_b a \quad \textcircled{1} \\ \Theta(n^k \log n) & \text{if } k = \log_b a \quad \textcircled{2} \\ \Theta(n^{\log_b a}) & \text{if } k < \log_b a \quad \textcircled{3} \end{cases}$$

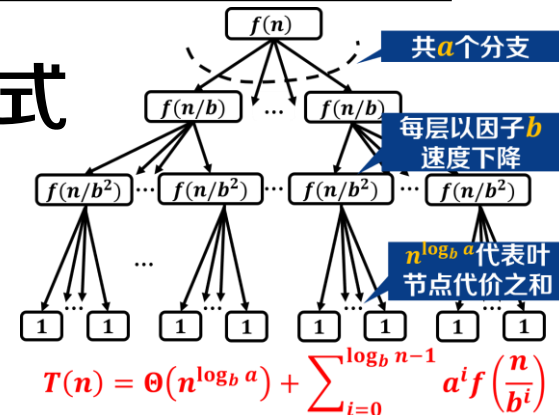


- 例二: $T(n) = 5T\left(\frac{n}{2}\right) + n^3$
 - $k = 3$
 - $a = 5, b = 2, \log_b a = \log_2 5$
 - $k > \log_b a$, 属于情况①
 - $T(n) = \Theta(n^k) = \Theta(n^3)$

主定理法：实例三

- 主定理(简化形式): 对形如 $T(n) = aT\left(\frac{n}{b}\right) + n^k$ 的递归式

$$T(n) = \begin{cases} \Theta(n^k) & \text{if } k > \log_b a \quad \textcircled{1} \\ \Theta(n^k \log n) & \text{if } k = \log_b a \quad \textcircled{2} \\ \Theta(n^{\log_b a}) & \text{if } k < \log_b a \quad \textcircled{3} \end{cases}$$



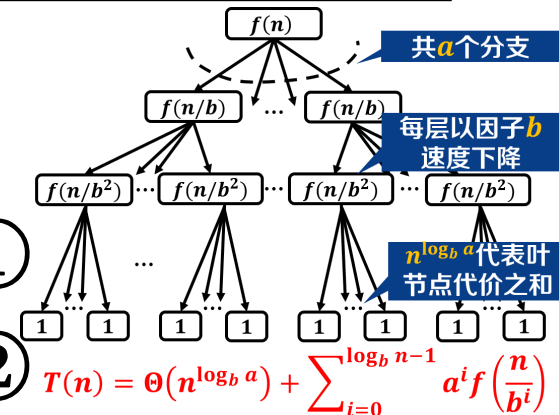
- 例三: $T(n) = 4T\left(\frac{n}{4}\right) + \sqrt{n}$
 - $k = \frac{1}{2}$
 - $a = 4, b = 4, \log_b a = \log_4 4 = 1$
 - $k < \log_b a$, 属于情况③
 - $T(n) = \Theta(n^{\log_b a}) = \Theta(n)$

主定理法：实例四



- 主定理：对形如 $T(n) = aT\left(\frac{n}{b}\right) + f(n)$ 的递归式

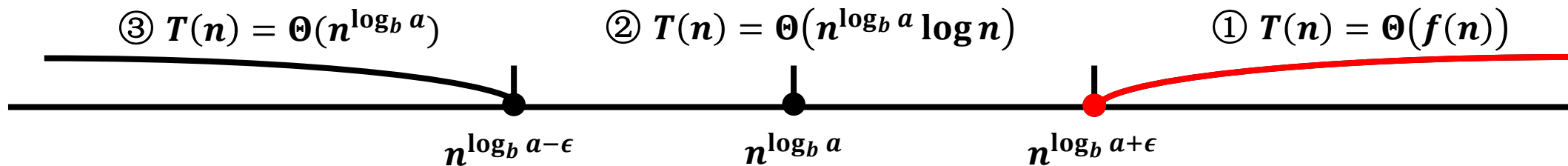
$$T(n) = \begin{cases} \Theta(f(n)) & \text{if } f(n) = \Omega(n^{\log_b a + \epsilon}) \quad \textcircled{1} \\ \Theta(n^{\log_b a} \log n) & \text{if } f(n) = \Theta(n^{\log_b a}) \quad \textcircled{2} \\ \Theta(n^{\log_b a}) & \text{if } f(n) = O(n^{\log_b a - \epsilon}) \quad \textcircled{3} \end{cases}$$



- 例四： $T(n) = 3T\left(\frac{n}{4}\right) + n \log n$

- $\log_b a = \log_4 3 < 1$, 则 $\exists \epsilon > 0$, 使得 $\log_b a + \epsilon < 1$, 故 $f(n) = \Omega(n^{\log_b a + \epsilon})$
- $\exists c = \frac{3}{4}$ 时, $af\left(\frac{n}{b}\right) = \frac{3n}{4} \log\left(\frac{n}{4}\right) < cf(n) = \frac{3}{4} n \log n$, 属于情况①
- $T(n) = \Theta(f(n)) = \Theta(n \log n)$

“正则”条件满足

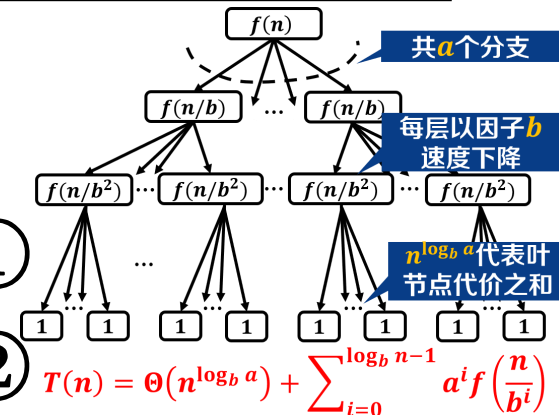


主定理法：实例五



- 主定理：对形如 $T(n) = aT\left(\frac{n}{b}\right) + f(n)$ 的递归式

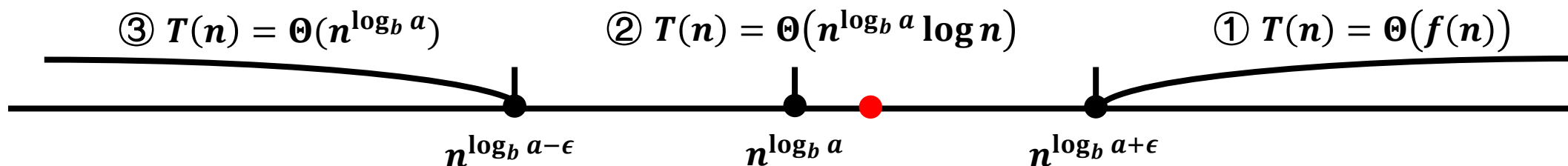
$$T(n) = \begin{cases} \Theta(f(n)) & \text{if } f(n) = \Omega(n^{\log_b a + \epsilon}) \quad ① \\ \Theta(n^{\log_b a} \log n) & \text{if } f(n) = \Theta(n^{\log_b a}) \quad ② \\ \Theta(n^{\log_b a}) & \text{if } f(n) = O(n^{\log_b a - \epsilon}) \quad ③ \end{cases}$$



- 例五： $T(n) = 2T\left(\frac{n}{2}\right) + n \log n$

- $\log_b a = \log_2 2 = 1, f(n) = \Omega(n^{\log_b a})$
- 然而对 $\forall \epsilon > 0, \log n$ 渐进小于 n^ϵ , 故 $\nexists \epsilon > 0$ 使 $f(n) = \Omega(n^{\log_b a + \epsilon})$
- 该情况落入①和②之间，不能使用主定理

上述主定理不适用
扩展形式主定理可解决



主定理法：例五



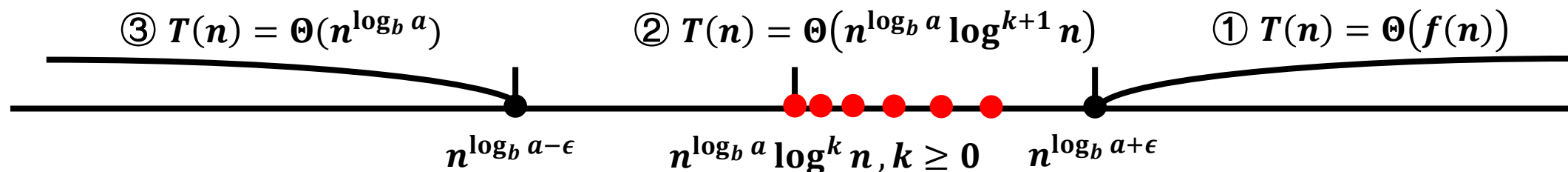
- 主定理(扩展形式): 对形如 $T(n) = aT\left(\frac{n}{b}\right) + f(n)$ 的递归式

$$T(n) = \begin{cases} \Theta(f(n)) & \text{if } f(n) = \Omega(n^{\log_b a + \epsilon}) \\ \Theta(n^{\log_b a} \log^{k+1} n) & \text{if } f(n) = \Theta(n^{\log_b a} \log^k n), k \geq 0 \\ \Theta(n^{\log_b a}) & \text{if } f(n) = O(n^{\log_b a - \epsilon}) \end{cases}$$

①
②
③

- 例五: $T(n) = 2T\left(\frac{n}{2}\right) + n \log n$

- $\log_b a = \log_2 2 = 1$
- $k = 1, f(n) = \Theta(n^{\log_b a} \log^k n)$, 属于情况②
- $T(n) = \Theta(n^{\log_b a} \log^{k+1} n) = \Theta(n \log^2 n)$



递归式分析：主定理法

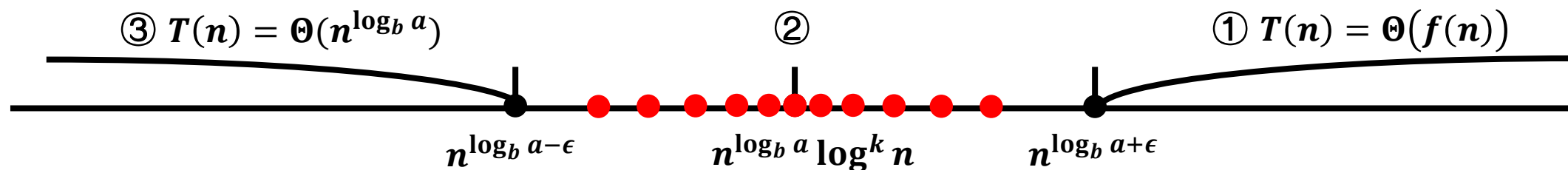


- 主定理(扩展形式): 对形如 $T(n) = aT\left(\frac{n}{b}\right) + f(n)$ 的递归式

$$T(n) = \begin{cases} \Theta(f(n)) & \text{if } f(n) = \Omega(n^{\log_b a + \epsilon}) \\ \Theta(n^{\log_b a} \log^{k+1} n) & \text{if } f(n) = \Theta(n^{\log_b a} \log^k n), k \geq 0 \\ \Theta(n^{\log_b a}) & \text{if } f(n) = O(n^{\log_b a - \epsilon}) \end{cases} \quad \begin{matrix} \textcircled{1} \\ \textcircled{2} \\ \textcircled{3} \end{matrix}$$

- 情况②的三种扩展

$$T(n) = \begin{cases} \Theta(n^{\log_b a} \log^{k+1} n) & k > -1 \\ \Theta(n^{\log_b a} \log \log n) & k = -1 \\ \Theta(n^{\log_b a}) & k < -1 \end{cases} \quad \textcircled{2}$$



- 递归式分析方法比较

分析方法	优点	缺点
递归树法	直观形象	难以展开
代入法	适用广泛	难猜通解
主定理法	形式简洁	适用有限

谢谢

