$\Theta(g(n))$

$$\Theta(g(n)) = \left\{ f(n) : \begin{array}{l} \exists c_1 > 0, c_2 > 0, n_0 > 0 \\ \forall n > n_0 \\ 0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n) \end{array} \right\}$$

Break this down into pieces. What does it mean?

(Slides courtesy of Russell Lewis)

$\Theta(g(n))$

$$\boxed{\Theta(g(n)) =} \left\{ f(n) : \begin{array}{l} \exists c_1 > 0, c_2 > 0, n_0 > 0 \\ \forall n > n_0 \\ 0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n) \end{array} \right\}$$

`Θ(g(n))` is defined as...

$\Theta(g(n))$

$$\Theta(g(n)) = \left\{ \boxed{f(n):} \begin{array}{l} \exists c_1 > 0, c_2 > 0, n_0 > 0 \\ \forall n > n_0 \\ 0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n) \end{array} \right\}$$

`Θ(g(n))` is defined as...a set of functions such that...

$\Theta(g(n))$

$$\Theta(g(n)) = \left\{ f(n) : \begin{array}{l} \boxed{\exists c_1 > 0, c_2 > 0, n_0 > 0} \\ \forall n > n_0 \\ 0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n) \end{array} \right\}$$

$\Theta(g(n))$ is defined as...a set of functions such that...there exist constants c_1, c_2, n_0 such that...

$\Theta(g(n))$

$$\Theta(g(n)) = \left\{ f(n) : \begin{array}{l} \exists c_1 > 0, c_2 > 0, n_0 > 0 \\ \boxed{\forall n > n_0} \\ 0 \le c_1 g(n) \le f(n) \le c_2 g(n) \end{array} \right\}$$

$\Theta(g(n))$ is defined as...a set of functions such that...there exist constants c_1, c_2, n_0 such that...for all n greater than n_0...

$\Theta(g(n))$

$$\Theta(g(n)) = \left\{ f(n): \begin{array}{l} \exists c_1 > 0, c_2 > 0, n_0 > 0 \\ \forall n > n_0 \\ \boxed{0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n)} \end{array} \right\}$$

`Θ(g(n))` is defined as...a set of functions such that...there exist constants `c`$_1$`, c`$_2$`, n`$_0$ such that...for all `n` greater than `n`$_0$, `f(n)` is bounded by `g(n)`.

$\Theta(g(n))$

$$\Theta(g(n)) = \left\{ f(n) : \begin{array}{l} \exists c_1 > 0, c_2 > 0, n_0 > 0 \\ \forall n > n_0 \\ \boxed{0 \leq c_1 g(n) \leq f(n)} \leq c_2 g(n) \end{array} \right\}$$

Lower bound on `f(n)`: `f(n)` cannot get too small.

$\Theta\left(g\left(n\right)\right)$

$$\Theta\left(g\left(n\right)\right)=\left\{ \begin{array}{ll} f\left(n\right): & \exists c_1>0, c_2>0, n_0>0 \\ & \forall n>n_0 \\ & 0\leq c_1 g\left(n\right)\leq \boxed{f\left(n\right)\leq c_2 g\left(n\right)} \end{array} \right\}$$

Upper bound on `f(n)`: `f(n)` cannot get too large.

$\Theta(g(n))$

Why so formal a definition?

So that we can prove theorems about it!

About this notation

- What does *n* mean when we write $\Theta(n)$?
  - Is it a number?
  - Is it a free variable?
- What does *n* mean when we write $\Theta(n^2)$?
- What does the one mean when we write $\Theta(1)$?

About this notation

- What does *n* mean when we write $\Theta(n)$?
  - Is it a number?
  - Is it a free variable?
- What does *n* mean when we write $\Theta(n^2)$?
- What does the one mean when we write $\Theta(1)$?

- Not a number, not a free variable.
- Each represents a *pure function.*

About this notation

- It's conventional to write the “=” symbol even though we really mean set-membership, “$\in$”
- For example,

  $37\ n^2 = \Theta(n^2)$.

$\Theta(g(n))$

$$q(n) = \Theta(n^2)$$
$$l(n) = \Theta(n)$$

Consider two functions, one quadratic and the other linear.

Imagine that the linear one has a large constant, and grows quickly.

Imagine that the quadratic one has a tiny constant, and hardly grows at all.

$\Theta(g(n))$

$$q(n)=\Theta(n^2)$$
$$l(n)=\Theta(n)$$

$$\lim_{n \to \infty} \frac{q(n)}{l(n)}$$

Can we prove the value of this limit?

I thought of a fuzzy answer, and a more precise one.

Let's work on this in groups.

$\Theta(g(n))$

$$q(n) = \Theta(n^2)$$
$$l(n) = \Theta(n)$$

NOTE:

We're using poor reasoning here, we'll do a better version next.

$$\lim_{n \to \infty} \frac{q(n)}{l(n)} \approx \lim_{n \to \infty} \frac{a\,n^2}{b\,n}$$

$\Theta(g(n))$

$$q(n) = \Theta(n^2)$$
$$l(n) = \Theta(n)$$

$$\lim_{n \to \infty} \frac{q(n)}{l(n)} \approx \lim_{n \to \infty} \frac{a\,n^2}{b\,n} = \lim_{n \to \infty} \frac{n}{b/a} = \infty$$

$\Theta(g(n))$

$$q(n) = \Theta(n^2)$$
$$l(n) = \Theta(n)$$

This argument has a fuzzy step. Can we make it more precise?

Let's use the formal definition of `Θ(g(n))` to solve this.

$$\lim_{n \to \infty} \frac{q(n)}{l(n)} \approx \lim_{n \to \infty} \frac{a\,n^2}{b\,n} = \lim_{n \to \infty} \frac{n}{b/a} = \infty$$

$\Theta(g(n))$

$$q(n) = \Theta(n^2)$$
$$l(n) = \Theta(n)$$

These are the two functions.

`q(n)` is quadratic.
`l(n)` is linear.

$\Theta(g(n))$

We start by using the definition of Θ to define a **lower bound** for `q(n)`

$$q(n) = \Theta(n^2)$$
$$l(n) = \Theta(n)$$

$$\exists c_q > 0, n_q > 0$$
$$\forall n > n_q$$
$$c_q n^2 \le q(n)$$

$\Theta\left(g\left(n\right)\right)$

$$q\left(n\right)=\Theta\left(n^{2}\right)$$
$$l\left(n\right)=\Theta\left(n\right)$$

We then use the definition of Θ to define an **upper bound** for `l(n)`

$$\exists c_q > 0, n_q > 0$$
$$\forall n > n_q$$
$$c_q n^2 \leq q\left(n\right)$$

$$\exists c_l > 0, n_l > 0$$
$$\forall n > n_l$$
$$0 \leq l\left(n\right) \leq c_l n$$

$\Theta(g(n))$

$$\exists c_q > 0, n_q > 0$$
$$\forall n > n_q$$
$$c_q n^2 \leq q(n)$$

$$\exists c_l > 0, n_l > 0$$
$$\forall n > n_l$$
$$0 \leq l(n) \leq c_l n$$

For some large-enough `n`, `q(n)` will always be **larger** than $c_q n^2$

$$\forall\, n > \max\{n_q, n_l\},$$

$$\frac{q(n)}{l(n)} \geq \frac{c_q n^2}{c_l n}$$

For some large-enough `n`, `l(n)` will always be **smaller** than $c_l n$

$\Theta\left(g\left(n\right)\right)$

$$\lim_{n \to \infty} \frac{q\left(n\right)}{l\left(n\right)} \geq \lim_{n \to \infty} \frac{c_q n^2}{c_l n} = \infty$$

Therefore, the limit of the ratio of the functions...

is a lower bound of the limit of ratio of the two bounds...

which diverges to plus infinity.

$$\Theta\left(g\left(n\right)\right)$$

$$\lim_{n \to \infty} \frac{q\left(n\right)}{l\left(n\right)} \geq \lim_{n \to \infty} \frac{c_q n^2}{c_l n} = \infty$$

Therefore, **no matter what the constants are,** a quadratic function is (eventually) **always larger** than a linear function.

Generalizing

- Higher-order polynomials are **always** upper bounds on lower-order polynomials.

stated another way …

- Higher-order polynomials are **always** larger than lower-order polynomials (eventually).

Topic 03: Asymptotic Notation

- Math Review
- An Example Method
- θ(g(n))
- **O(g(n))**
- Ω(g(n)), o(g(n)), ω(g(n))
- lg n
- Ordering functions by growth

O(g(n))

- Far more commonly used than θ(g(n))

  \- but -

- O(g(n)) only provides an **upper** bound

O(g(n))

$$O(g(n)) = \{\text{functions that grow no faster than } g(n)\}$$

$$O(g(n)) = \{f(n): \quad ??? \quad \}$$

O(g(n))

$$O\left(g\left(n\right)\right) = \left\{ \text{functions that grow no faster than } g\left(n\right) \right\}$$

$$O\left(g\left(n\right)\right) = \left\{ \begin{array}{ll} f\left(n\right): & \exists c > 0, n_0 > 0 \\ & \forall n \geq n_0 \\ & 0 \leq f\left(n\right) \leq c\, g\left(n\right) \end{array} \right\}$$

O(g(n))

- Why use `O(g(n))` when analyzing programs?
  - Do we even care about lower bound?
  - Special cases may be fast -- not our concern.
- Often, `O(g(n))` is the right choice for simple analysis
  - Θ`(g(n))` is useful for some proofs
- Be clear!

In Practice...

Formal Definition

$$n^2 = O\left(n^2\right)$$
$$n = O\left(n^2\right)$$
$$1 = O\left(n^2\right)$$

(this class)

Informal Usage

$$n^2 = O\left(n^2\right)$$
$$n \neq O\left(n^2\right)$$
$$1 \neq O\left(n^2\right)$$

(the rest of the world)

Asymptotically Tight?

- $\Theta(g(n))$ "provides bounds which are asymptotically tight"
  - Provides a precise statement of the growth pattern

- $O(g(n))$ may or may not

Asymptotically Tight?

Asymptotically Tight Bounds	**NOT** Asymptotically Tight Bounds
$n^2 = O(n^2)$	
$n = O(n)$	$n = O(n^2)$
$1 = O(1)$	$1 = O(n^2)$

A Metaphor

$\Theta\left(g\left(n\right)\right)$	$=$
$O\left(g\left(n\right)\right)$	$\leq$
?	$\geq$
?	$<$
?	$>$

Topic 03: Asymptotic Notation

- An Example Method
- $\theta(g(n))$
- $O(g(n))$
- **$\Omega(g(n))$, $o(g(n))$, $\omega(g(n))$**
- $\lg n$
- Ordering functions by growth

$\Omega(g(n))$

$$\Omega\left(g\left(n\right)\right)=\left\{\text{funcs that grow at least as fast as } g\left(n\right)\right\}$$

$$\Omega\left(g\left(n\right)\right)=\left\{\quad ??? \quad\right\}$$

Ω(g(n))

$$\Omega\left(g\left(n\right)\right)=\left\{\text{funcs that grow at least as fast as } g\left(n\right)\right\}$$

$$\Omega\left(g\left(n\right)\right)=\left\{\begin{array}{ll} f\left(n\right): & \exists c>0, n_0>0 \\ & \forall n \geq n_0 \\ & c\, g\left(n\right) \leq f\left(n\right) \end{array}\right\}$$

Θ, O, Ω

Θ: Upper and lower bounds

O: Upper bounds

Ω: Lower bounds

$$f(n) = \Theta(g(n)) \rightarrow f(n) = O(g(n))$$

$$f(n) = \Theta(g(n)) \rightarrow f(n) = \Omega(g(n))$$

Θ, O, Ω

Θ: Upper and lower bounds

O: Upper bounds

Ω: Lower bounds

$$\Theta\left(g\left(n\right)\right) \quad \subseteq \quad O\left(g\left(n\right)\right)$$

$$\Theta\left(g\left(n\right)\right) \quad \subseteq \quad \Omega\left(g\left(n\right)\right)$$

o(g(n))

- Provides an upper bound which is guaranteed **not** to be "asymptotically tight"

$$1 = o\left(n^2\right)$$

$$n = o\left(n^2\right)$$

$$n^2 \neq o\left(n^2\right)$$

o(g(n))

$$o(g(n)) = \{\text{funcs that grow more slowly than } g(n)\}$$

Concept:

Given some function f(n) = o(g(n)), no matter how small a scaling factor we put on g(n), and no matter how **large** a scaling factor we put on f(n), g(n) will eventually catch up, and pass f(n).

o(g(n))

$$o(g(n)) = \{\text{funcs that grow more slowly than } g(n)\}$$

$$o(g(n)) = \{f(n): \quad ??? \quad \}$$

Hint:

We only need a single scaling constant – which we'll apply to g(n), just like the previous definitions.

o(g(n))

$$o(g(n)) = \{\text{funcs that grow more slowly than } g(n)\}$$

$$o(g(n)) = \left\{ f(n) : \begin{array}{l} \forall c > 0 \\ \exists n_0 > 0 \\ \forall n \geq n_0 \\ 0 \leq f(n) < c\,g(n) \end{array} \right\}$$

o(g(n))

$$o\left(g\left(n\right)\right)=\left\{\text{funcs that grow more slowly than } g\left(n\right)\right\}$$

$$o\left(g\left(n\right)\right)=\left\{f\left(n\right):\lim_{n\to\infty}\frac{f\left(n\right)}{g\left(n\right)}=0\right\}$$

So Why Do We Care?

- Often, we want to explicitly state if two functions are **different**.
  - Suppose that we wanted to know the relationship between f(n) and h(n) ...

Assume:

$$l(n) = \Theta(lg^b n)$$

$$p(n) = \Theta(n^a)$$

$$a > 0$$

So Why Do We Care?

- Often, we want to explicitly state if two functions are **different**.
  - Suppose that we wanted to know the relationship between f(n) and h(n) ...

Assume:

$$a > 0$$

$$l(n) = \Theta(lg^b n)$$

$$p(n) = \Theta(n^a)$$

$$l(n) = o(p(n))$$

We'll discuss the relationship of polynomials and logarithms in more depth later.

ω(g(n))

$$\omega(g(n)) = \{\text{funcs that grow more quickly than } g(n)\}$$

$$\omega(g(n)) = \{f(n):\quad ???\quad\}$$

ω(g(n))

$$\omega(g(n)) = \{\text{funcs that grow more quickly than } g(n)\}$$

$$\omega(g(n)) = \left\{ f(n): \begin{array}{l} \forall c > 0 \\ \exists n_0 > 0 \\ \forall n \geq n_0 \\ 0 \leq c\,g(n) < f(n) \end{array} \right\}$$

o， ω

$$f(n)=\omega(g(n))$$

$$g(n) \ \textbf{?} \ f(n)$$

o, ω

$$f(n) = \omega(g(n))$$

$$g(n) = o(f(n))$$

Topic 03: Asymptotic Notation

- An Example Method
- $\Theta(g(n))$
- $O(g(n))$
- $\Omega(g(n))$, $o(g(n))$, $\omega(g(n))$
- **lg n**
- Ordering functions by growth

Some Definitions

$$\log n = \log_{10} n$$

$$\ln n = \log_e n$$

We'll mostly use this one. $\rightarrow$

$$lg\, n = \log_2 n$$

$$lg^k\, n = (lg\, n)^k$$

$$lg\, lg\, n = lg\,(lg\, n)$$

$$lg^*\, n = \text{"iterated logarithm"}$$

Log Identities

$$a = b^{\log_b a}$$

$$\log_c(ab) = \log_c a + \log_c b$$

$$\log_b a^n = n \log_b a$$

$$\log_b a = \frac{\log_c a}{\log_c b}$$

$$\log_b a = \frac{1}{\log_a b}$$

$$a^{\log_b c} = c^{\log_b a}$$

lg n

- Algorithms with logs in them are very common
- Logarithms grow very slowly

n	n^2	$lg\ n$
1	1	0
2	4	1
4	16	2
8	64	3
16	256	4
32	1024	5

Θ - equivalence

$$lg\, n = \Theta(\log n) = \Theta(\ln n)$$

Why?

lg n

- We've already seen how polynomials and logs relate:

$$lg^b n = o(n^a)$$

Assume:

$$a > 0$$

$$lg^{100} n = o(n)$$

lg n

- So how do we compare these pairs of functions?

$$n^a \quad ? \quad n^a \log^b n$$

$$n^a \log^b n \quad ? \quad n^{a+\epsilon}$$

Assume:

$$a, b, \epsilon > 0$$

lg n

- So how do we compare these pairs of functions?

$$n^a \quad = o\left(n^a \log^b n\right)$$

$$n^a \log^b n \quad = o\left(n^{a+\epsilon}\right)$$

Assume:

$$a, b, \epsilon > 0$$

2^n

- 2^n is very huge – far huger than any polynomial.

$$n^a = o\left(2^n\right)$$

2^n

- 2^n is very huge – far huger than any polynomial.

NOTE:

I'm not claiming that I've proved this step. But it's a reasonable transform.

$$n^a = o(2^n)$$

$$lg\, n^a = o(lg\, 2^n)$$
$$a\, lg\, n = o(n)$$

Topic 03: Asymptotic Notation

- An Example Method
- $\Theta(g(n))$
- $O(g(n))$
- $\Omega(g(n))$, $o(g(n))$, $\omega(g(n))$
- $\lg n$
- **Ordering functions by growth**

Ordering Functions by Growth

- We can organize functions into groups
  - Some you've seen already
  - Many you haven't yet

Some Famous Families

<u>What it's Famous For</u>

$$\Theta(1)$$

$$\Theta(\log n)$$

$$\Theta(n)$$

$$\Theta(n \log n)$$

$$\Theta(n^2)$$

$$\Theta(2^n)$$

$$\Theta(n^n)$$

(In Ascending Order)

Some Famous Families

	What it's Famous For
$\Theta(1)$	simple ops
$\Theta(\log n)$	binary search
$\Theta(n)$	inspect all elements
$\Theta(n \log n)$	the best sorts
$\Theta(n^2)$	check all pairs
$\Theta(2^n)$	all boolean combinations
$\Theta(n^n)$	give up, you've lost

(In Ascending Order)

Some Famous Families

	$n=1$	$n=10$	$n=1000$
$\Theta(1)$	1	1	1
$\Theta(\log n)$	0	1	3
$\Theta(n)$	1	10	1000
$\Theta(n\log n)$	0	10	3000
$\Theta(n^2)$	1	100	1000000
$\Theta(2^n)$	2	1024	googol, cubed
$\Theta(n^n)$	1	10000000000	???

Noteworthy Names

$\Theta(1)$	"constant time"
$\Theta(\log n)$	"log"
$o(n)$	"sublinear"
$\Theta(n)$	"linear"
$\Theta(n \log n)$	"log-linear"
$\Theta(n^k)$	"polynomial"
$\Theta\left(2^{(n^k)}\right)$	"exponential"

Topic 03: Asymptotic Notation

- An Example Method
- $\Theta(g(n))$
- $O(g(n))$
- $\Omega(g(n))$, $o(g(n))$, $\omega(g(n))$
- $\lg n$
- Ordering functions by growth

Summary