2 GROWTH RATES AND ASYMPTOTIC NOTATIONS

Goal:

- Categorize algorithms based on their asymptotic growth rate.
- Ignore (small) constant of proportionality & small inputs
- Estimate upper bound on growth rate of time complexity function

def1: (For arbitrary functions) $f(n) \in O(g(n))$ if there exist two positive constants c and n_0 such that

$$|f(n)| \leq c|g(n)|$$

for all $n \geq n_0$.

eg.

1. $f(n) = 5n \;\; g(n) = n^2$
   $5n_0 \leq cn_0^2$
   $\Rightarrow 5 \leq cn_0$
   $\Rightarrow n_0 = 5, c = 1$
   Thus, $5n = O(n^2)$
2. $f(n) = 10^6 n^2 \;\; g(n) = n^2$
   To prove: $(10^6 n^2) \leq cn^2$ for $n \geq n_0$
   choose $C = 10^6$
   Then $n_0 = 1$
   $\Rightarrow (10^6 n^2) = O(n^2)$

3. $A(n) = a_m n^m + a_{m-1} n^{m-1} + \ldots + a_1 n + a_0$

a polynominal of degree m, $n, m \geq 0$ for all i, $a_i \in R$

To Prove: $A(n) = O(n^m) = a_m n^m + O(n^{m-1})$

Example: $5n^3 + 2n^2 - 5 \;=\; O(n^3) \;=\; 5n^3 + O(n^2)$

Proof:

To Prove: $|A(n)| \;\leq\; c|n^m|$

$|A(n)| \leq |a_m| n^m + |a_{m-1}| n^{m-1} + \ldots + |a_1| n + |a_0|$

$\leq n^m (|a_m| + \frac{|a_{m-1}|}{n} + \ldots + \frac{|a_1|}{n^{m-1}} + \frac{|a_0|}{n^m})$

$\leq n^m (|a_m| + |a_{m-1}| + \ldots + |a_0|)$

$c = |a_m| + |a_{m-1}| + \cdots + |a_0|$

$n_0 = 1$

Example: $5n^3 + 2n^2 - 5 \;=\; O(n^3)$
c $= 5 + 2 + |-5| = 12$

$5n^3 + 2n^2 - 5 \;\leq\; 12n^3$
for all $n \geq 1$.

2.1 ORDER NOTATION FOR POSITVE FUNCTIONS

def2: Let

$$f : N \to R^*$$

$$g : N \to R^*$$

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

, ie.,

$$g(n) \in O(f(n))$$

,
if for some $c \in R^+$ and some $n_0 \in N$

$$g(n) \leq cf(n)$$

for all $n \geq n_0$, where
$R^* = R^+ \cup \{0\}$,
R^+ =set of positive reals
$N = \{0, 1, 2, 3, \ldots\}$.

def3: $f \in O(g)$ if $\lim_{n\to\infty} \frac{f(n)}{g(n)} \to c$ for some $c \in R^*$.

Examples:

1. $10^6 n^2 = O(n^2)$
   $\lim_{n\to\infty} \frac{10^6 n^2}{n^2} \to 10^6$

2. $5n \in O(n^2)$
   $\lim_{n\to\infty} \frac{5n}{n^2} \to 0$

3. Prove that $2^n \neq O(n^m)$ for any integer m.

   (By Contradiction:) Suppose $2^n = O(n^m)$

   $\lim_{n\to\infty} \frac{2^n}{n^m} = \lim_{n\to\infty} \frac{2^n \log 2}{mn^{m-1}}$

   $= \lim_{n\to\infty} \frac{(\log 2)^2 \; 2^n}{m(m-1)n^{m-2}}$

   $= \lim_{n\to\infty} \frac{(\log 2)^m 2^n}{m(m-1)\cdots(2)1} \to \infty$

   Thus, polynomial algorithms are distinctly superior to exponential-time algorithms.

(Project should have polynomial algo's only)

2.2 Θ AND OTHER NOTATIONS

def: If $f(n) = O(g(n))$
then $g(n) = \Omega(f(n))$.

Thus, $O(f)$ contains all function with growth rate "equal or slower."

$\Omega(f)$ contains all function with growth rate "equal or faster."

def: If $f(n) = O(g(n))$ and $g(n) = O(f(n))$
then $f(n) = \Theta(g(n))$.

Equivalently, $\lim_{n\to\infty} \frac{f(n)}{g(n)} = c \in R^+$.

Thus,

$$\Theta(f)$$

$= O(f) \cap \Omega(f)$

In terms of growth rate,
$\Theta(f)$ contains functions growing at the same rate as f.

$O(f)$ contains function growing no faster than f

$\Omega(f)$ contains function at least as fast as f

def: (Small o and ω notations):

$f = o(g)$ if $\lim_{n\to\infty} f/g \to 0$

$f = \omega(g)$ if and only if $g \in o(f)$

2.3 $\sqrt(n)$ VERSUS $\log_2^3 n$

show $\sqrt{n} = \Omega(\log_2^3 n)$

$\Rightarrow \ln^3 n = O(\sqrt{n})$

$\frac{\ln^3 n}{\sqrt{n}} = \lim_{n\to\infty} \frac{3\log^2 n 1/n}{1/2n^{-1/2}}$

$= \lim_{n\to\infty} 6\frac{\log^2 n}{\sqrt{n}} = \lim_{n\to\infty} 6\frac{2\log n 1/n}{1/2n^{-1/2}}$

$= \lim_{n\to\infty} 24\frac{\log n}{\sqrt{n}} = \lim_{n\to\infty} 24\frac{1/n}{1/2n^{-1/2}}$

$= \lim_{n\to\infty} \frac{48}{\sqrt{n}} \to 0$

$\Rightarrow \log^3 n = O(\sqrt{n})$

3 H.W.1

- Use defn. 2 to show that $\sqrt{n} = \Omega(\log_2^3 n)$

- Q. 1-1, p. 13
- Q. 3-2, p. 58