Chapter 3: Growth of Functions

About this lecture

- Introduce Asymptotic Notation
  - $\Theta(\ )$, $O(\ )$, $\Omega(\ )$, $o(\ )$, $\omega(\ )$

Dominating Term

Recall that for input size n,

- Insertion Sort 's running time is:

  $An^2 + Bn + C$, (A, B, C are constants)

- Merge Sort 's running time is:

  $Dn \log n + En + F$, (D, E, F are constants)

- To compare their running times for large n, we can just focus on the dominating term **(the term that grows fastest when n increases)**
  - An^2 vs $Dn \log n$

Dominating Term

- If we look more closely, the leading constants in the dominating term does not affect much in this comparison (if n is sufficiently large)
  - We may as well compare n^2 vs $n \log n$ (instead of An^2 vs $Dn \log n$)

- As a result, we conclude that Merge Sort is better than Insertion Sort when n is sufficiently large

Asymptotic Efficiency

- The previous comparison studies the asymptotic efficiency of two algorithms
- If algorithm P is asymptotically faster than algorithm Q, P is often a better choice
- To aid (and simplify) our study in the asymptotic efficiency, we now introduce some useful asymptotic notations

Big-O notation

✓ Definition: Given a function g(n), we denote $O(g(n))$ to be the set of functions:

$\{f(n) \mid$ there exists positive constants c and n_0 such that $0 \le f(n) \le c\,g(n)$ for all $n \ge n_0\}$

➢ Rough Meaning: $O(g(n))$ includes all functions that are upper bounded by $g(n)$

Big-O notation (example)

- $4n \in O(5n)$ [proof: $c = 1$, $n \geq 1$]
- $4n \in O(n)$ [proof: $c = 4$, $n \geq 1$]
- $4n + 3 \in O(n)$ [proof: $c = 5$, $n \geq 3$]
- $n \in O(0.001n^2)$ [proof: $c = 10$, $n \geq 100$]
- $\log_e n \in O(\lg n)$ [proof: $c = 1$, $n \geq 1$]
- $\lg n \in O(\log_e n)$ [proof: $c = \lg e$, $n \geq 1$]
- Remark: Usually, we will slightly abuse the notation, and write $f(n) = O(g(n))$ to mean $f(n) \in O(g(n))$

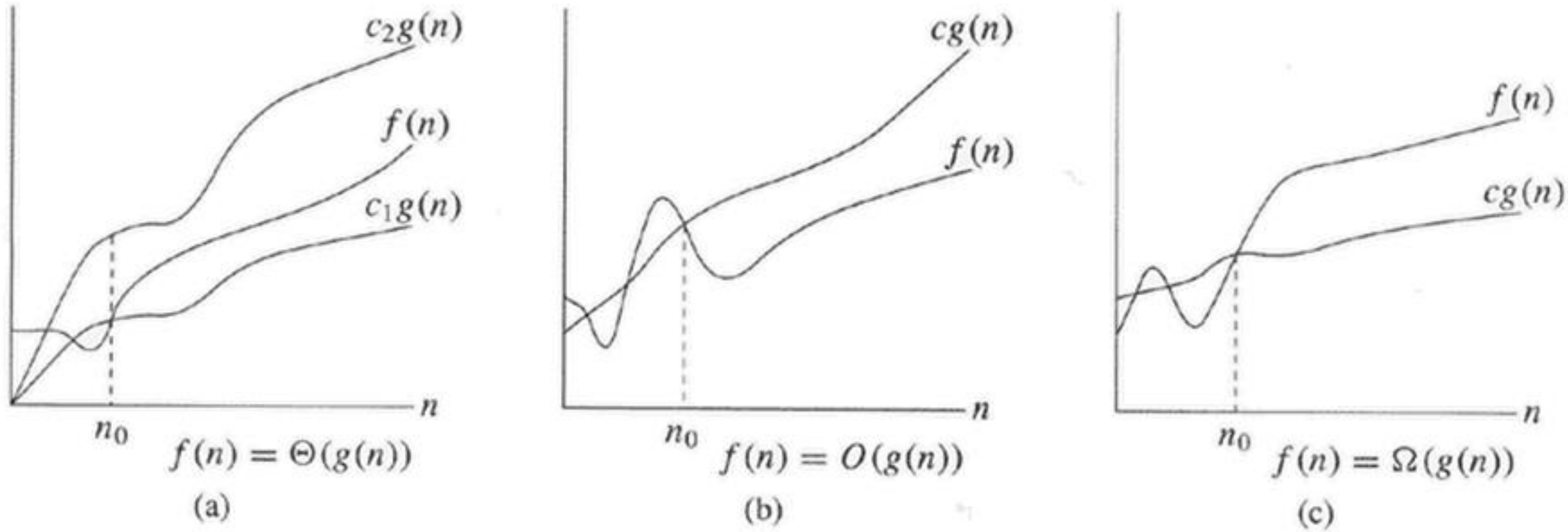

Figure 3.1 Graphic examples of the Θ, O, and Ω notations. In each part, the value of n_0 shown is the minimum possible value; any greater value would also work. **(a)** Θ-notation bounds a function to within constant factors. We write $f(n) = \Theta(g(n))$ if there exist positive constants n_0, c_1, and c_2 such that to the right of n_0, the value of $f(n)$ always lies between $c_1g(n)$ and $c_2g(n)$ inclusive. **(b)** O-notation gives an upper bound for a function to within a constant factor. We write $f(n) = O(g(n))$ if there are positive constants n_0 and c such that to the right of n_0, the value of $f(n)$ always lies on or below $cg(n)$. **(c)** Ω-notation gives a lower bound for a function to within a constant factor. We write $f(n) = \Omega(g(n))$ if there are positive constants n_0 and c such that to the right of n_0, the value of $f(n)$ always lies on or above $cg(n)$.

Big-Omega notation

- ✓ Definition: Given a function $g(n)$, we denote $\Omega(g(n))$ to be the set of functions

  $\{f(n) \mid$ there exists positive constants c and n_0

  such that $0 \le c\,g(n) \le f(n)$ for all $n \ge n_0 \}$

- Rough Meaning: $\Omega(g(n))$ includes all functions that are lower bounded by $g(n)$

Big-Ω notation (example)

- $5n \in \Omega(4n)$ [proof: $c = 1$, $n \geq 1$]
- $n \in \Omega(4n)$ [proof: $c = 1/4$, $n \geq 1$]
- $4n + 3 \in \Omega(n)$ [proof: $c = 1$, $n \geq 1$]
- $0.001n^2 \in \Omega(n)$ [proof: $c = 1/10$, $n \geq 100$]
- $\log_e n \in \Omega(\lg n)$ [proof: $c = 1/\lg e$, $n \geq 1$]
- $\lg n \in \Omega(\log_e n)$ [proof: $c = 1$, $n \geq 1$]

Big-O and Big-Omega

- ✓ Similar to Big-O, we will slightly abuse the notation, and write $f(n) = \Omega(g(n))$ to mean $f(n) \in \Omega(g(n))$

- ✓ Relationship between Big-O and Big-Ω :

$$f(n) = \Omega(g(n)) \Leftrightarrow g(n) = O(f(n))$$

Θ notation (Big-O \ Big-Ω)

✓ Definition: Given a function g(n), we denote $\Theta(g(n))$ to be the set of functions

{f(n) | there exists positive constants c_1, c_2, and n_0 such that $0 \le c_1 g(n) \le f(n) \le c_2 g(n)$ for all $n \ge n_0$ }

➢ *Meaning: Those functions which can be both upper bounded and lower bounded by of g(n)*

Big-O, Big-Ω, and Θ

✓Similarly, we write $f(n) = \Theta(g(n))$ to mean $f(n) \in \Theta(g(n))$

✓Relationship between Big-O, Big-Ω, and Θ:

$$f(n) = \Theta(g(n))$$

$$\Leftrightarrow$$

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

Θ notation (example)

- $4n = \Theta(n)$ [$c_1 = 1$, $c_2 = 4$, $n \geq 1$]
- $4n + 3 = \Theta(n)$ [$c_1 = 1$, $c_2 = 5$, $n \geq 3$]
- $\log_e n = \Theta(\lg n)$ [$c_1 = 1/\lg e$, $c_2 = 1$, $n \geq 1$]
- Running Time of Insertion Sort = $\Theta(n^2)$
  - If not specified, running time refers to the worst-case running time
- Running Time of Merge Sort = $\Theta(n \lg n)$

To remember the notation

O is like $\le$: $f(n) = O(g(n))$ means $f(n) \le cg(n)$

Ω is like $\ge$: $f(n) = \Omega(g(n))$ means $f(n) \ge cg(n)$

Θ is like $=$: $f(n) = \Theta(g(n)) \Leftrightarrow g(n) = \Theta(f(n))$

o is like $<$: $f(n) = o(g(n))$ means $f(n) < cg(n)$

ω is like $>$: $f(n) = \omega(g(n))$ means $f(n) > cg(n)$

✓ Note: Not any two functions can be compared asymptotically (E.g., sin x vs. cos x)

Little-o notation

- ✓ Definition: Given a function $g(n)$, we denote $o(g(n))$ to be the set of functions

  $\{f(n) \mid$ for any positive c, there exists positive constant n_0 such that $0 \leq f(n) < c\, g(n)$, for all $n \geq n_0$ $\}$

Note the similarities and differences with Big O

Little-o (equivalent definition)

✓ Definition: Given a function $g(n)$, $o(g(n))$ is the set of functions

$$\{f(n) \mid \lim_{n \to \infty} \left(f(n)/g(n)\right) = 0\}$$

Examples:

- $4n = o(n^2)$
- $n \lg n = o(n^{1.000001})$
- $n \lg n = o(n \lg^2 n)$

Little-omega notation

- ✓ Definition: Given a function $g(n)$, we denote $\omega(g(n))$ to be the set of functions

  $\{f(n) \mid$ for any positive c, there exists positive constant n_0 such that $0 \le c\, g(n) < f(n)$, for all $n \ge n_0$ $\}$

Note the similarities and differences with the Big-Omega definition

Little-omega (equivalent definition)

- ✓ Definition: Given a function $g(n)$, $\omega(g(n))$ is the set of functions

$$\{f(n) \mid \lim_{n \to \infty} \left(g(n)/f(n)\right) = 0\}$$

- ✓ Relationship between Little-o and Little-ω :

$$f(n) = \omega(g(n)) \Leftrightarrow g(n) = o(f(n))$$

Practice at home

- Exercises: 3.1-1, 3.1-3,3.1-4
- Problem: 3-1 (a, d), 3-2 (b, f), 3-4(c, f, g)

Practice at home

1. Given that

$$f(n) = a_m n^m + a_{m-1} n^{m-1} + \ldots + a_1 n + a_0, \quad \text{where } a_m > 0.$$

Show that $f(n) = \Theta(n^m)$.

2. Write a program to solve the Hanoi Tower Problem

3. What is wrong with the following argument?

"Since $n = O(n)$, and $2n = O(n)$, $\ldots$, we have

$$\sum_{k=1}^{n} k \cdot n = \sum_{k=1}^{n} O(n) = O(n^2).\text{"}$$