Topic 03: Asymptotic Notation

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

```c
void myMethod(int n)
{
    for (int i=0; i<n; i++)
    {
        for (int j=0; j<n; j++)
        {
            ... doWork ...
        }
        methodA();
    }
    methodB();
}
```
void myMethod(int n)
{
    for (int i=0; i<n; i++)
    {
        for (int j=0; j<n; j++)
        {
            ... doWork ...
        }
        methodA();
    }
    methodB();
}

Can we write an expression to give the overall time?
An Example Method

```java
void myMethod(int n) {
    for (int i=0; i<n; i++) {
        for (int j=0; j<n; j++) {
            ... doWork ...
        }
        methodA();
    }
    methodB();
}
```

Time to Execute

\[ n^2 (c_2 + c_3) + n (c_1 + c_4) + c_5 \]
Function Growth

Q: How does this function grow as $n$ gets larger?

$$n^2(c_2 + c_3) + n(c_1 + c_4) + c_5$$
Function Growth

\[ n^2(c_2 + c_3) + n(c_1 + c_4) + c_5 \]

Q: How does this function grow as \( n \) gets larger?

A: Eventually, the \( n^2 \) term gets so large that nothing else matters.
Function Growth

\[
\lim_{{n \to \infty}} \frac{n^2(c_2 + c_3) + n(c_1 + c_4) + c_5}{n^2(c_2 + c_3)} = ?
\]

Here's a way to see it as a limit...
Here's a way to see it as a limit...

As n gets large, the difference between the two expressions becomes negligible.
Function Growth

\[
\lim_{n \to \infty} \frac{a_1 n^2 + b_1 n + c_1}{a_2 n^2 + b_2 n + c_2} = ?
\]

What about when we want to compare two different algorithms? What is the ratio between two \( n^2 \) algorithms?
What about when we want to compare two different algorithms? What is the ratio between two $n^2$ algorithms?

All that really matters is the constants on the quadratic term!
Topic 03: Asymptotic Notation

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

- \( \Theta(g(n)) \) is a way of expressing how fast a function grows.

- You've seen \( O(g(n)) \)?
  - This is the generalization of that.

- To the first approximation, \( \Theta(g(n)) \) is:
  - The fastest-growing term of a polynomial
  - Minus the constant
\[ \Theta\left(g\left(n\right)\right) \]

\[ n^2(c_2 + c_3) + n(c_1 + c_4) + c_5 = \Theta\left(n^2\right) \]

\[ n^2 = \Theta\left(n^2\right) \]

\[ n^2 - 1000n = \Theta\left(n^2\right) \]

\[ n(c_1 + c_4) + c_5 = \Theta\left(n\right) \]

\[ 10n = \Theta\left(n\right) \]

\[ n - 1 = \Theta\left(n\right) \]
\[ \Theta(g(n)) \]

- Technically, \( \Theta(g(n)) \) is a **set of functions**.

- Our use of the = sign is not exactly what you think it is!

\[
a(n) = \Theta(n^2)\\
b(n) = \Theta(n^2)
\]

\[ a(n) \neq b(n) \]
\[ \Theta(g(n)) \]

\[ n^2 - 1000n = \Theta(n^2) \]
\[ n - 1 = \Theta(n) \]

The equality sign actually means “element of” in this case.

\[ n^2 - 1000n \in \{ \text{functions that grow like } n^2 \} \]
\[ n - 1 \in \{ \text{functions that grow like } n \} \]
Time for the formal definition...
\[\Theta(g(n)) = \{ f(n) : \exists c_1 > 0, c_2 > 0, n_0 > 0 \forall n > n_0 \}
\]

\[0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n)\]

Break this down into pieces. What does it mean?
\( \Theta(g(n)) \) is defined as...

\[
\Theta(g(n)) = \left\{ f(n) : \exists c_1 > 0, c_2 > 0, n_0 > 0 \right. \\
\forall n > n_0 \\
0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n) \right. 
\]
\[ \Theta(g(n)) = \{ f(n) : \exists c_1 > 0, c_2 > 0, n_0 > 0 \quad \forall n > n_0 \quad 0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n) \} \]

\( \Theta(g(n)) \) is defined as...a set of functions such that...
\( \Theta(g(n)) \)

\[ \begin{align*}
\Theta(g(n)) = & \{ f(n) : \exists c_1 > 0, c_2 > 0, n_0 > 0 \\
& \forall n > n_0 \}
\end{align*} \]

\( 0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n) \)

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

\[
\Theta(g(n)) = \left\{ f(n) : \exists c_1 > 0, c_2 > 0, n_0 > 0 \forall n > n_0 \right\}
\]

\[
0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n)
\]

\( \Theta(g(n)) \) 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 \( n_0 \)...
\[ \Theta(g(n)) \]

\[
\Theta(g(n)) = \{ f(n) : \exists c_1 > 0, c_2 > 0, n_0 > 0 \forall n > n_0 0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n) \}
\]

\( \Theta(g(n)) \) 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 \( n_0 \), \( f(n) \) is bounded by \( g(n) \).
Lower bound on $f(n)$: $f(n)$ cannot get too small.
\[ \Theta(g(n)) \]

\[ f(n): \exists c_1 > 0, c_2 > 0, n_0 > 0 \]
\[ \forall n > n_0 \]
\[ 0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n) \]

Upper bound on \( f(n) \): \( f(n) \) cannot get too large.
\( \Theta\left(g\left(n\right)\right) \)

Why so formal a definition?

So that we can prove theorems about it!
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.
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.
\[ \Theta(g(n)) \]

\[ q(n) = \Theta(n^2) \]
\[ l(n) = \Theta(n) \]

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

**NOTE:**
We're using poor reasoning here, we'll do a better version next.
\[ \Theta(g(n)) \]

\[ q(n) = \Theta(n^2) \]
\[ l(n) = \Theta(n) \]

\[ \lim_{n \to \infty} \frac{q(n)}{l(n)} \approx \lim_{n \to \infty} \frac{a n^2}{b n} = \lim_{n \to \infty} \frac{n}{b/a} = \infty \]
\( \Theta(g(n)) \)

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

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

\[
q(n) = \Theta(n^2) \\
l(n) = \Theta(n)
\]

\[
\lim_{n \to \infty} \frac{q(n)}{l(n)} \approx \lim_{n \to \infty} \frac{a n^2}{b n} = \lim_{n \to \infty} \frac{n}{b/a} = \infty
\]
\[ \Theta(g(n)) \]

\[ q(n) = \Theta(n^2) \]
\[ l(n) = \Theta(n) \]

These are the two functions.

\( q(n) \) is quadratic.
\( l(n) \) is linear.
We start by using the definition of $\Theta$ to define a **lower bound** for $q(n)$

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

\[
\Theta(g(n))
\]

\[
q(n) = \Theta(n^2) \\
l(n) = \Theta(n)
\]
\[ \Theta(g(n)) \]

\[ q(n) = \Theta(n^2) \]
\[ l(n) = \Theta(n) \]

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

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

We then use the definition of \( \Theta \) to define an upper bound for \( l(n) \).
\[ \Theta\left( g\left( n \right) \right) \]

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

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

For some large-enough \( n \), \( q\left( n \right) \) will always be larger than \( c_q n^2 \)

For some large-enough \( n \), \( l\left( n \right) \) will always be smaller than \( c_l n \)

\[ \frac{q\left( n \right)}{l\left( n \right)} \geq \frac{c_q n^2}{c_l n} \]
Therefore, the limit of the ratio of the functions...

is $\geq$ the limit of ratio of the two bounds...

which is infinity.
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
- $\Theta(g(n))$
- $O(g(n))$
- $\Omega(g(n))$, $o(g(n))$, $\omega(g(n))$
- $\log n$
- Ordering functions by growth
\( O(g(n)) \)

- Far more commonly used than \( \Theta(g(n)) \)

- but -

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

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

\[ O(g(n)) = \{ f(n) : ??? \} \]
\( O(g(n)) \) = \{ \text{functions that grow no faster than } g(n) \}

\[ f(n): \ \exists c > 0, n_0 > 0 \]
\[ O(g(n)) = \{ \forall n \geq n_0 \]
\[ 0 \leq f(n) \leq c g(n) \]
Why use $O(g(n))$ when analyzing programs?
- Do we even care about lower bound?
- Special cases may be fast.

Often, $O(g(n))$ is the right choice for simple analysis
- $\Theta(g(n))$ is useful for some proofs

Be clear!
### In Practice...

<table>
<thead>
<tr>
<th>Formal Definition</th>
<th>Informal Usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>$n^2 = O(n^2)$</td>
<td>$n^2 = O(n^2)$</td>
</tr>
<tr>
<td>$n = O(n^2)$</td>
<td>$n \neq O(n^2)$</td>
</tr>
<tr>
<td>$1 = O(n^2)$</td>
<td>$1 \neq O(n^2)$</td>
</tr>
</tbody>
</table>

(this class)  (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?

<table>
<thead>
<tr>
<th>Asymptotically Tight Bounds</th>
<th>NOT Asymptotically Tight Bounds</th>
</tr>
</thead>
<tbody>
<tr>
<td>$n^2 = O(n^2)$</td>
<td>$n = O(n^2)$</td>
</tr>
<tr>
<td>$n = O(n)$</td>
<td>$1 = O(n^2)$</td>
</tr>
<tr>
<td>$1 = O(1)$</td>
<td></td>
</tr>
</tbody>
</table>
A Metaphor

\[ \Theta(g(n)) \equiv O(g(n)) \land \land \land \land < \land > \]
Topic 03: Asymptotic Notation

- An Example Method
- $\Theta(g(n))$
- $O(g(n))$
- $\Omega(g(n)), \omega(g(n))$
- $\lg n$
- Ordering functions by growth
\( \Omega(g(n)) \) = \{ \text{funcs that grow at least as fast as } g(n) \}
\[ \Omega(g(n)) \]

\[ \Omega(g(n)) = \{ \text{funcs that grow at least as fast as } g(n) \} \]

\[ f(n) : \exists c > 0, n_0 > 0 \]

\[ \Omega(g(n)) = \{ \forall n \geq n_0 \]
\[ \quad c \cdot g(n) \leq f(n) \} \]
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: Upper and lower bounds
O: Upper bounds
Ω: Lower bounds

\[ \Theta(g(n)) \subseteq O(g(n)) \]
\[ \Theta(g(n)) \subseteq \Omega(g(n)) \]
\( o(g(n)) \)

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

\[
1 = o(n^2) \\
n = o(n^2) \\
n^2 \neq o(n^2)
\]
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 \( f(n) \).
\( \mathcal{o}(g(n)) \)

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

\[
o(g(n)) = \{ f(n) : \text{???} \}
\]

**Hint:**

We only need a single scaling constant – which we'll apply to \( g(n) \), just like the previous definitions.
$o(g(n))$ = \{funcs that grow more slowly than $g(n)$\}

\[f(n): \quad \forall c > 0 \quad \exists n_0 > 0 \quad \forall n \geq n_0 \quad 0 \leq f(n) < c g(n)\]
$o(g(n))$ = \{funcs that grow more slowly than $g(n)$\}

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

\[
l(n) = \Theta(lg^b n) \quad p(n) = \Theta(n^a)\]

Assume: $a > 0$
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 $h(n)$ ...

\[
\begin{align*}
l(n) &= \Theta(lg^bn) \\
p(n) &= \Theta(n^a) \\
l(n) &= o(p(n))
\end{align*}
\]

Assume: $a > 0$

We'll discuss the relationship of polynomials and logarithms in more depth later.
\omega(g(n)) = \{ \text{funcs that grow more quickly than } g(n) \}

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

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

\[ f(n): \quad \forall c > 0 \]
\[ \exists n_0 > 0 \]
\[ \forall n \geq n_0 \]
\[ 0 \leq c g(n) < f(n) \]
\[ o, \quad \omega \]

\[ f(n) = \omega(g(n)) \]

\[ g(n) \quad ? \quad f(n) \]
\[ f(n) = \omega(g(n)) \]

\[ g(n) = o(f(n)) \]
Topic 03: Asymptotic Notation

- An Example Method
- $\Omega(g(n))$
- $O(g(n))$
- $\Theta(g(n)), \omega(g(n)), \omega(g(n))$
- $\log n$

- Ordering functions by growth
Some Definitions

\[ \log n = \log_{10} n \]
\[ \ln n = \log_e n \]
\[ \lg n = \log_2 n \]

\[ \lg^k n = (\lg n)^k \]
\[ \lg \lg n = \lg (\lg n) \]

\[ \lg^* n = "iterated logarithm" \]
Log Identities

\[ a = b^{\log_b a} \]

\[ \log_c (ab) = \log_c a + \log_c b \]

\[ \log_b a^n = n \log_b a \]

\[ \log_b a = \frac{\log_c a}{\log_c b} \]

\[ \log_b a = \frac{1}{\log_a b} \]

\[ a^{\log_b c} = c^{\log_b a} \]
- Algorithms with logs in them are very common
- Logarithms grow very slowly

<table>
<thead>
<tr>
<th>n</th>
<th>( n^2 )</th>
<th>( \lg n )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>16</td>
<td>2</td>
</tr>
<tr>
<td>8</td>
<td>64</td>
<td>3</td>
</tr>
<tr>
<td>16</td>
<td>256</td>
<td>4</td>
</tr>
<tr>
<td>32</td>
<td>1024</td>
<td>5</td>
</tr>
</tbody>
</table>
\( \lg n = \Theta(\log n) = \Theta(\ln n) \)

Why?
We've already seen how polynomials and logs relate:

\[ \lg^n n = o(n^a) \quad \text{Assume:} \quad a > 0 \]

\[ \lg^{100} n = o(n) \]
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:

\[ a, b, \epsilon > 0 \]
So how do we compare these pairs of functions?

\[ n^a = o\left(n^a \log^b n\right) \]

\[ n^a \log^b n = o\left(n^{a+\epsilon}\right) \quad \text{Assume:} \quad a, b, \epsilon > 0 \]
$2^n$

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

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

- $2^n$ 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.

\[ n^a = o\left(2^n\right) \]
\[ \lg n^a = o\left(\lg 2^n\right) \]
\[ a \lg n = o\left(n\right) \]
Topic 03: Asymptotic Notation

- An Example Method
- \( \Omega(g(n)) \)
- \( O(g(n)) \)
- \( \Theta(g(n)), o(g(n)), \omega(g(n)) \)
- \( \log n \)

- 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

What it's Famous For

\( \Theta(1) \)
\( \Theta(\log n) \)
\( \Theta(n) \)
\( \Theta(n \log n) \)
\( \Theta(n^2) \)
\( \Theta(2^n) \)
\( \Theta(n^n) \)
Some Famous Families

<table>
<thead>
<tr>
<th>Time Complexity</th>
<th>What it's Famous For</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\Theta(1)$</td>
<td>simple ops</td>
</tr>
<tr>
<td>$\Theta(\log n)$</td>
<td>binary search</td>
</tr>
<tr>
<td>$\Theta(n)$</td>
<td>inspect all elements</td>
</tr>
<tr>
<td>$\Theta(n \log n)$</td>
<td>the best sorts</td>
</tr>
<tr>
<td>$\Theta(n^2)$</td>
<td>check all pairs</td>
</tr>
<tr>
<td>$\Theta(2^n)$</td>
<td>all boolean combinations</td>
</tr>
<tr>
<td>$\Theta(n^n)$</td>
<td>give up, you've lost</td>
</tr>
</tbody>
</table>
### Some Famous Families

<table>
<thead>
<tr>
<th></th>
<th>$n=1$</th>
<th>$n=10$</th>
<th>$n=1000$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\Theta(1)$</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>$\Theta(\log n)$</td>
<td>0</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>$\Theta(n)$</td>
<td>1</td>
<td>10</td>
<td>1000</td>
</tr>
<tr>
<td>$\Theta(n \log n)$</td>
<td>0</td>
<td>10</td>
<td>3000</td>
</tr>
<tr>
<td>$\Theta(n^2)$</td>
<td>1</td>
<td>100</td>
<td>10000000</td>
</tr>
<tr>
<td>$\Theta(2^n)$</td>
<td>2</td>
<td>1024</td>
<td>googol, cubed</td>
</tr>
<tr>
<td>$\Theta(n^n)$</td>
<td>1</td>
<td>100000000000</td>
<td>??</td>
</tr>
</tbody>
</table>
Noteworthy Names

\[ \Theta(1) \quad \text{"constant time"} \]
\[ \Theta(\log n) \quad \text{"log"} \]
\[ o(n) \quad \text{"sublinear"} \]
\[ \Theta(n) \quad \text{"linear"} \]
\[ \Theta(n \log n) \quad \text{"log-linear"} \]
\[ \Theta(n^k) \quad \text{"polynomial"} \]
\[ \Theta(2^{n^k}) \quad \text{"exponential"} \]
Topic 03: Asymptotic Notation

- An Example Method
- $\Omega(g(n))$
- $O(g(n))$
- $\omega(g(n))$, $o(g(n))$, $\omega(g(n))$
- $\lg n$

- Ordering functions by growth

Summary