

## Chapter 2. 函数增长率 Growth of functions

### Topics:

- Growth of functions
- $O/o/\Theta/\Omega/\omega$  notations

### Asymptotic Growth

- In the insertion-sort example we discussed that when analyzing algorithms we are
  - ✓ interested in worst-case running time as function of input size  $n$ .
  - ✓ not interested in exact constants in bound.
  - ✓ not interested in lower order terms (低阶项)
- A good reason for not caring about constants and lower order terms is that the RAM model is not completely realistic anyway.

### Machine-independent time

- What is insertion sort's worst-case time?
  - It depends on the speed of our computer:
    - >> relative speed (on the same machine)
    - >> absolute speed (on different machine)
- BIG IDEA:
  - Ignore machine-dependent constants.
  - Look at growth of  $T(n)$  as  $n \rightarrow \infty$

“Asymptotic Analysis”

### § 2.1 渐近表示法

- 算法的渐近时间定义为一个函数，定义域为自然数集合  $N=\{0,1,2,\dots\}$  ( $n$  表示 Size)。但有时也将其扩展到实数或限制到自然数的某子集上。(P25)

#### a. $\theta$ 记号 ( $\theta$ -notation)

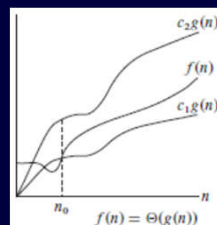
- ❖ Def: 给定一个函数  $g(n)$ ,  $\theta(g(n))$  表示一个函数的集合:

$$\theta(g(n)) = \{f(n) \mid \exists \text{ 常数 } c_1, c_2, n_0 > 0, \text{ 使得对所有的 } n \geq n_0 \text{ 有 } 0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n)\} / n \text{ 为 size}$$

- ❖ 即  $f(n) \in \theta(g(n))$  表示存在正常数  $c_1, c_2$  及足够大的  $n$ , 使得  $f(n)$  夹在  $c_1 g(n)$  和  $c_2 g(n)$  之间
- ❖ 通常  $f(n) \in \theta(g(n))$  表示为  $f(n) = \theta(g(n))$ , 其实  $f(n)$  是  $\theta(g(n))$  的成员 (可以理解为簇中的一员)

- 大  $\Theta$  的数学定义(渐进紧致界) 与上一页的  $\theta(g(n))$  无区别 对一个给定函数  $g(n)$ , 用  $\Theta(g(n))$  来表示以下的函数集合:

$$\Theta(g(n)) = \{f(n) \mid \text{存在正常量 } c_1, c_2, n_0, \text{ 使得对所有 } n \geq n_0, \text{ 有 } 0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n)\}$$



- 存在正常量  $c_1, c_2$  使得对于足够大的  $n$ , 函数  $f(n)$  能够被“夹在”  $c_1 g(n)$  和  $c_2 g(n)$  之间, 则  $f(n)$  属于集合  $\Theta(g(n))$ ,  $f(n) = \Theta(g(n))$ .
- 因为  $\Theta(g(n))$  是一个集合, 所以我们可以记为  $f(n) \in \Theta(g(n))$
- 我们称  $g(n)$  是  $f(n)$  的一个渐进紧致(确)界 (asymptotically tight bound)

## § 2.1 渐近表示法(续)

- 意义: 对所有的  $n \geq n_0$ , 函数  $f(n)$  在一个常数因子范围内等于  $g(n)$

$g(n)$  是  $f(n)$  的一个渐近紧致(确)界, 即  $g(n)$  是  $f(n)$  的渐近上界和渐近下界

$f(n)$ —算法的计算时间

$g(n)$ —算法时间的数量级

不同的输入实例, 一个算法的计算时间  $f(n)$  不一定相同, 故算法的计算时间应该是一个函数集合。

**Note:**  $\Theta$  定义中要求  $f(n)$  和  $g(n)$  是渐近非负的 ( $n$  足够大时函数值非负)

否则  $\theta(g(n))$  是空集

7

## § 2.1 渐近表示法(续)

- 例 (Page 27 in textbook):

$$\diamond T(n) = an^2 + bn + c \quad (a > 0)$$

$$\text{则 } T(n) = \theta(n^2)$$

$$\because \text{取 } c_1 = \frac{a}{4}, c_2 = 7\frac{a}{4}, n_0 = 2 \max\left(\frac{|b|}{a}, \sqrt{\frac{|c|}{a}}\right)$$

$$\text{对所有 } n > n_0, \text{ 有 } 0 \leq c_1 n^2 \leq an^2 + bn + c \leq c_2 n^2 \text{ 成立}$$

- 记号  $\theta(1)$ :

◇ 表示算法的运行时间与问题规模  $n$  无关

◇ 亦可理解为常值函数  $\theta(n^0)$  (任何常数是 0 次多项式)

8

## Asymptotic Notation in Equations

- Used to replace functions of lower-order terms to simplify equations/expressions.

- For example,

$$4n^3 + 3n^2 + 2n + 1 = 4n^3 + 3n^2 + \theta(n) \\ = 4n^3 + \theta(n^2) = \theta(n^3)$$

Or we can do the following:  $4n^3 + 3n^2 + 2n + 1 = 4n^3 + f(n^2)$ , where  $f(n^2)$  simplifies the equation

课本P27, 直觉上.....

## § 2.1 渐近表示法(续)

- b.  $O$ -notation(渐近上界)

◇ Def: 对给定函数  $g(n)$ ,  $O(g(n))$  是一个函数集合

$$O(g(n)) = \{f(n) \mid \exists \text{ 常数 } c, n_0 > 0, \text{ such that } 0 \leq f(n) \leq cg(n) \text{ for all } n \geq n_0\}$$

◇ 即在一个常数因子范围内  $g(n)$  是  $f(n)$  的渐近上界

◇  $\Theta$  记号强于  $O$  记号

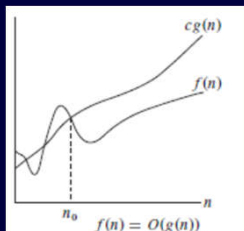
$$\because \theta(g(n)) \subseteq O(g(n))$$

$$\therefore f(n) = \theta(g(n)) \Rightarrow f(n) = O(g(n))$$

10

## ■ 大O的数学定义(渐进上界)

若  $g(n)$  和  $f(n)$  是定义在正整数集合上的两个函数, 则  $f(n) = O(g(n))$  表示存在两个正的常数  $c$  和  $n_0$ , 使得当  $n \geq n_0$  时都满足  $0 \leq f(n) \leq c \cdot g(n)$ 。



- 函数  $f(n)$  是集合  $O(g(n))$  的成员;
- $f(n) = \Theta(g(n))$  蕴含  $f(n) = O(g(n))$ , 因为  $\Theta$  是一个比  $O$  记号更强的概念;
- 我们称  $g(n)$  是  $f(n)$  的一个渐近上界(asymptotically upper bound), 限制算法最坏情况运行时间;

11

## § 2.1 渐近表示法(续)

- Note:

◇  $O$  描述上界, 当用于界定一个最坏运行时间时, 蕴含着该算法在任意输入上的运行时间都围于此界

◇  $\theta$  则不然, 一个算法的最坏运行时间是  $\theta(g(n))$ , 并非蕴含着该算法对每个输入实例的运行时间均围于  $\theta(g(n))$

- 例:

◇ 插入排序当输入初始有序时, 它的运行时间是  $\theta(n)$ , 而不是  $\theta(n^2)$ , 但  $O(n^2)$  对任何输入实例都成立。

$$\because n = O(n^2) \text{ 及 } n^2 = O(n^2) \text{ 均成立,}$$

但是

$$n^2 = \theta(n^2) \text{ 而 } n \neq \theta(n^2)$$

12

### Example

■  $an^3+bn^2+cn+d = O(n^3), a>0$

Proof: 前面的例子已经表明  $an^3+bn^2+cn+d = \Theta(n^3)$ , 又由于集合  $\Theta(n^3)$  是被  $O(n^3)$  包含的, 所以得到证明

■  $an^2+bn+d = O(n^3), a>0$

### Example

$1/3n^2 - 3n \in O(n^2)$  because  $1/3n^2 - 3n \leq cn^2$  if  $c \geq 1/3 - 3/n$  which holds for  $c = 1/3$  and  $n > 1$

$k_1n^2+k_2n+k_3 \in O(n^2)$  because  $k_1n^2+k_2n+k_3 \leq (k_1+|k_2|+|k_3|)n^2$  and for  $c > k_1 + |k_2| + |k_3|$  and  $n \geq 1$ ,  $k_1n^2 + k_2n + k_3 \leq cn^2$

$k_1n^2+k_2n+k_3 \in O(n^3)$  as  $k_1n^2+k_2n+k_3 \leq (k_1+|k_2|+|k_3|)n^3$

### O-notation

#### Note:

--When we say "the running time is  $O(n^2)$ " we mean that the worst-case running time is  $O(n^2)$  -- the best case might be better.

--Use of  $O$ -notation often makes it much easier to analyze algorithms; we can easily prove the  $O(n^2)$  insertion-sort time bound.

--We often abuse the notation a little:

>> We often write  $f(n) = O(g(n))$  instead of  $f(n) \in O(g(n))$

>> We often use  $O(n)$  in equations: e.g.  $2n^2 + 3n + 1 = 2n^2 + O(n)$  (meaning that  $2n^2 + 3n + 1 = 2n^2 + f(n)$  where  $f(n)$  is some function in  $O(n)$ )

>> We use  $O(1)$  to denote constant time.

### § 2.1 渐近表示法(续)

■ 当说算法的运行时间上界是  $O(n^2)$  往往是指其最坏运行时间, 无须修饰语, 对那些最好、最坏、平均时间数量级不同者均成立, 而  $\Theta$  则要分开表达、加修饰语。

#### c. $\Omega$ 记号 (渐近下界, lower bound)

❖ Def: 对给定函数  $g(n)$ ,  $\Omega(g(n))$  是一个函数集合:

$$\Omega(g(n)) = \{f(n) \mid \exists \text{ 常数 } c, n_0 > 0 \text{ such that } 0 \leq cg(n) \leq f(n) \text{ for all } n \geq n_0\}$$

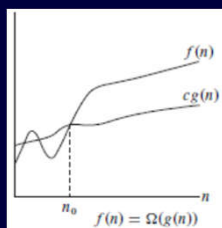
❖ 意义: 对所有  $n > n_0$ ,  $f(n)$  在  $cg(n)$  上方。即在一个常数因子范围内  $g(n)$  是  $f(n)$  的渐近下界

16

#### ■ 大 $\Omega$ 的数学定义(渐近下界)

对一个给定函数  $g(n)$ , 用  $\Omega(g(n))$  来表示以下的函数集合:

$\Omega(g(n)) = \{f(n) \mid \text{存在正常量 } c, n_0, \text{ 使得对所有 } n \geq n_0 \geq 0, \text{ 有 } 0 \leq cg(n) \leq f(n)\}$



■ 存在正常量  $c$  使得对于  $n_0$  及其右边的所有  $n$  值, 函数  $f(n)$  值总大于等于  $cg(n)$ , 则  $f(n)$  属于  $\Omega(g(n))$ 。

■ 我们称  $g(n)$  是  $f(n)$  的 **渐近下界** (asymptotically lower bound)

17

### § 2.1 渐近表示法(续)

■ Th2.1 对任意函数  $f(n)$  和  $g(n)$ ,  $f(n) = \theta(g(n))$  当且仅当  $f(n) = O(g(n))$  和  $f(n) = \Omega(g(n))$ 。(Page 28)

即:  $g(n)$  是  $f(n)$  的渐紧界当且仅当  $g(n)$  是  $f(n)$  的渐近上界和渐近下界

■ 当  $\Omega$  用来界定一个算法的最好情况下的运行时间时, 蕴含着该算法在任意输入上的运行时间都围于此界。

#### ■ 例:

❖ 插入排序的下界是  $\Omega(n)$ , 对任何实例成立(即插入排序的最好运行时间是  $\Omega(n)$ )

$$\therefore n = \Omega(n) \quad n^2 = \Omega(n)$$

18

## § 2.1 渐近表示法(续)

## d. 方程中的渐近记号 (P28~P29)

- ❖ 例：基于比较的排序时间下界是

$$\lg n! = n \lg n - 1.44n + O(\lg n)$$

- ❖ 可消去不必要的细节，突出主项的常数因子等。

## e. 小 o 记号(渐近非紧上界)

- ❖ 大 O 记号表示的渐近上界可以是渐近紧致的，也可以是渐近非紧界

$$2n^2 = O(n^2) \quad \because \frac{2n^2}{n^2} \rightarrow \text{常数} 2, \text{ 紧致界}$$

$$2n = O(n^2) \quad \because \frac{2n}{n^2} \rightarrow 0, \text{ 非紧界}$$

- ❖ 小 o 记号用来表示——函数的渐近非紧致上界

19

## § 2.1 渐近表示法(续)

## e. 小 o 记号(渐近非紧上界)

- ❖ Def: 找出在形式化定义上与大 O 记号的两处差别

$$o(g(n)) = \{f(n) \mid \forall \text{ 常数 } c > 0, \exists \text{ 常数 } n_0 > 0 \text{ such that } 0 \leq f(n) < cg(n) \text{ for all } n \geq n_0\}$$

只要 n 足够大, g(n) 是 f(n) 的上界

例  $2n = o(n^2)$  但  $2n^2 \neq o(n^2)$ 。

直观上, 当  $n \rightarrow \infty$ , f(n) 相对于 g(n) 是可忽略的

即:  $f(n) = o(g(n))$  蕴含着  $\lim_{n \rightarrow \infty} \frac{f(n)}{g(n)} = 0$

或说 f 和 g 数量级不同, 否则不可能对于任意常数 c, 都有  $cg(n)$  严格大于 f(n)

## § 2.1 渐近表示法(续)

f.  $\omega$ -记号(渐近非紧下界)

- ❖ Def: 找出在形式化定义上与大 O 记号的两处差别

$$\omega(g(n)) = \{f(n) \mid \forall \text{ 常数 } c > 0, \exists \text{ 常数 } n_0 > 0 \text{ such that } 0 \leq cg(n) < f(n) \text{ for all } n \geq n_0\}$$

即: 对任意常数  $c > 0$ ,  $cg(n)$  对足够大的 n 要严格小于 f(n)。∴ g 和 f 必定不是同数量级, (f 量级 > g 量级)

- ❖ 例:

$$\frac{n^2}{2} = \omega(n) \quad \text{但} \quad \frac{n^2}{2} \neq \omega(n^2)$$

$$f(n) = \omega(g(n)) \Rightarrow \lim_{n \rightarrow \infty} \frac{f(n)}{g(n)} = \infty$$

21

## § 2.1 渐近表示法(续)

## g. 函数间比较

- ❖ 许多实数的关系性质可用(引申)到渐近比较, 下面假定 f(n) 和 g(n) 是渐近正的

- ❖ 传递性 (对于五种渐进记号均适用)

$$\begin{aligned} f(n) = \theta(g(n)) \text{ and } g(n) = \theta(h(n)) &\Rightarrow f(n) = \theta(h(n)) // \text{渐紧界} \\ f(n) = O(g(n)) \text{ and } g(n) = O(h(n)) &\Rightarrow f(n) = O(h(n)) \\ f(n) = \Omega(g(n)) \text{ and } g(n) = \Omega(h(n)) &\Rightarrow f(n) = \Omega(h(n)) \\ f(n) = o(g(n)) \text{ and } g(n) = o(h(n)) &\Rightarrow f(n) = o(h(n)) // \text{渐近非紧上界} \\ f(n) = \omega(g(n)) \text{ and } g(n) = \omega(h(n)) &\Rightarrow f(n) = \omega(h(n)) // \text{渐近非紧下界} \end{aligned}$$

22

## § 2.1 渐近表示法(续)

## g. 函数间比较

- ❖ 自反性

$$f(n) = \theta(f(n)), f(n) = O(f(n)), f(n) = \Omega(f(n))$$

渐近非紧界无自反性

- ❖ 对称性 (仅对渐近紧确界成立)

$$f(n) = \theta(g(n)) \text{ iff } g(n) = \theta(f(n))$$

紧致界有对称性

23

## § 2.1 渐近表示法(续)

## g. 函数间比较

- ❖ 转置对称性

$$f(n) = O(g(n)) \text{ iff } g(n) = \Omega(f(n))$$

// 大 O 与大 Ω 对调时, f、g 对称

$$f(n) = o(g(n)) \text{ iff } g(n) = \omega(f(n))$$

// g 是 f 的非紧上界等价于 f 是 g 的非紧下界

24

## § 2.1 渐近表示法(续)

## 9. 函数间比较

## ❖ 算数运算 (请同学课下证明)

- ❖  $O(f(n)) + O(g(n)) = O(\max\{f(n), g(n)\})$  ;
- ❖  $O(f(n)) + O(g(n)) = O(f(n) + g(n))$  ;
- ❖  $O(f(n)) \cdot O(g(n)) = O(f(n) \cdot g(n))$  ;
- ❖  $O(cf(n)) = O(f(n))$  ;
- ❖  $g(n) = O(f(n)) \Rightarrow O(f(n)) + O(g(n)) = O(f(n))$

25

## § 2.1 渐近表示法(续)

由上述4个性质, 可将两函数间的渐近比较类比于两个实数间的比较。

$$\begin{aligned} f(n) = O(g(n)) &\approx a \leq b / " \approx " \text{ 类似于 } f \sim a, g \sim b \\ f(n) = \Omega(g(n)) &\approx a \geq b \\ f(n) = \theta(g(n)) &\approx a = b \\ f(n) = o(g(n)) &\approx a < b \\ f(n) = \omega(g(n)) &\approx a > b \end{aligned}$$

但实数的三岐性(三分性质)不能类比到渐近表示中

三岐性  $\forall a, b \in \mathbb{R}$ , 下述三种情况必有一个成立:

$$a < b, a = b, \text{ or } a > b$$

即任意两实数间是可比较的

26

## § 2.1 渐近表示法(续)

并非所有函数都是渐近可比较的

即  $\exists f(n)$  和  $g(n)$ ,

可能  $f(n) = O(g(n))$  不成立,

而  $f(n) = \Omega(g(n))$  也不成立

$\therefore$  由 Th2.1 知,  $f \neq \theta(g)$

例: 函数  $n$  和  $n^{1+\sin n}$  之间是无法渐近比较的

$$1 + \sin n \Rightarrow 0 \sim 2$$

即  $n^{1+\sin n}$  在  $O(1) \sim O(n^2)$  之间波动

27

## Asymptotic order of growth

■ A way of comparing functions that ignores constant factors and small input sizes

- ❖  $O(g(n))$ : class of functions  $f(n)$  that grow no faster than  $g(n)$
- ❖  $\Theta(g(n))$ : class of functions  $f(n)$  that grow at same rate as  $g(n)$
- ❖  $\Omega(g(n))$ : class of functions  $f(n)$  that grow at least as fast as  $g(n)$

28

## § 2.2 标准记号和常用函数

司特林公式, 重叠对数等

见教材P31.

29