Part I

Preface

Chapter 1

Purpose and Scope of This Book

1.1 What This Book Covers

1.2 What This Book Does Not Cover

1.3 Target Audience: Students, Researchers, and Practitioners

Chapter 2

Why "Precise Analysis" Matters — From Theory to Engineering

2.1 The Gap Between Theoretical Complexity and Real-World Performance

2.2 Case Studies: When Big-O Isn't Enough

2.3 The Role of Constants, Lower-Order Terms, and Hardware

Chapter 3

Mathematical and Algorithmic Prerequisites

3.1 Discrete Mathematics

3.1.1 Sets, Functions, and Relations

3.1.2 Combinatorics: Permutations, Combinations, and Binomial Coefficients

3.1.3 Graph Theory Basics

3.1.4 Proof Techniques: Induction, Contradiction, and Contrapositive

3.2 Elementary Probability Theory

3.2.1 Sample Spaces, Events, and Probability Measures

3.2.2 Random Variables and Expectations

3.2.3 Basic Distributions: Uniform, Bernoulli, Geometric, Binomial

3.2.4 Linearity of Expectation

3.2.5 Conditional Probability and Independence

3.2.6 Variance and Standard Deviation

3.2.7 Moment Generating Functions (Brief Introduction)

3.3 Mathematical Analysis

3.3.1 Limits, Continuity, and Asymptotic Behavior

Chapter 4

Structure of the Book: Theorems, Proofs, Examples, and Exercises

4.1 How to Read This Book

4.2 Notation and Conventions

4.3 Types of Exercises: Conceptual, Computational, and Proof-Based

4.4 Using Examples Effectively

4.5 The Role of Rigor vs. Intuition

Chapter 5

Primary References and Parallel Reading Guide

5.1 Classic Textbooks (CLRS, Sedgewick, Kleinberg-Tardos)

5.2 Research Papers and Monographs

5.3 Online Resources and Lecture Notes

5.4 Recommended Reading Order and Study Plans

Part II

Foundations of Algorithmic Analysis

Chapter 6

Introduction to Algorithm Analysis

6.1 What Is Algorithm Analysis?

6.1.1 Correctness vs. Efficiency

6.1.2 Resource Measures: Time, Space, Energy, I/O

6.1.3 The Need for Mathematical Models

6.2 The RAM Model of Computation

6.2.1 Basic Operations and Unit-Cost Assumption

6.2.2 Memory Access Model

6.2.3 Limitations and Extensions of the RAM Model

6.3 Measuring Algorithm Performance

6.3.1 Input Size and Problem Instances

6.3.2 Counting Basic Operations

6.3.3 Exact vs. Asymptotic Analysis

6.4 Overview of Complexity Classes

6.4.1 P, NP, NP-Complete, and NP-Hard (Brief Introduction)

6.4.2 Why We Focus on Polynomial-Time Algorithms

Chapter 7

Asymptotic Notation

7.1 The Need for Asymptotic Analysis

7.1.1 Why Exact Counts Are Often Impractical

7.1.2 Growth Rates and Scalability

7.2 Big-O Notation (O)

7.2.1 Formal Definition

7.2.2 Intuition: Upper Bounds

7.2.3 Common Functions and Their Growth Rates

7.2.4 Examples and Non-Examples

7.2.5 Properties of Big-O

Transitivity

Addition and Multiplication Rules

Reflexivity and Asymmetry

7.3 Big-Omega Notation (Ω)

7.3.1 Formal Definition

7.3.2 Intuition: Lower Bounds

7.3.3 Examples and Applications

7.3.4 Relationship Between O and Ω

7.4 Big-Theta Notation (Θ)

7.4.1 Formal Definition

Chapter 8

Recurrence Relations and Their Solutions

8.1 Introduction to Recurrence Relations

8.1.1 What Are Recurrences?

8.1.2 Why They Arise in Algorithm Analysis

8.1.3 Examples from Divide-and-Conquer Algorithms

8.2 The Substitution Method

8.2.1 Guessing the Solution

8.2.2 Proving by Induction

8.2.3 Examples: Mergesort, Binary Search

8.2.4 Strengthening the Inductive Hypothesis

8.3 The Recursion-Tree Method

8.3.1 Visualizing the Recurrence

8.3.2 Summing Over Levels

8.3.3 Examples and Illustrations

8.3.4 Limitations and When to Use

8.4 The Master Theorem

8.4.1 Statement of the Master Theorem (Standard Form)

8.4.2 Three Cases and Their Intuition

Chapter 9

Best-Case, Worst-Case, and Average-Case Analysis

9.1 Defining Input Classes

9.1.1 What Constitutes an "Input"?

9.1.2 Problem Instances and Instance Distributions

9.2 Best-Case Analysis

9.2.1 Definition and Purpose

9.2.2 Examples: Insertion Sort, Linear Search

9.2.3 When Best-Case Matters (and When It Doesn't)

9.3 Worst-Case Analysis

9.3.1 Definition and Motivation

9.3.2 Guarantees and Robustness

9.3.3 Examples: Quicksort, Searching in Unsorted Arrays

9.3.4 Lower Bounds and Optimality

9.4 Average-Case Analysis

9.4.1 Definition: Expected Running Time

9.4.2 Assumptions About Input Distributions

9.4.3 Probabilistic Models: Uniform, Gaussian, etc.

9.4.4 Examples: Quicksort, Hashing, Skip Lists

Chapter 10

Probabilistic Analysis of Algorithms

10.1 Foundations of Probabilistic Analysis

10.1.1 Random Variables in Algorithm Analysis

10.1.2 Indicator Random Variables

10.1.3 Linearity of Expectation

10.2 Expected Running Time

10.2.1 Formal Definition

10.2.2 Computing Expectations via Indicator Variables

10.2.3 Examples: Hiring Problem, Randomized Quicksort

10.3 Probabilistic Bounds

10.3.1 Markov's Inequality

10.3.2 Chebyshev's Inequality

10.3.3 Chernoff Bounds

10.3.4 Applications to Load Balancing and Hashing

10.4 Randomized Algorithms

10.4.1 Randomized Quicksort (Detailed Analysis)

10.4.2 Randomized Selection (Quickselect)

10.4.3 Hashing and Universal Hash Functions

10.4.4 Bloom Filters and Probabilistic Data Structures

Part III

Advanced Analysis Techniques

Chapter 11

Amortized Analysis

11.1 Introduction to Amortized Analysis

11.1.1 Motivation: Why Average Per-Operation Cost?

11.1.2 Amortized vs. Average-Case Analysis

11.1.3 When to Use Amortized Analysis

11.2 Aggregate Analysis

11.2.1 Definition and Methodology

11.2.2 Example: Dynamic Array (Vector) Resizing

11.2.3 Example: Binary Counter Increment

11.2.4 Example: Stack with Multipop

11.3 The Accounting Method

11.3.1 Conceptual Framework: Credits and Debits

11.3.2 Defining Amortized Costs

11.3.3 Example: Dynamic Array via Accounting

11.3.4 Example: Splay Trees (Introduction)

11.3.5 Ensuring Non-Negative Credit Balance

11.4 The Potential Method

11.4.1 Potential Functions: Definition and Intuition

11.4.2 Relating Amortized Cost to Actual Cost

Chapter 12

Space Complexity Analysis

12.1 Introduction to Space Complexity

12.1.1 Why Space Matters

12.1.2 Types of Memory: Stack, Heap, Static

12.1.3 In-Place vs. Out-of-Place Algorithms

12.2 Measuring Space Usage

12.2.1 Auxiliary Space vs. Total Space

12.2.2 Recursive Call Stack Depth

12.2.3 Implicit vs. Explicit Data Structures

12.3 Examples of Space Complexity Analysis

12.3.1 Iterative Algorithms: Loops and Arrays

12.3.2 Recursive Algorithms: Mergesort, Quicksort

12.3.3 Dynamic Programming: Memoization vs. Tabulation

12.3.4 Graph Algorithms: BFS, DFS, Shortest Paths

12.4 Space-Time Tradeoffs

12.4.1 Caching and Memoization

12.4.2 Lookup Tables and Precomputation

12.4.3 Compression and Succinct Data Structures

12.5 Streaming and Online Algorithms

Chapter 13

Cache-Aware and I/O Complexity

13.1 Introduction to the Memory Hierarchy

13.1.1 Registers, Cache (L1, L2, L3), RAM, Disk

13.1.2 Latency and Bandwidth Characteristics

13.1.3 Why Algorithm Design Must Consider Memory

13.2 The External Memory Model (I/O Model)

13.2.1 Parameters: N (data size), M (memory size), B (block size)

13.2.2 I/O Complexity: Counting Block Transfers

13.2.3 Comparison with RAM Model

13.3 I/O-Efficient Algorithms

13.3.1 Scanning and Sorting

External Merge Sort

I/O Complexity: $O((N/B)\log_{M/B}(N/B))$

13.3.2 Matrix Operations

Matrix Transposition

Matrix Multiplication

13.3.3 Graph Algorithms

I/O-Efficient BFS and DFS

Minimum Spanning Tree

13.4 Cache-Oblivious Algorithms

Chapter 14

Cache-Aware Scheduling and Analysis for Multicores

14.1 Introduction to Multicore and Parallel Computing

14.1.1 Shared vs. Distributed Memory

14.1.2 Parallel Models: PRAM, Fork-Join, Work-Stealing

14.1.3 Performance Metrics: Work, Span, Parallelism

14.2 Cache Coherence and Consistency

14.2.1 MESI and MOESI Protocols

14.2.2 False Sharing in Multicore Systems

14.2.3 Impact on Algorithm Design

14.3 Cache-Aware Parallel Algorithms

14.3.1 Parallel Sorting with Cache Awareness

14.3.2 Parallel Matrix Multiplication (Strassen, Coppersmith-Winograd)

14.3.3 Load Balancing and Task Granularity

14.4 Real-Time and Embedded Systems

14.4.1 WCET Analysis in Cache-Aware Contexts

14.4.2 Predictability vs. Average-Case Performance

14.4.3 Cache Partitioning and Locking

Part IV

Lower Bounds and Optimality

Chapter 15

Lower Bounds for Comparison-Based Algorithms

15.1 Decision Trees

15.1.1 Modeling Algorithms as Decision Trees

15.1.2 Height of Decision Trees and Worst-Case Complexity

15.2 Sorting Lower Bound

15.2.1 Information-Theoretic Argument

15.2.2 $\Omega(n \log n)$ Lower Bound for Comparison Sorting

15.2.3 Implications and Optimal Algorithms

15.3 Selection and Searching Lower Bounds

15.3.1 Finding the Minimum: $\Omega(n)$

15.3.2 Finding Median: Adversary Arguments

15.3.3 Searching in Sorted Arrays: $\Omega(\log n)$

15.4 Adversary Arguments

15.4.1 General Framework

15.4.2 Examples: Merging, Element Uniqueness

15.5 Exercises

Chapter 16

Algebraic and Non-Comparison Lower Bounds

16.1 Algebraic Decision Trees

16.1.1 Extending Beyond Comparisons

16.1.2 Element Distinctness Lower Bound

16.2 Communication Complexity

16.2.1 Models and Definitions

16.2.2 Applications to Data Structures

16.3 Cell-Probe Model

16.3.1 Lower Bounds for Data Structures

16.3.2 Dynamic vs. Static Data Structures

16.4 Exercises

Part V

Specialized Topics and Applications

Chapter 17

Analysis of Specific Algorithm Paradigms

17.1 Divide-and-Conquer Algorithms

17.1.1 General Framework and Recurrence Relations

17.1.2 Examples: Mergesort, Quicksort, Strassen's Algorithm

17.1.3 Optimality and Lower Bounds

17.2 Greedy Algorithms

17.2.1 Correctness via Exchange Arguments

17.2.2 Matroid Theory (Brief Introduction)

17.2.3 Examples: Huffman Coding, Kruskal's MST

17.3 Dynamic Programming

17.3.1 Optimal Substructure and Overlapping Subproblems

17.3.2 Memoization vs. Tabulation

17.3.3 Time and Space Complexity Analysis

17.3.4 Examples: Knapsack, Edit Distance, Matrix Chain Multiplication

17.4 Backtracking and Branch-and-Bound

17.4.1 Pruning the Search Space

17.4.2 Worst-Case Exponential, Average-Case Better

Chapter 18

Online Algorithms and Competitive Analysis

18.1 Introduction to Online Algorithms

18.1.1 Online vs. Offline Problems

18.1.2 Competitive Ratio

18.2 Examples of Online Problems

18.2.1 Paging and Caching (LRU, FIFO, LFU)

18.2.2 Load Balancing

18.2.3 Online Scheduling

18.3 Competitive Analysis Techniques

18.3.1 Deterministic vs. Randomized Algorithms

18.3.2 Lower Bounds via Adversary Arguments

18.4 Exercises

Chapter 19

Approximation Algorithms

19.1 Introduction to Approximation

19.1.1 NP-Hardness and Intractability

19.1.2 Approximation Ratios

19.2 Examples of Approximation Algorithms

19.2.1 Vertex Cover (2-Approximation)

19.2.2 Set Cover (Greedy, $\log n$-Approximation)

19.2.3 Traveling Salesman Problem (Metric TSP)

19.3 Analysis Techniques

19.3.1 Bounding Optimal Solutions

19.3.2 Linear Programming Relaxations

19.4 Exercises

Chapter 20

Parameterized Complexity

20.1 Introduction to Parameterized Algorithms

20.1.1 Fixed-Parameter Tractability (FPT)

20.1.2 Kernelization

20.2 Examples and Analysis

20.2.1 Vertex Cover Parameterized by Solution Size

20.2.2 Treewidth and Graph Algorithms

20.3 W-Hierarchy and Hardness

20.3.1 W[1], W[2], and Beyond

20.4 Exercises

Part VI

Practical Considerations and Case Studies

Chapter 21

From Theory to Practice

21.1 Hidden Constants and Lower-Order Terms

21.1.1 When $O(n \log n)$ Beats $O(n)$ in Practice

21.1.2 Empirical Performance Measurements

21.2 Algorithm Engineering

21.2.1 Profiling and Benchmarking

21.2.2 Tuning for Specific Hardware

21.2.3 Libraries and Implementations (STL, Boost, etc.)

21.3 Parallel and Distributed Algorithm Analysis

21.3.1 Scalability and Speedup

21.3.2 Amdahl's Law and Gustafson's Law

21.4 Energy Efficiency

21.4.1 Energy as a Resource

21.4.2 Green Computing and Mobile Devices

21.5 Exercises

Chapter 22

Case Studies

22.1 Sorting Algorithms in Practice

22.1.1 Timsort, Introsort, Radix Sort

22.1.2 Comparison of Theoretical vs. Empirical Performance

22.2 Graph Algorithms in Large-Scale Systems

22.2.1 Web Graphs and PageRank

22.2.2 Social Network Analysis

22.3 Machine Learning and Data Science

22.3.1 Complexity of Training Algorithms

22.3.2 SGD, AdaGrad, Adam: Time and Space Analysis

22.4 Database Systems

22.4.1 Query Optimization

22.4.2 Indexing Structures (B-Trees, LSM-Trees)

22.5 Exercises

Appendix A

Mathematical Background

A.1 Summation Formulas

A.2 Logarithms and Exponentials

A.3 Recurrence Relations (Quick Reference)

A.4 Probability Distributions

A.5 Matrix Operations

Appendix B

Pseudocode Conventions

B.1 Notation and Style

B.2 Common Data Structures

Appendix C

Solutions to Selected Exercises

Appendix D

Glossary of Terms

Appendix E

Index of Algorithms

Appendix F

Annotated Bibliography

F.1 Foundational Texts

F.2 Research Papers by Topic

F.3 Online Courses and Resources