

# 第2讲 基础知识

罗国杰

[gluo@pku.edu.cn](mailto:gluo@pku.edu.cn)

2025年春季学期

# 数学基础

- 函数的渐近的界
- 常见函数的阶
- 递推方程求解
  - ▶ 递归树
  - ▶ 主定理证明及应用
  - ▶ 生成函数

# 函数的渐近的界

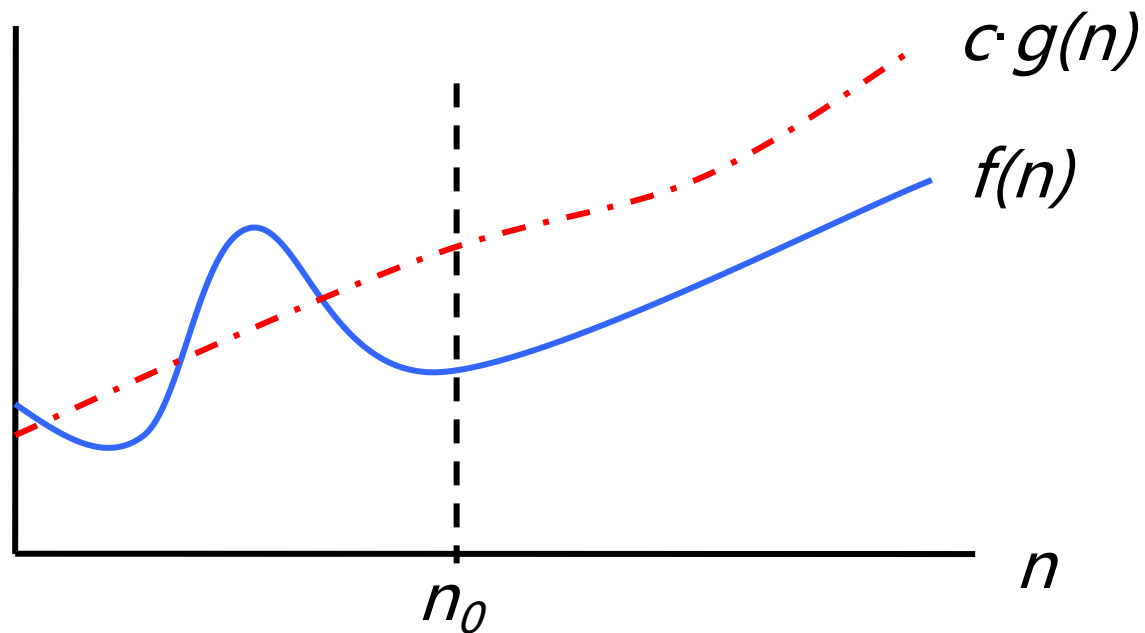
➡ 下列公式表示的确切含义是什么？

- ▶  $f(n) = O(g(n))$  : Big Oh  
 $f(n) = \Omega(g(n))$  : Big Omega
- ▶  $f(n) = o(g(n))$  : little oh  
 $f(n) = \omega(g(n))$  : little omega
- ▶  $f(n) = \Theta(g(n))$  : Big Theta

# 大O记号 (渐近上界)

►  $f(n) = O(g(n))$

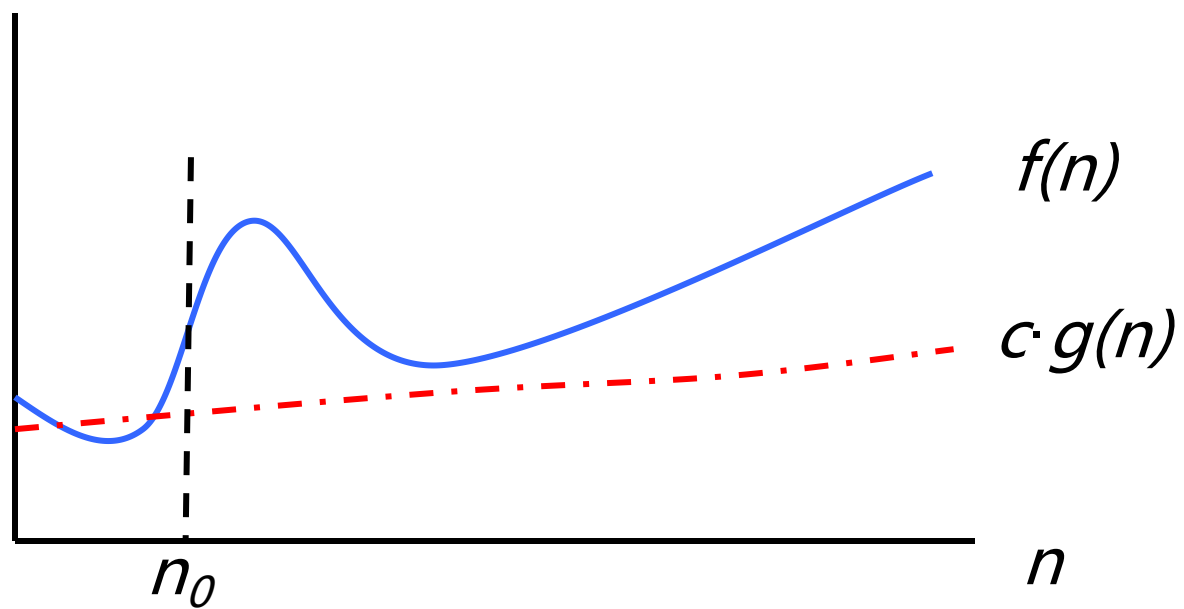
► iff  $\exists c, n_0 > 0$ , s.t.  $\forall n \geq n_0: 0 \leq f(n) \leq c \cdot g(n)$



# 大 $\Omega$ 记号 (渐近下界)

►  $f(n) = \Omega(g(n))$

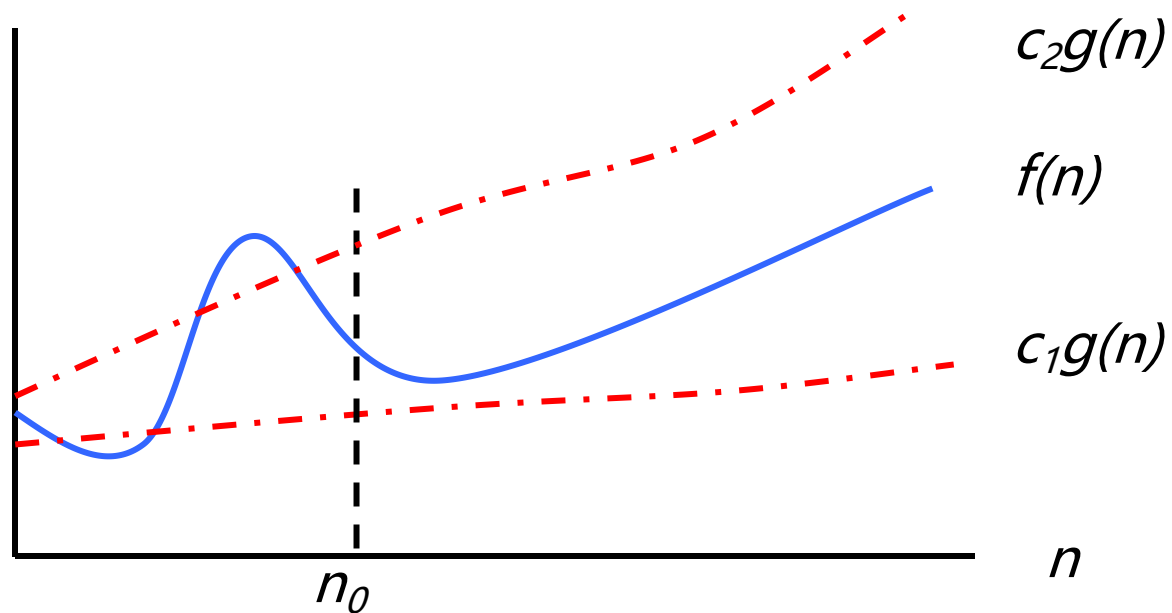
► iff  $\exists c, n_0 > 0$ , s.t.  $\forall n \geq n_0, 0 \leq c \cdot g(n) \leq f(n)$



# 大 $\Theta$ 记号 (渐近紧确界)

►  $f(n) = \Theta(g(n))$

iff  $\exists c_1, c_2, n_0 > 0$ , s.t.  $\forall n > n_0: 0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n)$



# 函数的渐近的界

**定义** 设  $f$  和  $g$  是定义域为自然数集  $N$  上的函数

1. 若存在正数  $c$  和  $n_0$ , 使得对一切  $n > n_0$  有  $0 \leq f(n) \leq cg(n)$  成立, 则称  $f(n)$  的渐近的上界是  $g(n)$ , 记作  $f(n) = O(g(n))$ 。// 也记作  $f(n) \in O(g(n))$
2. 若存在正数  $c$  和  $n_0$ , 使得对一切  $n > n_0$  有  $0 \leq cg(n) \leq f(n)$  成立, 则称  $f(n)$  的渐近的下界是  $g(n)$ , 记作  $f(n) = \Omega(g(n))$ 。// 也记作  $f(n) \in \Omega(g(n))$
3. 若  $f(n) = O(g(n))$  且  $f(n) = \Omega(g(n))$ , 则记作  $f(n) = \Theta(g(n))$ 。// 也记作  $f(n) \in \Theta(g(n))$

# 函数的渐近的界

4. 若对任意正数  $c$  都存在  $n_0$ , 使得当  $n \geq n_0$  时有  $0 \leq f(n) < cg(n)$  成立, 则记作  $f(n) = o(g(n))$ .

即  $f(n) = O(g(n))$  但  $f(n) \neq \Theta(g(n))$

5. 若对任意正数  $c$  都存在  $n_0$ , 使得当  $n \geq n_0$  时有  $0 \leq cg(n) < f(n)$  成立, 则记作  $f(n) = \omega(g(n))$ .

即  $f(n) = \Omega(g(n))$  但  $f(n) \neq \Theta(g(n))$

**例 函数**  $f(n) = n^2 + n$ ,

$$f(n) = O(n^2), f(n) = O(n^3), f(n) = o(n^3), f(n) = \Theta(n^2)$$



## 有关定理

**定理** 设  $f$  和  $g$  是定义域为自然数集合的函数.

(1) 如果  $\lim_{n \rightarrow \infty} \frac{f(n)}{g(n)} = c > 0$ , 那么  $f(n) = \Theta(g(n))$ .

(2) 如果  $\lim_{n \rightarrow \infty} \frac{f(n)}{g(n)} = 0$ , 那么  $f(n) = o(g(n))$ .

(3) 如果  $\lim_{n \rightarrow \infty} \frac{f(n)}{g(n)} = +\infty$ , 那么  $f(n) = \omega(g(n))$ .

## 有关阶的一些性质

► **定理** 设  $f, g, h$  是定义域为自然数集合的函数,

1. 如果  $f = O(g)$  且  $g = O(h)$ , 那么  $f = O(h)$ .
2. 如果  $f = \Omega(g)$  且  $g = \Omega(h)$ , 那么  $f = \Omega(h)$ .
3. 如果  $f = \Theta(g)$  和  $g = \Theta(h)$ , 那么  $f = \Theta(h)$ .

► **定理** 假设  $f$  和  $g$  是定义域为自然数集合的函数, 若对某个其它的函数  $h$ , 有  $f = O(h)$  和  $g = O(h)$ , 那么  $f + g = O(h)$ .

## 例题

**例题** 设  $f(n) = \frac{1}{2}n^2 - 3n$ , 证明  $f(n) = \Theta(n^2)$ 。

证：因为

$$\lim_{n \rightarrow \infty} \frac{f(n)}{n^2} = \lim_{n \rightarrow \infty} \frac{\frac{1}{2}n^2 - 3n}{n^2} = \frac{1}{2}$$

根据定理有  $f(n) = \Theta(n^2)$ 。

**可以证明：多项式函数，幂函数的阶低于指数函数**

$$n^d = o(r^n), \quad r > 1, \quad d > 0$$

## 对数函数的阶

符号：

$$\log n = \log_2 n$$

$$\log^k n = (\log n)^k$$

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

性质：

$$\log_b n = o(n^\alpha) \quad \alpha > 0$$

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

$$\log_k n = \Theta(\log_l n)$$

# 阶乘函数的阶

Stirling公式 
$$n! = \sqrt{2\pi n} \left(\frac{n}{e}\right)^n (1 + \Theta(\frac{1}{n}))$$

$$n! = o(n^n)$$

$$n! = \Omega(2^n)$$

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

$$\lim_{n \rightarrow +\infty} \frac{\log(n!)}{n \log n} = \lim_{n \rightarrow +\infty} \frac{\ln(n!) / \ln 2}{n \ln n / \ln 2} = \lim_{n \rightarrow +\infty} \frac{\ln(n!)}{n \ln n}$$

$$= \lim_{n \rightarrow +\infty} \frac{\ln(\sqrt{2\pi n} (\frac{n}{e})^n (1 + (\frac{c}{n})))}{n \ln n} = \lim_{n \rightarrow +\infty} \frac{\ln \sqrt{2\pi n} + n \ln \frac{n}{e}}{n \ln n} = 1$$

上述的 $c$ 为某个常数

## 例题：函数的阶

按照阶从高到低对以下函数排序：

$$\log^2 n, 1, n!, n2^n, n^{n/\log n}, \left(\frac{3}{2}\right)^n, \sqrt{\log n}, (\log n)^{\log n}, 2^{2^n}, \\ n^{\log \log n}, n^3, \log \log n, n \log n, n, 2^{\log n}, \log n, \log(n!)$$

结果：

$$2^{2^n}, n!, n2^n, n^{n/\log n}, \left(\frac{3}{2}\right)^n, (\log n)^{\log n} = n^{\log \log n}, n^3, \log(n!) = \\ \Theta(n \log n), n = 2^{\log n}, \log^2 n, \log n, \sqrt{\log n}, \log \log n$$

# 多项式函数与指数函数

时间复杂度函数	问题规模					
	10	20	30	40	50	60
$n$	10 ns	20 ns	30 ns	40 ns	50 ns	60 ns
$n^2$	0.1 $\mu$ s	0.4 $\mu$ s	0.9 $\mu$ s	1.6 $\mu$ s	2.5 $\mu$ s	3.6 $\mu$ s
$n^3$	1 $\mu$ s	8 $\mu$ s	27 $\mu$ s	64 $\mu$ s	125 $\mu$ s	216 $\mu$ s
$n^5$	0.1 ms	3.2 ms	24.3 ms	0.10 s	0.31 s	0.78 s
$2^n$	1 $\mu$ s	1 ms	1.1 s	18 min	13 d	37 y
$3^n$	59 $\mu$ s	3.5 s	57 h	3.9 C	$2 \times 10^5$ C	$1.3 \times 10^{10}$ C

\* 假设单个操作耗时1ns

# 10亿次/秒机器求解的问题

- 快速排序算法给10万个数据排序, 运算量约为 $10^5 \times \log_2 10^5 \approx 1.7 \times 10^6$ , 仅需 $1.7 \times 10^6 / 10^9 = 1.7 \times 10^{-3}$ 秒.
- Dijkstra算法求解1万个顶点的图的单源最短路径问题, 运算量约为 $(10^4)^2 = 10^8$ , 约需 $10^8 / 10^9 = 0.1$ 秒.
- 回溯法解100个顶点的图的最大团问题, 运算量为 $100 \times 2^{100} \approx 1.8 \times 10^{32}$ , 需要 $1.8 \times 10^{32} / 10^9 = 1.8 \times 10^{21}$ 秒  
 $= 5.7 \times 10^{15}$ 年, 即5千7百万亿年!
- 1分钟能解多大的问题: 1分钟=60秒, 这台计算机可做用快速排序算法可给 $2 \times 10^9$  (20亿) 个数据排序, 用 Dijkstra算法可解 $2.4 \times 10^5$ 个顶点的图的单源最短路径问题, 而用回溯法一天只能解41个顶点的图的最大团问题



# 多项式函数与指数函数

时间复杂度函数	1小时可解的问题实例的最大规模		
	计算机	快100倍的计算机	快1000倍的计算机
$n$	$N_1$	$100 N_1$	$1000 N_1$
$n^2$	$N_2$	$10 N_2$	$31.6 N_2$
$n^3$	$N_3$	$4.64 N_3$	$10 N_3$
$n^5$	$N_4$	$2.5 N_4$	$3.98 N_4$
$2^n$	$N_5$	$N_5 + 6.64$	$N_5 + 9.97$
$3^n$	$N_6$	$N_6 + 4.19$	$N_6 + 6.29$

# 快1000倍的计算机

- ➡ 😐? 求解规模更大的难题
- ➡ 😊! 求解更多规模适中的难题
  - ▶ 或将原问题分解为更多、更精确的规模适中的难题

```

1  #include <assert.h>
2  #include <stdint.h>
3
4  uint32_t inline div_255_fast(uint32_t x) { return (x + ((x + 257) >> 8)) >> 8; }
5
6  void test(uint32_t x) {
7      if (x <= 65536) {
8          uint32_t ret = div_255_fast(x);
9          assert(ret == (x / 255));
10     }
11 }

```

```

> cbmc example.c --function test --unsigned-overflow-check
CBMC version 5.10 (cbmc-5.10) 64-bit x86_64 linux
Parsing example.c
Converting
Type-checking example
Generating GOTO Program
Adding CPROVER library (x86_64)
Removal of function pointers and virtual functions
Generic Property Instrumentation
Running with 8 object bits, 56 offset bits (default)
Starting Bounded Model Checking
size of program expression: 45 steps
simple slicing removed 4 assignments
Generated 3 VCC(s), 3 remaining after simplification
Passing problem to propositional reduction
converting SSA
Running propositional reduction
Post-processing
Solving with MiniSAT 2.2.1 with simplifier
1085 variables, 5069 clauses
SAT checker: instance is UNSATISFIABLE
Runtime decision procedure: 0.0192695s

** Results:
[div_255_fast.overflow.1] arithmetic overflow on unsigned + in x + (unsigned int)257: SUCCESS
[div_255_fast.overflow.2] arithmetic overflow on unsigned + in x + (x + (unsigned int)257 >> 8): SUCCESS
[test.assertion.1] assertion ret == (x / 255): SUCCESS

** 0 of 3 failed (1 iteration)
VERIFICATION SUCCESSFUL

```

# 问题的复杂度分析

多项式时间可解的问题与难解的问题

- 多项式时间可解的问题  $P$

存在着解  $P$  的多项式时间的算法

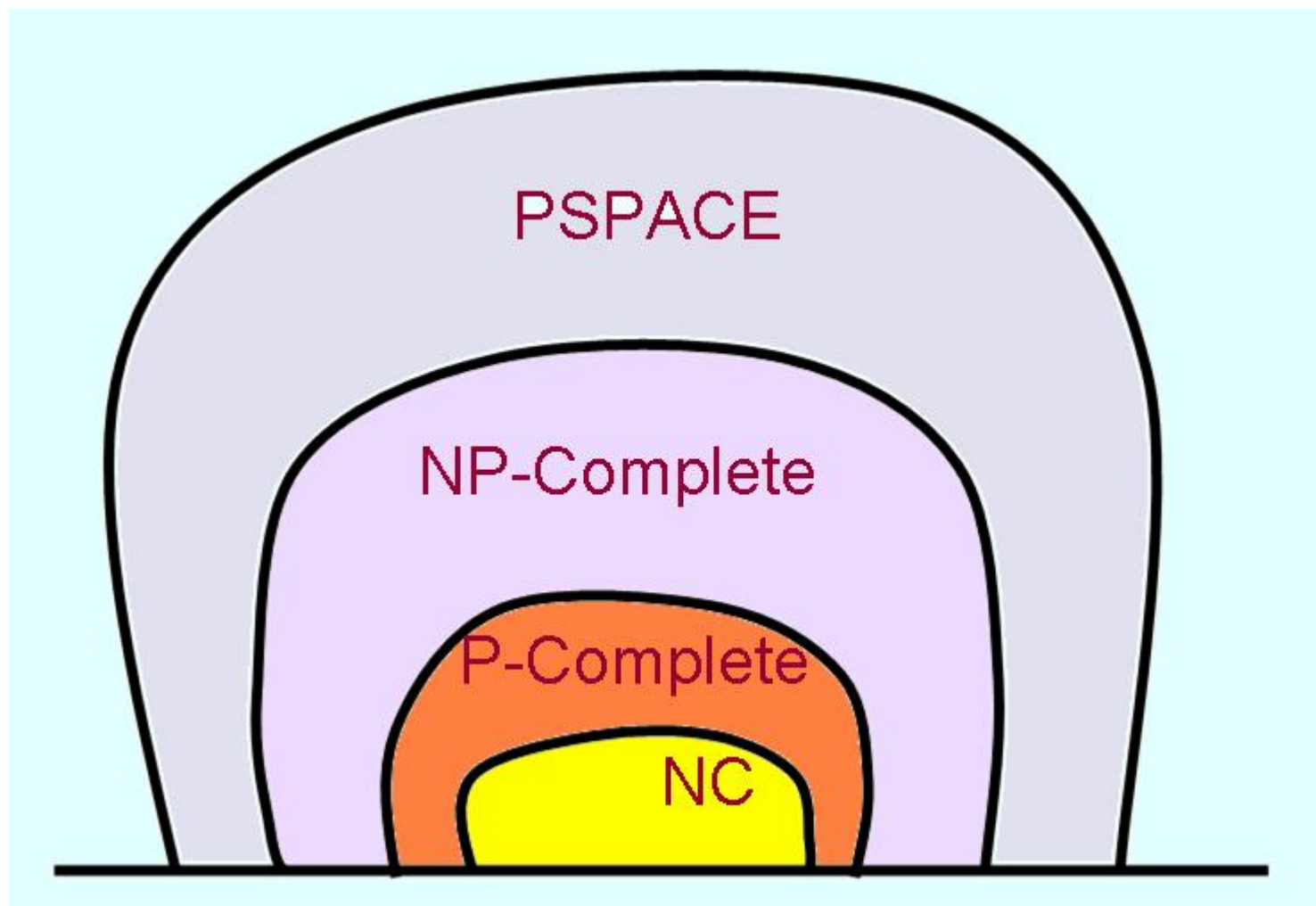
- 难解的问题  $P$

不存在解  $P$  的多项式时间的算法

- 实际上可计算的问题

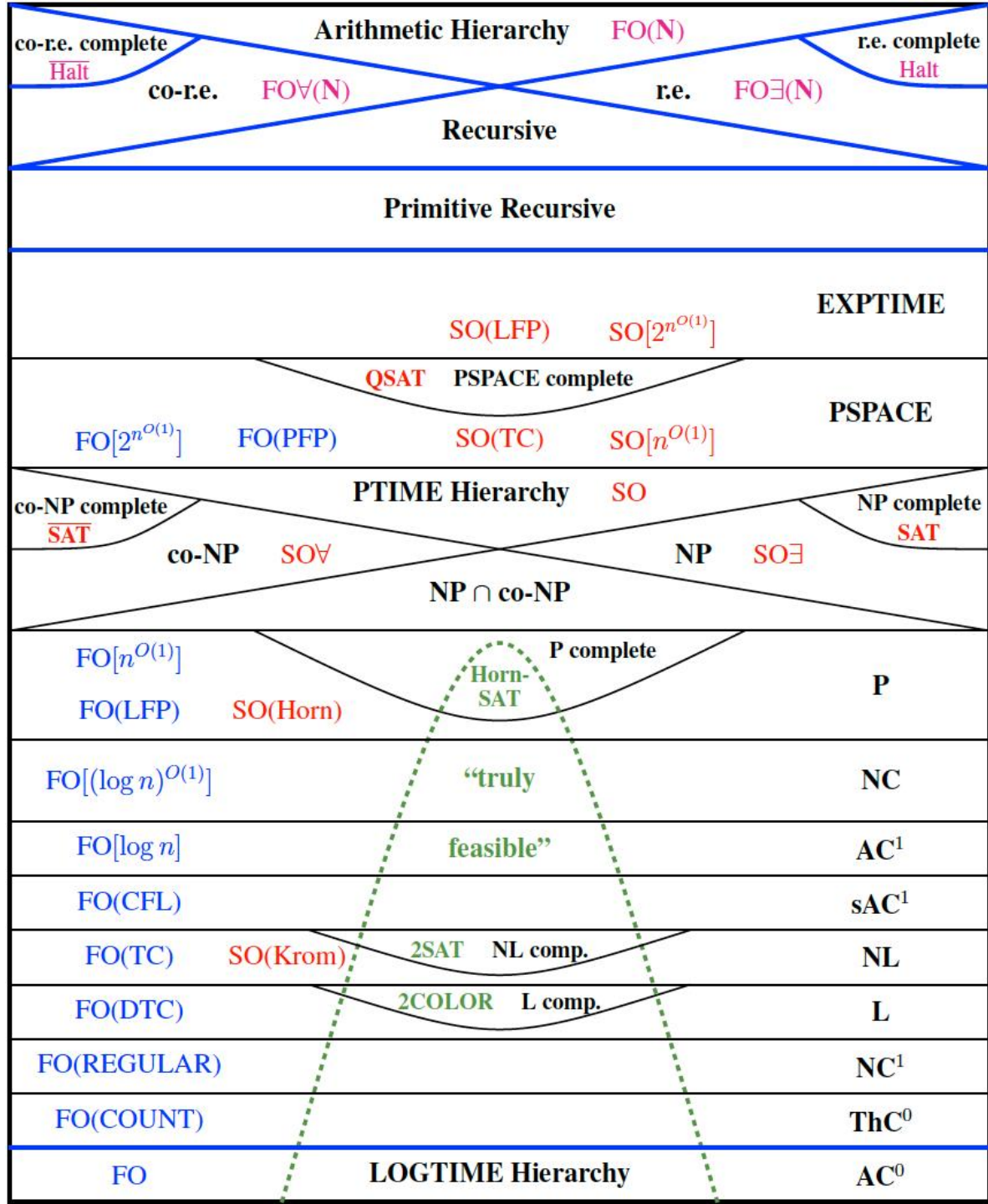
多项式时间可解的问题

## 不同复杂性类的基本层次结构



# 计算复杂性类的谱系

[https://people.cs.umass.edu/~immerman/descriptive\\_complexity.html](https://people.cs.umass.edu/~immerman/descriptive_complexity.html)

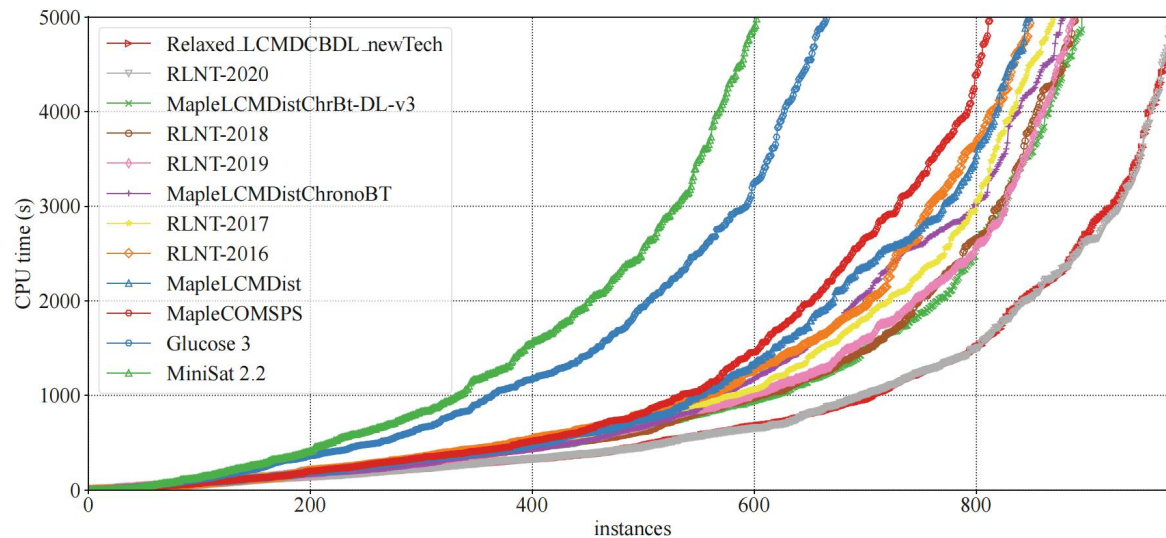


# 部分NPC问题的实例可在合理时间内取得结果

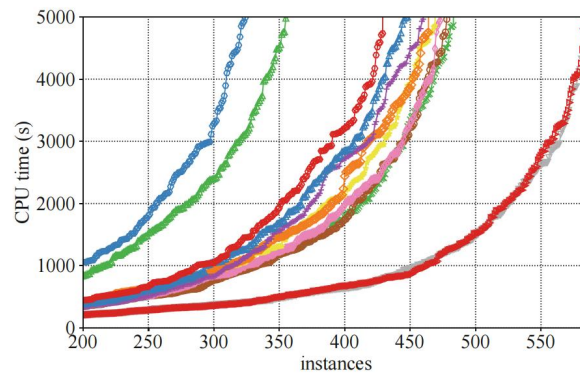
## ► 可满足性问题 SAT

► 某类问题的变量和子句平均数目可高达  $10^7$

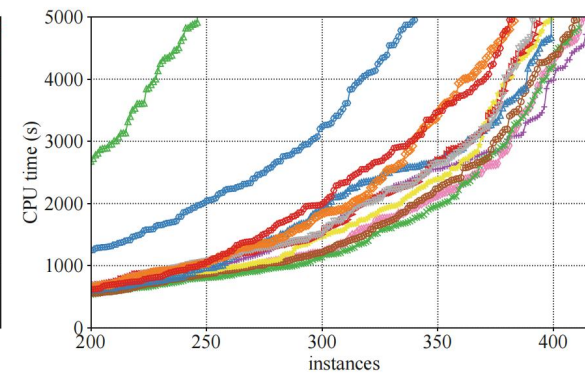
Family	#instances	avg. #vars.	avg. #clauses
2d-strip-packing	5	12753.20	1267404.40
bio	5	46376.40	471256.00
crypto-aes	11	25285.90	82980.81
crypto-des	9	31590.22	95015.66
crypto-gos	30	1892.93	22883.60
crypto-md5	11	66894.00	267579.54
crypto-sha	30	4773.33	151027.86
crypto-vmpc	8	1061.50	171947.00
diagnosis	26	235279.42	1129797.46
hardware-bmc-ibm	4	1335863.00	5234846.00
hardware-bmc	3	114448.33	341918.00
hardware-cec	30	255757.16	754358.03
hardware-velev	21	282973.85	7478281.09
planning	25	469285.60	3323685.24
scheduling-pesp	30	37454.30	270133.10
scheduling	30	160310.13	722325.73
software-bit-verif	14	131492.71	378409.78
software-bmc	3	11216817.00	32697150.00
termination	5	219846.80	950546.60



(a) All benchmarks



(b) Satisfiable benchmarks



(c) Unsatisfiable benchmarks

# 递推方程的求解



# 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

例子:  $a_n = a_{n-1} + a_{n-2}, a_0 = 0, a_1 = 1$

求解目标: 给出  $a_n$  的关于  $n$  的显式公式

$$a_n = \frac{1}{\sqrt{5}} (\phi^n - \hat{\phi}^n)$$
$$\phi = \frac{1 + \sqrt{5}}{2}, \hat{\phi} = \frac{1 - \sqrt{5}}{2}$$



# 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

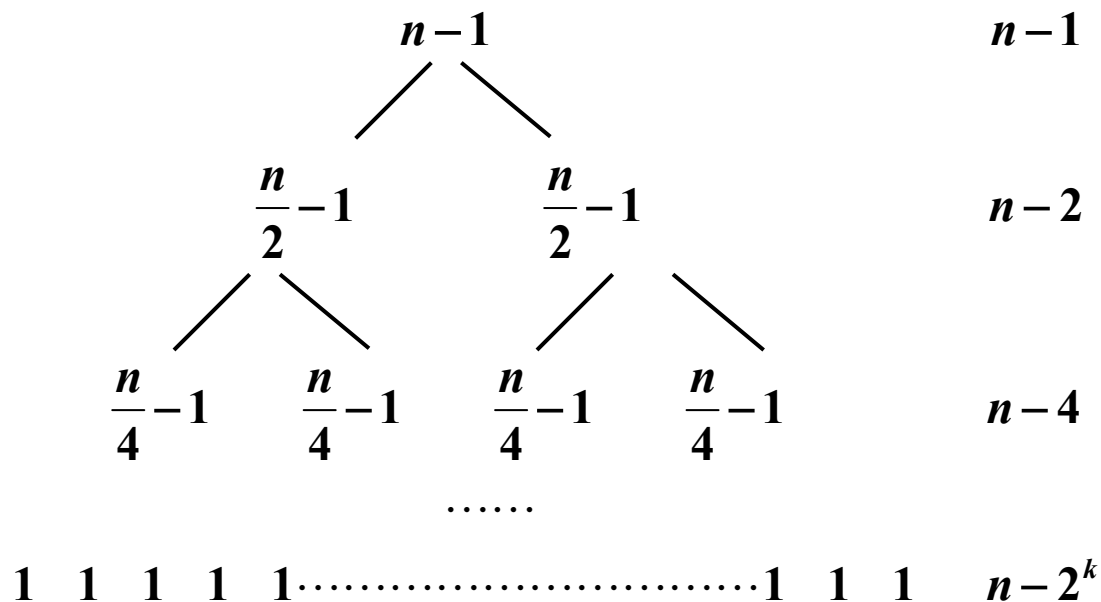
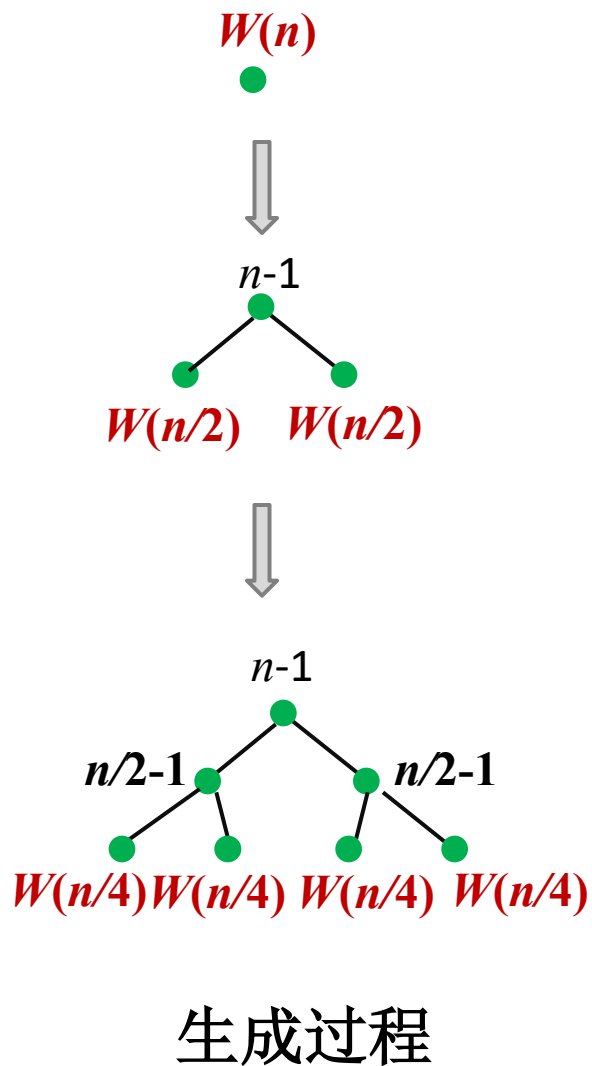
### 迭代模型: 递归树

### 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

### 生成函数

# 迭代模型：递归树



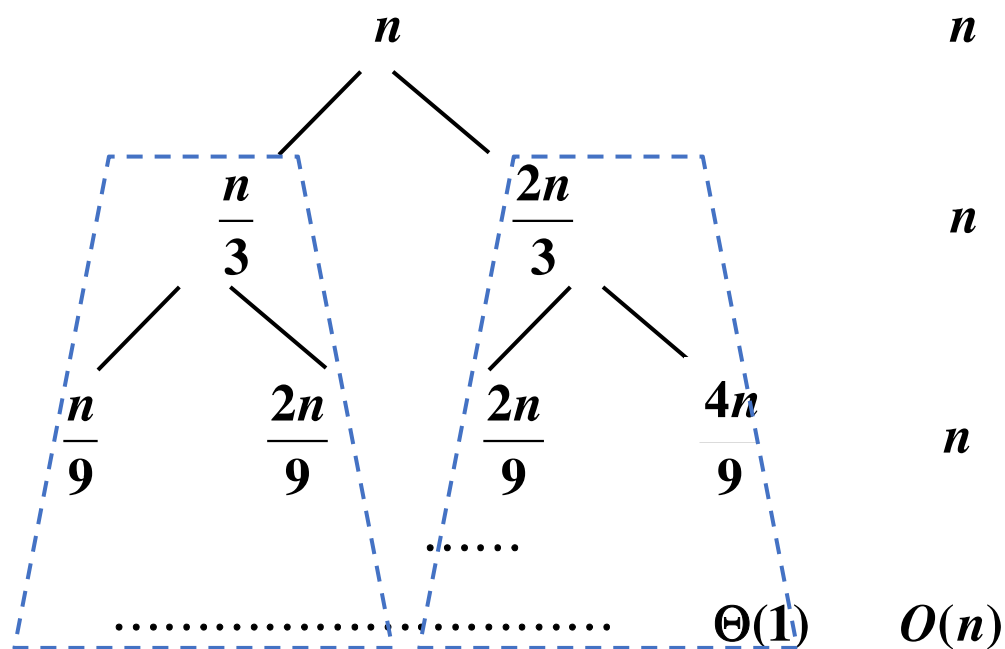
$$W(n) = 2W(n/2) + n - 1, \quad n = 2^k, \quad W(1) = 0$$

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

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

# 递归树的应用实例

求解:  $a_n = a_{n/3} + a_{2n/3} + n$



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

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

# 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= $cn$

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

(3)  $T(n)=cn^2$ , 左边= $cn^2$

$$\text{右边} = \frac{2}{n} \sum_{i=1}^{n-1} ci^2 + O(n) = \frac{2}{n} \left[ \frac{cn^3}{3} + O(n^2) \right] + O(n) = \frac{2c}{3} n^2 + O(n)$$

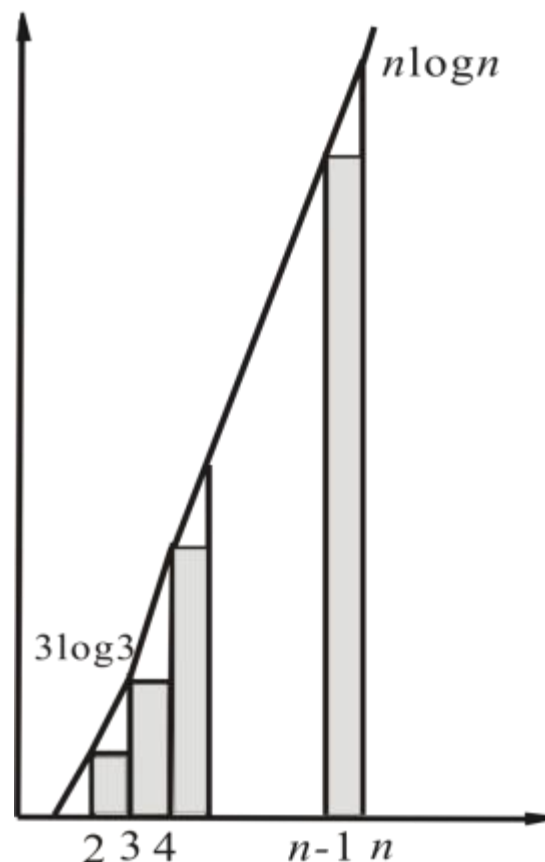
(4)  $T(n)=cn \log n$ , 左边= $cn \log n$

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

# 以积分作为求和的近似

$$\int_1^{n-1} x \log x dx \leq \sum_{i=1}^{n-1} i \log i \leq \int_2^n x \log x dx$$

$$\begin{aligned} \int_2^n x \log x dx &= \int_2^n \frac{x}{\ln 2} \ln x dx \\ &= \frac{1}{\ln 2} \left[ \frac{x^2}{2} \ln x - \frac{x^2}{4} \right] \Big|_2^n \\ &= \frac{1}{\ln 2} \left( \frac{n^2}{2} \ln n - \frac{n^2}{4} \right) - \frac{1}{\ln 2} \left( \frac{4}{2} \ln 2 - \frac{4}{4} \right) \end{aligned}$$



# 递推方程的归纳证明

- ➡ 尝试（猜测）递推方程的解应用归纳法严格证明
- ➡ 归纳法求解递推方程的三个步骤
  - ▶ 猜测解的形式
  - ▶ 用数学归纳法证明
    - 找出使解有效的常数
  - ▶ 确定常数使边界条件成立
- ➡ 常用技巧
  - ▶ 通过引入低阶项获得更紧的解的形式

# 递推方程的归纳证明

例:  $T(n) = 4T(n/2) + n$

- [假定  $T(1) = \Theta(1)$ ]
- 猜测  $T(n) = O(n^3)$  (分别证明  $O$  和  $\Omega$  关系)
- 假设, 对于所有的  $k < n$

$$T(k) \leq ck^3$$

- 通过归纳法证明

$$T(n) \leq cn^3$$

例:  $T(n) = 4T(n/2) + n$

$$T(n) = 4T(n/2) + n$$

$$\leq 4c(n/2)^3 + n$$

$$= (c/2)n^3 + n$$

$$= cn^3 - ((c/2)n^3 - n) \quad \longleftarrow \text{期望的形式} - \text{余项}$$

$$\leq cn^3 \quad \longleftarrow \text{期望的形式}$$

这里要保证:  $((c/2)n^3 - n) \geq 0$ ,

这只需要:  $c \geq 2$  并且  $n \geq 1$

于是有  $T(n) = O(n^3)$

余项



例：  $T(n) = 4T(n/2) + n$

- 还必须处理初始情形，才能使归纳成立。
- 注意到，因为对所有的  $1 \leq n < n_0$  都有  $T(n) = \Theta(1)$ （其中  $n_0$  是某个适当的常数）
- 于是当  $1 \leq n < n_0$  时，只要  $c$  足够大，就有

$$\text{“}\Theta(1)\text{”} \leq cn^3$$

- 但  $T(n) = O(n^3)$  这个界并不够紧

## 更紧的上界

- 我们来证明  $T(n) = O(n^2)$
- 假设对于所有的  $k < n$ , 有  $T(k) \leq ck^2$

$$T(n) = 4T(n/2) + n$$

$$\leq 4c(n/2)^2 + n = cn^2 + n$$

$$= O(n^2)$$

 错! 必须证明完全一致的形式



## 更紧的上界

- 我们来证明  $T(n) = O(n^2)$
- 假设对于所有的  $k < n$ , 有  $T(k) \leq ck^2$

$$T(n) = 4T(n/2) + n$$

$$\leq 4c(n/2)^2 + n = cn^2 + n$$

$$= cn^2 - (-n) \quad \longleftarrow \text{期望的形式} - \text{余项}$$

$$\leq cn^2$$

但对任何  $c > 0$ , 上式最后一步不可能成立!

## 更紧的上界

要点：加强归纳假设

\*减去一个低阶项

假设：对于  $k < n$ , 有  $T(k) \leq c_1 k^2 - c_2 k$

$$T(n) = 4T(n/2) + n$$

$$\leq 4(c_1(n/2)^2 - c_2(n/2)) + n$$

$$= c_1 n^2 - 2c_2 n + n$$

$$= c_1 n^2 - c_2 n - (c_2 n - n)$$

$$\leq c_1 n^2 - c_2 n \quad \text{当 } c_2 > 1$$

可以取  $c_1$  足够大来处理初始情况。

例：  $T(n) = 4T(n/2) + n = \Omega(n^2)$

► 再证明：  $T(n) = \Omega(n^2)$

► 假设对于  $k < n$ , 有  $T(k) \geq ck^2$

$$T(n) = 4T(n/2) + n$$

$$\geq 4c(n/2)^2 + n$$

$$= cn^2 + n$$

► 取  $c$  足够小来处理初始情况。

►  $T(n) = O(n^2)$  且  $T(n) = \Omega(n^2)$  得  $T(n) = \Theta(n^2)$

# 换元法的求解递推方程

- 例:  $T(n)=2T(\sqrt{n})+\log n$
- 通过改变变量转化递归式, 将  $\sqrt{n}$  转化为整数。  
令  $m = \log n$ , 于是

$$T(2^m) = 2T(2^{m/2}) + m$$

- 再令  $S(m) = T(2^m)$ , 于是

$$S(m) = 2S(m/2) + m$$

$$= \Theta(m \log m)$$

$$T(n) = T(2^m) = S(m) = \Theta(m \log m)$$

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

# 主定理 (Master Theorem)

- Bentley, Haken, Saxe, “A general method for solving divide-and-conquer recurrences,” *ACM SIGACT News*, 12 (3): 36–44, 1980.
- Popularized by the CLRS Textbook



# 主定理

**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 $f(n) = \Omega(n^{\log_b a + \varepsilon})$ ,  $\varepsilon > 0$ ,  
且对某个常数  $c < 1$  和 所有充分大的  $n$  有  
 $a f(n/b) \leq c f(n)$ , 那么  $T(n) = \Theta(f(n))$



# 主定理的应用

**例9** 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

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

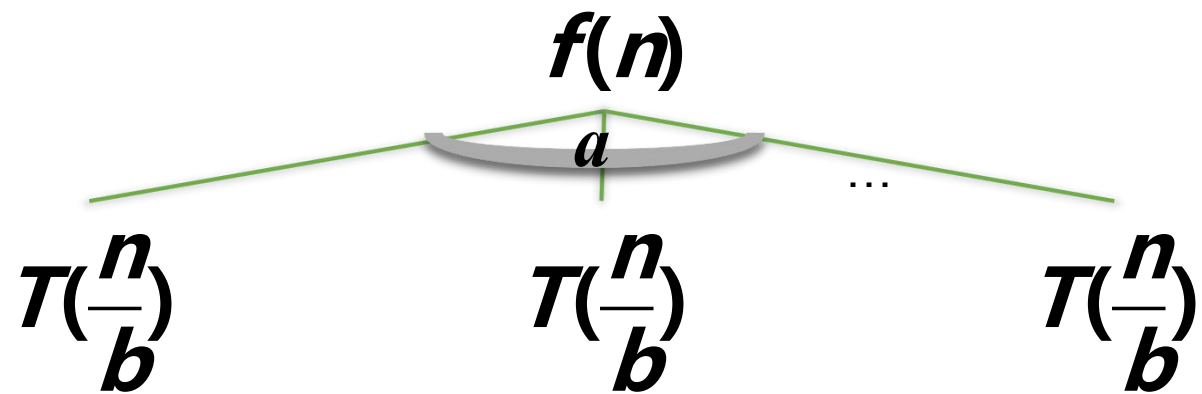
**例10** 求解递推方程  $T(n) = T(2n/3) + 1$

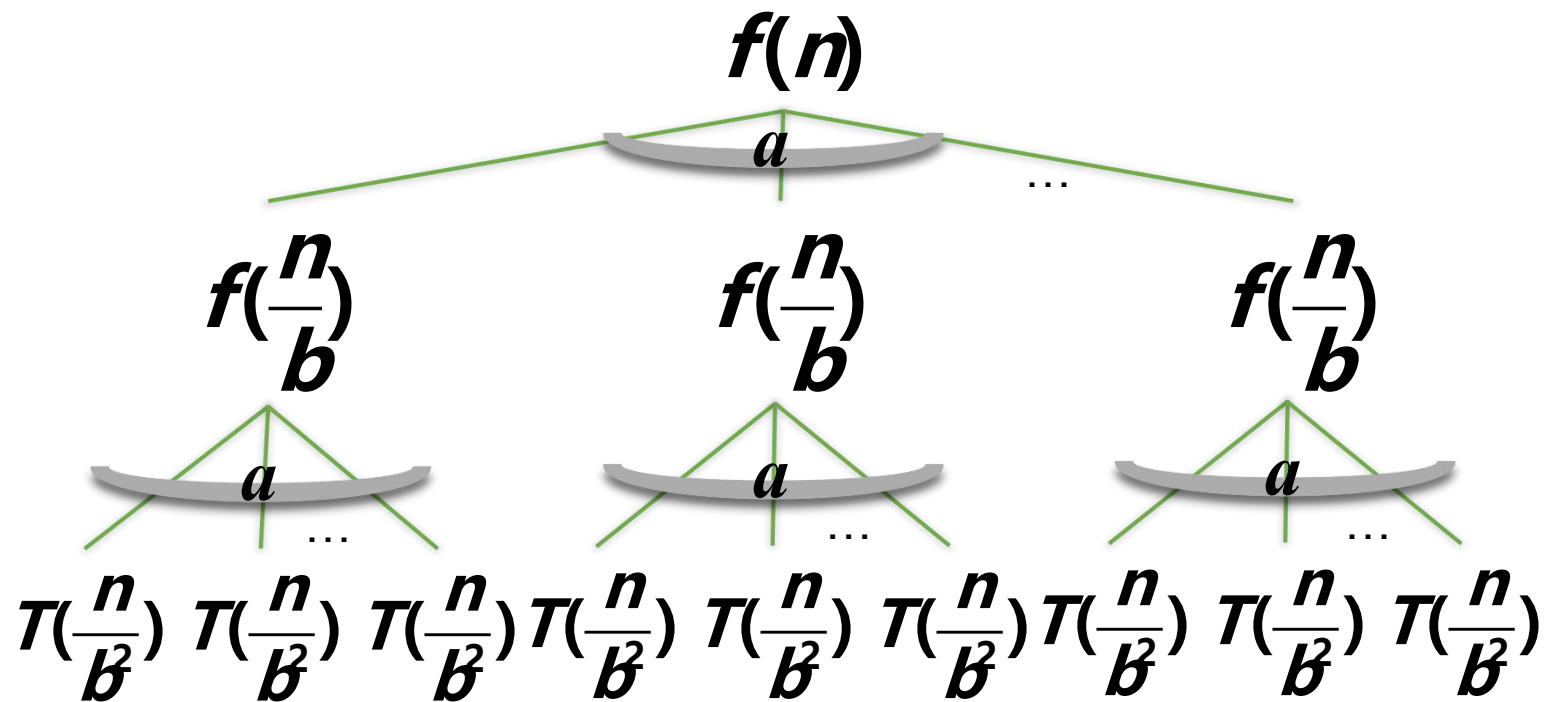
解 上述递推方程中的  $a=1, b=3/2, f(n)=1$ , 那么

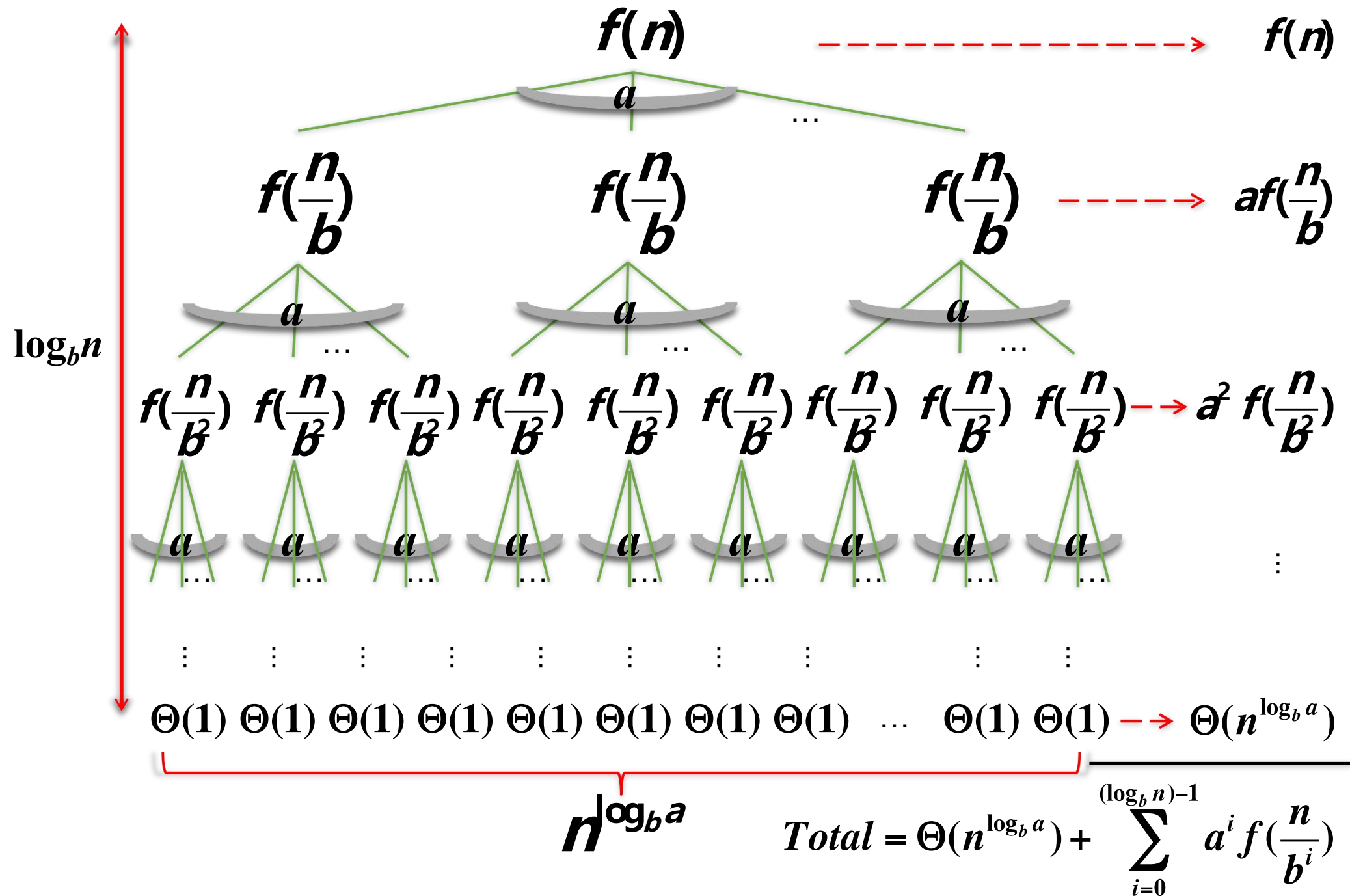
$$n^{\log_{3/2} 1} = n^0 = 1, \quad f(n) = 1$$

相当于主定理的第二种情况. 根据定理得到.

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







# 主定理的证明

不妨设  $n=b^k$

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

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

= ...

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

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

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

# 情况1

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

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

## 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

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

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

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

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

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

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



## 应用实例

### 例11 求解递推方程

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

解 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.

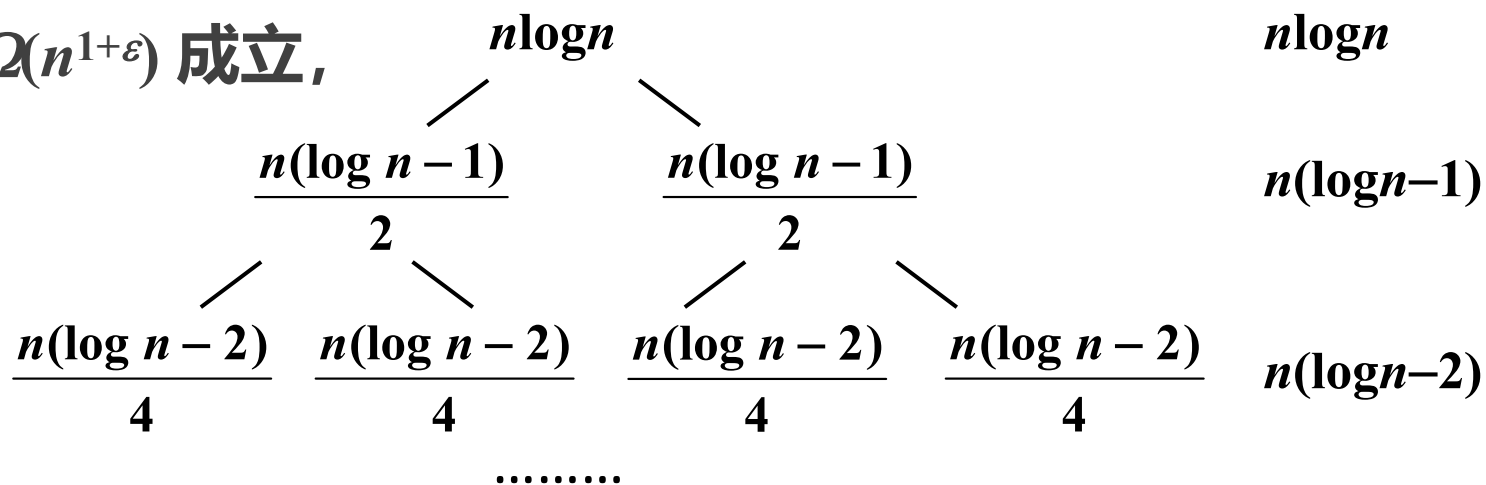
相当于主定理的第三种情况. 因此有  $T(n) = \Theta(f(n)) = \Theta(n \log n)$

# 不能直接使用主定理的例子

**例12** 求解  $T(n) = 2T(n/2) + n \log n$

$$a=b=2, \quad n^{\log_2 2} = n, \quad f(n)=n \log n$$

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



$$\begin{aligned}
 T(n) &= n \log n + n(\log n - 1) + n(\log n - 2) + \dots + n(\log n - k + 1) \\
 &= (n \log n) \log n - n(1 + 2 + \dots + k - 1) \\
 &= n \log^2 n - nk(k-1)/2 = O(n \log^2 n)
 \end{aligned}$$

# 生成函数求解递推方程

# 生成函数

## ► 生成函数 / 形式幂级数

► 带形式变量（无需考虑级数的收敛性）的幂级数，以幂级数系数表示递推序列

## ► 定义 给的序列 $a_0, a_1, a_2, a_3, \dots, a_n, \dots$ （即 $\{a_n\}$ ），称函数

$$A(z) = \sum_{k \geq 0} a_k z^k$$

为该序列的**常规生成函数**（OGF），  
用记号  $[z^k]A(z)$  表示系数  $a_k$ 。

## ► 生成函数求解递推方程的基本思想

► 表示：序列  $\sim$  生成函数系数

► 求解：递推方程  $\Rightarrow$  数列相邻项方程  $\Rightarrow$  生成函数方程  $\Rightarrow$  生成函数  $\sim$  序列

# 表1：基本的常规生成函数

$$1, 1, 1, \dots, 1, \dots$$

$$\sum_{N \geq 0} 1 \cdot z^N = \frac{1}{1 - z}$$

$$0, 1, 2, 3, 4, \dots, N, \dots$$

$$\sum_{N \geq 1} N \cdot z^N = \frac{z}{(1 - z)^2}$$

$$0, 0, 1, 3, 6, 10, \dots, \binom{N}{2}, \dots$$

$$\sum_{N \geq 2} \binom{N}{2} \cdot z^N = \frac{z^2}{(1 - z)^3}$$

$$0, \dots, 0, 1, M + 1, \dots, \binom{N}{M}, \dots$$

$$\sum_{N \geq M} \binom{N}{M} \cdot z^N = \frac{z^M}{(1 - z)^{M+1}}$$

$$1, M, \binom{M}{2}, \dots, \binom{M}{N}, \dots, M, 1$$

$$\sum_{N \geq 0} \binom{M}{N} \cdot z^N = (1 + z)^M$$

$$1, M + 1, \binom{M + 2}{2}, \binom{M + 3}{3}, \dots$$

$$\sum_{N \geq 0} \binom{N + M}{N} \cdot z^N = \frac{1}{(1 - z)^{M+1}}$$

# 表1：基本的常规生成函数（续）

$$1, 0, 1, 0, \dots, 1, 0, \dots$$

$$\sum_{N \geq 0} z^{2N} = \frac{1}{1 - z^2}$$

$$1, c, c^2, c^3, \dots, c^N, \dots$$

$$\sum_{N \geq 0} c^N \cdot z^N = \frac{1}{1 - cz}$$

$$1, 1, \frac{1}{2!}, \frac{1}{3!}, \dots, \frac{1}{N!}, \dots$$

$$\sum_{N \geq 0} \frac{1}{N!} \cdot z^N = e^z$$

$$0, 1, \frac{1}{2}, \frac{1}{3}, \frac{1}{4}, \dots, \frac{1}{N}, \dots$$

$$\sum_{N \geq 1} \frac{1}{N} \cdot z^N = \ln \frac{1}{1 - z}$$

$$0, 1, 1 + \frac{1}{2}, 1 + \frac{1}{2} + \frac{1}{3}, \dots, H_N, \dots$$

$$\sum_{N \geq 1} H_N \cdot z^N = \frac{1}{1 - z} \ln \frac{1}{1 - z}$$

$$0, 0, 1, 2 \left( \frac{1}{2} + \frac{1}{3} \right), \dots, N(H_N - 1), \dots$$

$$\sum_{N \geq 0} N(H_N - 1) \cdot z^N = \frac{z}{(1 - z)^2} \ln \frac{1}{1 - z}$$

## 表2：关于常规生成函数的运算

	$A(z) = \sum_{n \geq 0} a_n z^n$	$a_0, a_1, a_2, \dots, a_n, \dots$
	$B(z) = \sum_{n \geq 0} b_n z^n$	$b_0, b_1, b_2, \dots, b_n, \dots$
右移	$zA(z) = \sum_{n \geq 1} a_{n-1} z^n$	$0, a_0, a_1, a_2, \dots, a_{n-1}, \dots$
左移	$\frac{A(z) - a_0}{z} = \sum_{n \geq 0} a_{n+1} z^n$	$a_1, a_2, a_3, \dots, a_{n+1}, \dots$
下标乘 (微分)	$A'(z) = \sum_{n \geq 0} (n+1) a_{n+1} z^n$	$a_1, 2a_2, 3a_3, \dots, (n+1)a_{n+1}, \dots$
下标除 (积分)	$\int_0^z A(t) dt = \sum_{n \geq 1} \frac{a_{n-1}}{n} z^n$	$0, a_0, \frac{a_1}{2}, \frac{a_2}{3}, \dots, \frac{a_{n-1}}{n}, \dots$

# 表2：关于常规生成函数的运算（续）

比例  
因子

$$A(\lambda z) = \sum_{n \geq 0} \lambda^n a_n z^n$$

$$a_0, \lambda a_1, \lambda^2 a_2, \dots, \lambda^n a_n, \dots$$

相加

$$A(z) + B(z) = \sum_{n \geq 0} (a_n + b_n) z^n$$

$$a_0 + b_0, a_1 + b_1, \dots, a_n + b_n, \dots$$

差分

$$(1 - z)A(z) = a_0 + \sum_{n \geq 1} (a_n - a_{n-1}) z^n$$

$$a_0, a_1 - a_0, \dots, a_n - a_{n-1}, \dots$$

卷积

$$A(z)B(z) = \sum_{n \geq 0} \left( \sum_{0 \leq k \leq n} a_k b_{n-k} \right) z^n$$

$$a_0 + b_0, a_1 b_0 + a_0 b_1, \dots, \sum_{0 \leq k \leq n} a_k b_{n-k}, \dots$$

部分和

$$\frac{A(z)}{1 - z} = \sum_{n \geq 0} \left( \sum_{0 \leq k \leq n} a_k \right) z^n$$

$$a_0, a_0 + a_1, \dots, \sum_{0 \leq k \leq n} a_k, \dots$$



# 例子：若干数列的生成函数

$$\frac{1}{(1-3z)^4}$$

$$a_n = \binom{n+3}{n} 3^n$$

$$(1-z^2) \ln \frac{1}{1-z}$$

$$0, 1, \frac{1}{2}, \dots, \left( \frac{1}{N} - \frac{1}{N-2} \right)$$

$$\frac{1}{(1-2z^2)^2}$$

$$\left( \frac{1}{2} + \frac{(-1)^n}{2} \right) \left( \frac{n}{2} + 1 \right) \sqrt{2}^n$$

# 指数生成函数

► **定义** 给的序列  $a_0, a_1, a_2, a_3, \dots, a_n, \dots$  (即  $\{a_n\}$ ) , 称函数

$$A(z) = \sum_{k \geq 0} a_k \frac{z^k}{k!}$$

► 为该序列的**指数生成函数** (EGF) ,  
用记号  $k![z^k]A(z)$  表示系数  $a_k$ 。

$$1, 1, 1, \dots, 1, \dots$$

$$0, 1, 2, 3, 4, \dots, N, \dots$$

$$0, 0, 1, 3, 6, 10, \dots, \binom{N}{2}, \dots$$

$$0, \dots, 0, 1, M + 1, \dots, \binom{N}{M}, \dots$$

$$1, 0, 1, 0, \dots, 1, 0, \dots$$

$$1, c, c^2, c^3, \dots, c^N, \dots$$

$$1, \frac{1}{2}, \frac{1}{3}, \dots, \frac{1}{N+1}, \dots$$

$$1, 2, 6, 24, \dots, N!, \dots$$

$$e^z = \sum_{N \geq 0} \frac{z^N}{N!}$$

$$ze^z = \sum_{N \geq 1} \frac{z^N}{(N-1)!}$$

$$\frac{1}{2}z^2e^z = \frac{1}{2} \sum_{N \geq 2} \frac{z^N}{(N-2)!}$$

$$\frac{1}{M!}z^Me^z = \frac{1}{M!} \sum_{N \geq M} \frac{z^N}{(N-M)!}$$

$$\frac{1}{2}(e^z + e^{-z}) = \sum_{N \geq 0} \frac{1 + (-1)^N}{2} \frac{z^N}{N!}$$

$$e^{cz} = \sum_{N \geq 0} \frac{c^N z^N}{N!}$$

$$\frac{e^z - 1}{z} = \sum_{N \geq 1} \frac{z^{N-1}}{(N-1)!}$$

$$\frac{1}{1-z} = \sum_{N \geq 0} \frac{z^N}{N!}$$

# 基本的指数生成函数

# 表4 关于指数生成函数的运算

	$A(z) = \sum_{n \geq 0} a_n \frac{z^n}{n!}$	$a_0, a_1, a_2, \dots, a_n, \dots$
	$B(z) = \sum_{n \geq 0} b_n \frac{z^n}{n!}$	$b_0, b_1, b_2, \dots, b_n, \dots$
右移 (积分)	$\int_0^z A(t) dt = \sum_{n \geq 1} a_{n-1} \frac{z^n}{n!}$	$0, a_0, a_1, a_2, \dots, a_{n-1}, \dots$
左移 (微分)	$A'(z) = \sum_{n \geq 0} a_{n+1} \frac{z^n}{n!}$	$a_1, a_2, a_3, \dots, a_{n+1}, \dots$
下标乘	$zA(z) = \sum_{n \geq 0} na_{n-1} \frac{z^n}{n!}$	$0, a_0, 2a_1, 3a_2, \dots, na_{n-1}, \dots$
下标除	$\frac{A(z) - A(0)}{z} = \sum_{n \geq 1} \frac{a_{n+1}}{n+1} \frac{z^n}{n!}$	$a_1, \frac{a_2}{2}, \frac{a_3}{3}, \dots, \frac{a_{n+1}}{n+1}, \dots$

# 表4 关于指数生成函数的运算 (续)

相加

$$A(z) + B(z) = \sum_{n \geq 0} (a_n + b_n) \frac{z^n}{n!}$$

$$a_0 + b_0, \dots, a_n + b_n, \dots$$

差分

$$A'(z) - A(z) = \sum_{n \geq 0} (a_{n+1} - a_n) \frac{z^n}{n!}$$

$$a_1 - a_0, \dots, a_{n+1} - a_n, \dots$$

卷积

$$A(z)B(z) = \sum_{n \geq 0} \left( \sum_{0 \leq k \leq n} \binom{n}{k} a_k b_{n-k} \right) \frac{z^n}{n!}$$

$$a_0 + b_0, a_1 b_0 + a_0 b_1, \dots, \sum_{0 \leq k \leq n} \binom{n}{k} a_k b_{n-k}, \dots$$

部分和

$$e^z A(z) = \sum_{n \geq 0} \left( \sum_{0 \leq k \leq n} \binom{n}{k} a_k \right) \frac{z^n}{n!}$$

$$a_0, a_0 + a_1, \dots, \sum_{0 \leq k \leq n} \binom{n}{k} a_k, \dots$$

# 生成函数求解递推方程

- 利用常规生成函数求解递推方程的机械步骤
  - ▶ 在递推方程的两边乘以  $z^n$ ，然后关于  $n$  求和。
  - ▶ 处理所得的各个和，导出一个关于 **OGF** 的函数方程。
  - ▶ 解这个方程，导出 **OGF** 的显示公式。
  - ▶ 将 **OGF** 展开为一个幂级数，从而得到系数表达式。
  - ▶ 这些系数就是原序列中的元素
- 同样的步骤也适用于指数生成函数
  - ▶ 只是递推方程两边乘以  $z^n/n!$ ，然后在关于  $n$  求和。

## 例题：平凡线性递推

► 解递推方程：  $a_n = a_{n-1} + 1, \quad a_0 = 0$

► 两边乘以  $z^n$ ，然后关于  $n$  求和

$$\sum_{n \geq 1} a_n z^n = \sum_{n \geq 1} a_{n-1} z^n + \frac{z}{1-z}$$

► 由生成函数：  $A(z) = \sum_{n \geq 0} a_n z^n$

► 求得方程

$$A(z) = zA(z) + \frac{z}{1-z}$$

► 于是

$$A(z) = \frac{z}{(1-z)^2} \Rightarrow a_n = n$$

## 例题：简单指数型递推 (1/2)

► 解递推方程：  $a_n = 2a_{n-1} + 1, \quad a_0 = 1$

► 两边乘以  $z^n$ ，然后关于  $n$  求和

$$\sum_{n \geq 1} a_n z^n = 2z \sum_{n \geq 1} a_{n-1} z^{n-1} + \frac{z}{1-z}$$

► 由生成函数：  $A(z) = \sum_{n \geq 0} a_n z^n$

► 求得方程

$$\begin{aligned} A(z) - 1 &= 2zA(z) + \frac{z}{1-z} \\ A(z) &= \frac{1}{(1-z)(1-2z)} \end{aligned}$$



## 例题：简单指数型递推 (2/2)

► 应用分式和

$$A(z) = \frac{1}{(1-z)(1-2z)} = \frac{2}{1-2z} - \frac{1}{1-z}$$

► 求和展开

$$[z^n]A(z) = [z^n] \left( \frac{2}{1-2z} - \frac{1}{1-z} \right) = 2^{n+1} - 1$$

## 例题：斐波那契数列

► 解递推方程：  $F_n = F_{n-1} + F_{n-2}, F_0 = 0, F_1 = 1$

► 由生成函数：  $F(z) = \sum_{n \geq 0} F_n z^n$

► 满足

$$F(z) = zF(z) + z^2F(z) + z$$

► 导出

$$F(z) = \frac{z}{1 - z - z^2} = \frac{1}{\sqrt{5}} \left( \frac{1}{1 - \phi z} - \frac{1}{1 - \hat{\phi} z} \right)$$

► 两项对应的数列相加/减

$$F_n = \frac{1}{\sqrt{5}} (\phi^n - \hat{\phi}^n)$$

## 例题：高阶线性递推

► 解递归方程

$$a_n = 5a_{n-1} + 6a_{n-2}, a_0 = 0, a_1 = 1$$

► 由生成函数

$$A(z) = \sum_{n \geq 0} a_n z^n$$

► 满足

$$A(z) - z = 5zA(z) - 6z^2A(z)$$

► 导出

$$A(z) = \frac{z}{1 - 5z + 6z^2} = \frac{1}{1 - 3z} - \frac{1}{1 - 2z}$$

► 于是

$$a_n = (3^n - 2^n)$$

# 定理 求解高阶递推方程

- 对于高阶方程:

$$a_n = x_1 a_{n-1} + x_2 a_{n-2} + \dots + x_t a_{n-t}$$

- 其生成函数

$$a(z) = \sum_{n \geq 0} a_n z^n$$

- 为有理函数

$$a(z) = \frac{f(z)}{g(z)}$$

- 其中

$$g(z) = 1 - x_1 z - x_2 z^2 - \dots - x_t z^t$$

- 而 $f(z)$ 由数列初值

$$a_0, a_1, \dots, a_t$$

决定, 且次数小于 $t$ 。

# 证明

$$\begin{aligned} \sum_{n \geq t} a_n z^n &= x_1 \sum_{n \geq t} a_{n-1} z^n + x_2 \sum_{n \geq t} a_{n-2} z^n + \dots + x_t \sum_{n \geq t} a_{n-t} z^n \\ a(z) - u_0(z) &= (x_1 z a(z) - u_1(z)) + (x_2 z^2 a(z) - u_2(z)) + \dots + (x_t z^t a(z) - u_t(z)) \end{aligned}$$

其中  $u_i(z)$ ,  $0 \leq i \leq t$  至多是  $t-1$  次的。

设

$$f(z) \equiv u_0(z) - u_1(z) - u_2(z) - \dots - u_t(z)$$

则

$$a(z) = \frac{u_0(z) - u_1(z) - u_2(z) - \dots - u_t(z)}{1 - x_1 z - x_2 z^2 - \dots - x_t z^t} = \frac{f(z)}{g(z)}$$



# 例题

由 $f(z)$ 次数小于 $t$ , 并且 $f(z) = a(t)g(t)$ , 则必然有

$$f(z) = g(t) \sum_{0 \leq n < t} a_n z^n \pmod{z^t}$$

例题

$$a_n = 2a_{n-1} + a_{n-2} - 2a_{n-3} \quad (n > 2, a_0 = 0, a_1 = a_2 = 1)$$

$$a_n = 2a_{n-1} + a_{n-2} - 2a_{n-3} \quad (n > 2, a_0 = a_1 = a_2 = 1)$$

$$a_n = 2a_{n-1} - a_{n-2} + 2a_{n-3} \quad (n > 2, a_0 = 1, a_1 = 0, a_2 = -1)$$

$$a_n = 2a_{n-1} - a_{n-2} + 2a_{n-3} \quad (n > 2, a_0 = 0, a_1 = 1, a_2 = 4)$$

# 求解高阶递推方程的一般方法

- 由递推方程导出  $g(z)$
- 由  $g(z)$  和初始条件计算  $f(z)$
- 消去  $f(z)/g(z)$  中的公共因子
- 用部分分式将  $f(z)/g(z)$  表示为形如  $(1-\beta z)^{-j}$  的项的线性组合
- 将每个部分分式按以下公式展开

$$[z^n](1 - \beta z)^{-j} = \binom{n + j - 1}{j - 1} \beta^n$$

## 例题：快速排序的递推求解（1/2）

$$C_n = n + 1 + \frac{2}{n} \sum_{1 \leq k \leq n} C_{k-1}, n \geq 1, C_0 = 0$$

$$nC_n = (n+1)n + 2 \sum_{1 \leq k \leq n} C_{k-1}$$

定义生成函数：

$$C(z) = \sum_{n \geq 0} C_n z^n$$

关于  $n$  求和：

$$\sum_{n \geq 1} nC_n z^n = \sum_{n \geq 1} (n+1)n z^n + 2 \sum_{n \geq 1} \sum_{1 \leq k \leq n} C_{k-1} z^n$$



## 例题：快速排序的递推求解 (2/2)

由于

$$\begin{aligned}\sum_{n \geq 1} n C_n z^n &= z \sum_{n \geq 1} n C_n z^{n-1} = z \sum_{n \geq 0} (n+1) C_{n+1} z^n = z C'(z) \\ \sum_{n \geq 1} (n+1) n z^n &= z \sum_{n \geq 0} (n+2)(n+1) z^n = \frac{2z}{(1-z)^3} \\ \sum_{n \geq 1} \sum_{1 \leq k \leq n} C_{k-1} z^n &= z \sum_{n \geq 0} \sum_{0 \leq k \leq n} C_k z^n = \frac{z C(z)}{1-z}\end{aligned}$$

可得生成函数的微分方程：

$$C'(z) = \frac{2}{(1-z)^3} + \frac{2C(z)}{1-z}$$

# 递推方程与生成函数简史

- 四大文明古国的数学家们研究数列问题
- 1202年 Leonardo Bonacci (a.k.a., Fibonacci / 斐波那契) 研究 Fibonacci 数列
  - ▶ Leonardo Bonacci. “Liber Abaci,” 1202.
- 1722年 De Moivre (棣莫弗) 提出用生成函数方法研究 Fibonacci 数列
  - ▶ Abraham de Moivre. “De fractionibus algebraicis radicalitate immunibus ad fractiones simpliciores reducendis, deque summandis terminis quarundam serierum aequali intervallo a se distantibus.” Philosophical Transactions, 32(373):162–168, 1722.
- 生成函数的扩展阅读材料 (除用于解递推方程, 也广泛用于组合计数)
  - ▶ Sec. 7.3 “Solving Recurrence” in Graham, Knuth, Patashnik, “Concrete Mathematics (2nd ed.),” 2013
  - ▶ Donald E. Knuth. Fundamental Algorithms, volume 1 of The Art of Computer Programming. Addison-Wesley, third edition, 1997.
  - ▶ C. L. Liu. Introduction to Combinatorial Mathematics. McGraw-Hill, 1968.

# 内容小结

- 函数的渐近的界
- 常见函数的阶
- 递推方程求解
  - ▶ 递归树
  - ▶ 主定理证明及应用
  - ▶ 生成函数

## 教材和扩展阅读

- 1.3 算法的数学基础
  - ▶ 1.3.1 函数的渐近的界
  - ▶ 1.3.2 求和的方法
  - ▶ 1.3.3 递推方程求解方法
- 生成函数
  - ▶ <https://aofa.cs.princeton.edu/30gf/>