Math Fundamentals Part 2

- Functions
- Order of Growth

What is a function?

input $\longrightarrow$ output

(domain) (range / codomain)

How do you compare and/or classify functions?

$f(n) = 3n^2 + 4n + 1$ $g(n) = 5n^2$ $\boxed{h(n) = 100n + 100}$ $i(n) = ½\, n^3$ $j(n) = ⅛\, n^4$ $\boxed{k(n) = 0.00000001(2^n)}$

power of n

coefficient

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

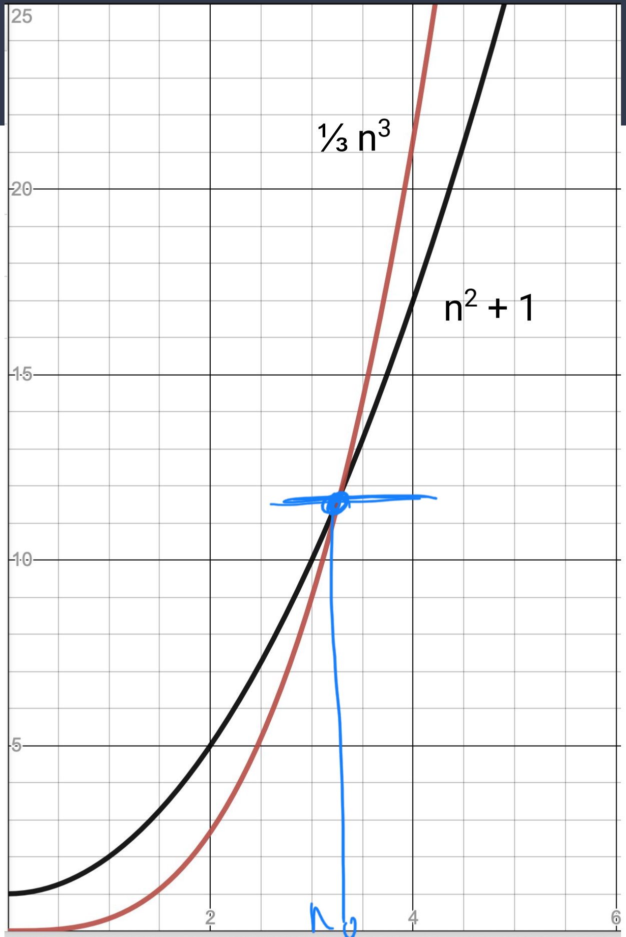
n	f(n)	g(n)
0	1	0
1	2	⅓
2	5	2 ⅔
4	17	21 ⅓
8	65	170 ⅔
16	257	1365 ⅓

Comparing Functions: by graphing

$f(n) = n^2 + 1$ vs. $g(n) = ⅓\, n^3$

$\underline{\underline{g(n) \geq f(n)}}$

for $n \geq n_0$



Comparing Functions: by ratio

$f(n) = n^2 + 1$ vs. $g(n) = ⅓\, n^3$

$$\frac{n^2 + 1}{\frac{1}{3} n^3} = \frac{\cancel{n^2}}{\frac{1}{3} n^{\cancel{3}}} + \frac{1}{\frac{1}{3} n^3}$$

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

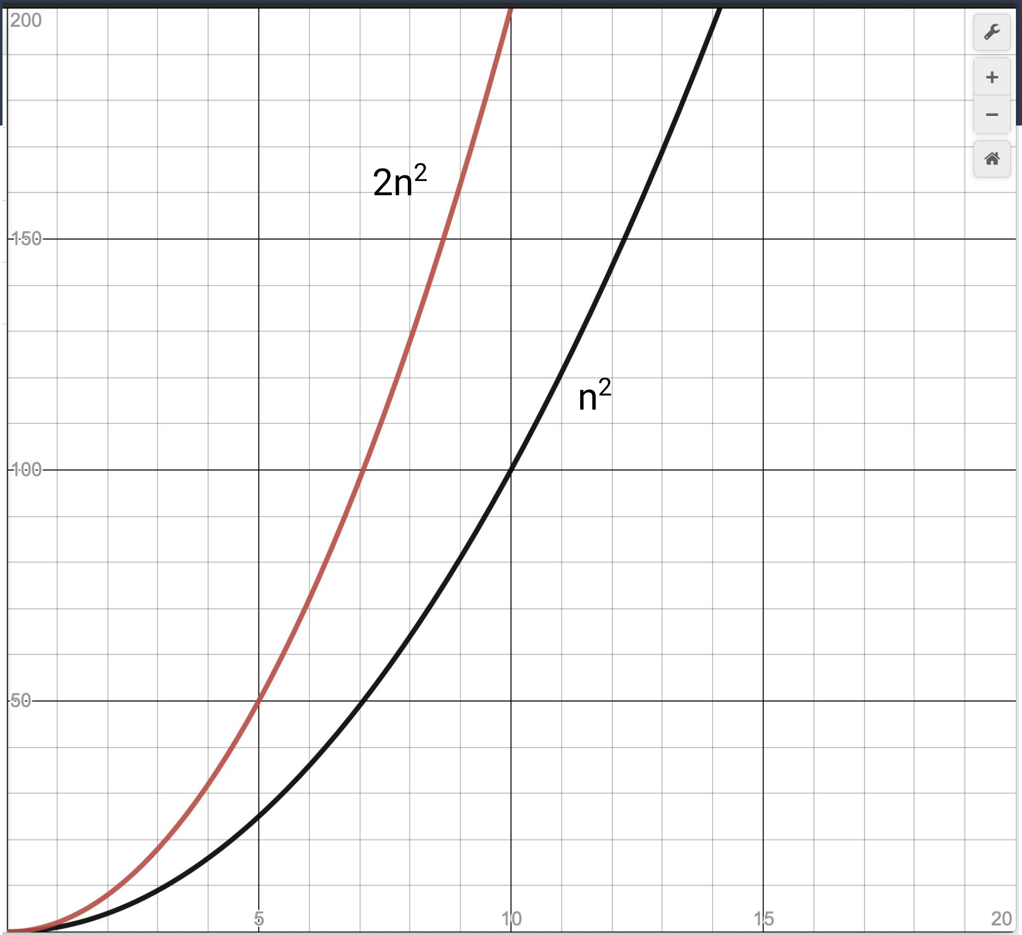
Comparing Functions: by specific values

$f(n) = n^2$ 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

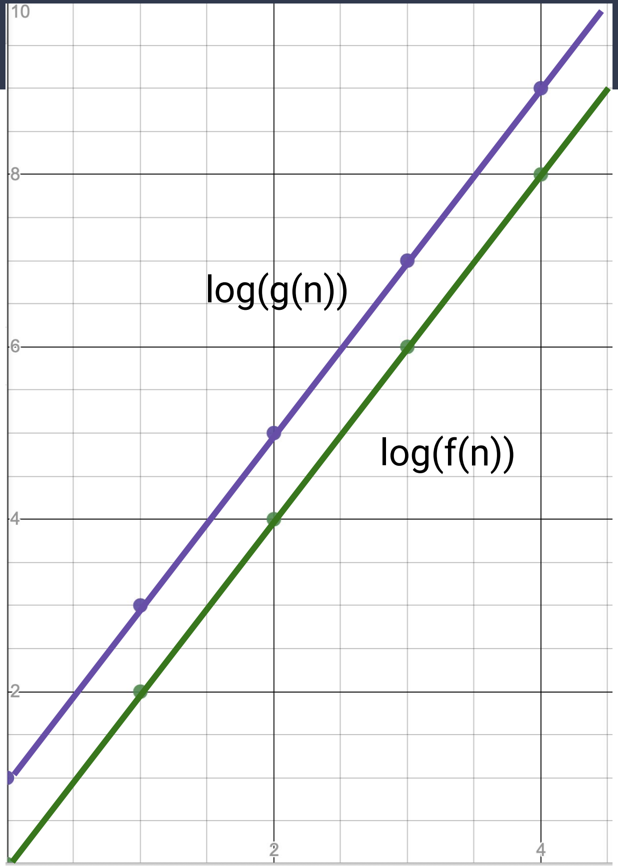
Comparing Functions: by graphing

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

$$\frac{2\cancel{n^2}}{\cancel{n^2}} = 2$$

Classifying functions

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

Classifying functions

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

How do we know if they grow at the same rate?

Classifying functions

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

How do we know if they grow at the same rate?

What does it actually mean to grow at the same rate?

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 bound* 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)

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)

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

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

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

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

2. $f(n) = 2n^2$ and $g(n) = n^2 logn$

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

$$\frac{\cancel{n^2}\log n}{2\cancel{n^2}} = \frac{\log n}{2}$$

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

Big-Oh: Formal Definition

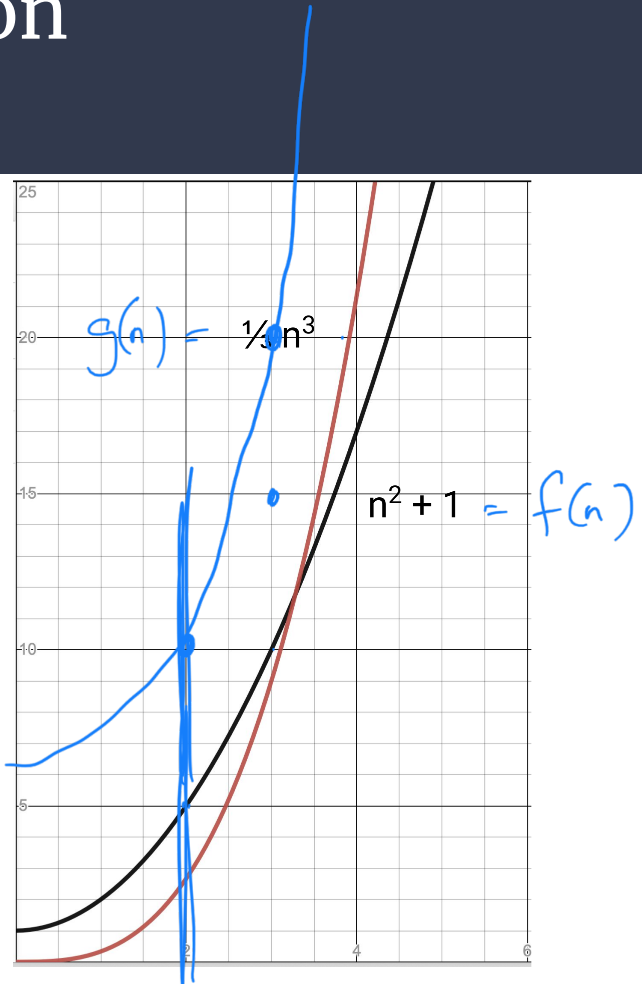
f(n) is *O(g(n))* if there exist positive constants *c* and n_0 such that $f(n) \le c \cdot g(n)$ for $n \ge n_0$.

Big-Oh: Formal Definition

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

$c = 2$

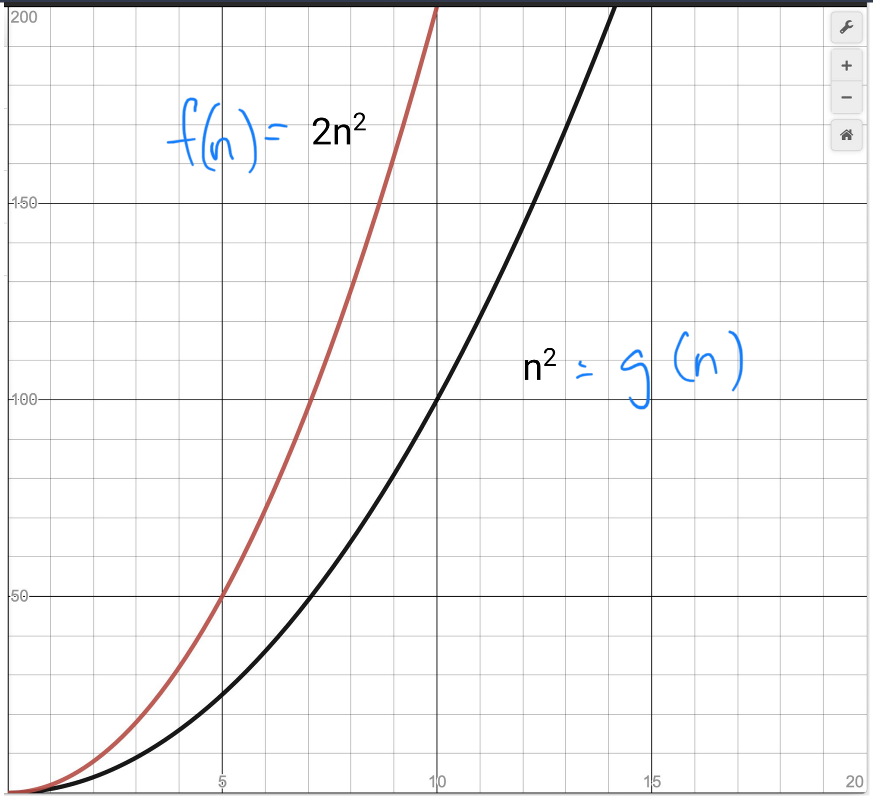
$n_0 = 2$



Big-Oh: Formal Definition

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

c = 3

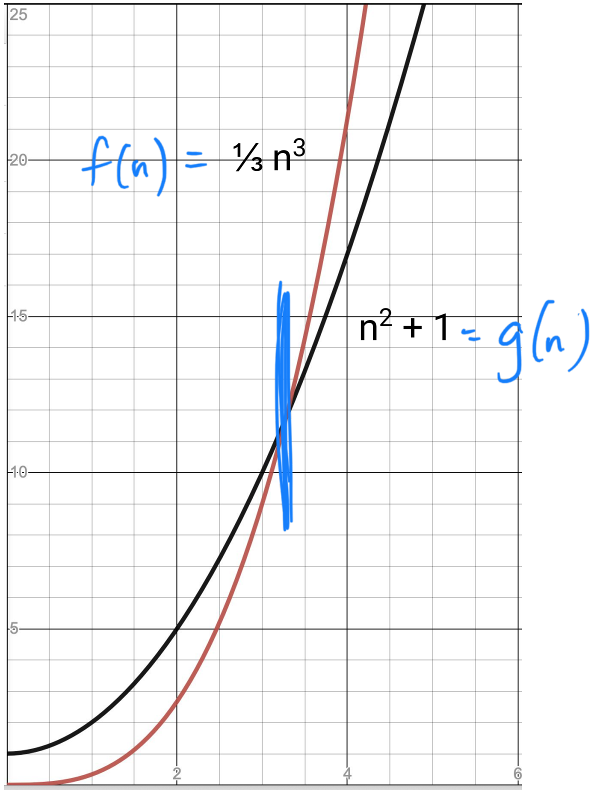


Big-Omega: Formal Definition

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

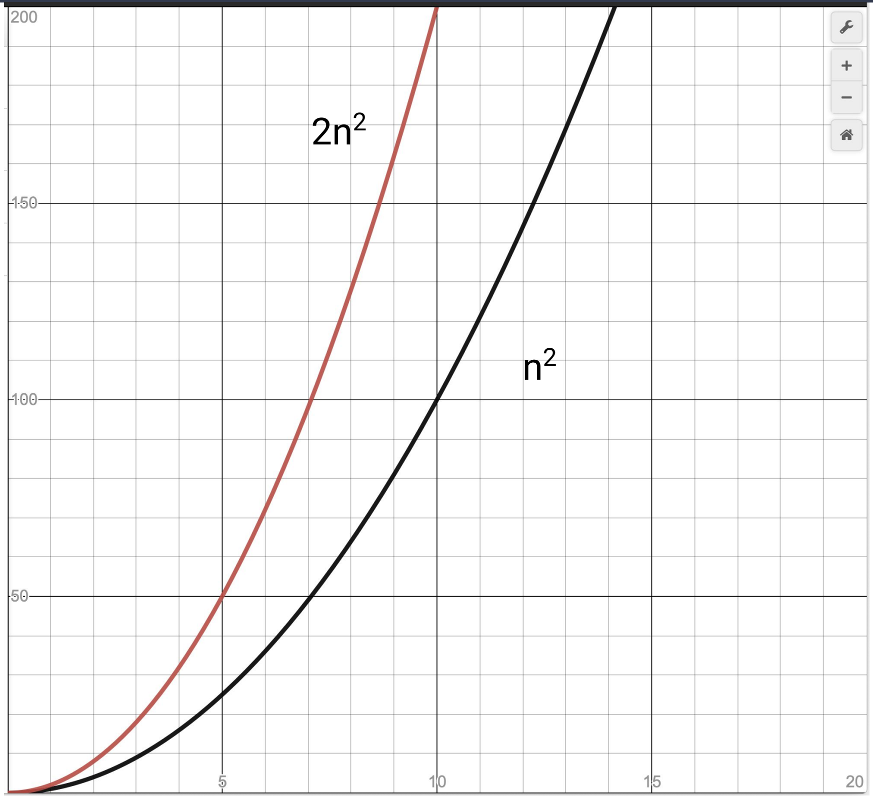
Big-Omega: Formal Definition

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



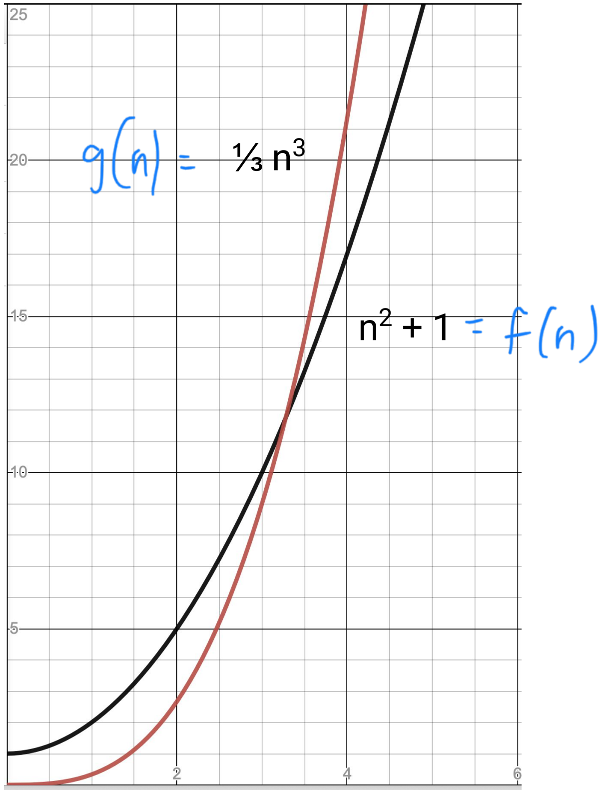
Big-Theta: Formal Definition

$f(n)$ is $\theta(g(n))$ if $f(n)$ is $O(g(n))$ and $f(n)$ is $\Omega(g(n))$.

Big-Theta: Formal Definition

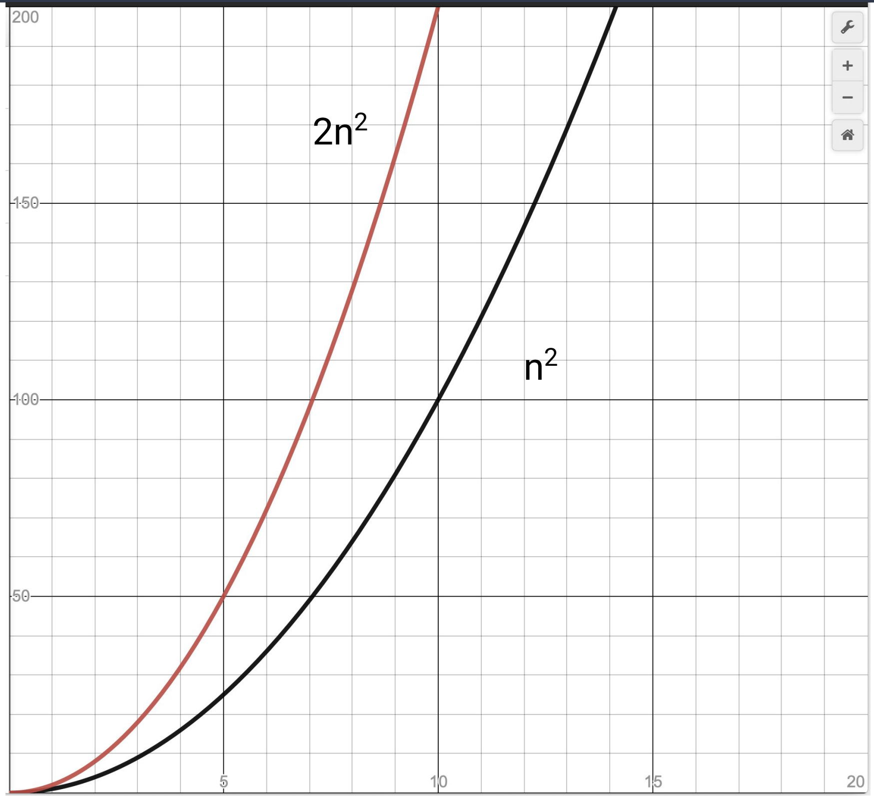
f(n) is *θ(g(n))* if *f(n)* is *O(g(n))* and *f(n)* is *Ω (g(n))*.

~~f(n) is Ω(g(n))~~



Big-Theta: Formal Definition

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
  - ignore lower terms
  - drop the starting 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 algorithms according to order of growth. This is because since it establishes both an upper and a lower bound, that bound has to be a tight bound.

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

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

Claim: $f(n)$ is $\Theta(n^2)$

$f(n)$ is $O(n^2)$: $5n^2 + 4n + 1 \le 5n^2 + 5n^2 + 5n^2 \le 15n^2$, $c = 15$, $n_0 = 1$

$f(n)$ is $\Omega(n^2)$: $5n^2 + 4n + 1 \ge n^2$ for $n \ge 1$

2. $f(n) = 4n^2 logn^{2} + 5n + 20logn$

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

$f(n)$ is $O(n^2 \log n)$: $8n^2 \log n + 5n + 20 \log n \le 20n^2 \log n + 20n^2 \log n + 20n^2 \log n \le 60n^2 \log n$ for $n \ge 1$

$f(n)$ is $\Omega(n^2 \log n)$: $8n^2 \log n + 5n + 20 \log n \ge n^2 \log n$ for $n \ge 1$

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

$\Theta(n)$

$3^{\log_3 n + 1} = 3 \cdot 3^{\log_3 n} = 3n$

$\frac{3n-1}{2} = \frac{3n}{2} - \frac{1}{2} \le \frac{3n}{2} + \frac{3n}{2} \le 3n$

$\frac{3n-1}{2} = \frac{3n}{2} - \frac{1}{2} \ge \frac{1}{2}n$ for $n \ge 1$

Common Order of Growth Classifications

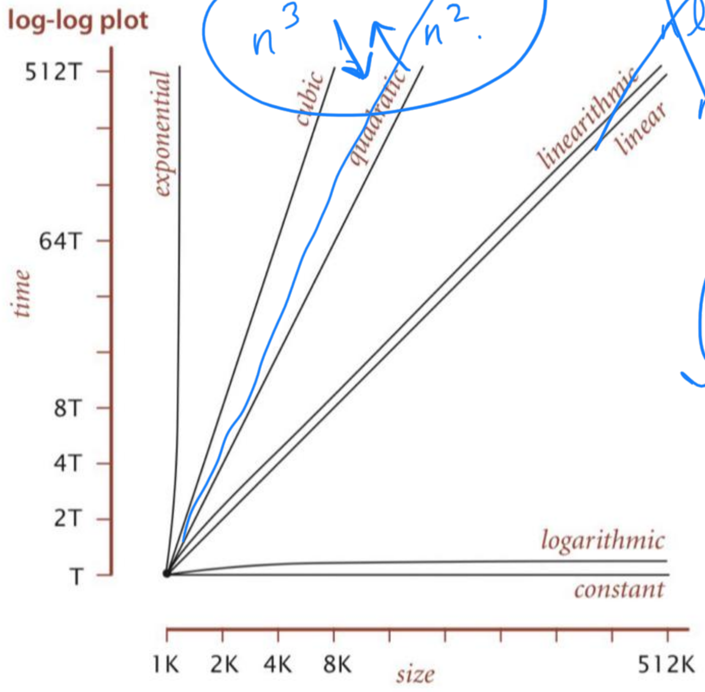
slowest-growing

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	$\theta(2^n)$	$2^{n+1},\ 8(2^n)$

faster-growing

factorial

Common Order of Growth Classifications



Typical orders of growth

$\frac{\log n}{n}$

$$\frac{\cancel{n^2}\log n}{\cancel{n^2}} \quad \text{vs. } n^2$$

$$\frac{n^{\cancel{3}}}{\cancel{n^2}\log n} = \frac{n}{\log n}$$

image from [2]

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

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

T

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

F

3. $6n^2 log2^n + 4n^3$ is $O(n^2 logn)$

F

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

T

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

F $O(n)$

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

T

$4(logn)^2$

$logn$

n vs.

$n\,logn$

$logn$ $\quad 1$

Some useful things to keep in mind.

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 $\theta(g(n))$, then $g(n)$ is $\theta(f(n))$.

4. Every order of growth function in this class will always be $\Omega(1)$.

5. If $f(n)$ is $O(g(n))$, then $f(n) + g(n)$ is $\theta(g(n))$.

Given:

$f(x)$ is $O(g(x))$ $\quad$ g(x) is Ω(f(x))

$h(x)$ is $\Omega(g(x))$ $\quad$ g(x) is O(h(x))

$i(x)$ is $\Theta(f(x))$

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

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

DT

i(x) is Θ(f(x))

f(x) is O(g(x)) → g(x) is Ω(~~f(x)~~ i(x))

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

DT

h(x)

g(x)

f(x)

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. $f(x) + \cancel{g(x)} + \cancel{h(x)} + i(x)$ is $O(i(x))$

PT

References & Resources

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