Analysis of Algorithms

Quick Reference Guide

Notations

Symbol	Concept	Definition
Θ	Tight Bound	$c_1 g \leq f \leq c_2 g$
O	Upper Bound	$f \leq cg$
Ω	Lower Bound	$cg \leq f$
o	Strict Upper	$f < cg$
ω	Strict Lower	$cg < f$

Interpretations

- $O(g(n))$: f grows no faster than g.
- $\Omega(g(n))$: f grows at least as fast as g.
- $\Theta(g(n))$: f grows at the same rate as g.

Complexity Classes

Constant $O(1)$ Operations independent of input size. *Ex: Hash table lookup, Array index.*
Logarithmic $O(\lg n)$ Problem size reduced by constant fraction. *Ex: Binary Search.*
Linear $O(n)$ Processing each element once. *Ex: Linear Search, Iteration.*
Linearithmic $O(n \lg n)$ Divide and conquer algorithms. *Ex: Merge Sort, Heap Sort.*
Quadratic $O(n^2)$ Nested loops over data. *Ex: Bubble Sort, Matrix Add.*

Fundamental Laws

Transitivity If $f(n) \in O(g(n))$ and $g(n) \in O(h(n))$, then $f(n) \in O(h(n))$.
Symmetry $f(n) \in \Theta(g(n))$ if and only if $g(n) \in \Theta(f(n))$.
Rule of Sums $O(f(n) + g(n)) = O(\max(f(n), g(n)))$.

Limit Comparisons

To compare $f(n)$ and $g(n)$, evaluate:

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

- If $L = 0$: $f \in o(g)$ (f is smaller)
- If $0 < L < \infty$: $f \in \Theta(g)$ (Same growth)
- If $L = \infty$: $f \in \omega(g)$ (f is larger)

Mathematical Identities

Logarithms $a = b^{\log_b a}$
$\log_b a = \frac{\ln a}{\ln b}$
$\log(n!) = \Theta(n \lg n)$
Exponentials $a^{m+n} = a^m a^n$
$(a^m)^n = a^{mn}$
$n^b \ll a^n$ for any $b, a > 1$.
Series $\sum_{i=1}^{n} i = \frac{n(n+1)}{2} \approx \frac{n^2}{2}$
$\sum_{i=0}^{n} x^i = \frac{x^{n+1} - 1}{x - 1}$

The Master Theorem

Used to solve recurrences of the form:

$$T(n) = aT(n/b) + f(n)$$

1. If $f(n)$ is polynomially smaller than $n^{\log_b a}$, then $T(n) = \Theta(n^{\log_b a})$.
2. If $f(n)$ is approx equal to $n^{\log_b a}$, then $T(n) = \Theta(n^{\log_b a} \lg n)$.
3. If $f(n)$ is polynomially larger, then $T(n) = \Theta(f(n))$.

Guidelines

Drop constants. Efficiency scales with N.
Dominant terms only. $n^2 + n \approx n^2$.
Base is irrelevant. $\log_2 n \approx \log_{10} n$.