CS581 -- Algorithms

Spring 2014

Prof. Lynne E. Parker

9-Jan-2014

Reading Assignments

- Today's class:
  - Chapter 1, Chapter 3

- Reading assignment for next class:
  - Chapter 2, 4.0, 4.4

Asymptotic Complexity

- Running time of an algorithm as a function of input size *n* **for large *n***.
- Expressed using only the **highest-order term** in the expression for the exact running time.
  - Instead of exact running time, say $\Theta(n^2)$.
- Describes behavior of function in the limit.
- Written using ***Asymptotic Notation.***

Asymptotic Notation

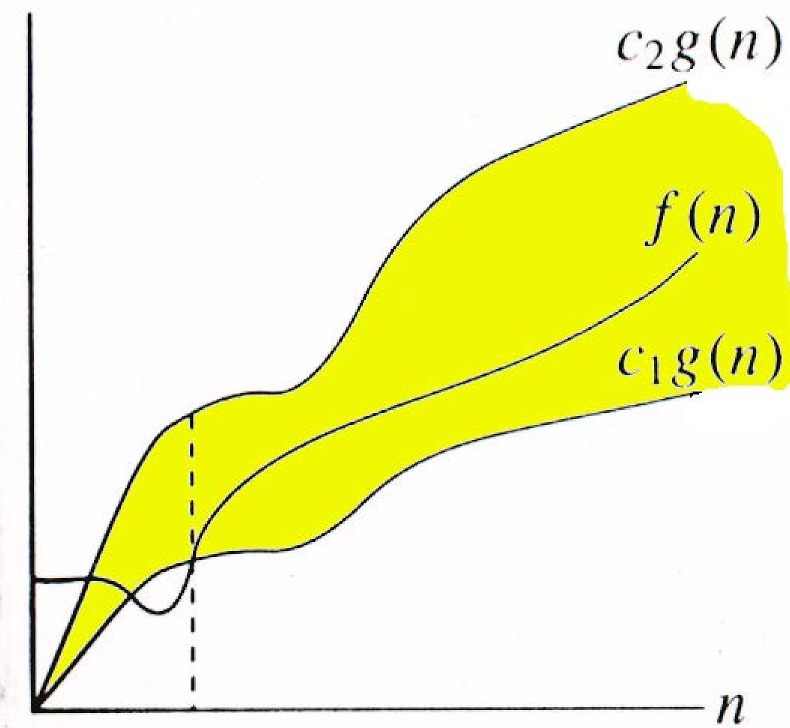
- T(n) = worst case run time, defined on integers
- **Θ, *O*, Ω, *o*, ω**
- Defined for functions over the natural numbers.
  - **Ex:** $f(n) = \Theta(n^2)$.
  - Describes how $f(n)$ grows in comparison to n^2.
- Define a ***set*** of functions; in practice used to compare two function sizes.
- The notations describe different rate-of-growth relations between the defining function and the defined set of functions.

$\theta(g(n))$ (Tight Bound)

$\Theta(g(n))$ = {$f(n)$: $\exists$ positive constants c_1, c_2, and n_0, such that $\forall n \geq n_0$, $0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n)$}

We write: $f(n) = \theta(g(n))$

(**not** $f(n) \in \theta(g(n))$)



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

Intuitively: Set of all functions that have the same *rate of growth* as $g(n)$.

$f(n)$ and $g(n)$ are nonnegative, for large n.

Example

$\Theta(g(n)) = \{f(n) :$
$\exists$ positive constants c_1, c_2, and
n_0, such that $\forall n \geq n_0$,
$0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n)\}$

$$f(n) = \tfrac{1}{2}n^2 - 3n = \theta(n^2)$$

Find c_1, c_2, n_0 that makes this true:

“Eye-balling” order of growth

- Look at leading term
- Ignore constants

- E.g., $n^2/2 - 3n = \theta(n^2)$

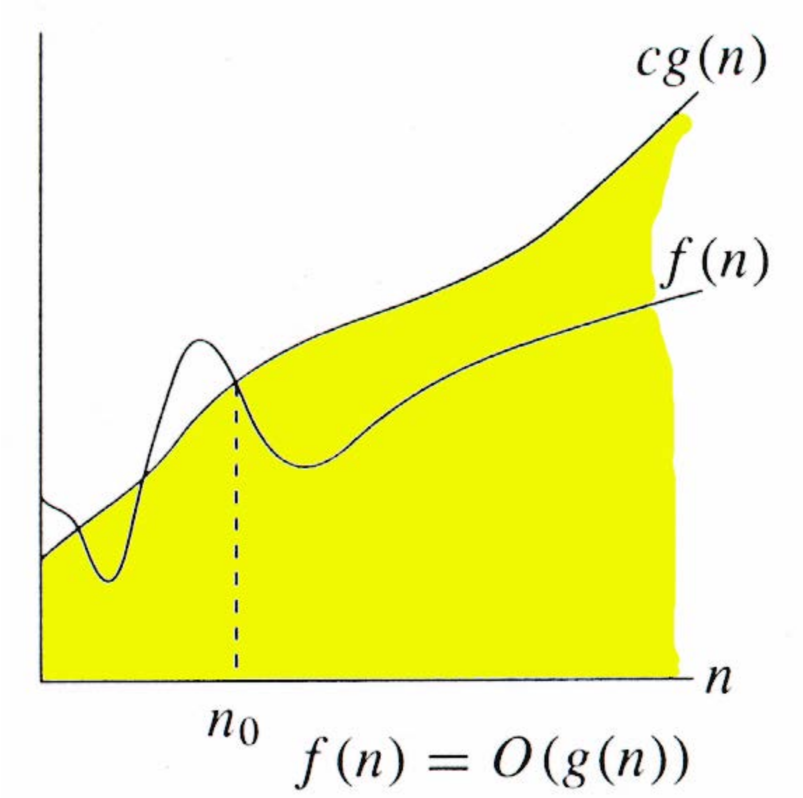
- Is $3n^3 = \theta(n^4)$?

$O\big(g(n)\big)$ (Upper Bound)

$O(g(n)) = \{f(n) :\ \exists$ positive constants c and n_0, such that $\forall n \geq n_0$, we have $0 \leq f(n) \leq cg(n)\ \}$

We write: $f(n) = O\big(g(n)\big)$

Intuitively: Set of all functions whose *rate of growth* is the same as or lower than that of $g(n)$.

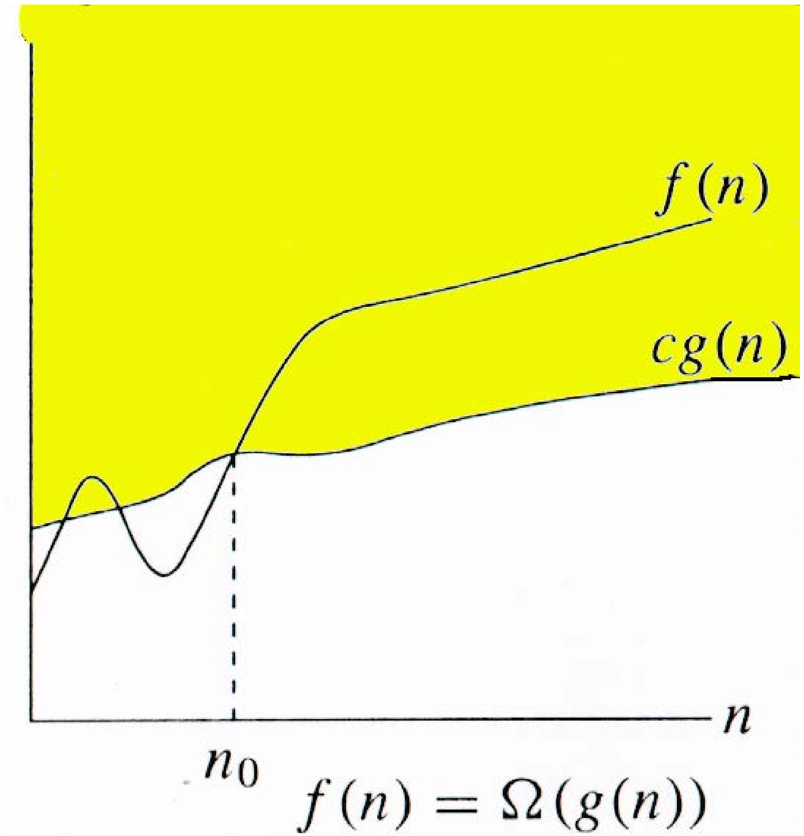


$\Omega\big(g(n)\big)$ (Lower Bound)

$\Omega(g(n))$ = $\{f(n)$: $\exists$ positive constants c and n_0, such that $\forall n \geq n_0$, we have $0 \leq cg(n) \leq f(n)\}$

We write: $f(n) = \Omega\big(g(n)\big)$

Intuitively: Set of all functions whose *rate of growth* is the same as or higher than that of $g(n)$.

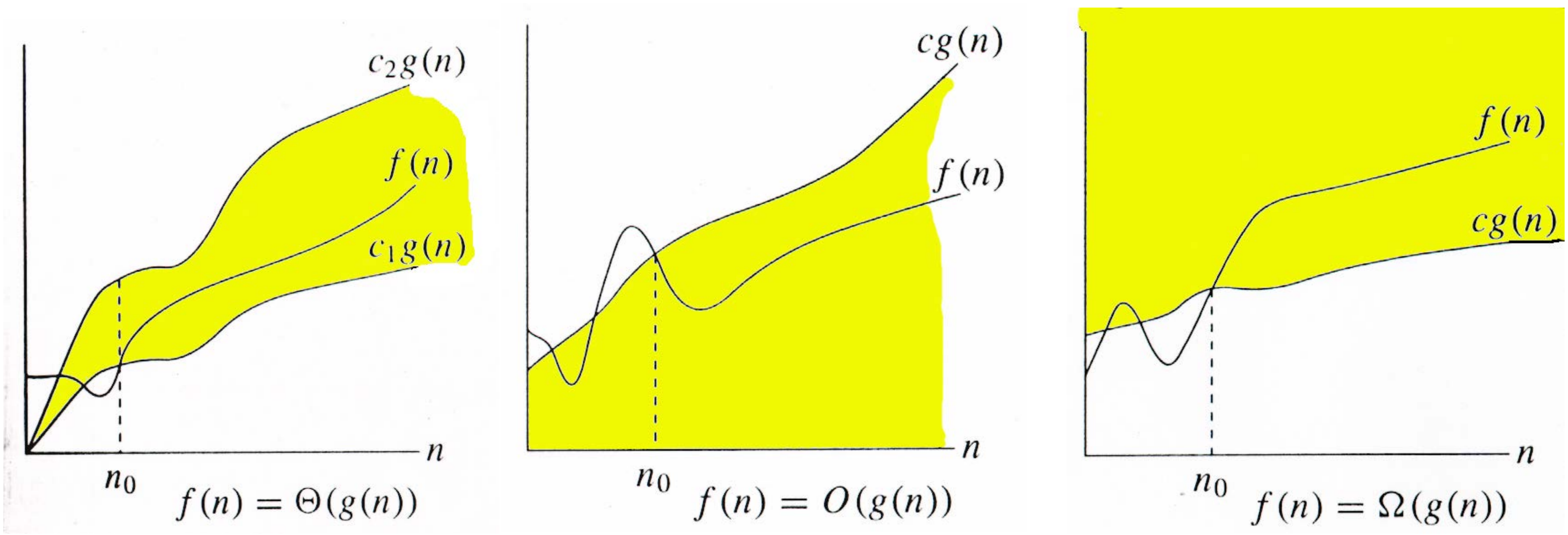


Comparing Θ, Ω, *O*

Theorem : For any two functions $g(n)$ and $f(n)$,

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

$f(n) = O(g(n))$ and $f(n) = \Omega(g(n))$.



o: Non-asymptotic tight bound

$\boldsymbol{o(g(n))} = \{f(n)$: $\forall$ $\boldsymbol{c > 0}$, $\exists$ $\boldsymbol{n_0 > 0}$ such that $\forall$ $n \geq n_0$, we have $0 \leq f(n) < cg(n)\}$.

Note that $f(n) = o\big(g(n)\big) \Rightarrow \lim_{n \to \infty} \frac{f(n)}{g(n)} = 0$

ω : Non-asymptotic lower bound

$\omega(g(n))$ = $\{f(n)$: $\forall$ **$c > 0$**, $\exists$ **$n_0 > 0$** such that
$\forall\ n \geq n_0$, we have $0 \leq cg(n) < f(n)\}$.

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

Limits

- $f(n) = o\big(g(n)\big) \Rightarrow \lim_{n\to\infty} \frac{f(n)}{g(n)} = 0$
- $f(n) = \omega\big(g(n)\big) \Rightarrow \lim_{n\to\infty} \frac{f(n)}{g(n)} = \infty$

- $f(n) = \theta\big(g(n)\big) \Rightarrow 0 < \lim_{n\to\infty} \frac{f(n)}{g(n)} < \infty$
- $f(n) = O\big(g(n)\big) \Rightarrow \lim_{n\to\infty} \frac{f(n)}{g(n)} < \infty$
- $f(n) = \Omega\big(g(n)\big) \Rightarrow 0 < \lim_{n\to\infty} \frac{f(n)}{g(n)}$

Running Times

- "Running time is $O(f(n))$" $\Rightarrow$ Worst case is $O(f(n))$
- $O(f(n))$ bound on the worst-case running time $\Rightarrow$ $O(f(n))$ bound on the running time of every input.
- $\Theta(f(n))$ bound on the worst-case running time $\not\Rightarrow$ $\Theta(f(n))$ bound on the running time of every input.
- "Running time is $\Omega(f(n))$" $\Rightarrow$ Best case is $\Omega(f(n))$
- Can still say "Worst-case running time is $\Omega(f(n))$"
  - Means worst-case running time is given by some unspecified function $g(n) \in \Omega(f(n))$.

Asymptotic Notation in Equations

- Can use asymptotic notation in equations to replace expressions containing lower-order terms.
- For example,

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

  $= 4n^3 + \Theta(n^2) = \Theta(n^3)$. **How to interpret?**
- In equations, $\Theta(f(n))$ always stands for an ***anonymous function*** $g(n) \in \Theta(f(n))$
  - In the example above, $\Theta(n^2)$ stands for $3n^2 + 2n + 1$.

Relational Properties

- Transitivity:

$$f(n) = \theta\big(g(n)\big) \text{ and } g(n) = \theta\big(h(n)\big)$$
$$\Rightarrow f(n) = \theta(h(n))$$

(similarly for Ω, O, ω, o)

Relational Properties

- Reflexivity:

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

Relational Properties

- Symmetry:

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

- Transpose Symmetry:

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

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

Monotonicity

- $f(n)$ is
  - **monotonically increasing** if $m \leq n \Rightarrow f(m) \leq f(n)$.
  - **monotonically decreasing** if $m \geq n \Rightarrow f(m) \geq f(n)$.
  - **strictly increasing** if $m < n \Rightarrow f(m) < f(n)$.
  - **strictly decreasing** if $m > n \Rightarrow f(m) > f(n)$.

Example

- True or false?

  For 2 functions $f(n)$ and $g(n)$, either $f(n) = O(g(n))$ or $f(n) = \Omega(g(n))$.

Example

- Let:

$$f(n) = n^3 \lg^4 n$$
$$g(n) = n^4 \lg^3 n$$
$$h(n) = n^5 / \lg n$$

We also state the following mathematical property:

$\lim_{n \to \infty} \frac{\lg^b n}{n^a} = 0$, for any real constants $a > 0$ and b.

True or false?

- $f(n) \in O(g(n))$
- $h(n) \in O(g(n))$
- $f(n) \in \Theta(g(n))$
- $g(n) \in \omega(f(n))$
- $h(n) \in o(f(n)$

Exponentials

- Useful Identities:

$$a^{-1} = \frac{1}{a}$$

$$(a^m)^n = a^{mn}$$

$$a^m a^n = a^{m+n}$$

- Exponentials and polynomials

$$\lim_{n\to\infty} \frac{n^b}{a^n} = 0$$

$$\Rightarrow n^b = o(a^n)$$

Logarithms

$x = \log_b a$ is the exponent for $a = b^x$.

Natural log: $\ln a = \log_e a$

Binary log: $\lg a = \log_2 a$

$\lg^2 a = (\lg a)^2$

$\lg \lg a = \lg (\lg a)$

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

$$\log_c (ab) = \log_c a + \log_c b$$

$$\log_b a^n = n \log_b a$$

$$\log_b a = \frac{\log_c a}{\log_c b}$$

$$\log_b (1/a) = -\log_b a$$

$$\log_b a = \frac{1}{\log_a b}$$

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

Bases of logs and exponentials

- If the base of a logarithm is changed from one constant to another, the value is altered by a constant factor.
  - **Ex:** $\log_{10} n$ * **$\log_2 10$** = $\log_2 n$.
  - Base of logarithm is not an issue in asymptotic notation.

- Exponentials with different bases differ by a exponential factor (not a constant factor).
  - **Ex:** 2^n = **$(2/3)^n$**$*3^n$.

Polylogarithms

- **For $a \geq 0$, $b > 0$,** $\lim_{n\to\infty} (\lg^a n \,/\, n^b) = 0$, so $\lg^a n = o(n^b)$, and $n^b = \omega(\lg^a n)$
  – Prove using L'Hopital's rule repeatedly

- $\lg(n!) = \Theta(n \lg n)$
  – Prove using Stirling's approximation (in the text) for $\lg(n!)$.

Exercise

- Express functions in A in asymptotic notation using function in B.

A	B
$5n^2 + 100n$	$3n^2 + 2$
$\log_3(n^2)$	$\log_2(n^3)$
$n^{\lg 4}$	$3^{\lg n}$
$\lg^2 n$	$n^{1/2}$

Remember Reading Assignments

- Today's class:
  - Chapter 1, Chapter 3
- Reading assignment for next class:
  - Chapter 2, 4.0, 4.4