Algorithm Design & Analysis I

Functions & Order of Growth

How would you compare and/or classify these functions?

$f(n) = 3n^2 + 4n + 1$ $g(n) = 5n^2$ $h(n) = 100n + 100$ $i(n) = \frac{1}{2}\, n^3$ $j(n) = \frac{1}{8}\, n^4$ $k(n) = 0.00000001(2^n)$

Key Idea: We compare and classify functions by focusing on *the way the function grows.*

Key Idea: We compare and classify functions by focusing on *the way the function grows.*

That means looking at the relationship between the input and the output–how the output grows as the input gets larger.

Comparing Functions

$f(n) = n^2 + 1$ vs. $g(n) = \frac{1}{3} n^3$

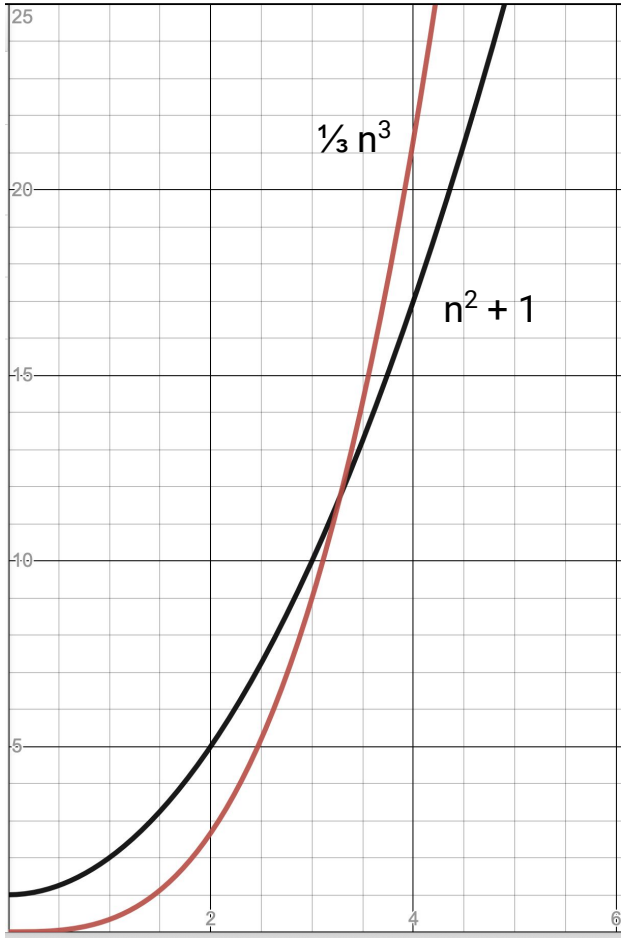
Comparing Functions: by specific values

$f(n) = n^2 + 1$ vs. $g(n) = \frac{1}{3} n^3$

n	f(n)	g(n)
0	1	0
1	2	⅓
2	5	2 ⅔
4	17	21 ⅓
8	65	170 ⅔
16	257	1365 ⅓

Comparing Functions: by graphing

$f(n) = n^2 + 1$ vs. $g(n) = \frac{1}{3} n^3$



Comparing Functions: by ratio

$f(n) = n^2 + 1$ vs. $g(n) = \frac{1}{3} n^3$

Comparing Functions: by specific values

$f(n) = n^2$ vs. $g(n) = 2n^2$

n	f(n)	g(n)
0	0	0
1	1	2
2	4	8
4	16	32
8	64	128
16	256	512

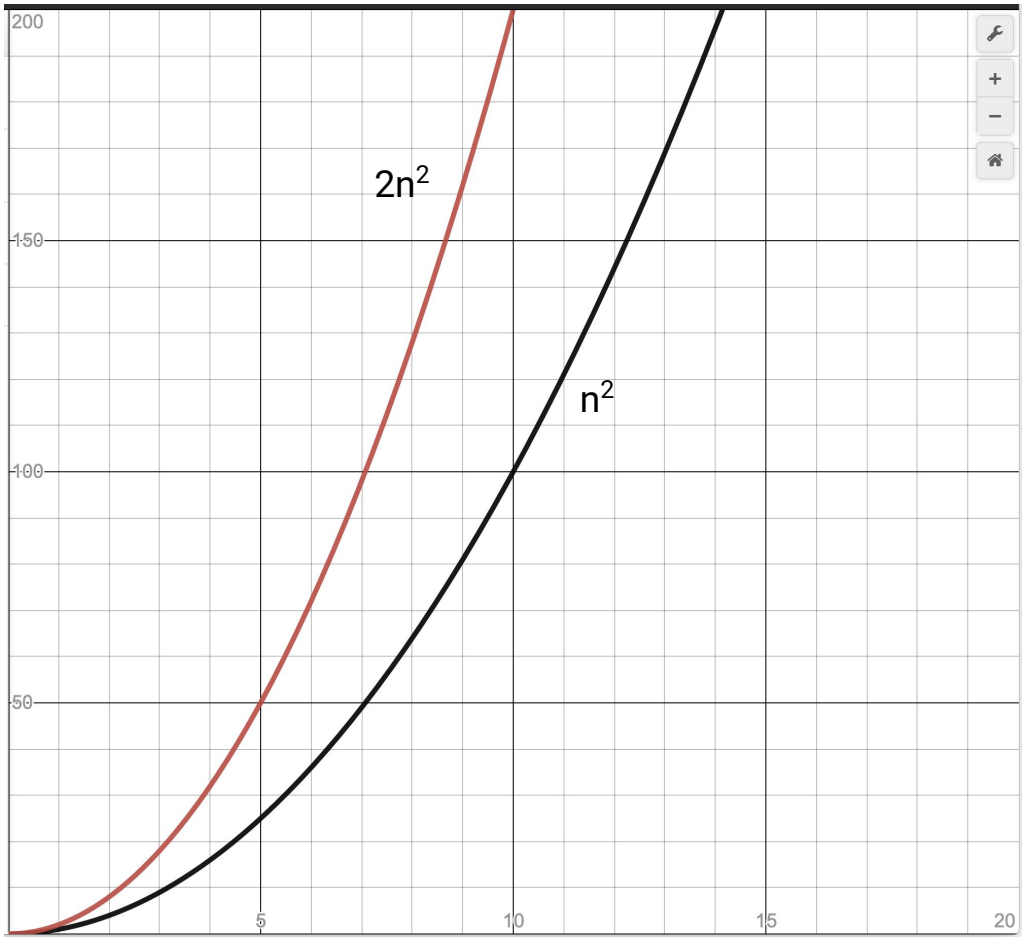
Comparing Functions: by specific values

$f(n) = n^2$ vs. $g(n) = 2n^2$

n	f(n)	f(2n)/f(n)	g(n)	g(2n)/g(n)
0	0	-	0	-
1	1	-	2	-
2	4	4	8	4
4	16	4	32	4
8	64	4	128	4
16	128	4	512	4

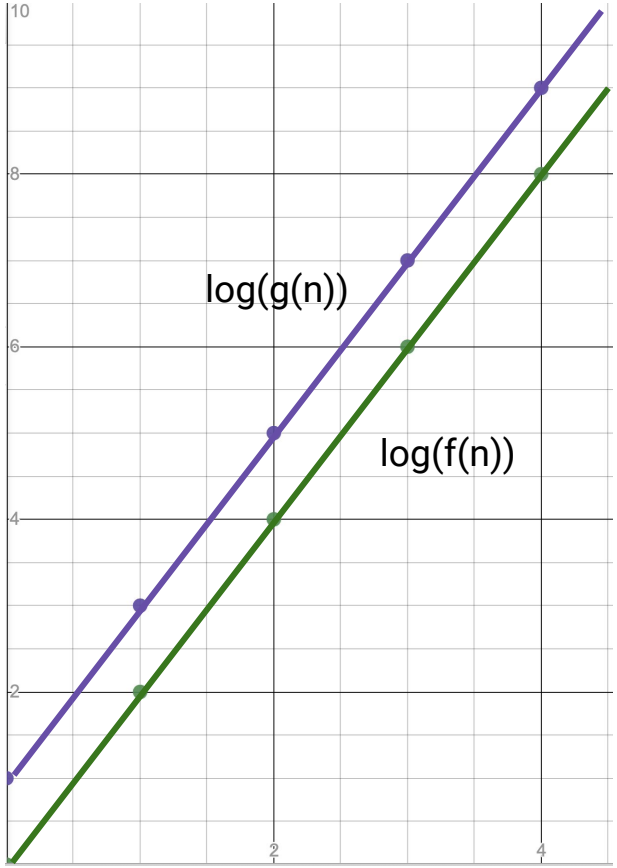
Comparing Functions: by graphing

$f(n) = n^2$ vs. $g(n) = 2n^2$



Comparing Functions: by (log-log) graphing

n	lgn	f(n)	lg(f(n))	g(n)	lg(g(n))
0	-	0	-	0	-
1	0	1	0	2	1
2	1	4	2	8	3
4	2	16	4	32	5
8	3	64	6	128	7
16	4	256	8	512	9



Comparing Functions: by ratio

$f(n) = n^2$ vs. $g(n) = 2n^2$

Classifying functions

Two functions are in the same order of growth classification if they grow at the same rate.

This means that the difference between their growth rates is a constant factor.

Big-Oh, Big-Omega, and Big-Theta.

- Big-Oh, Big-Omega, and Big-Theta are notations with formal definitions that help us compare and classify functions according to order of growth.
- Big-Oh is used to denote that one function can be an upper bound for another function.
- Big-Omega is used to denote that one function can be a lower bound for another function.
- Big-Theta is used to denote that one function can be both an upper and a lower bound for another function. (This also means that they grow at the same rate and are therefore in the same order of growth classification.)

Big-Oh, Big-Omega, and Big-Theta.

f(x) is O(g(x)) means
- g(x) is an upper bound for f(x)
- g(x) does not grow more slowly than f(x)
- f(x) does not grow more quickly than g(x)

f(x) is Ω(g(x)) means
- g(x) is a lower bound for f(x)
- g(x) does not grow more quickly than f(x)
- f(x) does not grow more slowly than g(x)

f(x) is θ(g(x)) means
- g(x) is a tight upper bound and a tight lower bound for f(x)
- f(x) is a tight upper bound and a tight lower bound for g(x)
- f(x) and g(x) grow at the same rate

Example 1. Using the proper notation, compare each pair of functions.

1. $f(n) = 4n^3 + n^2$ and $g(n) = 24n^3 + 6n^2$

$f(n)$ is $\Theta(g(n)) \longrightarrow$ $f(n)$ is $O(g(n))$ and $f(n)$ is $\Omega(g(n))$

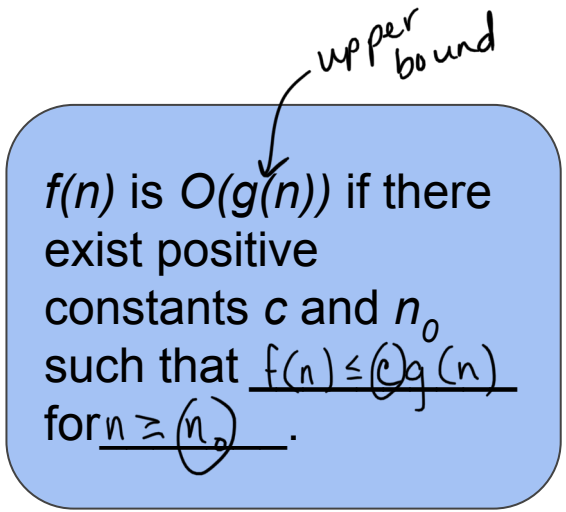
$g(n)$ is $\Theta(f(n)) \longrightarrow$ $g(n)$ is $O(f(n))$ and $g(n)$ is $\Omega(f(n))$

2. $f(n) = 2n^2$ and $g(n) = n^2 \log n$

$f(n)$ is $O(g(n))$

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

Big-Oh: Formal Definition



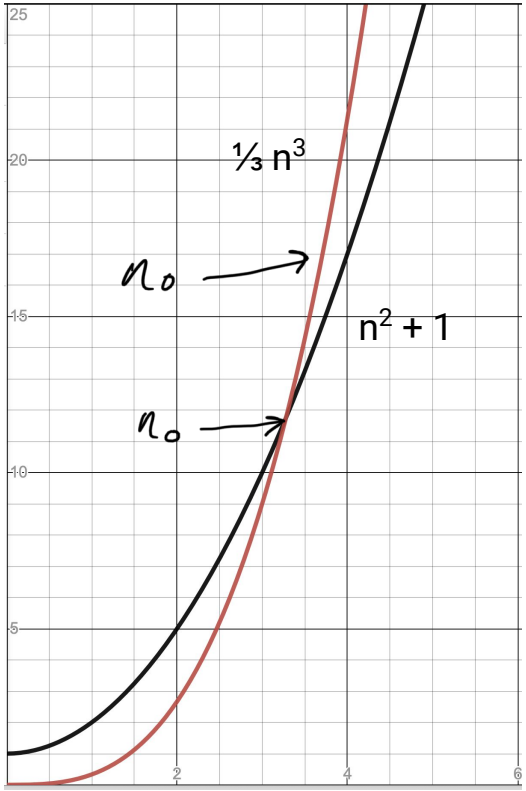
Big-Oh: Formal Definition

$f(n) = n^2 + 1$

$g(n) = \frac{1}{3}n^3$

$f(n)$ is $O(g(n))$

f(n) is *O(g(n))* if there exist positive constants *c* and n_0 such that $f(n) \leq c \cdot g(n)$ for $n \geq n_0$.

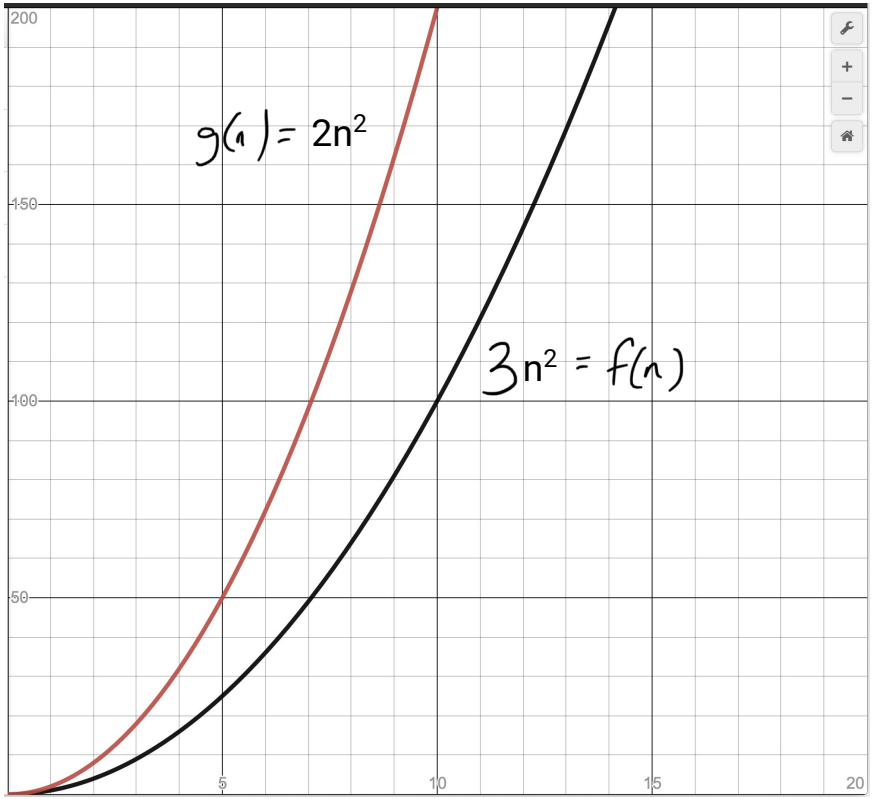


Big-Oh: Formal Definition

$f(n)$ is $O(g(n))$

$g(n)$ is $O(f(n))$

f(n) is *O(g(n))* if there exist positive constants *c* and n_0 such that $f(n) \leq c \cdot g(n)$ for $n \geq n_0$.



Big-Omega: Formal Definition

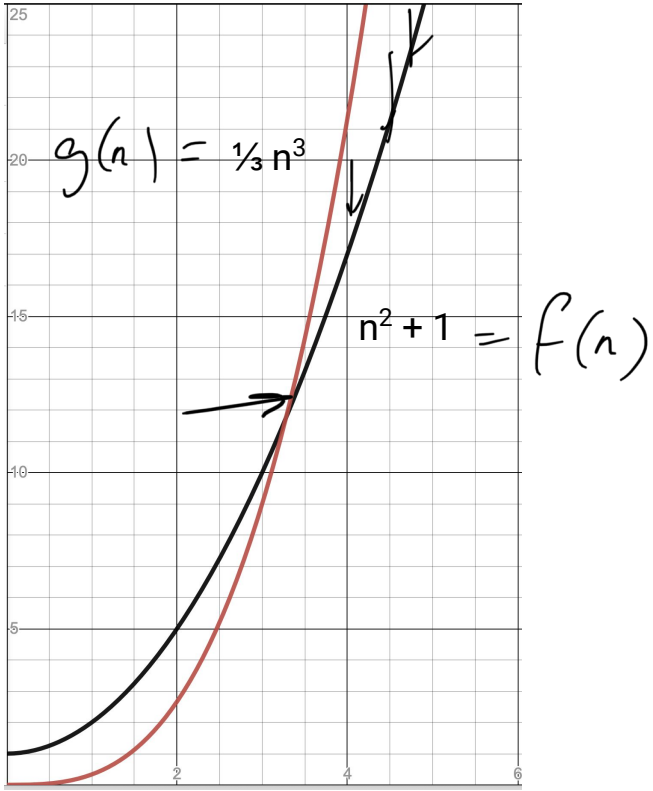
lower bound

f(n) is $\Omega(g(n))$ if there exist positive constants *c* and n_0 such that $f(n) \geq cg(n)$ for $n \geq n_0$.

Big-Omega: Formal Definition

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

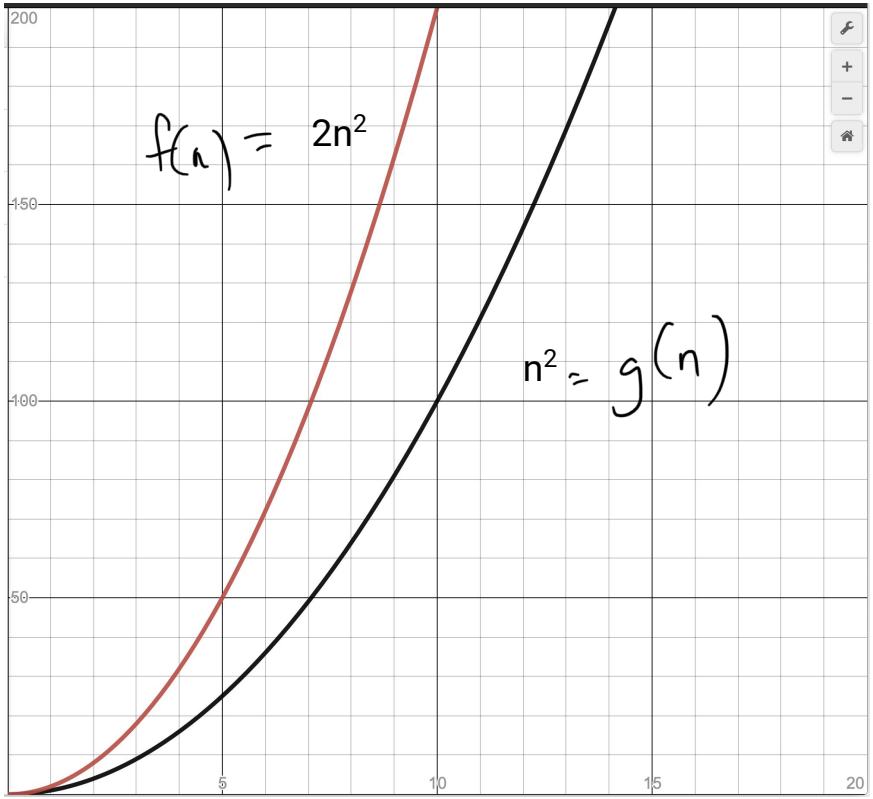
f(n) is *Ω(g(n))* if there exist positive constants *c* and n_0 such that $f(n) \geq c \cdot g(n)$ for $n \geq n_0$.



Big-Omega: Formal Definition

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

f(n) is *Ω(g(n))* if there exist positive constants *c* and n_0 such that $f(n) \geq c \cdot g(n)$ for $n \geq n_0$.



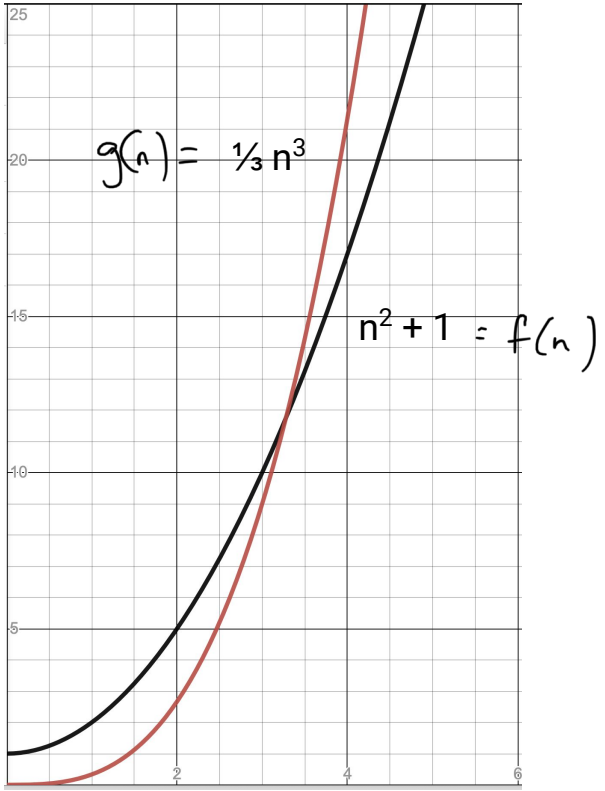
Big-Theta: Formal Definition

f(n) is *θ(g(n))* if
f(n) is O(g(n))
and f(n) is Ω(g(n)).

Big-Theta: Formal Definition

$f(n)$ is not $\Theta(g(n))$

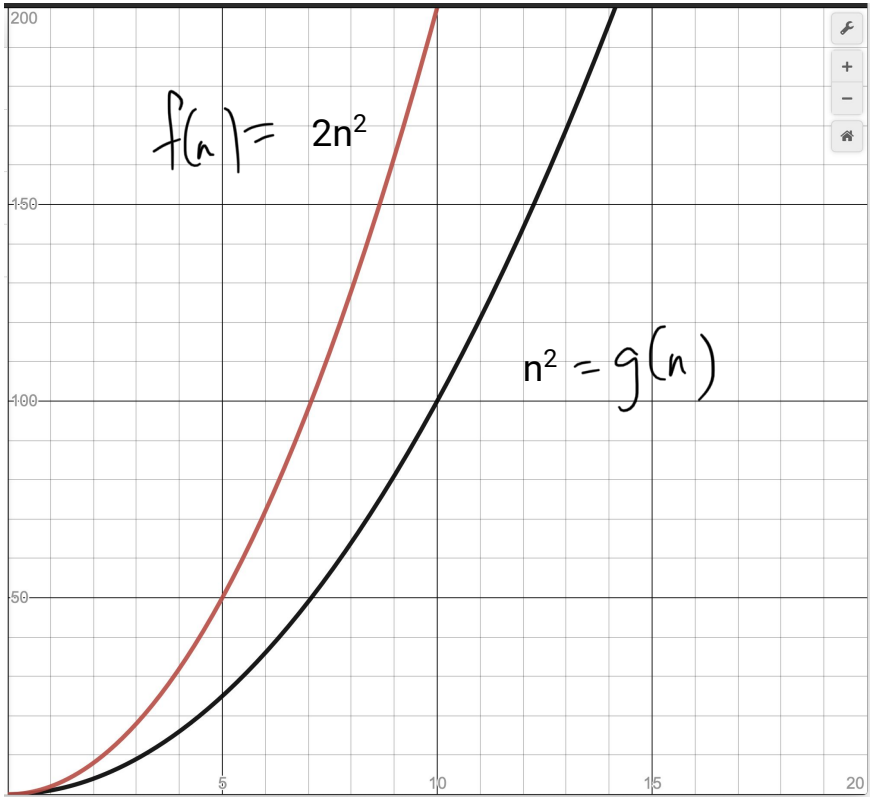
f(n) is *θ(g(n))* if *f(n)* is *O(g(n))* and *f(n)* is *Ω (g(n))*.



Big-Theta: Formal Definition

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

f(n) is *θ(g(n))* if *f(n)* is *O(g(n))* and *f(n)* is *Ω (g(n))*.



Classifying functions…

- Note that the formal definitions of big-Oh, big-Omega, and bit-Theta only require that a c and an n_0 exist.
- This means we can pick any c and n_0 that work.
- This also means that when we classify functions in this way we typically
  - drop the lower terms
  - drop the coefficient

- Big-Oh is used to describe an upper bound (does not have to be tight).
- Big-Omega is used to describe a lower bound (does not have to be tight).
- Big-Theta is used to classify functions according to order of growth. This is because since it establishes both an upper and a lower, that bound has to be a tight bound
- A tight upper bound is the "smallest" possible upper bound and a tight lower bound is the "largest" possible lower bound.

Example 2. Classify each function and justify your answer according to the formal definition of big-Theta.

$\Theta(n^2)$ → $f(n) \le cg(n)$ for $n \ge n_0$ $O(g(n))$

$f(n) \ge c_2 g(n)$ for $n \ge n_1$ $\Omega(g(n))$

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

1. $f(n) = 5n^2 + 4n + 1$

$f(n) = 5n^2 + 4n + 1$
$\le 5n^2 + 5n^2 + 5n^2$ for $n \ge 1$
$\le 15n^2$ for $n \ge 1$

$c = 15, n = 1$

$f(n) = 5n^2 + 4n + 1 \ge n^2$ for $n \ge 1$

$c = 1, n_0 = 1$

2. $f(n) = 4n^2 \log n^2 + 5n + 20 \log n$

3. $f(n) = \frac{3^{\log_3 n + 1} - 1}{2}$

Example 2. Classify each function and justify your answer according to the formal definition of big-Theta.

1. $f(n) = 5n^2 + 4n + 1$

2. $f(n) = 4n\log n + 5n + 20\log n$

$= 8n^2\log n + 5n + 20\log n$

$\Theta(n^2\log n)$

f(n) is $O(g(n))$

$f(n) = 8n^2\log n + 5n + 20\log n$

$\leq 20n^2\log n + 20n^2\log n + 20n^2\log n$

for $n \geq 10$

$= 60n^2\log n$

3. $f(n) = \frac{3^{\log_3 n+1} - 1}{2}$

f(n) is $\Omega(g(n))$:

$f(n) = 8n^2\log n + 5n + 20\log n$

$\geq 1n^2\log n$

for all $n \geq 1$

Example 2. Classify each function and justify your answer according to the formal definition of big-Theta.

1. $f(n) = 5n^2 + 4n + 1$

$f(n)$ is $O(n)$:

$f(n) = \frac{3}{2}n + \frac{1}{2}$

$\leq \frac{3}{2}n + \frac{3}{2}n$

$= 3n$ for $n \geq 1$

2. $f(n) = 4n^2 \log n^2 + 5n + 20 \log n$

3. $f(n) = \frac{3^{\log_3 n+1} - 1}{2} = \frac{3n-1}{2} = \frac{3}{2}n - \frac{1}{2}$ $\Theta(n)$

$f(n)$ is $\Omega(n)$:

$f(n) = \frac{3}{2}n - \frac{1}{2}$

$\geq n$ for $n \geq 1$

Common Order of Growth Classifications

Name	In Big-Theta	Examples
constant	$\theta(1)$	$100, 256, 300000, 1/2$
logarithmic	$\theta(logn)$	$log_2 n, 4log_3 n, logn^6, log(n/4)$
linear	$\theta(n)$	$5n + 2, 8n + 100, \frac{4n-1}{5}$
linearithmic	$\theta(nlogn)$	$4nlogn^3 + 5n + 1, \frac{nlog_4 n}{2} + 100$
quadratic	$\theta(n^2)$	$8n^2 + 8n + 1, 300nlog(2^n)$
cubic	$\theta(n^3)$	$n^3 + n^2 + n + 1, n^2log(16^n)$
exponential	$\theta(2^n)$	$2^{n+1}, 8(2^n)$

Common Order of Growth Classifications

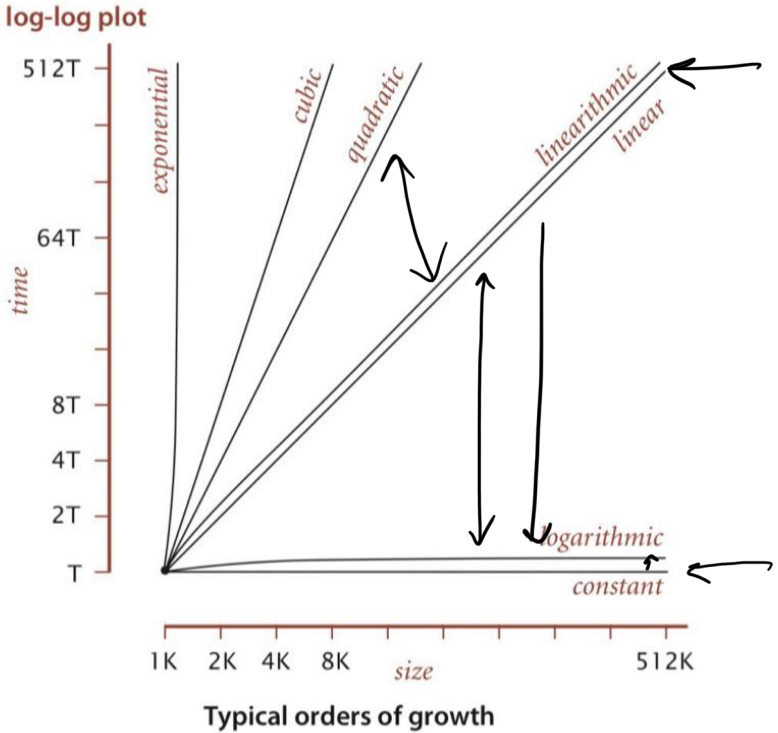
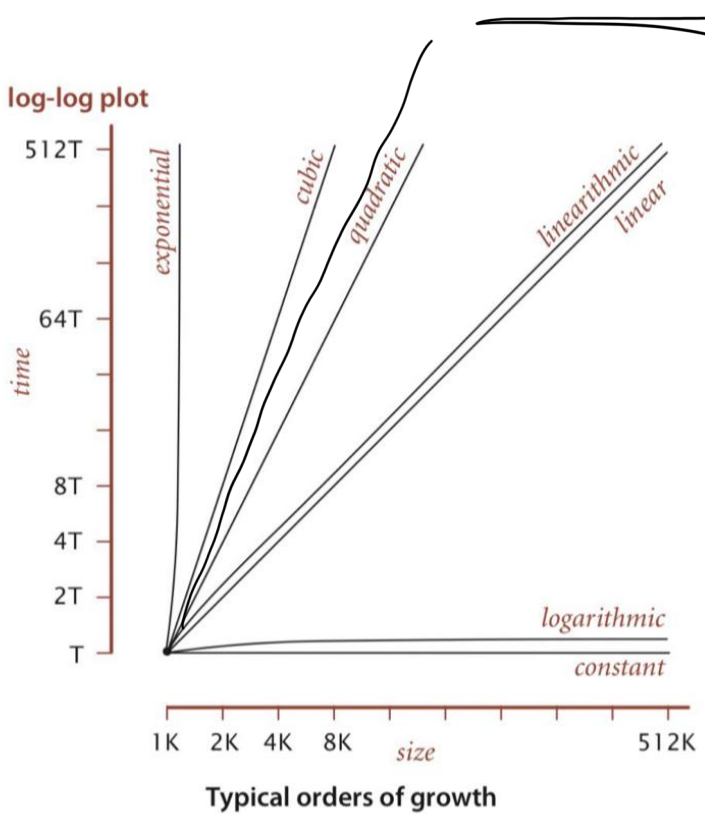


image from [2]

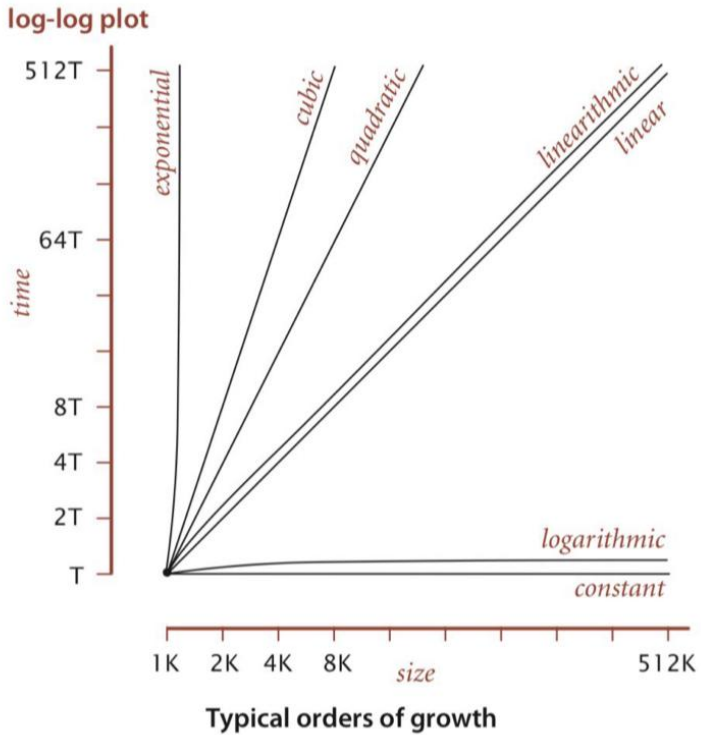
Common Order of Growth Classifications



Where would $n^2 \log n$ fall?

image from [2]

Common Order of Growth Classifications



Typical orders of growth

Where would $n^2 \log n$ fall?

between n^2 and n^3

What about $\log^2 n$?

between $\log n$ and n

(but closer to $\log n$)

image from [2]

Example 3. Determine if these statements are true or false and justify your answer.

1. $3n^2 \log n^2 + 4n$ is $\Omega(\log n)$

T

2. $3n^2 \log n + 4n$ is $O(n^2)$

F

3. $6n^2 \log 2^n + 4n^3$ is $O(n^2 \log n)$

$= 6n^3 + 4n^3$

F

4. $3n^2 + 5n + 7$ is $\theta(n^2)$

T

5. $4\log^2 n + n + 2^n$ is $\theta(n)$

F

6. $15n^3$ is $\Omega(1)$

T

Some useful things…

1. If f(n) is O(g(n)), then g(n) is $\Omega(f(n))$.
2. If f(n) is O(g(n)) and g(n) is O(h(n)), then f(n) is $O(h(n))$.
3. If f(n) is Θ(g(n)), then g(n) is $\Theta(f(n))$.
4. Every non-decreasing function is $\Omega(1)$.
5. If f(n) is O(g(n)), then f(n) + g(n) is $O(g(n))$.
6. f(n) + g(n) is $\Omega(f(n))$ and $\Omega(g(n))$.

Example 4. Determine if each statement is definitely true (DT), definitely false (DF), or possibly true/possibly false (PT) and justify your answer.

Given:

$f(x)$ *is* $O(g(x))$

$h(x)$ *is* $\Omega(g(x))$ ✓

$i(x)$ *is* $\theta(f(x))$

1. $f(x)$ *is* $O(h(x))$

DT

2. $g(x)$ *is* $\Omega(i(x))$

DT

3. $f(x) + g(x) + h(x) + i(x)$ *is* $O(h(x))$

DT

4. $f(x) + g(x) + h(x) + i(x)$ *is* $\Omega(f(x))$

DT

5. $f(x) + g(x) + h(x) + i(x)$ *is* $O(i(x))$

PT

References & Resources

1. https://www.desmos.com/calculator
2. Algorithms by Sedgewick & Wayne