Analysis of Algorithms

COMP.4040, Summer 2019

Chapter 3: Growth of Functions

By: Sirong Lin, PhD

University of Massachusetts Lowell

UMASS LOWELL
Learning with Purpose

Announcement

No class on May 27 (Monday)

My today's office hours cancelled

TA (Yan Li)'s

- 1-2pm today, Dandeneau Hall 420

Homework 2

Due Date: May 30, 2019 (Th), Before the class starts

Honor Statement needs to be enclosed for each assignment, otherwise the homework will not be graded

Outline

Introduce various asymptotic notations

Growth of functions

Asymptotic Notation

Asymptotical Analysis

BIG IDEA: **Asymptotical Analysis**

- rate of growth/order of growth of function as n —> ∞
- the running time is dominated by the leading terms
  - the higher-order terms (leading terms) grow much faster than the lower terms

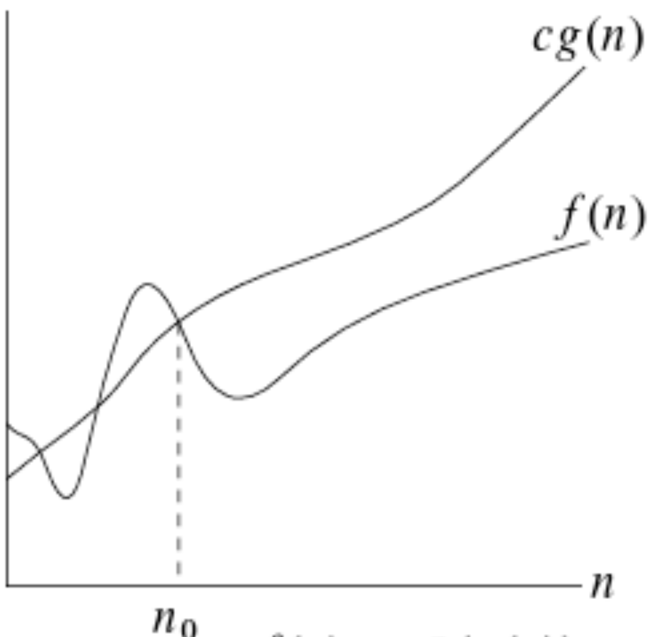
Asymptotical Analysis

Use asymptotic notations to simplify the asymptotical analysis :

- mostly refer to running time, could refer to space or other factors
- Conventions:
  - O-notation, Ω-notation, Θ-notation, o-notation, ω-notation

O-notation

$$O(g(n)) = \{ f(n) : \text{ there exist positive constants } c \text{ and } n_0 \text{ such that } 0 \le f(n) \le cg(n) \text{ for all } n \ge n_0 \} .$$



$f(n) = O(g(n))$

for all values *n* at and to the right of n_0 (i.e., n >= n_0), the value of the function *f(n)* is on or below *cg(n)*

O-notation (Cont'd)

What is $O(g(n))$?

the SET of ALL the functions that satisfy: there exists positive constants c and n_0, such that for all $n \geq n_0$, $0 \leq f(n) \leq cg(n)$

What is $f(n)$?

a member of $O(g(n))$, belongs to the set of $O(g(n))$

O-notation (Cont'd)

Examples of O-notation (details in class)

- $2n^2 = O(n^3)$, with $c = 1$ and $n_0 = 2$
- examples of functions in $O(n^2)$:
  - n^2, $n^{2+}n$, $n^2+1000n$, $1000n^2+1000n$, n, $n^{1.99}$, $n^2/\lg\lg\lg n$

a lot of examples in class

O-notation Summary

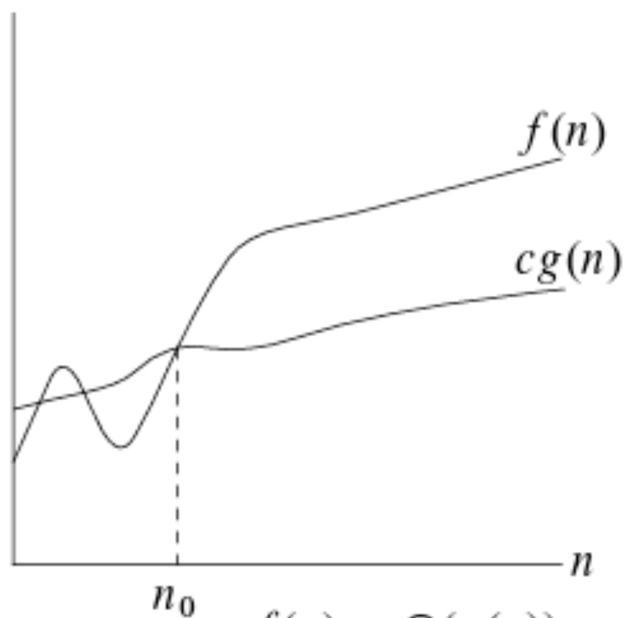
There are infinite number of set of functions for both f(n) and g(n)

O-notation does **not say WHAT functions are**, just say the relationship between two functions

O-notation claims an **asymptotic upper bound** on a function f(n), but does **not claim** about <u>HOW TIGHT</u> an upper bound it is

Ω-notation

$$\Omega(g(n)) = \{f(n) : \text{there exist positive constants } c \text{ and } n_0 \text{ such that } 0 \le cg(n) \le f(n) \text{ for all } n \ge n_0\}.$$



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

Ω-notation gives an **asymptotic lower bound** on a function

Ω-notation (Cont'd)

What is $\Omega(g(n))$?

the SET of ALL the functions that satisfy: there exists positive constants c and n_0, such that for all $n \geq n_0$, $0 \leq cg(n) \leq f(n)$

What is f(n)?

a member of $\Omega(g(n))$, belongs to the set of $\Omega(g(n))$

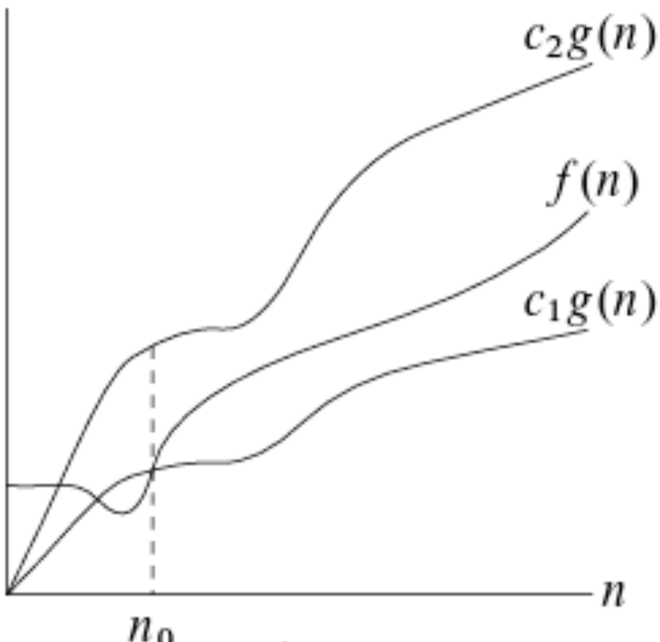
Ω-notation (Cont'd)

Examples in class

$\Omega(g(n))$ says, one algorithm at least needs this time (e.g., at least this bad)

Θ-notation

$\Theta(g(n)) = \{f(n) :$ there exist 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 $n \ge n_0\}$.[1]



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

need to choose different constants c1 and c2

Θ-notation (Cont'd)

Theorem 3.1

For any two functions $f(n)$ and $g(n)$, we have $f(n) = \Theta(g(n))$ if and only if $f(n) = O(g(n))$ and $f(n) = \Omega(g(n))$. ■

Θ-notation gives an asymptotically **tight** bound for f(n)

Θ-notation (Cont'd)

$\Theta(1)$: $\Theta(n^0)$

Θ is symmetric:

if $f(n) = \Theta(g(n))$, then $g(n) = \Theta(f(n))$

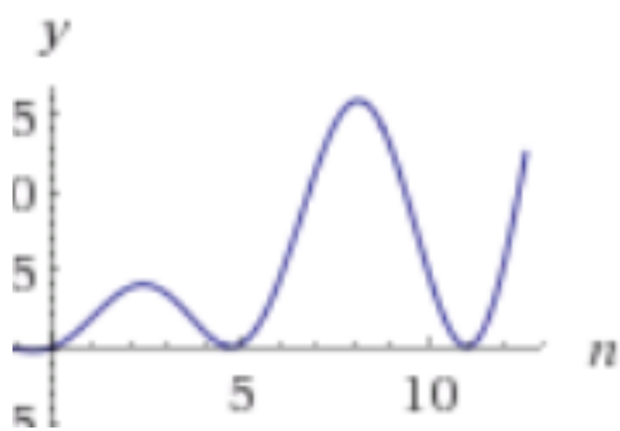
e.g., $n^3 = \Theta(3n^3-n^2)$, $3n^3-n^2 = \Theta(n^3)$

$n^3 \neq \Theta(n)$, $n \neq \Theta(n^3)$

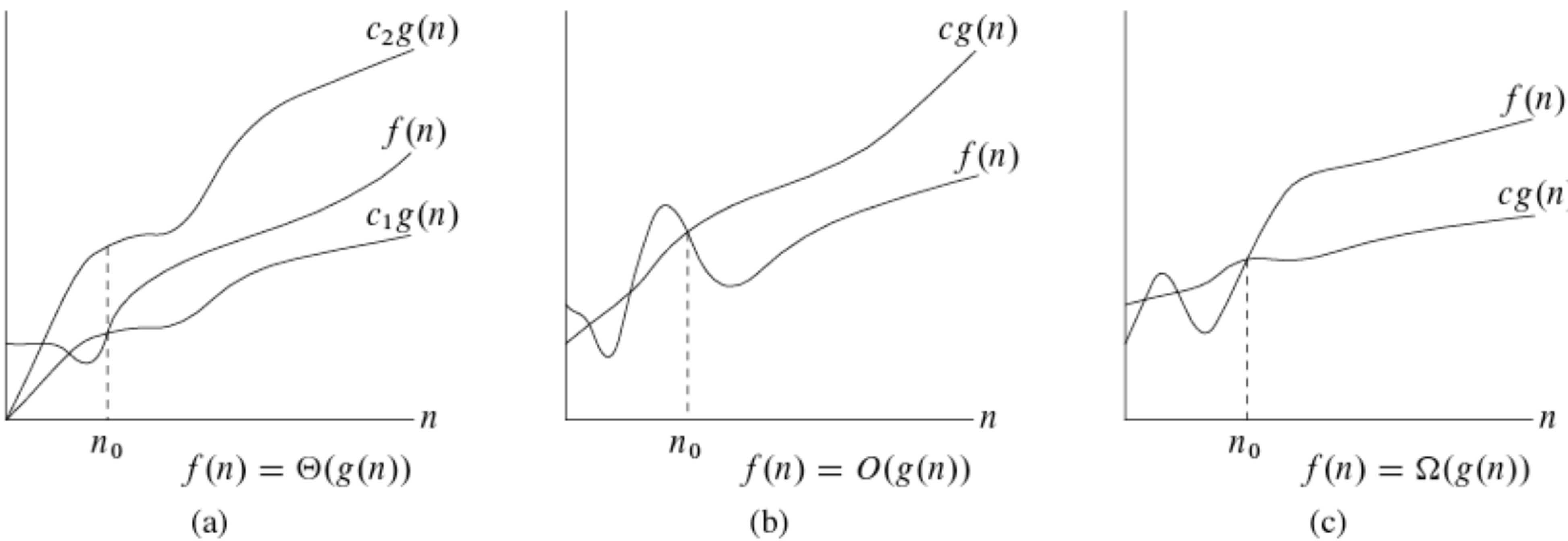
Θ-notation (Cont'd)

not all functions have Θ, e.g., $f(n) = n(1+ \sin n)$

$f(n) \in O(n)$, $f(n) \in \Omega(0)$, not in $\Theta(n)$ or $\Theta(0)$



Comparison of Θ-, O-, Ω-,

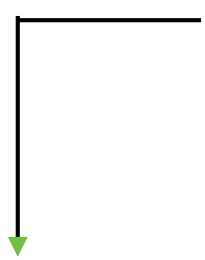


Comparison of O-, Ω-,Θ-

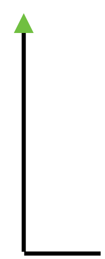
Arrow diagram (idea from Prof. Karen Daniels)

help to compare functions in different notations

O-notation



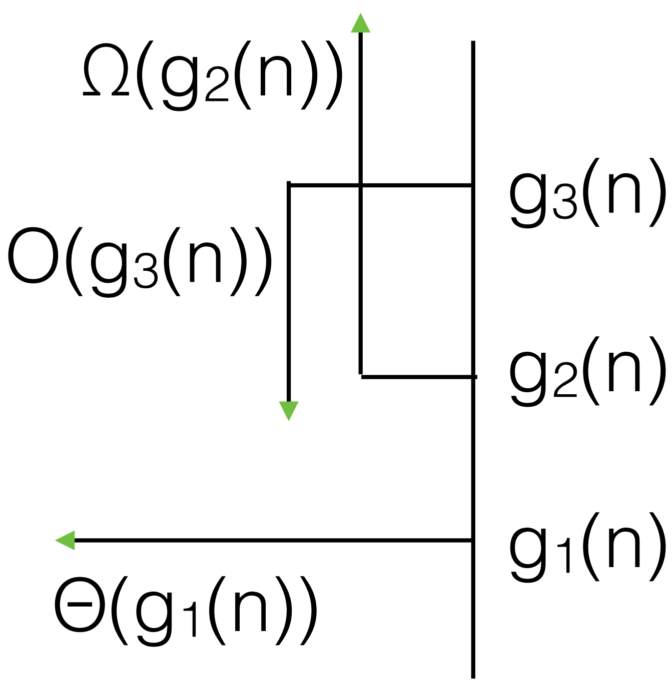
Ω-notation



Θ-notation

Comparison of O-, Ω-,Θ-

Arrow diagram, e.g.,



$g_3(n)$ grows faster than $g_2(n)$
$g_2(n)$ grows faster than $g_1(n)$

Comparison of O-, Ω-,Θ-

Examples with the Arrow diagram

$f_1(n) = \lg^2 n$, $f_2(n) = \lg(n^2)$

Is $f_2(n) \in O(f_1(n))$ true or false?

Is $f_2(n) \in \Theta(f_1(n))$ true or false?

Asymptotic notation in equations (Cont'd)

$2n^2+3n+1 = 2n^2+\Theta(n)$

there exists a function, $f(n) \in \Theta(n)$, such that $2n^2+3n+1 = 2n^2 + f(n)$. In particular, $f(n) = 3n+1$

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

for all functions $g(n) \in \Theta(n)$, there exists a function $h(n) \in \Theta(n^2)$, such that $2n^2+g(n) = h(n)$

chain together: $2n^2+3n+1 = 2n^2+\Theta(n) = \Theta(n^2)$

When asymptotic notation appears in formula, we interpret as "anonymous function that we don't care to name"

o-notation (little-o)

$$o(g(n)) = \{ f(n) : \text{ for any positive constant } c > 0, \text{ there exists a constant } n_0 > 0 \text{ such that } 0 \leq f(n) < cg(n) \text{ for all } n \geq n_0 \} .$$

Difference between O- & o-?

- O-notation denotes an upper bound that may or may not be asymptotically tight
- o-notation denotes an **upper bound that is NOT asymptotically tight**
  - no matter what c we choose, when $n \geq n_0$, we have $f(n) < c(g(n))$
  - n_0 is a value in terms of c

$$\lim_{n \to \infty} \frac{f(n)}{g(n)} = 0 .$$

o-notation examples

$3n=O(n^2)$ and $3n^2=O(n^2)$ are both true

$3n = o(n^2)$, true or false? $n_0 = 3/c$

$3n^2 = o(n^2)$, true or false?

ω-notation (little-ω)

$$\omega(g(n)) = \{ f(n) : \text{for any positive constant } c > 0, \text{ there exists a constant } n_0 > 0 \text{ such that } 0 \le cg(n) < f(n) \text{ for all } n \ge n_0 \} .$$

Difference between Ω- & ω-?

Ω-notation denotes a lower bound that may or may not be asymptotically tight

ω-notation denotes an **lower bound that is NOT asymptotically tight**

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

ω-notation examples

$n^2 \lg n = \omega(n^2)$

$n^2 \neq \omega(n^2)$

$n^2/1000 \neq \omega(n^2)$

Comparisons of functions

Analogy of notations (a way to compare "sizes" of functions)

$O \approx \leq$

$\Omega \approx \geq$

$\Theta \approx =$

$o \approx <$

$\omega \approx >$

Another look of Insertion Sort

Best-case running time: T(n): an + b

Worst-case running time: T(n): an^2 + bn + c

belongs to both Ω(n) and O(n^2), and these bounds are asymptotically as tight as possible

O(n^2): bound on worst-case running time, and this also applies to its running time <u>on every input</u>

Θ(n^2): bound on worst-case running time, but this does <u>not imply a Θ(n^2) bound on its running time on every input</u>

Standard notations and common functions

See textbook (page 54-60) and notes

- Monotonicity
- Exponentials
- Logarithms
- Factorials

Logarithms and Exponents

1. $\log_b ac = \log_b a + \log_b c$
2. $\log_b (a / c) = \log_b a - \log_b c$
3. $\log_b (a^c) = c \log_b a$
4. $\log_b a = \log_c a / \log_c b$
5. $b^{\log_c a} = a^{\log_c b}$
6. $(b^a)^c = b^{ac}$
7. $b^a b^c = b^{a+c}$
8. $b^a / b^c = b^{a-c}$
9. $a^{-1} = \dfrac{1}{a}$

Standard notations and common functions

iterated logarithm function, lg*n (read “lg star of n”), the number of logarithms to make the result to be 1

lg*2 = 1

lg*4 = 2

lg*16 = 3

lg*65536 = lg* (2^{16}) = 4

lg*(2^{65536}) = 5

iterated logarithm is a VERY slow growing function