

Introduction to **The Design &
Analysis of Algorithms**

3RD EDITION

Vice President and Editorial Director, ECS	<i>Marcia Horton</i>
Editor-in-Chief	<i>Michael Hirsch</i>
Acquisitions Editor	<i>Matt Goldstein</i>
Editorial Assistant	<i>Chelsea Bell</i>
Vice President, Marketing	<i>Patrice Jones</i>
Marketing Manager	<i>Yezan Alayan</i>
Senior Marketing Coordinator	<i>Kathryn Ferranti</i>
Marketing Assistant	<i>Emma Snider</i>
Vice President, Production	<i>Vince O'Brien</i>
Managing Editor	<i>Jeff Holcomb</i>
Production Project Manager	<i>Kayla Smith-Tarbox</i>
Senior Operations Supervisor	<i>Alan Fischer</i>
Manufacturing Buyer	<i>Lisa McDowell</i>
Art Director	<i>Anthony Gemmellaro</i>
Text Designer	<i>Sandra Rigney</i>
Cover Designer	<i>Anthony Gemmellaro</i>
Cover Illustration	<i>Jennifer Kohnke</i>
Media Editor	<i>Daniel Sandin</i>
Full-Service Project Management	<i>Windfall Software</i>
Composition	<i>Windfall Software, using ZzT_EX</i>
Printer/Binder	<i>Courier Westford</i>
Cover Printer	<i>Courier Westford</i>
Text Font	<i>Times Ten</i>

Copyright © 2012, 2007, 2003 Pearson Education, Inc., publishing as Addison-Wesley. All rights reserved. Printed in the United States of America. This publication is protected by Copyright, and permission should be obtained from the publisher prior to any prohibited reproduction, storage in a retrieval system, or transmission in any form or by any means, electronic, mechanical, photocopying, recording, or likewise. To obtain permission(s) to use material from this work, please submit a written request to Pearson Education, Inc., Permissions Department, One Lake Street, Upper Saddle River, New Jersey 07458, or you may fax your request to 201-236-3290.

This is the eBook of the printed book and may not include any media, Website access codes or print supplements that may come packaged with the bound book.

Many of the designations by manufacturers and sellers to distinguish their products are claimed as trademarks. Where those designations appear in this book, and the publisher was aware of a trademark claim, the designations have been printed in initial caps or all caps.

Library of Congress Cataloging-in-Publication Data

Levitin, Anany.

Introduction to the design & analysis of algorithms / Anany Levitin. — 3rd ed.

p. cm.

Includes bibliographical references and index.

ISBN-13: 978-0-13-231681-1

ISBN-10: 0-13-231681-1

1. Computer algorithms. I. Title. II. Title: Introduction to the design and analysis of algorithms.

QA76.9.A43L48 2012

005.1—dc23

2011027089

15 14 13 12 11—CRW—10 9 8 7 6 5 4 3 2 1

PEARSON

ISBN 10: 0-13-231681-1

ISBN 13: 978-0-13-231681-1

Introduction to **The Design &
Analysis of Algorithms**

3RD EDITION

Anany Levitin

Villanova University

PEARSON

Boston Columbus Indianapolis New York San Francisco Upper Saddle River
Amsterdam Cape Town Dubai London Madrid Milan Munich Paris Montreal Toronto
Delhi Mexico City Sao Paulo Sydney Hong Kong Seoul Singapore Taipei Tokyo

This page intentionally left blank

Brief Contents

New to the Third Edition	xvii
Preface	xix
1 Introduction	1
2 Fundamentals of the Analysis of Algorithm Efficiency	41
3 Brute Force and Exhaustive Search	97
4 Decrease-and-Conquer	131
5 Divide-and-Conquer	169
6 Transform-and-Conquer	201
7 Space and Time Trade-Offs	253
8 Dynamic Programming	283
9 Greedy Technique	315
10 Iterative Improvement	345
11 Limitations of Algorithm Power	387
12 Coping with the Limitations of Algorithm Power	423
Epilogue	471
APPENDIX A	
 Useful Formulas for the Analysis of Algorithms	475
APPENDIX B	
 Short Tutorial on Recurrence Relations	479
 References	493
 Hints to Exercises	503
 Index	547

This page intentionally left blank

Contents

New to the Third Edition	xvii
Preface	xix
1 Introduction	1
1.1 What Is an Algorithm?	3
Exercises 1.1	7
1.2 Fundamentals of Algorithmic Problem Solving	9
Understanding the Problem	9
Ascertaining the Capabilities of the Computational Device	9
Choosing between Exact and Approximate Problem Solving	11
Algorithm Design Techniques	11
Designing an Algorithm and Data Structures	12
Methods of Specifying an Algorithm	12
Proving an Algorithm's Correctness	13
Analyzing an Algorithm	14
Coding an Algorithm	15
Exercises 1.2	17
1.3 Important Problem Types	18
Sorting	19
Searching	20
String Processing	20
Graph Problems	21
Combinatorial Problems	21
Geometric Problems	22
Numerical Problems	22
Exercises 1.3	23

1.4 Fundamental Data Structures	25
Linear Data Structures	25
Graphs	28
Trees	31
Sets and Dictionaries	35
Exercises 1.4	37
Summary	38
2 Fundamentals of the Analysis of Algorithm Efficiency	41
2.1 The Analysis Framework	42
Measuring an Input's Size	43
Units for Measuring Running Time	44
Orders of Growth	45
Worst-Case, Best-Case, and Average-Case Efficiencies	47
Recapitulation of the Analysis Framework	50
Exercises 2.1	50
2.2 Asymptotic Notations and Basic Efficiency Classes	52
Informal Introduction	52
O -notation	53
Ω -notation	54
Θ -notation	55
Useful Property Involving the Asymptotic Notations	55
Using Limits for Comparing Orders of Growth	56
Basic Efficiency Classes	58
Exercises 2.2	58
2.3 Mathematical Analysis of Nonrecursive Algorithms	61
Exercises 2.3	67
2.4 Mathematical Analysis of Recursive Algorithms	70
Exercises 2.4	76
2.5 Example: Computing the nth Fibonacci Number	80
Exercises 2.5	83
2.6 Empirical Analysis of Algorithms	84
Exercises 2.6	89
2.7 Algorithm Visualization	91
Summary	94

3	Brute Force and Exhaustive Search	97
3.1	Selection Sort and Bubble Sort	98
	Selection Sort	98
	Bubble Sort	100
	Exercises 3.1	102
3.2	Sequential Search and Brute-Force String Matching	104
	Sequential Search	104
	Brute-Force String Matching	105
	Exercises 3.2	106
3.3	Closest-Pair and Convex-Hull Problems by Brute Force	108
	Closest-Pair Problem	108
	Convex-Hull Problem	109
	Exercises 3.3	113
3.4	Exhaustive Search	115
	Traveling Salesman Problem	116
	Knapsack Problem	116
	Assignment Problem	119
	Exercises 3.4	120
3.5	Depth-First Search and Breadth-First Search	122
	Depth-First Search	122
	Breadth-First Search	125
	Exercises 3.5	128
	Summary	130
4	Decrease-and-Conquer	131
4.1	Insertion Sort	134
	Exercises 4.1	136
4.2	Topological Sorting	138
	Exercises 4.2	142
4.3	Algorithms for Generating Combinatorial Objects	144
	Generating Permutations	144
	Generating Subsets	146
	Exercises 4.3	148

4.4 Decrease-by-a-Constant-Factor Algorithms	150
Binary Search	150
Fake-Coin Problem	152
Russian Peasant Multiplication	153
Josephus Problem	154
Exercises 4.4	156
4.5 Variable-Size-Decrease Algorithms	157
Computing a Median and the Selection Problem	158
Interpolation Search	161
Searching and Insertion in a Binary Search Tree	163
The Game of Nim	164
Exercises 4.5	166
Summary	167
5 Divide-and-Conquer	169
5.1 Mergesort	172
Exercises 5.1	174
5.2 Quicksort	176
Exercises 5.2	181
5.3 Binary Tree Traversals and Related Properties	182
Exercises 5.3	185
5.4 Multiplication of Large Integers and Strassen's Matrix Multiplication	186
Multiplication of Large Integers	187
Strassen's Matrix Multiplication	189
Exercises 5.4	191
5.5 The Closest-Pair and Convex-Hull Problems by Divide-and-Conquer	192
The Closest-Pair Problem	192
Convex-Hull Problem	195
Exercises 5.5	197
Summary	198

6	Transform-and-Conquer	201
6.1	Presorting	202
	Exercises 6.1	205
6.2	Gaussian Elimination	208
	<i>LU</i> Decomposition	212
	Computing a Matrix Inverse	214
	Computing a Determinant	215
	Exercises 6.2	216
6.3	Balanced Search Trees	218
	AVL Trees	218
	2-3 Trees	223
	Exercises 6.3	225
6.4	Heaps and Heapsort	226
	Notion of the Heap	227
	Heapsort	231
	Exercises 6.4	233
6.5	Horner's Rule and Binary Exponentiation	234
	Horner's Rule	234
	Binary Exponentiation	236
	Exercises 6.5	239
6.6	Problem Reduction	240
	Computing the Least Common Multiple	241
	Counting Paths in a Graph	242
	Reduction of Optimization Problems	243
	Linear Programming	244
	Reduction to Graph Problems	246
	Exercises 6.6	248
	Summary	250
7	Space and Time Trade-Offs	253
7.1	Sorting by Counting	254
	Exercises 7.1	257
7.2	Input Enhancement in String Matching	258
	Horspool's Algorithm	259

Boyer-Moore Algorithm	263
Exercises 7.2	267
7.3 Hashing	269
Open Hashing (Separate Chaining)	270
Closed Hashing (Open Addressing)	272
Exercises 7.3	274
7.4 B-Trees	276
Exercises 7.4	279
Summary	280
8 Dynamic Programming	283
8.1 Three Basic Examples	285
Exercises 8.1	290
8.2 The Knapsack Problem and Memory Functions	292
Memory Functions	294
Exercises 8.2	296
8.3 Optimal Binary Search Trees	297
Exercises 8.3	303
8.4 Warshall's and Floyd's Algorithms	304
Warshall's Algorithm	304
Floyd's Algorithm for the All-Pairs Shortest-Paths Problem	308
Exercises 8.4	311
Summary	312
9 Greedy Technique	315
9.1 Prim's Algorithm	318
Exercises 9.1	322
9.2 Kruskal's Algorithm	325
Disjoint Subsets and Union-Find Algorithms	327
Exercises 9.2	331
9.3 Dijkstra's Algorithm	333
Exercises 9.3	337

9.4	Huffman Trees and Codes	338
	Exercises 9.4	342
	Summary	344
10	Iterative Improvement	345
10.1	The Simplex Method	346
	Geometric Interpretation of Linear Programming	347
	An Outline of the Simplex Method	351
	Further Notes on the Simplex Method	357
	Exercises 10.1	359
10.2	The Maximum-Flow Problem	361
	Exercises 10.2	371
10.3	Maximum Matching in Bipartite Graphs	372
	Exercises 10.3	378
10.4	The Stable Marriage Problem	380
	Exercises 10.4	383
	Summary	384
11	Limitations of Algorithm Power	387
11.1	Lower-Bound Arguments	388
	Trivial Lower Bounds	389
	Information-Theoretic Arguments	390
	Adversary Arguments	390
	Problem Reduction	391
	Exercises 11.1	393
11.2	Decision Trees	394
	Decision Trees for Sorting	395
	Decision Trees for Searching a Sorted Array	397
	Exercises 11.2	399
11.3	P , NP , and NP -Complete Problems	401
	P and NP Problems	402
	NP -Complete Problems	406
	Exercises 11.3	409

11.4 Challenges of Numerical Algorithms	412
Exercises 11.4	419
Summary	420

12 Coping with the Limitations of Algorithm Power **423**

12.1 Backtracking	424
n -Queens Problem	425
Hamiltonian Circuit Problem	426
Subset-Sum Problem	427
General Remarks	428
Exercises 12.1	430
12.2 Branch-and-Bound	432
Assignment Problem	433
Knapsack Problem	436
Traveling Salesman Problem	438
Exercises 12.2	440
12.3 Approximation Algorithms for <i>NP</i>-Hard Problems	441
Approximation Algorithms for the Traveling Salesman Problem	443
Approximation Algorithms for the Knapsack Problem	453
Exercises 12.3	457
12.4 Algorithms for Solving Nonlinear Equations	459
Bisection Method	460
Method of False Position	464
Newton's Method	464
Exercises 12.4	467
Summary	468

Epilogue **471**

APPENDIX A

Useful Formulas for the Analysis of Algorithms	475
Properties of Logarithms	475
Combinatorics	475
Important Summation Formulas	476
Sum Manipulation Rules	476

Approximation of a Sum by a Definite Integral	477
Floor and Ceiling Formulas	477
Miscellaneous	477

APPENDIX B

Short Tutorial on Recurrence Relations	479
Sequences and Recurrence Relations	479
Methods for Solving Recurrence Relations	480
Common Recurrence Types in Algorithm Analysis	485
 References	 493
 Hints to Exercises	 503
 Index	 547

This page intentionally left blank

Useful Formulas for the Analysis of Algorithms

This appendix contains a list of useful formulas and rules that are helpful in the mathematical analysis of algorithms. More advanced material can be found in [Gra94], [Gre07], [Pur04], and [Sed96].

Properties of Logarithms

All logarithm bases are assumed to be greater than 1 in the formulas below; $\lg x$ denotes the logarithm base 2, $\ln x$ denotes the logarithm base $e = 2.71828 \dots$; x, y are arbitrary positive numbers.

1. $\log_a 1 = 0$
2. $\log_a a = 1$
3. $\log_a x^y = y \log_a x$
4. $\log_a xy = \log_a x + \log_a y$
5. $\log_a \frac{x}{y} = \log_a x - \log_a y$
6. $a^{\log_b x} = x^{\log_b a}$
7. $\log_a x = \frac{\log_b x}{\log_b a} = \log_a b \log_b x$

Combinatorics

1. Number of permutations of an n -element set: $P(n) = n!$
2. Number of k -combinations of an n -element set: $C(n, k) = \frac{n!}{k!(n-k)!}$
3. Number of subsets of an n -element set: 2^n

Important Summation Formulas

1. $\sum_{i=l}^u 1 = \underbrace{1 + 1 + \cdots + 1}_{u-l+1 \text{ times}} = u - l + 1$ (l, u are integer limits, $l \leq u$); $\sum_{i=1}^n 1 = n$
2. $\sum_{i=1}^n i = 1 + 2 + \cdots + n = \frac{n(n+1)}{2} \approx \frac{1}{2}n^2$
3. $\sum_{i=1}^n i^2 = 1^2 + 2^2 + \cdots + n^2 = \frac{n(n+1)(2n+1)}{6} \approx \frac{1}{3}n^3$
4. $\sum_{i=1}^n i^k = 1^k + 2^k + \cdots + n^k \approx \frac{1}{k+1}n^{k+1}$
5. $\sum_{i=0}^n a^i = 1 + a + \cdots + a^n = \frac{a^{n+1} - 1}{a - 1}$ ($a \neq 1$); $\sum_{i=0}^n 2^i = 2^{n+1} - 1$
6. $\sum_{i=1}^n i2^i = 1 \cdot 2 + 2 \cdot 2^2 + \cdots + n2^n = (n-1)2^{n+1} + 2$
7. $\sum_{i=1}^n \frac{1}{i} = 1 + \frac{1}{2} + \cdots + \frac{1}{n} \approx \ln n + \gamma$, where $\gamma \approx 0.5772 \dots$ (Euler's constant)
8. $\sum_{i=1}^n \lg i \approx n \lg n$

Sum Manipulation Rules

1. $\sum_{i=l}^u ca_i = c \sum_{i=l}^u a_i$
2. $\sum_{i=l}^u (a_i \pm b_i) = \sum_{i=l}^u a_i \pm \sum_{i=l}^u b_i$
3. $\sum_{i=l}^u a_i = \sum_{i=l}^m a_i + \sum_{i=m+1}^u a_i$, where $l \leq m < u$
4. $\sum_{i=l}^u (a_i - a_{i-1}) = a_u - a_{l-1}$

Approximation of a Sum by a Definite Integral

$$\int_{l-1}^u f(x)dx \leq \sum_{i=l}^u f(i) \leq \int_l^{u+1} f(x)dx \quad \text{for a nondecreasing } f(x)$$

$$\int_l^{u+1} f(x)dx \leq \sum_{i=l}^u f(i) \leq \int_{l-1}^u f(x)dx \quad \text{for a nonincreasing } f(x)$$

Floor and Ceiling Formulas

The *floor* of a real number x , denoted $\lfloor x \rfloor$, is defined as the greatest integer not larger than x (e.g., $\lfloor 3.8 \rfloor = 3$, $\lfloor -3.8 \rfloor = -4$, $\lfloor 3 \rfloor = 3$). The *ceiling* of a real number x , denoted $\lceil x \rceil$, is defined as the smallest integer not smaller than x (e.g., $\lceil 3.8 \rceil = 4$, $\lceil -3.8 \rceil = -3$, $\lceil 3 \rceil = 3$).

1. $x - 1 < \lfloor x \rfloor \leq x \leq \lceil x \rceil < x + 1$
2. $\lfloor x + n \rfloor = \lfloor x \rfloor + n$ and $\lceil x + n \rceil = \lceil x \rceil + n$ for real x and integer n
3. $\lfloor n/2 \rfloor + \lceil n/2 \rceil = n$
4. $\lceil \lg(n+1) \rceil = \lfloor \lg n \rfloor + 1$

Miscellaneous

1. $n! \approx \sqrt{2\pi n} \left(\frac{n}{e}\right)^n$ as $n \rightarrow \infty$ (Stirling's formula)
2. Modular arithmetic (n, m are integers, p is a positive integer)

$$(n + m) \bmod p = (n \bmod p + m \bmod p) \bmod p$$

$$(nm) \bmod p = ((n \bmod p)(m \bmod p)) \bmod p$$