Chapter 2. 函数增长率
Growth of functions

1

与MIT算法导论教材对应关系

Chapter 3. 函数的渐进增长

2

Topics:

- Growth of functions
- O/o/Θ/Ω/ω notations

3

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)

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 → ∞

"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,…}(∵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. θ记号（θ-notation）

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

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

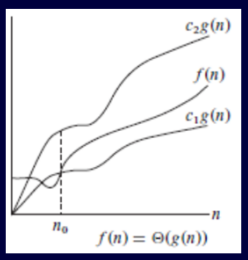
- 即 $f(n) \in \theta(g(n))$表示存在正常数c_1, c_2及足够大的n，使得f(n)夹在c_1g(n)和c_2g(n)之间
- 通常 $f(n) \in \theta(g(n))$ 表示为 $f(n) = \theta(g(n))$,其实f(n)是Θ(g(n))的成员 (可以理解为簇中的一员)

6

■ 大Θ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Θ(g(n))来表示以下的函数集合：

Θ(g(n))={f(n): 存在正常量c_1, c_2, n_0, 使得对所有n>=n_0, 有 $0 \le c_1 g(n) \le f(n) \le c_2 g(n)$}



- 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_1g(n)和c_2g(n)之间，则f(n)属于集合Θ(g(n)), f(n)=Θ(g(n))。
- 因为Θ(g(n))是一个集合，所以我们也可以记为f(n)∈Θ(g(n))
- 我们称g(n)是f(n)的一个渐进紧致(确)界(asymptotically tight bound)

7

§ 2.1 渐近表示法(续)

■ 意义：对所有的$n \ge n_0$，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： Θ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

8

§ 2.1 渐近表示法(续)

■ 例 (Page 27 in textbook)：

❖ $T(n)=an^2+bn+c$ (a>0)

则 $T(n) = \theta(n^2)$

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

对所有$n > n_0$, 有$0 \le c_1 n^2 \le an^2 + bn + c \le c_2 n^2$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)

9

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

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

❖ 即在一个常数因子范围内g(n)是f(n)的渐近上界

❖ Θ记号强于O 记号

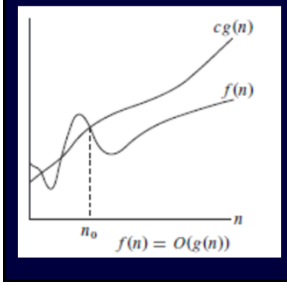
$\because \theta(g(n)) \subseteq O(g(n))$

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

11

■ 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$n \ge n_0$时都满足$0 \le f(n) \le c \cdot g(n)$。



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

12

§ 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囿于此界
  - θ 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囿于 $\theta(g(n))$
- 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。

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

    但是

    $n^2 = \theta(n^2)$而$n \neq \theta(n^2)$

13

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 $1/3n^2 - 3n \le cn^2$ if $c \ge 1/3 - 3/n$ which holds for $c = 1/3$ and $n > 1$

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

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

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Θ则要分开表达、加修饰语。
- c. Ω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：

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

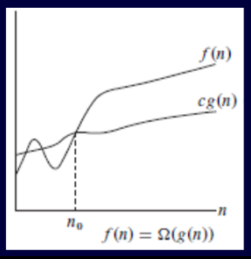
  - 意义：对所有$n>n_0$，f(n)在cg(n)上方。即在一个常数因子范围内g(n)是f(n)的渐近下界

17

- 大Ω的数学定义(渐进下界)

对一个给定函数g(n)，用Ω(g(n))来表示以下的函数集合：

Ω(g(n))={f(n): 存在正常量c, n_0, 使得对所有$n \ge n_0 \ge 0$, 有$0 \le cg(n) \le f(n)$}



- 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Ω(g(n))。
- 我们称g(n)是f(n)的渐进下界(asymptotically lower bound)

18

§ 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 Ω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囿于此界。
- 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 $\Omega(n)$)

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

19

§ 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

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

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

- 小 o 记号用来表示——函数的渐近非紧致上界

20

§ 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

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

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 $n\to\infty$ ，f(n)相对于g(n)是可忽略的

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

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)

21

§ 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Ω记号的两处差别

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

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：

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

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

22

§ 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)

$f(n)=\theta(g(n))$ and g(n)=θ(h(n)) $\Rightarrow f(n)=\theta(h(n))$ //渐紧界

$f(n)=O(g(n))$ and g(n)=O(h(n)) $\Rightarrow f(n)=O(h(n))$

$f(n)=\Omega(g(n))$ and g(n)=Ω(h(n)) $\Rightarrow f(n)=\Omega(h(n))$

$f(n)=o(g(n))$ and g(n)=o(h(n)) $\Rightarrow f(n)=o(h(n))$ //渐近非紧上界

$f(n)=\omega(g(n))$ and g(n)=ω(h(n)) $\Rightarrow f(n)=\omega(h(n))$ //渐近非紧下界

23

§ 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 (仅对渐进紧确界成立)

$f(n)=\theta(g(n))\ iff\ g(n)=\theta(f(n))$

紧致界有对称性

24

§ 2.1 渐近表示法(续)

g. 函数间比较

- 转置对称性

$$f(n)=O(g(n)) \ iff \ g(n)=\Omega(f(n))$$
//大O与大Ω对调时，f、g对称
$$f(n)=o(g(n)) \ iff \ g(n)=\omega(f(n))$$
//g是f的非紧上界等价于f是g的非紧下界

25

§ 2.1 渐近表示法(续)

g. 函数间比较

- 算数运算 (请同学课下证明)
  - $O(f(n))+O(g(n)) = O(\max\{f(n),g(n)\})$；
  - $O(f(n))+O(g(n)) = O(f(n)+g(n))$；
  - $O(f(n))*O(g(n)) = O(f(n)*g(n))$；
  - $O(cf(n)) = O(f(n))$；
  - $g(n)= O(f(n)) \Rightarrow O(f(n))+O(g(n)) = O(f(n))$

26

§ 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。

$$f(n)=O(g(n)) \approx a \le b \ / \text{“}\approx\text{”相似于} f\sim a, g\sim b$$
$$f(n)=\Omega(g(n)) \approx a \ge b$$
$$f(n)=\theta(g(n)) \approx a = b$$
$$f(n)=o(g(n)) \approx a < b$$
$$f(n)=\omega(g(n)) \approx a > b$$

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：

$$a<b, a=b, or\ a>b$$

即任意两实数间是可比较的

27

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

28

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 $g(n)$

29

§ 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.

30