Design and Analysis of Algorithms

L05: Complexity Analysis

Dr. Ram P Rustagi
Sem IV (2020-Even)
Dept of CSE, KSIT
rprustagi@ksit.edu.in

Resources

- Text Book I: Levitin
- Text Book II: Horowitz

Asymptotic Notation

- Focus of analysis framework
  - Order of growth of time complexity function
- Notation
  - `C(n)`: Count of basic operations of an algorithm
  - `g(n)`: Some simple function for comparison purpose
    - A non-negative function
  - `T(n)`: running time of algorithm indicated by C(n)
    - A non-negative function

Order of Running by a Person

- Order of running of a general person:
  - O(*Bolt*):
  - Ω(*Tortoise*)
  - Θ(*Common man*)

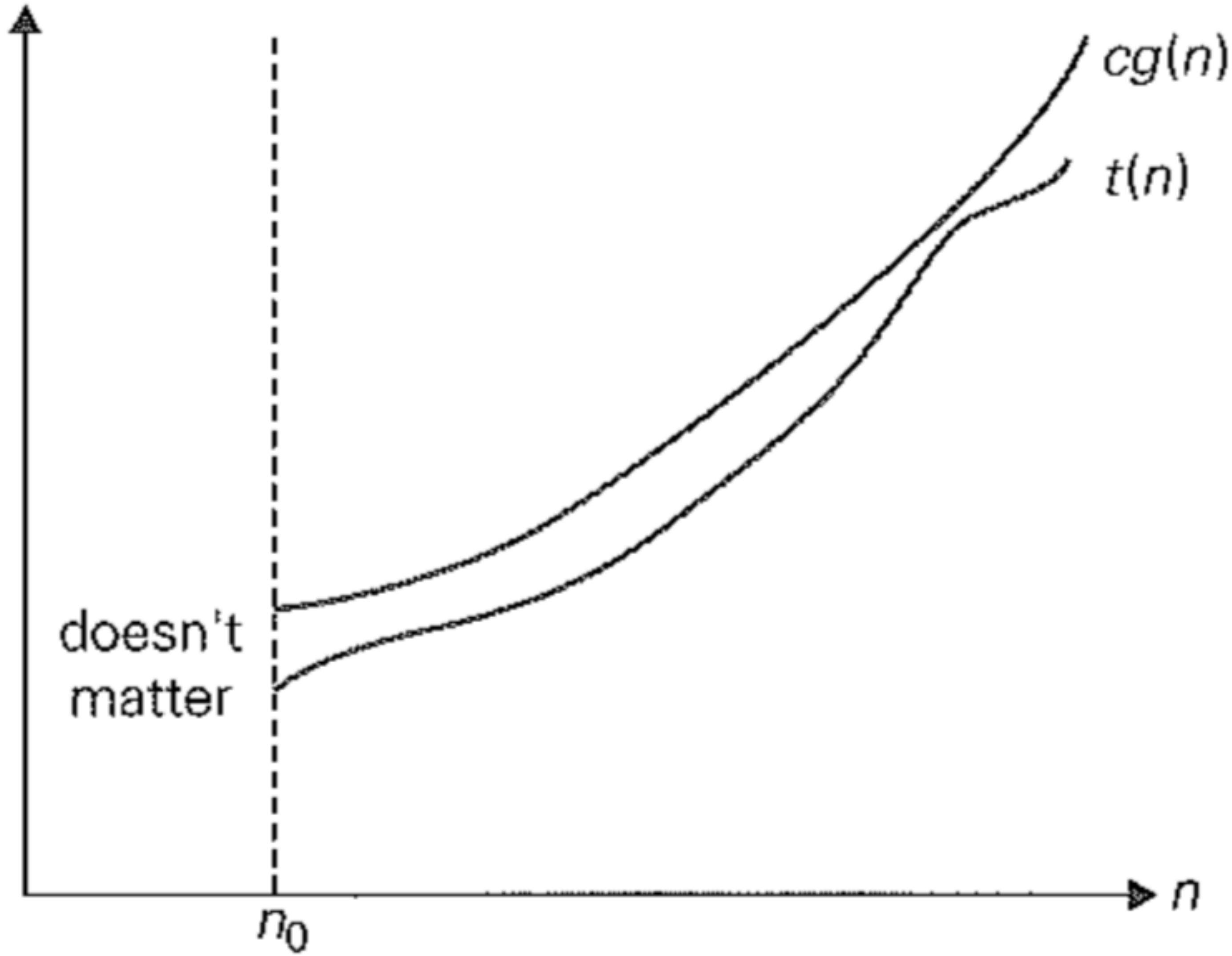
Asymptotic Notation

- 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 rat**e as $g(n)$
  - $\Omega(g(n))$: class of functions $f(n)$ that grow **at least as fast** as $g(n)$

Asymptotic Notation: Big-Oh

- $O(g(n))$: set of all functions with a smaller or same order of growth as $g(n)$
  - $n \in O(n^2)$
  - $100n+5 \in O(n^2)$
  - $n(n+1)/2 \in O(n^2)$
  - $n^3 \notin O(n^2)$
  - $0.000001n^3 \notin O(n^2)$
  - $n^4+n^2+c \notin O(n^2)$
- A function $t(n)$ is said to be in $O(g(n))$ if $t(n)$ is bounded above by some +ve constant multiple of $g(n)$ for large n, i.e. $t(n) \in O(g(n))$, if
  $t(n) \le cg(n)$ for all $n \ge n_0$

Big-Oh

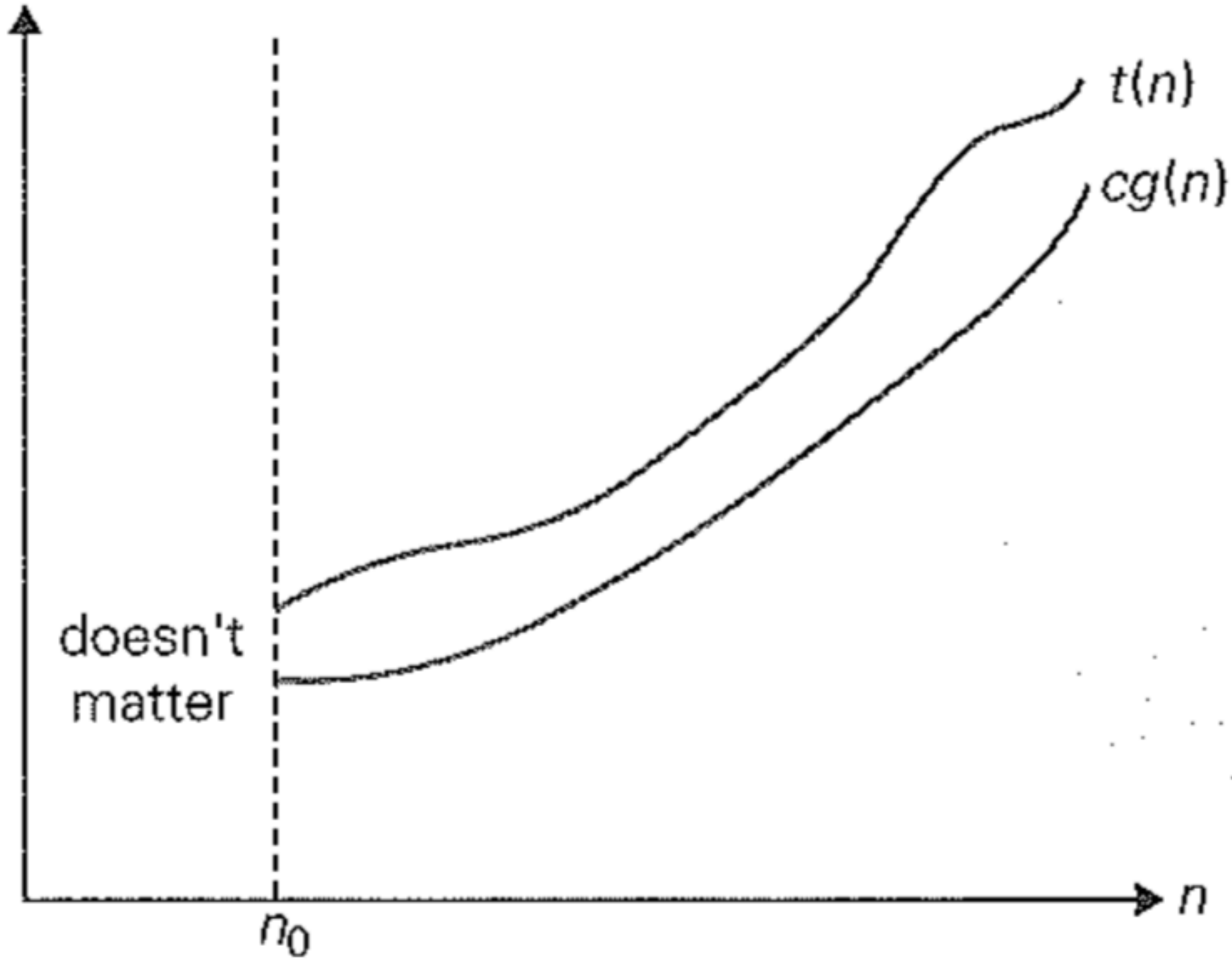


Big-Oh notation: $t(n) \in O(g(n))$

Asymptotic Notation: Ω

- $\Omega(g(n))$: set of all functions with a larger or same order of growth as $g(n)$
  - $n \notin \Omega(n^2)$
  - $100n+5 \notin \Omega(n^2)$
  - $n(n+1)/2 \in \Omega(n^2)$
  - $n^3 \in \Omega(n^2)$
  - $0.000001n^3 \in \Omega(n^2)$
  - $n^4+n^2+c \in \Omega(n^2)$
- A function $t(n)$ is said to be in $\Omega(g(n))$ if $t(n)$ is bounded below by some +ve constant multiple of $g(n)$ for large n, i.e. $t(n) \in \Omega(g(n))$, if
  $t(n) \geq cg(n)$ for all $n \geq n_0$

Big Omega Notation



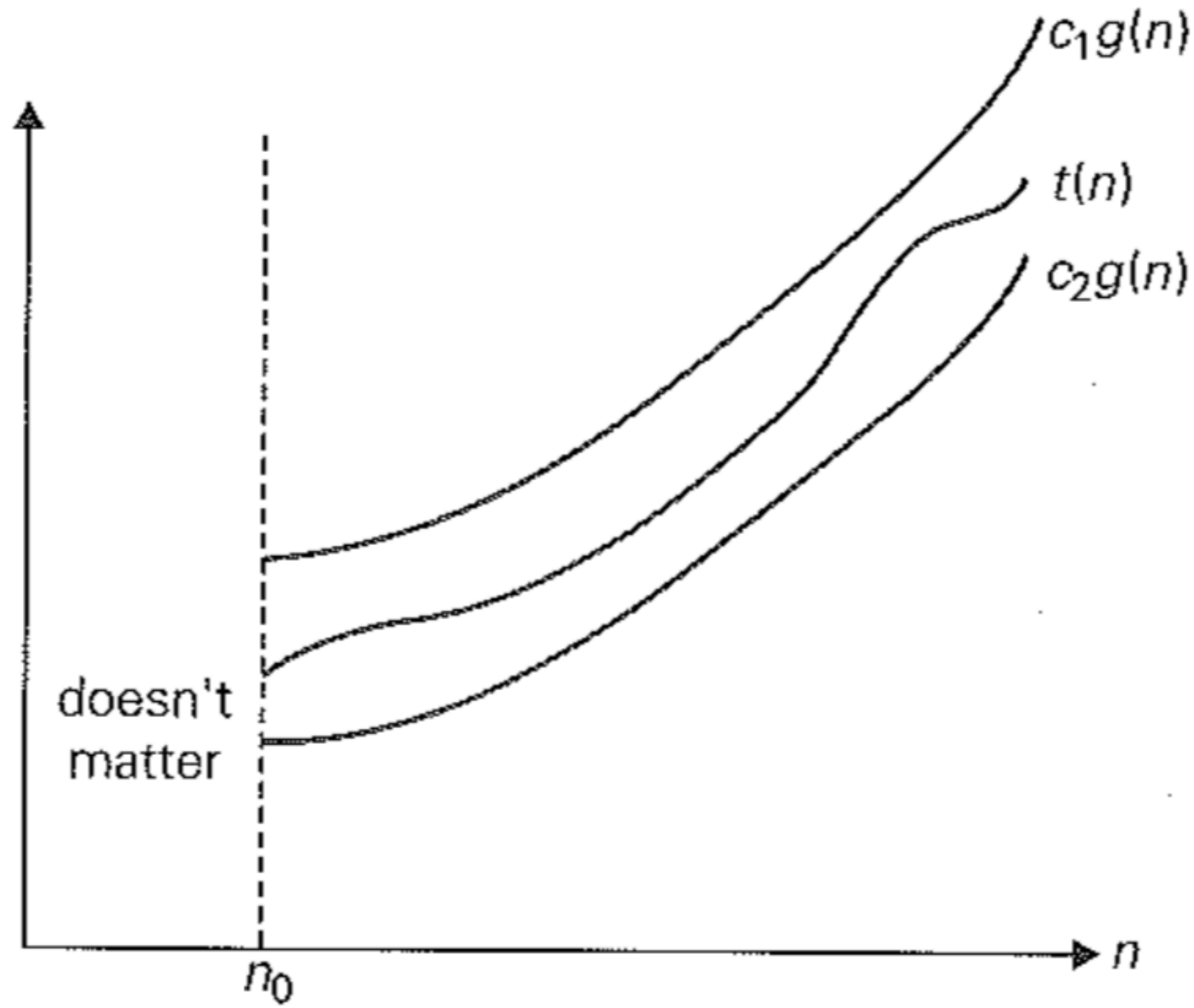
Big omega notation: $t(n) \in \Omega(g(n))$

Asymptotic Notation: Theta Θ

- $\Theta(g(n))$: set of all functions with a similar order of growth as $g(n)$ within a constant multiple
  - $n \notin \Theta(n^2)$
  - $100n+5 \notin \Theta(n^2)$
  - $n(n+1)/2 \in \Theta(n^2)$
  - $0.000001n^3 \notin \Theta(n^2)$
  - $n^4+n^2+c \notin \Theta(n^2)$
- A function $t(n)$ is said to be in $\Theta(g(n))$ if $t(n)$ is bounded both above and below by some +ve constant multiple of $g(n)$ for all large n,
  - i.e. $t(n) \in \Theta(g(n))$, if

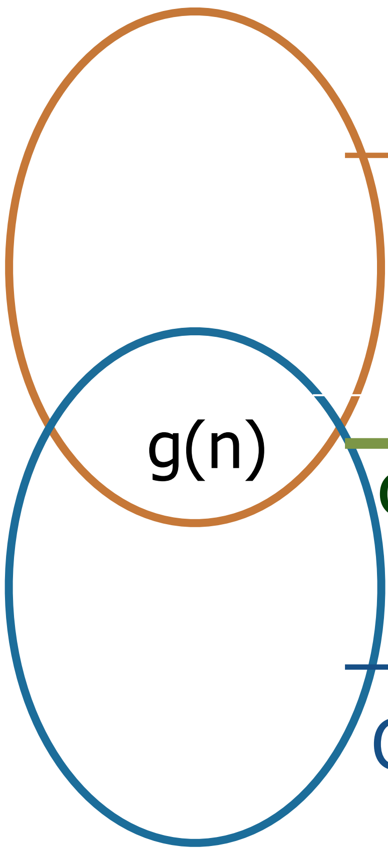
    $c_2 g(n) \leq t(n) \leq c_1 g(n)$ for all $n \geq n_0$

Big Theta Notation



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

Asymptotic Notation View



$>=$

$\Omega(g(n))$, functions that grow <u>at least as fast as</u> g(n)

$=$

$\Theta(g(n))$, functions that grow <u>at the same rate as</u> g(n)

$<=$

$O(g(n))$, functions that grow <u>no faster than</u> 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, e.g.
    - $5n^2 + 3n\log n \in O(n^2)$
- Some properties
    - $f(n) \in O(f(n))$
    - $f(n) \in O(g(n))$ iff $g(n) \in \Omega(f(n))$
    - *If* $f(n) \in O(g(n))$ *and* $g(n) \in O(h(n))$, *then* $f(n) \in O(h(n))$
    - $\Sigma_{1\le i\le n} \Theta(f(i)) = \Theta(\Sigma_{1\le i\le n} f(i))$

Exercises

- Compare the order of growth of
  1. n^2 and $n(n-1)/2$
  2. $\log_2 n$ and $\sqrt{n}$
  3. $n!$ and 2^n

Basic Asymptotic Efficiency Classes

- 1 Constant
- log n Logarithmic
- n Linear
- nlog n n-log-n
- n^2 Quadratic
- n^3 Cubic
- 2^n Exponential
- n! Factorial

Useful Summation Formulas

Expression	function
$\Sigma_{1\le i\le n} 1 = 1+1+\ldots+1$ (n times)	$n \in \Theta(n)$
$\Sigma_{1\le i\le n}\, i = 1+2+\ldots+n = n(n+1)/2$	$n^2/2 \in \Theta(n^2)$
$\Sigma_{1\le i\le n}\, i^2 = 1^2+2^2+\ldots+n^2$	$n(n+1)(2n+1)/6$
$\Sigma_{0\le i\le n} 2^i = 2^0 + 2^1 + \ldots + 2^n$	$2^{n+1}-1 \in \Theta(2^n)$
$\Sigma_{0\le i\le n}\, a^i = 1+a+\ldots+a^n$	$(a^{n+1}-1)/(a-1)$
$\Sigma(a_i \pm b_i)$	$\Sigma a_i \pm \Sigma b_i$
$\Sigma c a_i$	$c\Sigma a_i$
$\Sigma_{1\le i\le m} a_i + \Sigma_{m+1\le i\le u} a_i$	$\Sigma_{1\le i\le u} a_i$

Summary

- Order of growth
  - Big-Oh
  - Big-Omega
  - Big-Theta