Chpater 3. Characterizing Running Times

Joon Soo Yoo

March 20, 2025

Assignment

- Read §3.2, §3.3
- Problems:
  - §3.3 - #4(b)
  - Problems - #3-4,

Chapter 3: Characterizing Running Times

Overview of Asymptotic Notation

- Chapter 3.1: $O-$notation, $\Omega-$notation, and $\theta-$notation
- **Chapter 3.2: Asymptotic notation – formal definitions**
- Chapter 3.3: Standard notations and common functions

Big-O Notation: Formal Definition

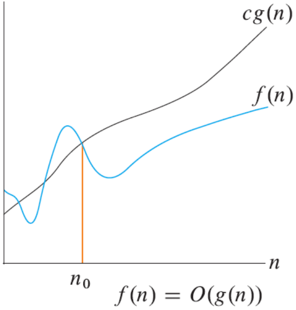
What is Big-O Notation?

- As seen in Section 3.1, O-notation provides an **asymptotic upper bound**.
- It describes how a function grows **at most** at the rate of another function, up to a constant factor.
- Formally, for a given function $g(n)$, the set $O(g(n))$ is defined as:

$$O(g(n)) = \{f(n) \mid \exists c > 0, n_0 > 0 \text{ such that } 0 \leq f(n) \leq cg(n), \forall n \geq n_0\}$$

- This definition ensures that $f(n)$ is bounded above by $g(n)$ for sufficiently large n.

Visualizing Big-O Notation



Interpretation:

- The function $f(n)$ is always below or equal to $cg(n)$ for $n \geq n_0$.
- This means $g(n)$ serves as an **upper bound** for $f(n)$ in the long run.

Key Assumptions in Big-O Notation

- The definition requires $f(n)$ to be **asymptotically nonnegative** for sufficiently large n.
- That means $f(n) \geq 0$ for all $n \geq n_0$.
- Likewise, the function $g(n)$ must also be **asymptotically nonnegative**.
- Otherwise, the set $O(g(n))$ would be empty.

Big-O Notation as a Set

- Mathematically, we define Big-O notation using **set notation**.
- A function $f(n)$ belongs to $O(g(n))$:

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

- However, in common usage, we often write:

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

- Even though this is an **abuse of equality**, it is widely accepted for simplicity.

Example: Showing $4n^2 + 100n + 500 = O(n^2)$

- Consider the function:

$$f(n) = 4n^2 + 100n + 500$$

- We need to find constants c and n_0 such that:

$$f(n) \leq cn^2, \quad \forall n \geq n_0$$

- Dividing by n^2:

$$4 + \frac{100}{n} + \frac{500}{n^2} \leq c$$

- Choosing $n_0 = 1$, $c = 604$ works.
- Choosing $n_0 = 10$, $c = 19$ also works.
- Thus, $f(n) \in O(n^2)$.

Big-Omega (Ω) Notation: Formal Definition

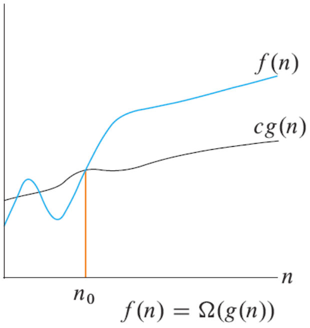
What is Big-Omega Notation?

- Just as O-notation provides an **asymptotic upper bound**, Ω-notation provides an **asymptotic lower bound**.
- It describes how a function grows **at least as fast** as another function, up to a constant factor.
- Formally, for a given function $g(n)$, the set $\Omega(g(n))$ is defined as:

$$\Omega(g(n)) = \{f(n) \mid \exists c > 0, n_0 > 0 \text{ such that } 0 \leq cg(n) \leq f(n), \forall n \geq n_0\}$$

- This definition ensures that $f(n)$ is bounded below by $g(n)$ for sufficiently large n.

Visualizing Big-Omega (Ω)



Interpretation:

- The function $f(n)$ is always above or equal to $cg(n)$ for $n \geq n_0$.
- This means $g(n)$ serves as a **lower bound** for $f(n)$ in the long run.

Example: Showing $4n^2 + 100n + 500 = \Omega(n^2)$

- We have the function:

$$f(n) = 4n^2 + 100n + 500$$

- We need to find constants c and n_0 such that:

$$cn^2 \leq f(n), \quad \forall n \geq n_0$$

- Dividing by n^2:

$$c \leq 4 + \frac{100}{n} + \frac{500}{n^2}$$

- Choosing $n_0 = 1$, $c = 4$ works.
- Thus, $f(n) \in \Omega(n^2)$.

Example: Showing $\frac{n^2}{100} - 100n - 500 = \Omega(n^2)$

- Consider:

$$f(n) = \frac{n^2}{100} - 100n - 500$$

- We divide by n^2:

$$\frac{1}{100} - \frac{100}{n} - \frac{500}{n^2} \leq c$$

- Choosing $n_0 = 10,005$, we can set $c = 2.49 \times 10^{-9}$.
- If we choose a larger n_0, we can increase c closer to $\frac{1}{100}$.
- This proves that $f(n) \in \Omega(n^2)$.

Theta (Θ) Notation: Formal Definition

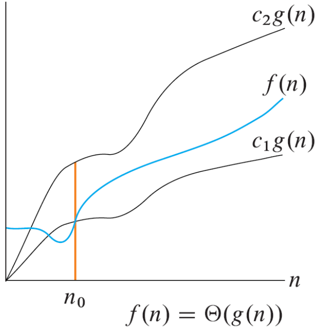
What is Theta Notation?

- We use Θ-notation to represent an **asymptotically tight bound**.
- It describes how a function grows **exactly at the rate** of another function, up to constant factors.
- Formally, for a given function $g(n)$, the set $\Theta(g(n))$ is defined as:

$$\Theta(g(n)) = \{f(n) \mid \exists c_1, c_2 > 0, n_0 > 0 \text{ such that } 0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n), \forall n \geq n_0\}$$

- This means $f(n)$ is bounded both **above and below** by $g(n)$ within constant factors.

Visualizing Theta (Θ)



Interpretation:

- The function $f(n)$ is sandwiched between two constant multiples of $g(n)$.
- This means $g(n)$ provides an **exact rate of growth** for $f(n)$.

Theorem: Relationship Between $O(g(n)), \Omega(g(n))$, and $\Theta(g(n))$

Theorem 3.1:

- A function $f(n) = \Theta(g(n))$ if and only if

$$f(n) = O(g(n)) \quad \text{and} \quad f(n) = \Omega(g(n))$$

- This means if $f(n)$ is both upper and lower bounded by $g(n)$, then $f(n)$ is tightly bound.

Proof of Theorem 3.1: If $f(n)$ is both $O(g(n))$ and $\Omega(g(n))$, then $f(n) = \Theta(g(n))$

Proof:

Backward direction: Assume $f(n) = O(g(n))$ and $f(n) = \Omega(g(n))$.

- Since $f(n) = O(g(n))$, there exist constants $c_2 > 0$ and $n_2 > 0$ such that:

$$f(n) \leq c_2 g(n), \quad \forall n \geq n_2$$

- Since $f(n) = \Omega(g(n))$, there exist constants $c_1 > 0$ and $n_1 > 0$ such that:

$$f(n) \geq c_1 g(n), \quad \forall n \geq n_1$$

- Let $n_0 = \max(n_1, n_2)$. Then, for $n \geq n_0$, we have:

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

- This matches the definition of $\Theta(g(n))$, so $f(n) = \Theta(g(n))$.

Proof of Theorem 3.1: If $f(n) = \Theta(g(n))$, then $f(n) = O(g(n))$ and $f(n) = \Omega(g(n))$

Proof:

Forward direction: Assume $f(n) = \Theta(g(n))$.

- By definition of $\Theta(g(n))$, there exist positive constants c_1, c_2 and n_0 such that:

$$c_1 g(n) \leq f(n) \leq c_2 g(n), \quad \forall n \geq n_0$$

- **To show $f(n) = O(g(n))$:**
  - From the upper bound $f(n) \leq c_2 g(n)$, we see that $f(n)$ is at most a constant multiple of $g(n)$ for sufficiently large n.
  - This satisfies the definition of $O(g(n))$, so $f(n) \in O(g(n))$.
- **To show $f(n) = \Omega(g(n))$:**
  - From the lower bound $c_1 g(n) \leq f(n)$, we see that $f(n)$ is at least a constant multiple of $g(n)$ for sufficiently large n.
  - This satisfies the definition of $\Omega(g(n))$, so $f(n) \in \Omega(g(n))$.
- Since we have shown both $f(n) = O(g(n))$ and $f(n) = \Omega(g(n))$, it follows that:

$$f(n) = O(g(n)) \quad \text{and} \quad f(n) = \Omega(g(n)) \Rightarrow f(n) = \Theta(g(n))$$

Problem: Prove Transitivity of Θ-Notation

Prove:

$$f(n) = \Theta(g(n)) \quad \text{and} \quad g(n) = \Theta(h(n)) \quad \Rightarrow \quad f(n) = \Theta(h(n)).$$

Proof of Transitivity of Θ-Notation

Proof:

- By the definition of Θ-notation, $\exists c_1, c_2, c_3, c_4 > 0$ and $n_1, n_2 > 0$ such that:

$$\forall n \geq n_1, \quad c_1 g(n) \leq f(n) \leq c_2 g(n).$$

$$\forall n \geq n_2, \quad c_3 h(n) \leq g(n) \leq c_4 h(n).$$

- Define $n_0 = \max(n_1, n_2)$. For all $n \geq n_0$
- Substituting $g(n)$ using the second inequality:

$$c_1(c_3 h(n)) \leq f(n) \leq c_2(c_4 h(n)).$$

- Simplifying:

$$(c_1 c_3)h(n) \leq f(n) \leq (c_2 c_4)h(n).$$

- Since $c_1 c_3 > 0$ and $c_2 c_4 > 0$, this shows:

$$f(n) = \Theta(h(n)) \quad \square$$

Asymptotic Notation in Equations and Inequalities

Key Idea:

- Asymptotic notation is formally defined in terms of **sets**.
- However, in equations, we often write:

$$f(n) = O(g(n)) \quad \text{instead of} \quad f(n) \in O(g(n)).$$

- This means we use the equal sign $(=)$ to represent **set membership** $(\in)$.

Interpreting Asymptotic Notation in Formulas

How to Interpret Asymptotic Notation in Equations:

- When asymptotic notation appears in a formula, it represents an anonymous function.
- Example:
$$2n^2 + 3n + 1 = 2n^2 + \Theta(n)$$
- This means there exists some function $f(n) \in \Theta(n)$ such that:
$$2n^2 + 3n + 1 = 2n^2 + f(n),$$
where $f(n) = 3n + 1$.

Why Use Asymptotic Notation in Formulas?

Eliminating Inessential Details

- Asymptotic notation simplifies expressions by ignoring **lower-order terms**.
- Example:

$$T(n) = 2T(n/2) + \Theta(n)$$

- Instead of writing out the full details, we use $\Theta(n)$ to hide unnecessary complexity.
- This makes equations **clearer** and **easier to analyze**.

Anonymous Functions in Asymptotic Notation

Key Rule:

- The number of **anonymous functions** in an expression equals the number of asymptotic notation terms.
- Example:

$$\sum_{i=1}^{n} O(i)$$

- This is **not** the same as:

$$O(1) + O(2) + \cdots + O(n),$$

since the latter has multiple anonymous functions.

Asymptotic Notation on the Left-Hand Side

Interpreting Equations with Asymptotic Notation:

- When notation appears on the left-hand side, it means:
- "For any function chosen on the left, there exists a function on the right to make the equation valid."
- Example:

$$2n^2 + \Theta(n) = \Theta(n^2)$$

- This means for any function $f(n) \in \Theta(n)$, there is a function $g(n) \in \Theta(n^2)$ such that:

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

Step 1 & Step 2: Chaining Asymptotic Notation

Step 1: Group Lower-Order Terms in $\Theta(n)$

- Since $3n + 1$ grows at most linearly, we recognize it belongs to $\Theta(n)$, so we rewrite:

$$2n^2 + 3n + 1 = 2n^2 + \Theta(n)$$

- Instead of keeping the exact form of $3n + 1$, we just say it belongs to $\Theta(n)$.

Step 2: Identify Dominant Term

- We now focus on:

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

- Since the quadratic term dominates for large n, the whole expression must behave like $\Theta(n^2)$, so:

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

Step 3: Conclude the Final Result

Final Step: Chaining the Results

- Since each step is valid, we chain them together to conclude:

$$2n^2 + 3n + 1 = \Theta(n^2).$$

- The asymptotic notation hides **lower-order terms** and focuses on the dominant growth rate.

Proper Abuses of Asymptotic Notation

Why Do We Abuse Asymptotic Notation?

- Asymptotic notation is formally defined using sets, but we often simplify notation for clarity.
- Some common "abuses" are widely accepted because they make expressions more readable.
- As long as we **don't misuse** these conventions, they help focus on **essential growth behavior**.

Abuse #1: Using $=$ Instead of $\in$

Set Membership vs. Equality

- Asymptotic notation represents **sets of functions**, so formally we should write:
$$f(n) \in O(g(n))$$
- However, for convenience, we usually write:
$$f(n) = O(g(n))$$
- This is not strict equality; it just means $f(n)$ belongs to the set $O(g(n))$.
- This abuse is **acceptable** as long as we interpret it correctly.

Abuse #2: Inferring the Variable from Context

Interpreting $O(1)$ in Context

- Asymptotic notation depends on the variable **tending to infinity**.
- Normally, we assume:

$$O(g(n)) \quad \text{means growth as } n \to \infty.$$

- Writing $T(n) = O(1)$ is **technically ambiguous**, as it does not explicitly state which variable is growing.
- However, by convention, we understand that n is the variable tending to infinity.
- This abuse is widely accepted because:
  - It simplifies notation.
  - The context usually makes it clear which variable is growing.

Abuse #3: Using $O(1)$ for Small n

Example:

- We often write:

$$T(n) = O(1) \quad \text{for } n < 3.$$

- **Problem** The formal definition of O-notation only applies for sufficiently large n, so this notation is technically meaningless.
- **What it actually means**

$$\exists c > 0 \text{ such that } T(n) \leq c \text{ for all } n < 3.$$

- This allows us to avoid explicitly naming constants while keeping expressions clean.

Abuse #4: Using Asymptotic Notation with Undefined Functions

What If a Function is Not Defined for Some n?

- Some algorithms are only defined for specific input sizes (e.g., powers of 2).
- We still use asymptotic notation, but only where the function is **defined**
- Example:

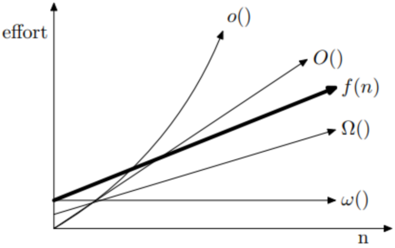
$$T(n) = O(n \log n), \quad \text{where } n \text{ is a power of 2.}$$

- **Interpretation**: The bound holds for all valid values of n, even if $T(n)$ is undefined for some inputs.

Key Takeaways: Properly Abusing Asymptotic Notation

- Abuse #1: We write $f(n) = O(g(n))$ instead of $f(n) \in O(g(n))$ for simplicity.
- Abuse #2: We assume the variable tending to infinity is clear from context.
- Abuse #3: We sometimes use $O(1)$ for small values of n, even though it's formally meaningless.
- Abuse #4: We use asymptotic notation even for functions that are only defined for certain input sizes.
- **Asymptotic notation is meant to simplify analysis**, but we must use it correctly to avoid incorrect conclusions.

Little-O (o)-Notation: Definition



What is Little-O Notation?

- $O(g(n))$ provides an upper bound, but it may or may not be **asymptotically tight**.
- $o(g(n))$ is a **strictly looser bound** than $O(g(n))$, meaning $f(n)$ grows **strictly slower** than $g(n)$.
- **Formal Definition:**

$$o(g(n)) = \{f(n) \mid \forall c > 0, \exists n_0 > 0 \text{ such that } 0 \leq f(n) < cg(n), \forall n \geq n_0\}$$

Intuition Behind Little-O Notation

Key Difference Between O and o

- $O(g(n))$ allows $f(n)$ to grow **at the same rate as** $g(n)$.
- $o(g(n))$ requires $f(n)$ to grow **strictly slower than** $g(n)$.

Mathematical Intuition:

- If $f(n) = o(g(n))$, then:

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

- This means $f(n)$ **becomes insignificant relative to** $g(n)$ as $n \to \infty$.

Little-Omega (ω)-Notation: Definition

What is Little-Omega Notation?

- Just as $o(g(n))$ is a **strict** upper bound, $\omega(g(n))$ is a **strict** lower bound.
- We write $f(n) = \omega(g(n))$ if $f(n)$ grows **strictly faster** than $g(n)$.
- **Formal Definition:**

$$\omega(g(n)) = \{f(n) \mid \forall c > 0, \exists n_0 > 0 \text{ such that } 0 \leq cg(n) < f(n), \forall n \geq n_0\}$$

Intuition Behind Little-Omega Notation

Key Difference Between Ω and ω

- $\Omega(g(n))$ allows $f(n)$ to grow **at least as fast as** $g(n)$.
- $\omega(g(n))$ requires $f(n)$ to grow **strictly faster than** $g(n)$.

Mathematical Intuition:

- If $f(n) = \omega(g(n))$, then:

$$\lim_{n \to \infty} \frac{f(n)}{g(n)} = \infty.$$

- This means $f(n)$ **becomes arbitrarily large relative to** $g(n)$ as $n \to \infty$.

Summary of O, Ω, o, ω Notation

- $O(g(n))$: Upper bound (may be tight).
- $\Omega(g(n))$: Lower bound (may be tight).
- $o(g(n))$: Strict upper bound (not tight).
- $\omega(g(n))$: Strict lower bound (not tight).

Mathematical Limits:

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} = \begin{cases} 0, & \text{if } f(n) = o(g(n)) \\ \infty, & \text{if } f(n) = \omega(g(n)) \\ \text{constant}, & \text{if } f(n) = \Theta(g(n)) \end{cases}$$

Transitivity of Asymptotic Notation

Transitivity Rules:

- If $f(n) = \Theta(g(n))$ and $g(n) = \Theta(h(n))$, then:
$$f(n) = \Theta(h(n)).$$
- If $f(n) = O(g(n))$ and $g(n) = O(h(n))$, then:
$$f(n) = O(h(n)).$$
- If $f(n) = \Omega(g(n))$ and $g(n) = \Omega(h(n))$, then:
$$f(n) = \Omega(h(n)).$$
- If $f(n) = o(g(n))$ and $g(n) = o(h(n))$, then:
$$f(n) = o(h(n)).$$
- If $f(n) = \omega(g(n))$ and $g(n) = \omega(h(n))$, then:
$$f(n) = \omega(h(n)).$$

Reflexivity of Asymptotic Notation

Reflexivity Rules:

- Every function is asymptotically related to itself:

$$f(n) = \Theta(f(n)).$$

$$f(n) = O(f(n)).$$

$$f(n) = \Omega(f(n)).$$

Proof of Reflexivity for Θ-Notation

To Prove: Every function satisfies:

$$f(n) = \Theta(f(n)).$$

Proof:

- By definition, $f(n) \in \Theta(f(n))$ if there exist **positive constants** $c_1, c_2 > 0$ and a **threshold** $n_0 > 0$ such that:

  $$c_1 f(n) \leq f(n) \leq c_2 f(n), \quad \forall n \geq n_0.$$

- Choosing $c_1 = 1$, $c_2 = 1$, and any $n_0 > 0$, we get:

  $$f(n) \leq f(n) \leq f(n),$$

  which trivially holds for all $n \geq n_0$.
- Hence, by definition of Θ-notation:

  $$f(n) = \Theta(f(n)).$$

Transpose Symmetry

Transpose Symmetry Rules:

- $$f(n) = O(g(n)) \iff g(n) = \Omega(f(n)).$$
- $$f(n) = o(g(n)) \iff g(n) = \omega(f(n)).$$

Analogy Between Asymptotic Notation and Real Numbers

Asymptotic notation can be compared to real number inequalities:

- $f(n) = O(g(n))$ is like $a \leq b$.
- $f(n) = \Omega(g(n))$ is like $a \geq b$.
- $f(n) = \Theta(g(n))$ is like $a = b$.
- $f(n) = o(g(n))$ is like $a < b$.
- $f(n) = \omega(g(n))$ is like $a > b$.

Trichotomy in Real Numbers vs. Asymptotic Notation

Trichotomy Property (Real Numbers)

- For any two real numbers a and b, exactly one of the following must hold:

$$a < b, \quad a = b, \quad a > b.$$

Why Trichotomy Does Not Hold in Asymptotic Notation

- Not all functions are asymptotically comparable.
- There exist functions $f(n)$ and $g(n)$ such that neither:

$$f(n) = O(g(n)) \quad \text{nor} \quad f(n) = \Omega(g(n))$$

holds.

Why Trichotomy Fails in Asymptotic Notation

Why Trichotomy Does Not Hold for Asymptotic Notation

- Consider $g(n) = n^{1+\sin n}$, where $\sin n$ oscillates between -1 and 1.
- This means:
$$1 \leq g(n) \leq n^2.$$
- If we try to compare $g(n)$ with $f(n) = n$, we find:
  - $g(n)$ is sometimes **larger than** n (i.e., $g(n) = n^2$).
  - $g(n)$ is sometimes **smaller than** n (i.e., $g(n) = 1$).
- Since neither $g(n) = O(n)$ nor $g(n) = \Omega(n)$ is always true, **trichotomy fails**.

Chapter 3: Characterizing Running Times

Overview of Asymptotic Notation

- Chapter 3.1: $O-$notation, $\Omega-$notation, and $\theta-$notation
- Chapter 3.2: Asymptotic notation – formal definitions
- **Chapter 3.3: Standard notations and common functions**

Monotonic Functions

Definition: A function $f(n)$ is:

- **Monotonically increasing** if:

$$m \leq n \Rightarrow f(m) \leq f(n).$$

- **Monotonically decreasing** if:

$$m \leq n \Rightarrow f(m) \geq f(n).$$

- **Strictly increasing** if:

$$m < n \Rightarrow f(m) < f(n).$$

- **Strictly decreasing** if:

$$m < n \Rightarrow f(m) > f(n).$$

Floor and Ceiling Functions

Definition:

- The **floor function** $\lfloor x \rfloor$ gives the greatest integer $\leq x$.
- The **ceiling function** $\lceil x \rceil$ gives the smallest integer $\geq x$.

Example Values:

- $\lfloor 3.7 \rfloor = 3, \quad \lceil 3.7 \rceil = 4.$
- $\lfloor -2.3 \rfloor = -3, \quad \lceil -2.3 \rceil = -2.$
- $\lfloor 5 \rfloor = 5, \quad \lceil 5 \rceil = 5.$

Properties of Floor and Ceiling Functions

Basic Properties: For any integer n,

$$\lfloor n \rfloor = n = \lceil n \rceil.$$

For all real numbers x, we have:

$$x - 1 < \lfloor x \rfloor \leq x \leq \lceil x \rceil < x + 1.$$

Negation Property:

$$-\lfloor x \rfloor = \lceil -x \rceil, \quad -\lceil x \rceil = \lfloor -x \rfloor.$$

Modular Arithmetic

Definition: For any integer a and any positive integer n,

$$a \mod n = a - n\lfloor a/n \rfloor.$$

Properties:

- $0 \leq a \mod n < n$, even for negative a.
- $a \equiv b \pmod{n}$ if $(a \mod n) = (b \mod n)$.
- $a \equiv b \pmod{n} \iff n$ divides $(a - b)$.

Polynomials

Definition: A polynomial of degree d is:

$$p(n) = \sum_{i=0}^{d} a_i n^i, \quad \text{where } a_d \neq 0.$$

Properties:

- $p(n)$ is **asymptotically positive** if $a_d > 0$.
- If $p(n)$ is asymptotically positive, then:

  $$p(n) = \Theta(n^d).$$

- n^a is **monotonically increasing** for $a \geq 0$, and **monotonically decreasing** for $a \leq 0$.
- A function is **polynomially bounded** if:

  $$f(n) = O(n^k) \text{ for some constant } k.$$

Exponentials

Basic Properties: For any $a > 0$, and integers m, n,

$$a^0 = 1, \quad a^1 = a, \quad a^{-1} = \frac{1}{a}, \quad (a^m)^n = a^{mn}.$$

$$a^m a^n = a^{m+n}, \quad \text{and} \quad \frac{a^m}{a^n} = a^{m-n}.$$

Growth Comparison:

- If $a > 1$, the function a^n is **monotonically increasing**.
- If $a < 1$, the function a^n is **monotonically decreasing**.
- For all $a > 1$ and any b, we have:

$$\lim_{n \to \infty} \frac{n^b}{a^n} = 0, \quad \text{so} \quad n^b = o(a^n).$$

- Any exponential function with $a > 1$ grows **faster than any polynomial**.

Logarithmic Functions

Notations:

$$\lg n = \log_2 n, \quad \ln n = \log_e n, \quad \lg^k n = (\lg n)^k.$$

Properties: For any $a, b, c > 0$ and n:

- $a = b^{\log_b a}$.
- $\log_c(ab) = \log_c a + \log_c b$.
- $\log_b a^n = n \log_b a$.
- $\log_b a = \frac{\log_c a}{\log_c b}$.

Growth Comparison:

$$\lg^b n = o(n^a) \quad \text{for any } a > 0.$$

- Any positive polynomial function grows **faster** than any polylogarithmic function.

Factorials and Stirling's Approximation

Definition: The factorial function $n!$ is defined as:

$$n! = \begin{cases} 1, & n = 0, \\ n \cdot (n-1)! & n > 0. \end{cases}$$

Growth Bounds:

- Weak upper bound: $n! \leq n^n$.
- Stirling's Approximation:

$$n! \approx \sqrt{2\pi n} \left(\frac{n}{e}\right)^n \left(1 + \Theta\left(\frac{1}{n}\right)\right).$$

Question?