UNIT I INTRODUCTION:

Introduction to Algorithms – Fundamentals of Algorithmic Problem Solving – Fundamentals of the Analysis of Algorithmic Efficiency – Analysis Framework – Asymptotic Notations and Basic Efficiency Classes – Mathematical Analysis of Recursive Algorithms – Mathematical Analysis of Non-recursive Algorithms

Asymptotic Notations and Basic Efficiency Classes

A way of comparing functions that ignores constant factors and small input sizes (because?)

- O(*g*(*n*)): class of functions *f*(*n*) that grow no faster than *g*(*n*)
- Θ(*g*(*n*)): class of functions *f*(*n*) that grow at same rate as *g*(*n*)
- Ω(*g*(*n*)): class of functions *f*(*n*) that grow at least as fast as *g*(*n*)

Big-oh

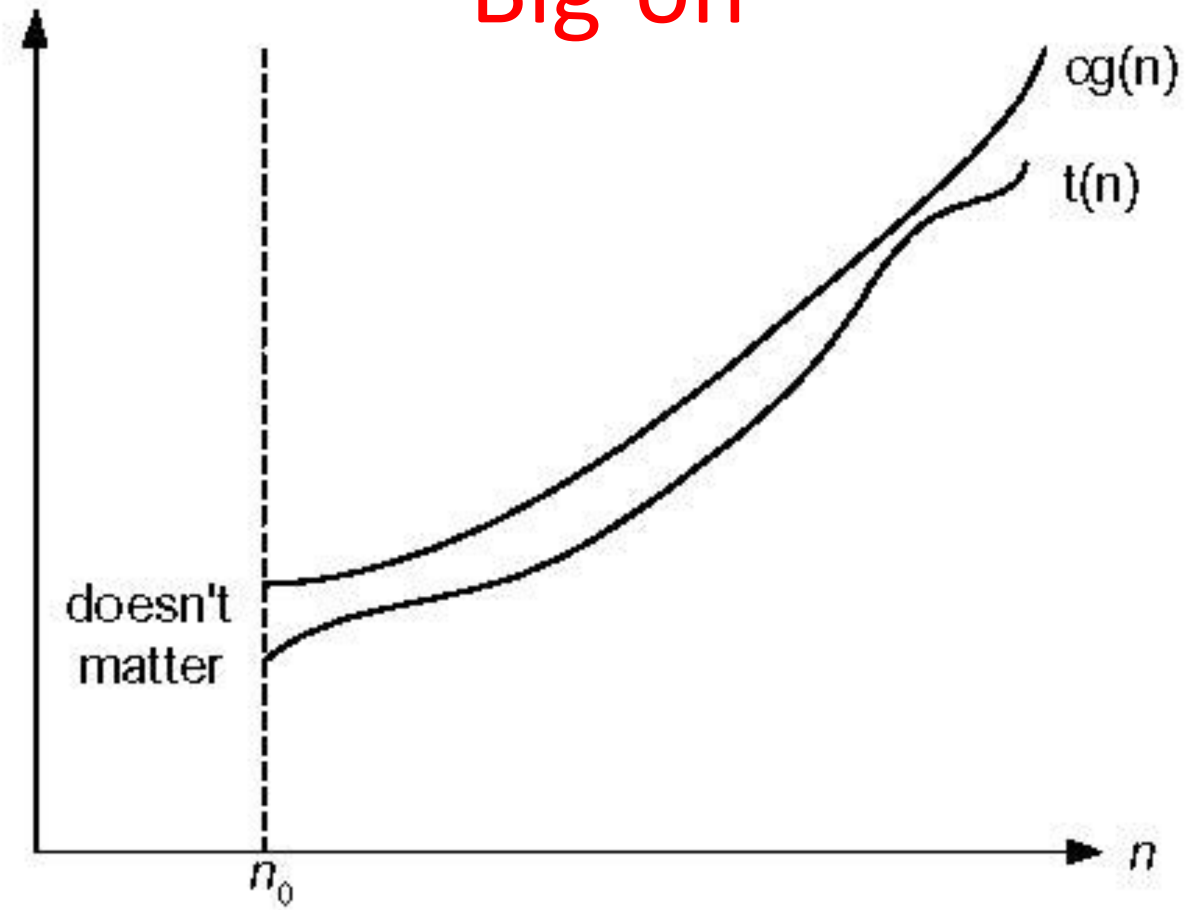


Figure 2.1 Big-oh notation: $t(n) \in O(g(n))$

Big-omega

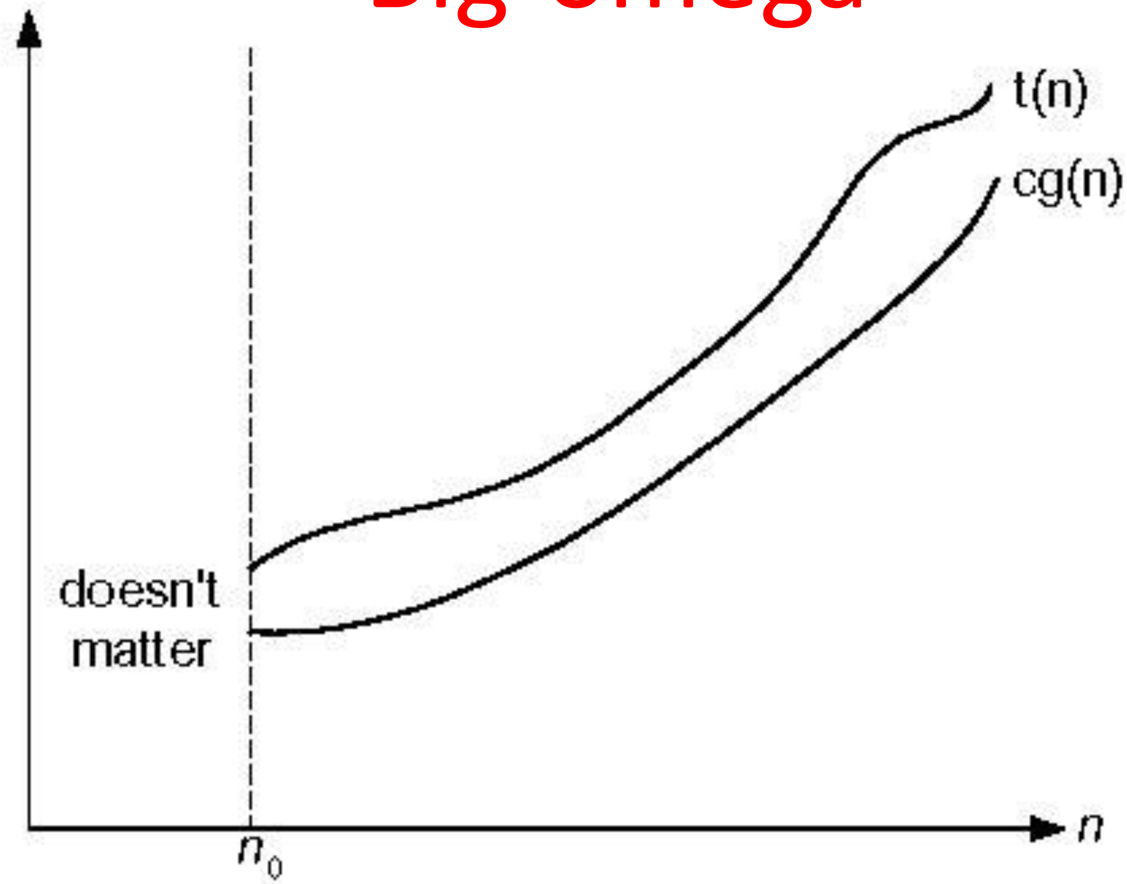


Fig. 2.2 Big-omega notation: $t(n) \in \Omega(g(n))$

Big-theta

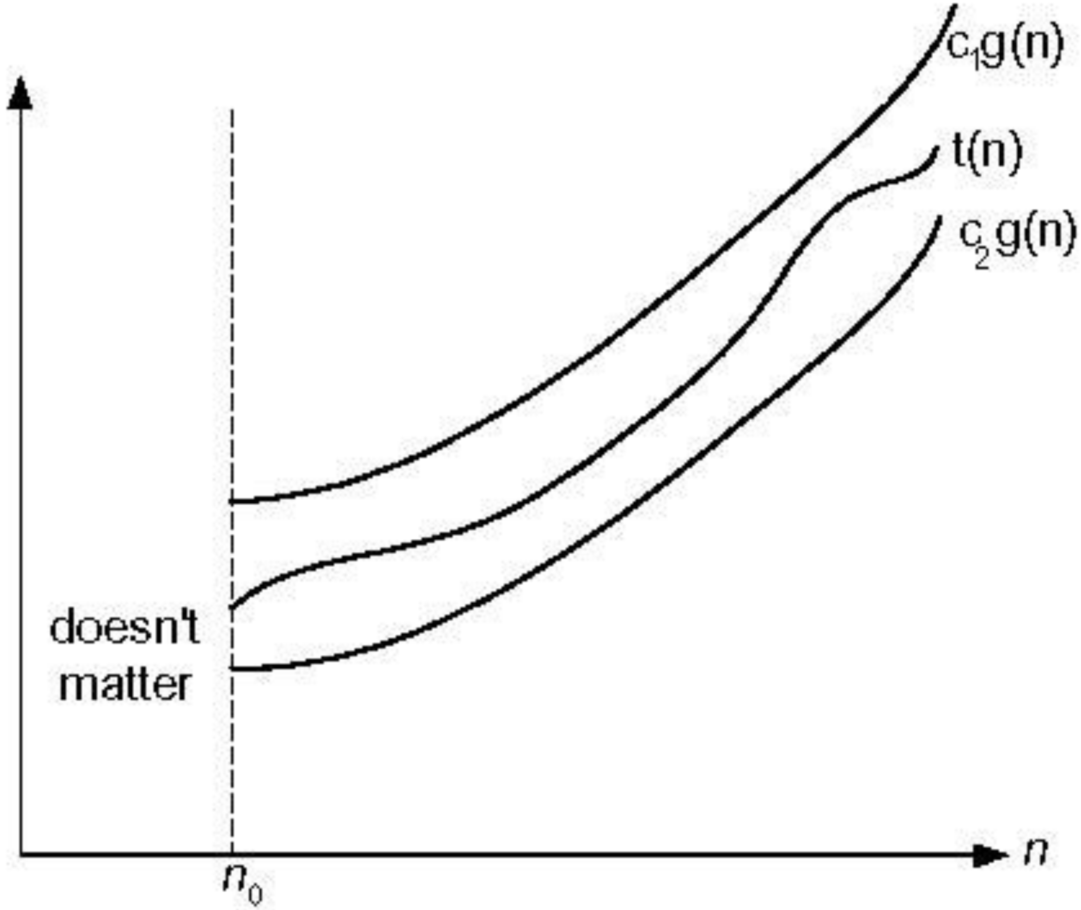


Figure 2.3 Big-theta notation: $t(n) \in \Theta(g(n))$

O-notation

Definition: $f(n)$ is in $O(g(n))$ if order of growth of $f(n) \leq$ order of growth of $g(n)$ (within constant multiple), i.e., there exist positive constant c and non-negative integer n_0 such that

$$f(n) \leq c\, g(n) \text{ for every } n \geq n_0$$

Examples:

- $10n$ is in $O(n^2)$

- $5n+20$ is in $O(n)$

Ω-notation

- Formal definition
  - A function $t(n)$ is said to be in $\Omega(g(n))$, denoted $t(n) \in \Omega(g(n))$, if $t(n)$ is bounded below by some constant multiple of $g(n)$ for all large n, i.e., if there exist some positive constant c and some nonnegative integer n_0 such that

    $t(n) \geq cg(n)$ for all $n \geq n_0$

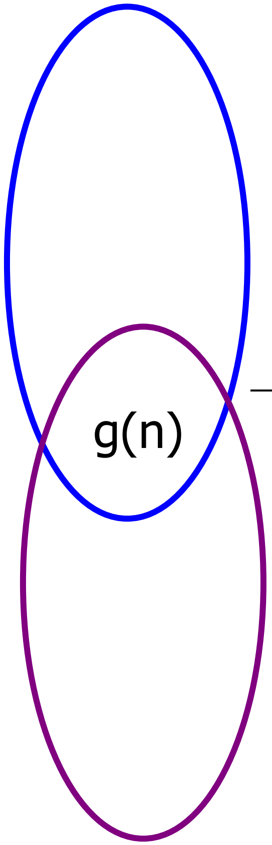
- Exercises: prove the following using the above definition
  - $10n^2 \in \Omega(n^2)$
  - $0.3n^2 - 2n \in \Omega(n^2)$
  - $10n^3 \in \Omega(n^2)$

Θ-notation

- Formal definition
  - A function *t(n)* is said to be in $\Theta(g(n))$, denoted $t(n) \in \Theta(g(n))$, if *t(n)* is bounded both above and below by some positive constant multiples of *g(n)* for all large *n*, i.e., if there exist some positive constant c_1 and c_2 and some nonnegative integer n_0 such that

    $c_2\, g(n) \leq t(n) \leq c_1\, g(n)$ for all $n \geq n_0$

- Exercises: prove the following using the above definition
  - $10n^2 \in \Theta(n^2)$
  - $0.3n^2 - 2n \in \Theta(n^2)$
  - $(1/2)n(n-1) \in \Theta(n^2)$



\>=

Ω(g(n)), functions that grow at least as fast as g(n)

=

Θ(g(n)), functions that grow at the same rate as g(n)

<=

O(g(n)), functions that grow no faster than g(n)

Theorem

- If $t_1(n) \in O(g_1(n))$ and $t_2(n) \in O(g_2(n))$, then $t_1(n) + t_2(n) \in O(max\{g_1(n), g_2(n)\})$.
  - The analogous assertions are true for the Ω-notation and Θ-notation.

- Implication: The algorithm's overall efficiency will be determined by the part with a larger order of growth, i.e., its least efficient part.
  - For example, $5n^2 + 3nlogn \in O(n^2)$

Proof. There exist constants *c1, c2, n1, n2* such that

$t1(n) \le c1*g1(n)$, **for all** $n \ge n1$

$t2(n) \le c2*g2(n)$, **for all** $n \ge n2$

Define $c3 = c1 + c2$ **and** $n3 = max\{n1,n2\}$. ***Then***

$t1(n) + t2(n) \le c3*max\{g1(n), g2(n)\}$, **for all** $n \ge n3$

Some properties of asymptotic order of growth

- $f(n) \in O(f(n))$
- $f(n) \in O(g(n))$ iff $g(n) \in \Omega(f(\mathrm{n}))$
- If $f(n) \in O(g(n))$ and $g(n) \in O(h(n))$, then $f(n) \in O(h(n))$

  Note similarity with $a \leq \mathrm{b}$
- If $f_1(n) \in O(g_1(n))$ and $f_2(n) \in O(g_2(n))$, then
  $$f_1(n) + f_2(n) \in O(\max\{g_1(n), g_2(n)\})$$

Exercise: Can you prove these properties?

Establishing order of growth using limits

$$\lim_{n\to\infty} T(n)/g(n) = \begin{cases} \mathbf{0} & \text{order of growth of } \boldsymbol{T(n)} < \text{ order of growth of } \boldsymbol{g(n)} \\ \boldsymbol{c > 0} & \text{order of growth of } \boldsymbol{T(n)} = \text{ order of growth of } \boldsymbol{g(n)} \\ \boldsymbol{\infty} & \text{order of growth of } \boldsymbol{T(n)} > \text{ order of growth of } \boldsymbol{g(n)} \end{cases}$$

Examples:

- **10*n*** **vs.** n^2
- $n(n+1)/2$ **vs.** n^2

Basic asymptotic efficiency classes

1	constant
$\log n$	logarithmic
n	linear
$n \log n$	n-log-n
n^2	quadratic
n^3	cubic
2^n	exponential
$n!$	factorial

Thank You