

# DESIGN AND ANALYSIS OF ALGORITHMS

S. Sridhar

*Professor*

*Department of Information Science and Technology*

*College of Engineering, Guindy Campus*

*Anna University, Chennai*

**OXFORD**  
UNIVERSITY PRESS

**OXFORD**  
UNIVERSITY PRESS

Oxford University Press is a department of the University of Oxford. It furthers the University's objective of excellence in research, scholarship, and education by publishing worldwide. Oxford is a registered trade mark of Oxford University Press in the UK and in certain other countries.

Published in India by  
Oxford University Press  
YMCA Library Building, 1 Jai Singh Road, New Delhi 110001, India

© Oxford University Press 2014

The moral rights of the author/s have been asserted.

First published in 2014

All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, without the prior permission in writing of Oxford University Press, or as expressly permitted by law, by licence, or under terms agreed with the appropriate reprographics rights organization. Enquiries concerning reproduction outside the scope of the above should be sent to the Rights Department, Oxford University Press, at the address above.

You must not circulate this work in any other form  
and you must impose this same condition on any acquirer.

ISBN-13: 978-0-19-809369-5  
ISBN-10: 0-19-809369-1

Typeset in Times New Roman  
by Ideal Publishing Solutions, Delhi  
Printed in India by Radha Press, New Delhi 110031

Third-party website addresses mentioned in this book are provided  
by Oxford University Press in good faith and for information only.  
Oxford University Press disclaims any responsibility for the material contained therein.

# Brief Contents

*Features of the Book* iv

*Preface* vi

*Detailed Contents* xi

1	Introduction to Algorithms	1
2	Basics of Algorithm Writing	22
3	Basics of Algorithm Analysis	58
4	Mathematical Analysis of Recursive Algorithms	98
5	Data Structures—I	141
6	Data Structures—II	194
7	Brute Force Approaches	231
8	Divide-and-conquer Approach	262
9	Decrease-and-conquer Approach	309
10	Time–Space Tradeoffs	342
11	Greedy Algorithms	372
12	Transform-and-conquer Approach	417
13	Dynamic Programming	455
14	Backtracking	517
15	Branch-and-bound Technique	543
16	String Algorithms	569
17	Iterative Improvement and Linear Programming	601
18	Basics of Computational Complexity	638
19	Randomized and Approximation Algorithms	663
20	Parallel Algorithms	705

*Appendix A—Mathematical Basics* 735

*Appendix B—Proof Techniques* 752

*Bibliography* 763

*Index* 766

# Detailed Contents

*Features of the Book* iv

*Preface* vi

*Brief Contents* x

<b>1</b>	<b>Introduction to Algorithms</b>	<b>1</b>	2.3	Non-recursive Algorithms	31
1.1	Introduction	1	2.3.1	<i>Flowcharts</i>	32
1.2	Need for Algorithmic Thinking	1	2.4	Basics of Recursion	42
1.3	Overview of Algorithms	3	2.5	Recursive Algorithms	43
	1.3.1 <i>Computational Problems, Instance, and Size</i>	5	2.6	Algorithm Correctness	52
1.4	Need for Algorithm Efficiency	7	<b>3</b>	<b>Basics of Algorithm Analysis</b>	<b>58</b>
1.5	Fundamental Stages of Problem Solving	9	3.1	Basics of Algorithm Complexity	58
	1.5.1 <i>Understanding the Problem</i>	9	3.2	Introduction to Time complexity	60
	1.5.2 <i>Planning an Algorithm</i>	9	3.2.1	<i>Random Access Machine</i>	60
	1.5.3 <i>Designing an Algorithm</i>	11	3.3	Analysis of Iterative Algorithms	62
	1.5.4 <i>Validating and Verifying an Algorithm</i>	11	3.3.1	<i>Measuring Input Size</i>	63
	1.5.5 <i>Analysing an Algorithm</i>	12	3.3.2	<i>Measuring Running Time</i>	63
	1.5.6 <i>Implementing an Algorithm and Performing Empirical Analysis</i>	14	3.3.3	<i>Best-, Worst-, and Average-case Complexity</i>	71
	1.5.7 <i>Post (or Postmortem) Analysis</i>	14	3.4	Rate of Growth	73
1.6	Classification of Algorithms	15	3.4.1	<i>Measuring Larger Inputs</i>	73
	1.6.1 <i>Based on Implementation</i>	15	3.4.2	<i>Comparison Framework</i>	74
	1.6.2 <i>Based on Design</i>	16	3.5	Asymptotic Analysis	76
	1.6.3 <i>Based on Area of Specialization</i>	17	3.5.1	<i>Asymptotic Notations</i>	78
	1.6.4 <i>Based on Tractability</i>	18	3.5.2	<i>Asymptotic Rules</i>	85
			3.5.3	<i>Asymptotic Complexity Classes</i>	87
<b>2</b>	<b>Basics of Algorithm Writing</b>	<b>22</b>	3.6	Space Complexity Analysis	88
2.1	Tools for Problem-solving	22	3.7	Empirical Analysis and Algorithm Visualization	88
	2.1.1 <i>Stepwise Refinement or Top-down design</i>	23	3.7.1	<i>Experimental Purpose</i>	89
	2.1.2 <i>Bottom-up Approach</i>	24	3.7.2	<i>Statistical Tests</i>	90
	2.1.3 <i>Structured Programming</i>	25	<b>4</b>	<b>Mathematical Analysis of Recursive Algorithms</b>	<b>98</b>
2.2	Algorithm Specifications	26	4.1	Introduction to Recurrence Equations	98
	2.2.1 <i>Guidelines for Writing Algorithms</i>	27	4.1.1	<i>Linear Recurrences</i>	99
			4.1.2	<i>Non-linear Recurrences</i>	101
			4.2	Formulation of Recurrence Equations	103

4.3	Techniques for Solving Recurrence Equations	106	<b>6</b>	<b>Data Structures—II</b>	<b>194</b>
4.3.1	<i>Guess-and-verify Method</i>	106	6.1	Introduction to Dictionary	194
4.3.2	<i>Substitution Method</i>	109	6.1.1	<i>Introduction to Binary Search Tree</i>	194
4.3.3	<i>Recurrence-tree Method</i>	112	6.1.2	<i>AVL Trees</i>	200
4.3.4	<i>Difference Method</i>	117	6.2	Priority Queues and Heaps	206
4.4	Solving Recurrence Equations Using Polynomial Reduction	118	6.2.1	<i>Binary Heaps</i>	207
4.4.1	<i>Solving Homogeneous Equations</i>	118	6.2.2	<i>Binomial Heaps</i>	212
4.4.2	<i>Solving Non-homogeneous Equations</i>	122	6.2.3	<i>Fibonacci heap</i>	218
4.5	Generating Functions	123	6.3	Disjoint Sets	221
4.5.1	<i>Properties of Generating Functions</i>	124	6.3.1	<i>Representation and Operations</i>	221
4.6	Divide-and-conquer Recurrences	127	6.4	Amortized Analysis	224
4.6.1	<i>Master Theorem</i>	127	6.4.1	<i>Aggregate Method</i>	225
4.6.2	<i>Transformations</i>	133	6.4.2	<i>Accounting Method</i>	226
4.6.3	<i>Conditional Asymptotics</i>	135	6.4.3	<i>Potential Method</i>	226
<b>5</b>	<b>Data Structures—I</b>	<b>141</b>	<b>7</b>	<b>Brute Force Approaches</b>	<b>231</b>
5.1	Data Structures and Algorithms	141	7.1	Introduction	231
5.2	Lists	142	7.1.1	<i>Advantages and Disadvantages of Brute Force Method</i>	232
5.2.1	<i>Linear Lists and Arrays</i>	143	7.2	Sequential Search	232
5.2.2	<i>Linked Lists</i>	146	7.2.1	<i>Analysis of Recursion Programs</i>	234
5.3	Stacks	152	7.2.2	<i>Recursive Form of Linear Search Algorithm</i>	235
5.3.1	<i>Representation of and Operations on Stacks</i>	152	7.3	Sorting Problem	236
5.4	Queues	154	7.3.1	<i>Classification of Sorting Algorithms</i>	237
5.4.1	<i>Queue Representation</i>	154	7.3.2	<i>Properties of Sorting Algorithms</i>	237
5.5	Trees	157	7.3.3	<i>Bubble Sort</i>	238
5.5.1	<i>Tree Terminologies</i>	157	7.3.4	<i>Selection Sort</i>	242
5.5.2	<i>Classification of Trees</i>	159	7.4	Computational Geometry Problems	245
5.5.3	<i>Binary Tree Representation</i>	161	7.4.1	<i>Closest-pair Problem</i>	245
5.5.4	<i>Binary Tree Operations</i>	163	7.4.2	<i>Convex Hull Problem</i>	247
5.6	Graphs	167	7.5	Exhaustive Searching	249
5.6.1	<i>Terminologies and Types of Graphs</i>	168	7.5.1	<i>15-puzzle Problem</i>	250
5.6.2	<i>Graph Representation</i>	172	7.5.2	<i>8-queen Problem</i>	251
5.6.3	<i>Graph Traversal</i>	174	7.5.3	<i>Magic Squares</i>	252
5.6.4	<i>Elementary Graph Algorithms</i>	178	7.5.4	<i>Container Loading Problem</i>	253
5.6.5	<i>Spanning Tree and Minimum-cost Spanning Tree</i>	186	7.5.6	<i>Assignment Problem</i>	256

<b>8</b>	<b>Divide-and-conquer Approach</b>	<b>262</b>		
8.1	Introduction	262		
	8.1.1 Recurrence Equation for Divide and Conquer	263		
	8.1.2 Advantages and Disadvantages of Divide-and-conquer Paradigm	264		
8.2	Merge Sort	264		
8.3	Quicksort	269		
	8.3.1 Partitioning Algorithms	270		
	8.3.2 Variants of Quicksort	275		
8.4	Finding Maximum and Minimum Elements	277		
8.5	Multiplication of Long Integers	280		
8.6	Strassen Matrix Multiplication	284		
8.7	Tiling Problem	288		
8.8	Closest-pair Problem	291		
	8.8.1 Using Divide-and-conquer Method	291		
8.9	Convex Hull	293		
	8.9.1 Quickhull	293		
	8.9.2 Merge Hull	295		
8.10	Fourier Transform	296		
	8.10.1 Polynomial Multiplication	296		
	8.10.2 Application of Fourier Transform	298		
	8.10.3 Fast Fourier Transform	302		
<b>9</b>	<b>Decrease-and-conquer Approach</b>	<b>309</b>		
9.1	Introduction	309		
9.2	Decrease by Constant Method	311		
	9.2.1 Insertion Sort	311		
	9.2.2 Topological Sort	315		
	9.2.3 Generating Permutations	321		
	9.2.4 Generating Subsets	323		
9.3	Decrease by Constant Factor Method	325		
	9.3.1 Binary Search	326		
	9.3.2 Fake Coin Detection	330		
	9.3.3 Russian Peasant Multiplication Problem	331		
9.4	Decrease by Variable Factor Method	332		
	9.4.1 Interpolation Search	333		
	9.4.2 Selection and Ordered Statistics	334		
	9.4.3 Finding Median	336		
<b>10</b>	<b>Time–Space Tradeoffs</b>	<b>342</b>		
10.1	Introduction to Time–Space Tradeoffs	342		
10.2	Linear Sorting	342		
	10.2.1 Counting Sort	342		
	10.2.2 Bucket Sort	347		
	10.2.3 Radix Sort	349		
10.3	Hashing and Hash Tables	351		
	10.3.1 Properties of Hash Functions	353		
	10.3.2 Hash Table Operations	354		
	10.3.3 Collision	356		
10.4	B-trees	359		
	10.4.1 B-tree Balancing Operations	361		
	10.4.2 B-tree Operations	363		
<b>11</b>	<b>Greedy Algorithms</b>	<b>372</b>		
11.1	Introduction to Greedy Approach	372		
	11.1.1 Components of Greedy Algorithms	373		
11.2	Suitability of Greedy Approach	374		
11.3	Coin Change Problem	375		
11.4	Scheduling Problems	376		
	11.4.1 Scheduling without Deadline	377		
	11.4.2 Scheduling with Deadline	379		
	11.4.3 Activity Selection Problem	382		
11.5	Knapsack Problem	384		
11.6	Optimal Storage of Tapes	388		
11.7	Optimal Tree Problems	390		
	11.7.1 Optimal Merge	390		
	11.7.2 Huffman Coding	393		
	11.7.3 Tree Vertex Splitting Problem	398		
11.8	Optimal Graph Problems	401		
	11.8.1 Minimum Spanning Trees	401		
	11.8.2 Single-source Shortest-path Problems	407		

<b>12 Transform-and-conquer Approach</b>	<b>417</b>		
12.1 Introduction to Transform and Conquer	417		
12.2 Introduction to Instance Simplification	418		
12.3 Matrix Operations	419		
12.3.1 Gaussian Elimination Method	420		
12.3.2 LU Decomposition	427		
12.3.3 Crout's Method of Decomposition	434		
12.3.4 Finding Matrix Inverse	436		
12.3.5 Finding Matrix Determinant	439		
12.4 Change of Representation	441		
12.4.1 Heap Sort	441		
12.4.2 Polynomial Evaluation Using Horner's Method	445		
12.4.3 Binary Exponentiation	447		
12.5 Problem Reduction	450		
<b>13 Dynamic Programming</b>	<b>455</b>		
13.1 Basics of Dynamic Programming	455		
13.1.1 Components of Dynamic Programming	456		
13.1.2 Characteristics of Dynamic Programming	458		
13.2 Fibonacci Problem	460		
13.3 Computing Binomial Coefficients	463		
13.4 Multistage Graph Problem	466		
13.4.1 Forward Computation Procedure	467		
13.4.2 Backward Computation Procedure	471		
13.5 Transitive Closure and Warshall Algorithm	472		
13.5.1 Finding Transitive Closure Using Brute Force Approach	473		
13.5.2 Finding Transitive Closure Using Warshall Algorithm	473		
13.5.3 Alternative Method to Warshall Algorithm for Finding Transitive Closure	476		
13.6 Floyd–Warshall All Pairs Shortest-path Algorithm	478		
13.6.1 Shortest-path Reconstruction	481		
13.7 Bellman–Ford Algorithm	481		
13.8 Travelling Salesperson Problem	486		
13.9 Chain Matrix Multiplication	488		
13.9.1 Dynamic Programming Approach for Solving Chain Matrix Multiplication Problem	490		
13.10 Knapsack Problem	496		
13.11 Optimal Binary Search Trees	500		
13.11.1 Brute Force Approach for Constructing Optimal BSTs	500		
13.11.2 Dynamic Programming Approach for Constructing Optimal BSTs	502		
13.12 Flow-shop Scheduling Problem	507		
13.12.1 Single-machine Sequencing Problem	508		
13.12.2 Two-machine Sequencing Problem	509		
<b>14 Backtracking</b>	<b>517</b>		
14.1 Introduction	517		
14.2 Basics of Backtracking	518		
14.3 <i>N</i> -queen Problem	522		
14.3.1 State Space of 4-queen Problem	523		
14.4 Sum of Subsets	525		
14.5 Vertex Colouring Problem	527		
14.6 Hamiltonian Circuit Problem	531		
14.6.1 Promising (or Bounding Function) for Hamiltonian Problem	531		
14.7 Generating Permutation	534		
14.8 Graham Scan	536		
<b>15 Branch-and-bound Technique</b>	<b>543</b>		
15.1 Introduction	543		
15.2 Search Techniques for Branch-and-bound Technique	545		

15.2.1	<i>BFS using Branch-and-bound algorithm—FIFOBB</i>	545	17.8	Max-flow Problem	623
15.2.2	<i>LIFO with Branch and Bound</i>	547	17.9	Bipartite Matching Problem	628
15.2.3	<i>Least Cost with Branch and Bound</i>	547	17.10	Stable Marriage Problem	630
15.3	15-puzzle Game	548	<b>18</b>	<b>Basics of Computational Complexity</b>	<b>638</b>
15.4	Assignment Problem	552	18.1	Introduction to Computational Complexity	638
15.5	Traveling Salesperson Problem	559	18.2	Algorithm Complexity, Upper and Lower Bound Theory	639
15.6	Knapsack Problem	562	18.2.1	<i>Upper and Lower Bounds</i>	640
<b>16</b>	<b>String Algorithms</b>	<b>569</b>	18.3	Decision Problems and Turing Machine	644
16.1	Introduction to String Processing	569	18.4	Complexity Classes	647
16.2	Basic String Algorithms	571	18.4.1	<i>Class P</i>	648
16.2.1	<i>Length of Strings</i>	571	18.4.2	<i>NP Class</i>	648
16.2.2	<i>Concatenation of Two Strings</i>	571	18.5	Theory of NP-complete Problems	650
16.2.3	<i>Finding Substrings</i>	572	18.6	Reductions	651
16.3	Longest Common Subsequences	573	18.6.1	<i>Turing Reduction</i>	651
16.4	naïve String Matching Algorithm	576	18.6.2	<i>Karp Reduction</i>	652
16.5	Pattern Matching Using Finite Automata	578	18.7	Satisfiability Problem and Cook's Theorem	654
16.6	Rabin–Karp Algorithm	580	18.8	Example Problems for Proving NP-completeness	655
16.7	Knuth–Morris–Pratt Algorithm	584	18.8.1	<i>SAT is NP-complete</i>	655
16.8	Harspool Algorithm	588	18.8.2	<i>Problem 3-CNF-SAT is NP-complete</i>	656
16.9	Boyer–Moore String Matching Algorithm	590	18.8.3	<i>Clique Decision Problem is NP-complete</i>	657
16.10	Approximate String Matching	594	18.8.4	<i>Sum of Subsets (from 3-CNF-SAT)</i>	658
<b>17</b>	<b>Iterative Improvement and Linear Programming</b>	<b>601</b>	<b>19</b>	<b>Randomized and Approximation Algorithms</b>	<b>663</b>
17.1	Introduction to Iterative Improvement	601	19.1	Dealing with NP-hard Problems	663
17.2	Linear Programming	601	19.2	Introduction to Randomized Algorithms	664
17.3	Formulation of LPPs	603	19.2.1	<i>Generation of Random Numbers</i>	666
17.4	Graphical Method for Solving LPPs	606	19.2.2	<i>Types of Randomized Algorithms</i>	669
17.5	Simplex Method	611			
17.6	Minimization Problems	615			
17.7	Principle of Duality	619			



19.3	Examples of Randomized Algorithms	669	20.2	Classification of Parallel Systems	705
	19.3.1 Hiring Problem	669		20.2.1 Flynn Classification	706
	19.3.2 Primality Testing Algorithm	672		20.2.2 Address-space (or Memory Mechanism-based) Classification	708
	19.3.3 Comparing Strings Using Randomization Algorithm	674		20.2.3 Classification Based on Interconnection Networks	710
	19.3.4 Randomized Quicksort	675	20.3	Introduction to PRAM Model	711
19.4	Introduction to Approximation Algorithms	678	20.4	Parallel Algorithm Specifications and Analysis	712
19.5	Types of Approximation Algorithms	679		20.4.1 Parallel Algorithm Analysis	714
19.6	Examples of Approximation Algorithms	680	20.5	Simple Parallel Algorithms	716
	19.6.1 Heuristic-based Approximation Algorithms	681		20.5.1 Prefix Computation	716
	19.6.2 Greedy Approximation Algorithms	688		20.5.2 List Ranking	718
	19.6.3 Approximation Algorithm Design Using Linear Programming	693		20.5.3 Euler Tour	719
	19.6.4 Designing Approximation Algorithms Using Dynamic Programming	696	20.6	Parallel Searching and Parallel Sorting	720
<b>20</b>	<b>Parallel Algorithms</b>	<b>705</b>		20.6.1 Parallel Searching	720
	20.1 Introduction to Parallel Processing	705		20.6.2 Odd–Even Swap Sort	722
				20.6.3 Parallel Merge–Split Algorithm	724
			20.7	Additional Parallel Algorithms	726
				20.7.1 Parallel Matrix Multiplication	726
				20.7.2 Parallel Graph Algorithms	729
	<i>Appendix A—Mathematical Basics</i>	735			
	<i>Appendix B—Proof Techniques</i>	752			
	<i>Bibliography</i>	763			
	<i>Index</i>	766			