第一章 算法分析的数学基础

东南大学计算机学院 方效林

本章内容

- 复杂性函数的阶
- 和的估计与界限
- 递归方程

一些记号

- $\lfloor x \rfloor$表示小于等于x的最大整数
- $\lceil x \rceil$表示大于等于x的最小整数
  - $x - 1 < \lfloor x \rfloor \le x \le \lceil x \rceil < x + 1$
- $\log n = \log_2 n$，$\lg n = \log_2 n$
- $\ln n = \log_e n$

复杂性函数的阶

- 渐近复杂性
  - 当输入规模趋于极限情形时(相当大)的复杂性
  - 表示复杂性阶的三个记号
  - $T(n)=O(f(n))$
    - 若存在$c > 0$，和正整数$n_0 \geq 1$，使得当$n \geq n_0$时，有$T(n) \leq c*f(n)$成立。
    - 给出算法复杂度的上界，不可能比$c*f(n)$更大
    - e.g. $T(n)=3n^3+2n^2$，取$c=5$，$n_0=1$，$f(n)=n^3$，则当$n \geq n_0(=1)$时，有$3n^3+2n^2 \leq 5n^3$. $\therefore T(n)= O(n^3)$

复杂性函数的阶

- 渐近复杂性
  - 当输入规模趋于极限情形时(相当大)的复杂性
  - 表示复杂性阶的三个记号
  - $T(n)=\Omega(f(n))$
    - 若存在c > 0，和正整数$n_0 \geq 1$，使得当$n \geq n_0$时，有$T(n) \geq c*f(n)$成立。
    - 给出算法复杂度的下界，不可能比c*f(n)更小
    - e.g. $T(n)=3n^3+2n^2$，取c=3，$n_0=1$，$f(n)=n^3$，则当$n \geq n_0(=1)$时，有$3n^3+2n^2 \geq 3n^3$，$\therefore T(n)=\Omega(n^3)$

复杂性函数的阶

- **渐近复杂性**
  - 当输入规模趋于极限情形时(相当大)的复杂性
  - 表示复杂性阶的三个记号
  - $T(n)=\Theta(f(n))$
    - 若存在$c_1,c_2>0$，和正整数$n_0\geq1$，使得当$n\geq n_0$时，总有$T(n)\leq c_1*f(n)$且$T(n)\geq c_2*f(n)$成立，即$T(n)=O(f(n))$与$T(n)=\Omega(f(n))$都成立。
    - 给出了算法时间复杂度的上界和下界
    - e.g.$T(n)= 3n^3+2n^2$，$c_1=5$，取$c_2=3$，$n_0=1$，$f(n)=n^3$，则当$n\geq n_0(=1)$时，有$3n^3+2n^2\leq5n^3$及$3n^3+2n^2\geq3n^3$（无穷多个），$\therefore T(n)= \Theta (n^3)$

多项式时间与指数时间

- 设每秒可做某基本运算10^9次，n=60

	算法1	算法2	算法3	算法4	算法5	算法6
复杂度	n	n^2	n^3	n^5	2^n	3^n
运算时	$6*10^{-8}$s	$3.6*10^{-6}$s	$2.16*10^{-4}$s	0.013min	3.66世纪	$1.3*10^{13}$世纪

- 两个结论
  - 多项式时间的算法互相之间虽有差距，一般可接受
  - 指数量级时间的算法对于较大的n无实用价值

和的估计与界限

- 直接求和的界限
  - $\sum_{k=1}^{n} k \le \sum_{k=1}^{n} n \le n^2$

和的估计与界限

- 直接求和的界限
  - $\sum_{k=1}^{n} a_k \leq n \max\limits_{1 \leq k \leq n} \{a_k\}$

和的估计与界限

- 直接求和的界限
  - 对于所有$k \geq 0$，有$\frac{a_{k+1}}{a_k} \leq r < 1$，求$\sum_{k=1}^{n} a_k$上界
    - $\frac{a_1}{a_0} \leq r \rightarrow a_1 \leq a_0 r$
    - $\frac{a_2}{a_1} \leq r \rightarrow a_2 \leq a_1 r \leq a_0 r^2$
    - …
    - $\frac{a_{k+1}}{a_k} \leq r \rightarrow a_{k+1} \leq a_k r \leq a_0 r^k$
    - $\sum_{k=1}^{n} a_k \leq \sum_{k=1}^{n} a_0 r^k = a_0 \sum_{k=1}^{n} r^k \leq \frac{a_0}{1-r}$

和的估计与界限

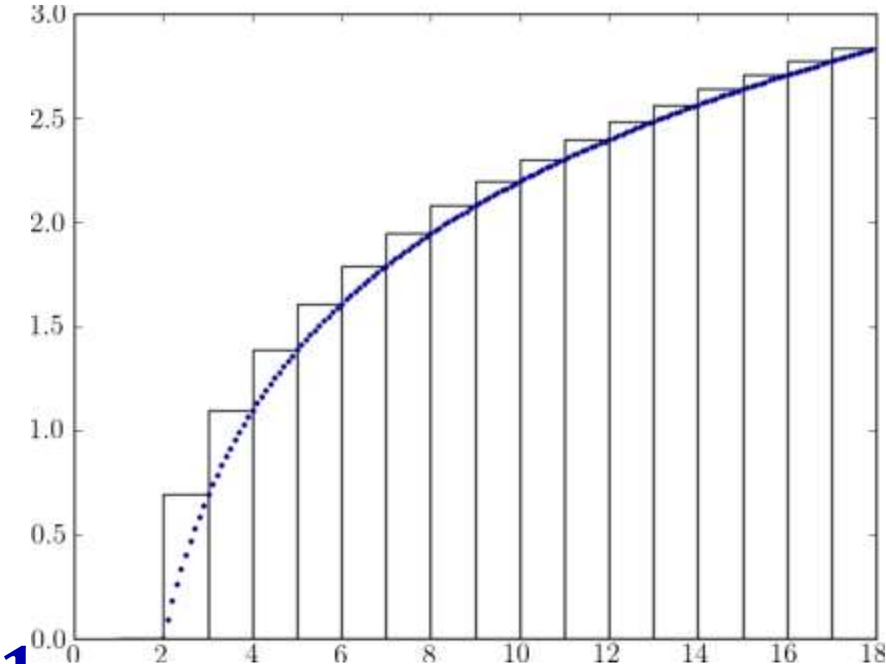
- **直接求和的界限**
  - 对于所有$k \geq 0$，有$\frac{a_{k+1}}{a_k} \leq r < 1$，求$\sum_{k=1}^{n} a_k$上界
  - 求$\sum_{k=1}^{\infty} k/3^k$上界
    - $\frac{\frac{k+1}{3^{k+1}}}{\frac{k}{3^k}} = \frac{1}{3}\frac{k+1}{k} \leq \frac{2}{3}$
    - $\sum_{k=1}^{\infty} k/3^k \leq \sum_{k=1}^{\infty} a_1 r^k = \sum_{k=1}^{\infty} \frac{1}{3}\left(\frac{2}{3}\right)^k = \frac{1}{3} \cdot \frac{1}{1-\frac{2}{3}} = 1$

和的估计与界限

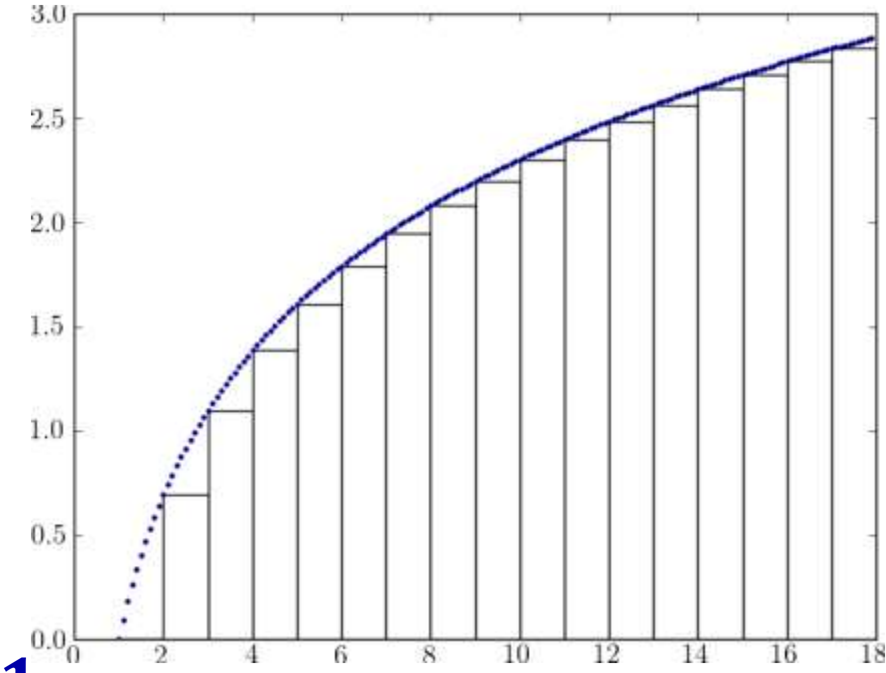
- 直接求和的界限
  - 对于所有$k \ge 0$，有$\frac{a_{k+1}}{a_k} \le r < 1$，求$\sum_{k=1}^{n} a_k$上界
  - 求$\sum_{k=0}^{\infty} k^2/2^k$上界
    - 当$k \ge 3$时，有$\frac{\frac{(k+1)^2}{2^{k+1}}}{\frac{k^2}{2^k}} = \frac{1}{2}\frac{(k+1)^2}{k^2} \le \frac{8}{9}$
    - $\sum_{k=0}^{\infty} k^2/2^k \le \sum_{k=0}^{2} \frac{k^2}{2^k} + \sum_{k=3}^{\infty} \frac{k^2}{2^k} \le \sum_{k=0}^{2} \frac{k^2}{2^k} + \sum_{k=3}^{\infty} \frac{9}{8}\left(\frac{8}{9}\right)^k = O(1)$

和的估计与界限

- 求和转换为求积分
  - $\log n! = \sum_{i=1}^{n} \log i$
  - 曲线之下面积
  - $\therefore \log n! > \int_1^n \log x \, dx$
  - $\because \log x = \log e \ln x$
  - $\int_1^n \ln x \, dx = n \ln n - n + 1$
  - $\therefore \log n! > (n \ln n - n + 1) \log e$
  - $\log n! = \Omega(n \log n)$



和的估计与界限



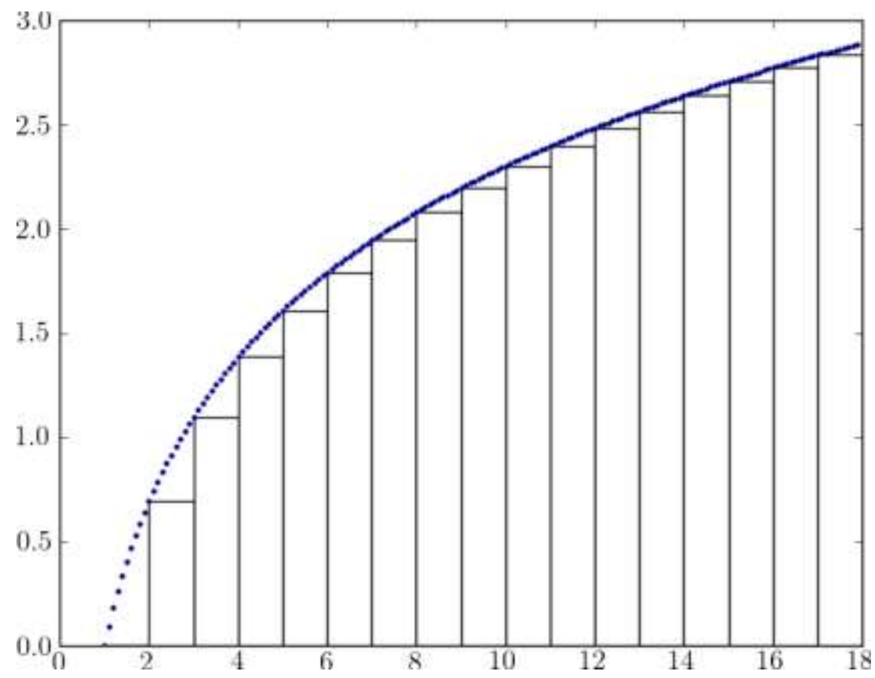
- 求和转换为求积分
  - $\log n! = \sum_{i=1}^{n} \log i$
  - 曲线之下面积
  - $\log n! < \int_{1}^{n+1} \log x \, dx$
  - $\because \log x = \log e \ln x$
  - $\int_{1}^{n} \ln x \, dx = n \ln n - n + 1$
  - $\therefore \log n! < [(n+1)\ln(n+1) - (n+1) + 1] \log e$
  - $\log n! = O(n \log n)$

$$\log n! = \Theta(n \log n)$$

和的估计与界限

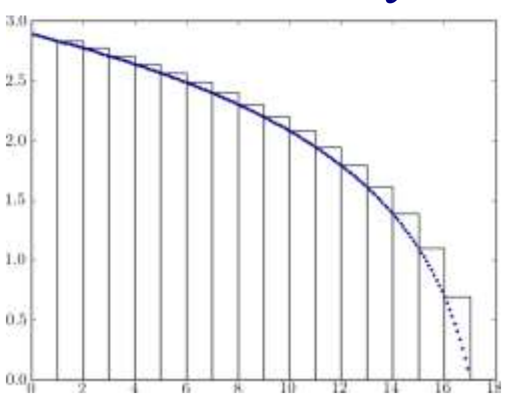
- 求和转换为求积分
  - 当$f(x)$单调递增时，有
  - $\int_{m-1}^{n} f(x)\,dx \le \sum_{m}^{n} f(x) \le \int_{m}^{n+1} f(x)\,dx$

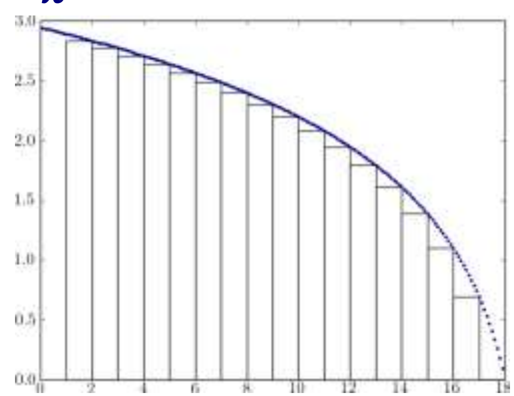
- 求和转换为求积分
  - 当$f(x)$单调递增时，有
  - $\int_{m-1}^{n} f(x)\,dx \le \sum_{m}^{n} f(x) \le \int_{m}^{n+1} f(x)\,dx$



和的估计与界限

- 求和转换为求积分
  - 类似地，当$f(x)$单调递减时，有
  - $\int_{m}^{n+1} f(x)\,dx \le \sum_{m}^{n} f(x) \le \int_{m-1}^{n} f(x)\,dx$
  - 例如：
    - $\sum_{i=1}^{n}\frac{1}{i} = 1 + \sum_{i=2}^{n}\frac{1}{i} < 1 + \int_{1}^{n}\frac{dx}{x} = 1 + \ln n$
    - $\sum_{i=1}^{n}\frac{1}{i} > \int_{1}^{n+1}\frac{dx}{x} = \ln(n+1)$





递归方程

- 例： Merge-sort排序算法的复杂性方程
  - $$T(n) = \begin{cases} \Theta(1), & n = 1 \\ 2T\left(\frac{n}{2}\right) + \Theta(n), & n > 1 \end{cases}$$
  - 解：$T(n) = \Theta(n \log n)$

递归方程

- 递归逐层展开求解
  - $T(n) = n + 3T\left(\left\lfloor\frac{n}{4}\right\rfloor\right)$
  - $= n + 3(\left\lfloor\frac{n}{4}\right\rfloor + 3T\left(\left\lfloor\frac{n}{16}\right\rfloor\right))$
  - $= n + 3(\left\lfloor\frac{n}{4}\right\rfloor + 3\left(\left\lfloor\frac{n}{16}\right\rfloor + 3T(\left\lfloor\frac{n}{64}\right\rfloor)\right))$
  - $= n + 3\left\lfloor\frac{n}{4}\right\rfloor + 3^2\left\lfloor\frac{n}{4^2}\right\rfloor + 3^3\left\lfloor\frac{n}{4^3}\right\rfloor + \cdots + 3^k T(\left\lfloor\frac{n}{4^k}\right\rfloor)$
  - 深度 $k = \log_4 n$，最底层有$3^k = 3^{\log_4 n} = n^{\log_4 3}$个
  - $T(n) = \sum_{i=0}^{(\log_4 n)-1} 3^i \frac{n}{4^i} + \Theta(n^{\log_4 3})$
  - $\leq 4n + \Theta(n^{\log_4 3}) = O(n)$

递归方程

- 变量替换法求解
  - $T(n) = 2T(\sqrt{n}) + \log n$
  - 令 $m = \log n$，则$n = 2^m$， $T(2^m) = 2T\left(2^{\frac{m}{2}}\right) + m$
  - 令 $S(m) = T(2^m)$，则 $S\left(\frac{m}{2}\right) = T\left(2^{\frac{m}{2}}\right)$
  - 于是 $S(m) = 2S\left(\frac{m}{2}\right) + m$
  - 显然 $S(m) = \Theta(m \log m)$
  - 则 $T(n) = \Theta(\log n \log(\log n))$

递归方程

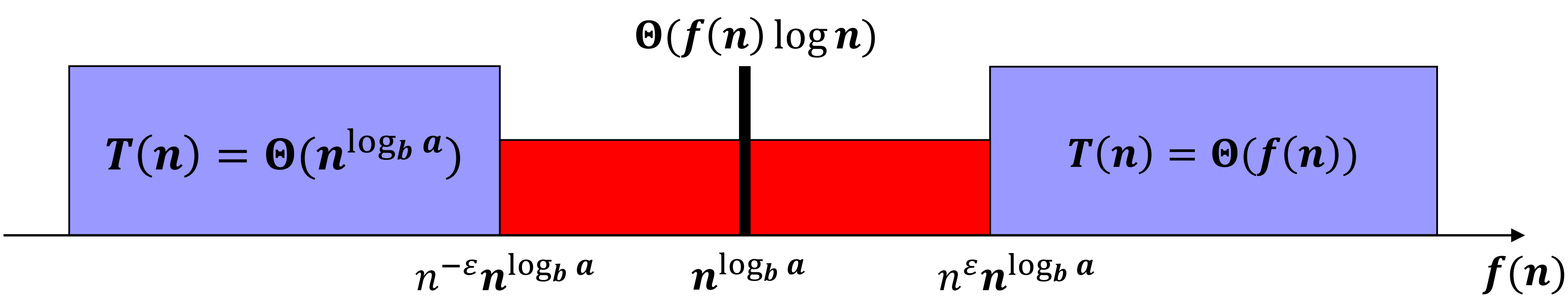
- **Master定理求解**
  - 求解 $T(n) = a \cdot T\left(\frac{n}{b}\right) + f(n)$ 型递归方程，其中 $a \geq 1$，$b > 1$ 是常数，$f(n)$ 是正函数
    - 记住三种情况，可快速求解

递归方程

- **Master定理求解**
  - 求解 $T(n) = a \cdot T\left(\frac{n}{b}\right) + f(n)$ 型递归方程，其中 $a \geq 1$，$b > 1$ 是常数，$f(n)$ 是正函数
    - 若 $f(n) = O(n^{(\log_b a)-\varepsilon})$，$\varepsilon > 0$ 是常数，则有$T(n) = \Theta(n^{\log_b a})$
    - 若 $f(n) = \Theta(n^{\log_b a})$，$\varepsilon > 0$ 是常数， 则有$T(n) = \Theta(n^{\log_b a} \log n)$
    - 若 $f(n) = \Omega(n^{(\log_b a)+\varepsilon})$，$\varepsilon > 0$ 是常数，且对所有充分大的 n 有 $af(\frac{n}{b}) \leq cf(n)$，$c < 1$ 是常数，则有 $T(n) = \Theta(f(n))$

递归方程

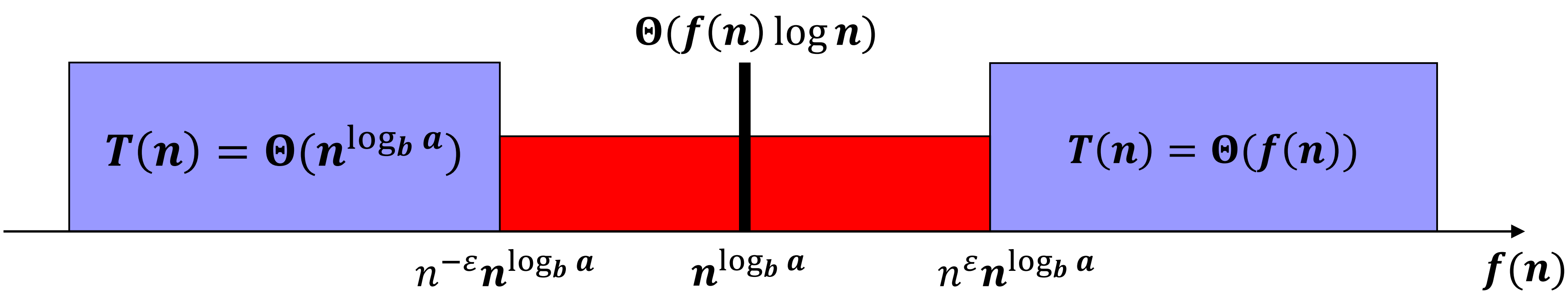
- **Master定理（直观理解，一般情况）**
  - 用 $f(n)$ 与 $n^{\log_b a}$ 的阶比较，
    - 若 $n^{\log_b a}$ 更大，则 $T(n) = \Theta(n^{\log_b a})$
    - 若 $f(n) = \Theta(n^{\log_b a})$，即 $f(n)$ 与 $n^{\log_b a}$ 同阶， 则有 $T(n) = \Theta\left(n^{\log_b a}\log n\right) = \Theta(f(n)\log n)$
    - 若 $f(n)$ 更大，则 $T(n) = \Theta(f(n))$



对于红色部分，Master定理无能为力

递归方程

- **Master定理（更进一步理解）**
  - 用 $f(n)$ 与 $n^{\log_b a}$ 的阶比较，
    - 第一种情况，$f(n)$不仅小于 $n^{\log_b a}$，而且要小于 $n^{\log_b a}/n^{\varepsilon}$，即$f(n) = O(n^{(\log_b a)-\varepsilon})$
    - 第三种情况，$f(n)$不仅大于 $n^{\log_b a}$，而且要大于 $n^{\log_b a} * n^{\varepsilon}$，即$f(n) = \Omega(n^{(\log_b a)+\varepsilon})$



对于红色部分，Master定理无能为力

递归方程

- **Master定理（例子）**
  - 求解 $T(n)=9T\left(\frac{n}{3}\right)+n$
  - $a=9, b=3, f(n)=n, n^{\log_b a}=n^2$
  - $\because f(n)=n=O(n^{(\log_b a)-\varepsilon})$, 即 $\varepsilon=1$
  - $\therefore T(n)=\Theta\left(n^{\log_b a}\right)=\Theta\left(n^2\right)$

递归方程

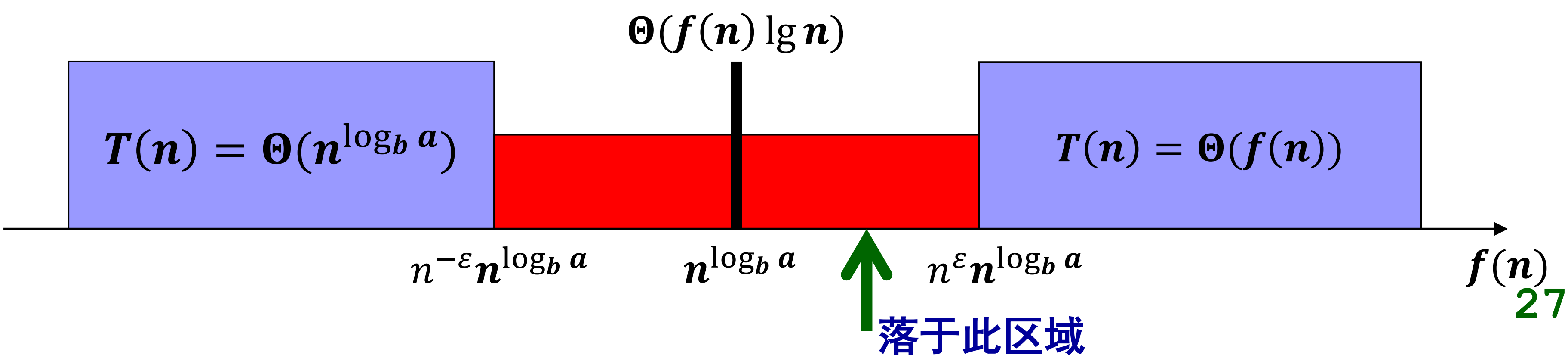
- **Master定理（例子）**
  - 求解 $T(n) = T\left(\frac{2n}{3}\right) + 1$
  - $a = 1,\ b = \frac{3}{2},\ f(n) = 1,\ n^{\log_b a} = n^{\log_{3/2} 1} = 1$
  - $\because f(n) = 1 = \Theta(n^{\log_b a}),$
  - $\therefore T(n) = \Theta\left(n^{\log_b a} \lg n\right) = \Theta(\lg n)$

递归方程

- Master定理（例子）
  - 求解 $T(n) = 3T\left(\frac{n}{4}\right) + n\log n$
  - $a = 3, b = 4,$
  - $f(n) = n\log n, \quad n^{\log_b a} = n^{\log_4 3} = O(n^{0.793})$
  - $\because f(n) = n\log n \geq n = \Theta(n^{\log_b a+\varepsilon}), \varepsilon \approx 0.2$
  - 对所有n有 $af\left(\frac{n}{b}\right) = 3\frac{n}{b}\log\frac{n}{b} \leq \frac{3}{4}n\log n = cf(n)$
  - $\therefore T(n) = \Theta(f(n)) = \Theta(n\log n)$

递归方程

- **Master定理（例子）**
  - 求解 $T(n) = 2T\left(\frac{n}{2}\right) + n\log n$
  - $a = 2,\ b = 2,$
  - $f(n) = n\log n,\quad n^{\log_b a} = n^{\log_2 2} = O(n)$
  - 虽然 $f(n) = n\log n \geq n^{\log_b a} = n$,但是$\frac{f(n)}{n^{\log_b a}} = \log n$渐近小于 n^{ε}



Master定理证明

- 证明思路
  - $a \geq 1$，$b > 1$ 是常数，$f(n)$ 是正函数
  - 对 $T(n) = a \cdot T\left(\frac{n}{b}\right) + f(n)$ 展开可得
  - $T(n) = \Theta\left(n^{\log_b a}\right) + \sum_{i=0}^{k-1} a^i f\left(\frac{n}{b^i}\right)$,
  - $n = b^k, k = \log_b n, a^k = a^{\log_b n} = n^{\log_b a}$
  - 令 $g(n) = \sum_{i=0}^{k-1} a^i f\left(\frac{n}{b^i}\right)$
  - $T(n) = \Theta\left(n^{\log_b a}\right) + g(n)$

Master定理证明

$$g(n)=\sum_{i=0}^{k-1}a^i f(\frac{n}{b^i})$$

- 证明思路

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

  - 若 $f(n)=O(n^{(\log_b a)-\varepsilon})$，$\varepsilon>0$ 是常数，则有
  - $g(n)=O(\sum_{i=0}^{k-1}a^i\left(\frac{n}{b^i}\right)^{(\log_b a)-\varepsilon})$
  - $=O(n^{(\log_b a)-\varepsilon}\sum_{i=0}^{k-1}\left(\frac{ab^\varepsilon}{b^{\log_b a}}\right)^i)$
  - $=O(n^{(\log_b a)-\varepsilon}\sum_{i=0}^{k-1}(b^\varepsilon)^i)$
  - $=O(n^{(\log_b a)-\varepsilon}\sum_{i=0}^{k-1}(b^\varepsilon)^i)$
  - $=O(n^{(\log_b a)-\varepsilon}\frac{n^\varepsilon-1}{b^\varepsilon-1})$
  - $=O\left(n^{\log_b a}\right)$
  - $\therefore T(n)=\Theta\left(n^{\log_b a}\right)+g(n)=\Theta\left(n^{\log_b a}\right)$

Master定理证明

$$g(n)=\sum_{i=0}^{k-1}a^i f(\frac{n}{b^i})$$

- **证明思路**

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

  - 若 $f(n)=\Theta(n^{\log_b a})$，$\varepsilon>0$ 是常数， 则有
  - $g(n)=\Theta(\sum_{i=0}^{k-1}a^i\left(\frac{n}{b^i}\right)^{\log_b a})$
  - $=\Theta(n^{\log_b a}\sum_{i=0}^{k-1}\left(\frac{a}{b^{\log_b a}}\right)^i)$
  - $=\Theta(n^{\log_b a}\sum_{i=0}^{k-1}1)$
  - $=\Theta(n^{\log_b a}k)$
  - $=\Theta(n^{\log_b a}\log_b n)$
  - $=\Theta\left(n^{\log_b a}\log n\right)$
  - $\therefore T(n)=\Theta\left(n^{\log_b a}\right)+g(n)=\Theta(n^{\log_b a}\log n)$

Master定理证明

$$g(n)=\sum_{i=0}^{k-1}a^i f(\frac{n}{b^i})$$

- **证明思路**

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

  - 若 $f(n)=\Omega(n^{(\log_b a)+\varepsilon})$，$\varepsilon>0$ 是常数，且对所有充分大的 n 有 $af(\frac{n}{b})\le cf(n)$，$c<1$ 是常数，则有
  - $af\left(\frac{n}{b^2}\right)\le cf\left(\frac{n}{b}\right)$
  - $\vdots$
  - $af\left(\frac{n}{b^i}\right)\le cf\left(\frac{n}{b^{i-1}}\right)$
  - 两边分别相乘，可得 $a^i f(\frac{n}{b^i})\le c^i f(n)$
  - $g(n)=\sum_{i=0}^{k-1}a^i f(\frac{n}{b^i})\le\sum_{i=0}^{k-1}c^i f(n)=f(n)\sum_{i=0}^{k-1}c^i$
  - $\le f(n)\frac{1}{1-c}=\Theta(f(n))$
  - $\therefore T(n)=\Theta\left(n^{\log_b a}\right)+g(n)=\Theta(f(n))$

$$T(n)=7T(n/7)+n$$

$$T(n)=8T(n/6)+n^{3/2}\log n$$

$$T(n)=2T(n^{1/3})+1$$