Antti Laaksonen

# Guide to Competitive Programming

## Learning and Improving Algorithms Through Contests

Second Edition

Springer

Antti Laaksonen
Department of Computer Science
University of Helsinki
Helsinki, Finland

ISSN 1863-7310 ISSN 2197-1781 (electronic)
Undergraduate Topics in Computer Science
ISBN 978-3-030-39356-4 ISBN 978-3-030-39357-1 (eBook)
https://doi.org/10.1007/978-3-030-39357-1

1st edition: © Springer International Publishing AG, part of Springer Nature 2017
2nd edition: © Springer Nature Switzerland AG 2020
This work is subject to copyright. All rights are reserved by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.
The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.
The publisher, the authors and the editors are safe to assume that the advice and information in this book are believed to be true and accurate at the date of publication. Neither the publisher nor the authors or the editors give a warranty, expressed or implied, with respect to the material contained herein or for any errors or omissions that may have been made. The publisher remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

This Springer imprint is published by the registered company Springer Nature Switzerland AG
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

# Contents

**1 Introduction** . . . . . 1
1.1 What Is Competitive Programming? . . . . . 1
1.1.1 Programming Contests . . . . . 2
1.1.2 Tips for Practicing . . . . . 3
1.2 About This Book . . . . . 3
1.3 CSES Problem Set . . . . . 5
1.4 Other Resources . . . . . 7

**2 Programming Techniques** . . . . . 9
2.1 Language Features . . . . . 9
2.1.1 Input and Output . . . . . 10
2.1.2 Working with Numbers . . . . . 12
2.1.3 Shortening Code . . . . . 14
2.2 Recursive Algorithms . . . . . 15
2.2.1 Generating Subsets . . . . . 16
2.2.2 Generating Permutations . . . . . 17
2.2.3 Backtracking . . . . . 18
2.3 Bit Manipulation . . . . . 20
2.3.1 Bit Operations . . . . . 21
2.3.2 Representing Sets . . . . . 23

**3 Efficiency** . . . . . 27
3.1 Time Complexity . . . . . 27
3.1.1 Calculation Rules . . . . . 28
3.1.2 Common Time Complexities . . . . . 30
3.1.3 Estimating Efficiency . . . . . 31
3.1.4 Formal Definitions . . . . . 32
3.2 Algorithm Design Examples . . . . . 32
3.2.1 Maximum Subarray Sum . . . . . 33
3.2.2 Two Queens Problem . . . . . 35
3.3 Code Optimization . . . . . 37
3.3.1 Compiler Output . . . . . 37
3.3.2 Processor Features . . . . . 40

**4 Sorting and Searching** . . . . . 43
4.1 Sorting Algorithms . . . . . 43
4.1.1 Bubble Sort . . . . . 44
4.1.2 Merge Sort . . . . . 45
4.1.3 Sorting Lower Bound . . . . . 46
4.1.4 Counting Sort . . . . . 47
4.1.5 Sorting in Practice . . . . . 47
4.2 Solving Problems by Sorting . . . . . 50
4.2.1 Sweep Line Algorithms . . . . . 50
4.2.2 Scheduling Events . . . . . 51
4.2.3 Tasks and Deadlines . . . . . 52
4.3 Binary Search . . . . . 53
4.3.1 Implementing the Search . . . . . 53
4.3.2 Finding Optimal Solutions . . . . . 55

**5 Data Structures** . . . . . 57
5.1 Dynamic Arrays . . . . . 57
5.1.1 Vectors . . . . . 58
5.1.2 Iterators and Ranges . . . . . 59
5.1.3 Other Structures . . . . . 60
5.2 Set Structures . . . . . 61
5.2.1 Sets and Multisets . . . . . 61
5.2.2 Maps . . . . . 63
5.2.3 Priority Queues . . . . . 64
5.2.4 Policy-Based Sets . . . . . 65
5.3 Experiments . . . . . 66
5.3.1 Set Versus Sorting . . . . . 66
5.3.2 Map Versus Array . . . . . 67
5.3.3 Priority Queue Versus Multiset . . . . . 67

**6 Dynamic Programming** . . . . . 69
6.1 Basic Concepts . . . . . 69
6.1.1 When Greedy Fails . . . . . 69
6.1.2 Finding an Optimal Solution . . . . . 70
6.1.3 Counting Solutions . . . . . 73
6.2 Further Examples . . . . . 75
6.2.1 Longest Increasing Subsequence . . . . . 75
6.2.2 Paths in a Grid . . . . . 76
6.2.3 Knapsack Problems . . . . . 77
6.2.4 From Permutations to Subsets . . . . . 79
6.2.5 Counting Tilings . . . . . 81

**7 Graph Algorithms** . . . . . 83
7.1 Basics of Graphs . . . . . 84
7.1.1 Graph Terminology . . . . . 84
7.1.2 Graph Representation . . . . . 86
7.2 Graph Traversal . . . . . 89
7.2.1 Depth-First Search . . . . . 89
7.2.2 Breadth-First Search . . . . . 91
7.2.3 Applications . . . . . 92
7.3 Shortest Paths . . . . . 93
7.3.1 Bellman–Ford Algorithm . . . . . 94
7.3.2 Dijkstra's Algorithm . . . . . 96
7.3.3 Floyd–Warshall Algorithm . . . . . 98
7.4 Directed Acyclic Graphs . . . . . 100
7.4.1 Topological Sorting . . . . . 100
7.4.2 Dynamic Programming . . . . . 102
7.5 Successor Graphs . . . . . 104
7.5.1 Finding Successors . . . . . 104
7.5.2 Cycle Detection . . . . . 105
7.6 Minimum Spanning Trees . . . . . 106
7.6.1 Kruskal's Algorithm . . . . . 107
7.6.2 Union-Find Structure . . . . . 110
7.6.3 Prim's Algorithm . . . . . 112

**8 Algorithm Design Topics** . . . . . 115
8.1 Bit-Parallel Algorithms . . . . . 115
8.1.1 Hamming Distances . . . . . 115
8.1.2 Counting Subgrids . . . . . 116
8.1.3 Reachability in Graphs . . . . . 118
8.2 Amortized Analysis . . . . . 118
8.2.1 Two Pointers Method . . . . . 119
8.2.2 Nearest Smaller Elements . . . . . 121
8.2.3 Sliding Window Minimum . . . . . 122
8.3 Finding Minimum Values . . . . . 122
8.3.1 Ternary Search . . . . . 123
8.3.2 Convex Functions . . . . . 124
8.3.3 Minimizing Sums . . . . . 125

**9 Range Queries** . . . . . 127
9.1 Queries on Static Arrays . . . . . 127
9.1.1 Sum Queries . . . . . 128
9.1.2 Minimum Queries . . . . . 129

9.2 Tree Structures . . . 130
9.2.1 Binary Indexed Trees . . . 130
9.2.2 Segment Trees . . . 133
9.2.3 Additional Techniques . . . 136

**10 Tree Algorithms** . . . 139
10.1 Basic Techniques . . . 139
10.1.1 Tree Traversal . . . 140
10.1.2 Calculating Diameters . . . 142
10.1.3 All Longest Paths . . . 143
10.2 Tree Queries . . . 145
10.2.1 Finding Ancestors . . . 145
10.2.2 Subtrees and Paths . . . 146
10.2.3 Lowest Common Ancestors . . . 147
10.2.4 Merging Data Structures . . . 150
10.3 Advanced Techniques . . . 152
10.3.1 Centroid Decomposition . . . 152
10.3.2 Heavy-Light Decomposition . . . 153

**11 Mathematics** . . . 155
11.1 Number Theory . . . 155
11.1.1 Primes and Factors . . . 156
11.1.2 Sieve of Eratosthenes . . . 158
11.1.3 Euclid’s Algorithm . . . 159
11.1.4 Modular Exponentiation . . . 161
11.1.5 Euler’s Theorem . . . 161
11.1.6 Solving Equations . . . 162
11.2 Combinatorics . . . 164
11.2.1 Binomial Coefficients . . . 164
11.2.2 Catalan Numbers . . . 167
11.2.3 Inclusion-Exclusion . . . 169
11.2.4 Burnside’s Lemma . . . 171
11.2.5 Cayley’s Formula . . . 172
11.3 Matrices . . . 172
11.3.1 Matrix Operations . . . 173
11.3.2 Linear Recurrences . . . 175
11.3.3 Graphs and Matrices . . . 177
11.3.4 Gaussian Elimination . . . 178
11.4 Probability . . . 181
11.4.1 Working with Events . . . 182
11.4.2 Random Variables . . . 183
11.4.3 Markov Chains . . . 185
11.4.4 Randomized Algorithms . . . 186

11.5 Game Theory . . . 188
11.5.1 Game States . . . 189
11.5.2 Nim Game . . . 190
11.5.3 Sprague–Grundy Theorem . . . 192
11.6 Fourier Transform . . . 194
11.6.1 Working with Polynomials . . . 195
11.6.2 FFT Algorithm . . . 196
11.6.3 Calculating Convolutions . . . 199

**12 Advanced Graph Algorithms . . . 201**
12.1 Strong Connectivity . . . 201
12.1.1 Kosaraju's Algorithm . . . 202
12.1.2 2SAT Problem . . . 203
12.2 Complete Paths . . . 206
12.2.1 Eulerian Paths . . . 206
12.2.2 Hamiltonian Paths . . . 207
12.2.3 Applications . . . 209
12.3 Maximum Flows . . . 210
12.3.1 Ford–Fulkerson Algorithm . . . 211
12.3.2 Disjoint Paths . . . 214
12.3.3 Maximum Matchings . . . 215
12.3.4 Path Covers . . . 218
12.4 Depth-First Search Trees . . . 220
12.4.1 Biconnectivity . . . 220
12.4.2 Eulerian Subgraphs . . . 221
12.5 Minimum Cost Flows . . . 222
12.5.1 Minimum Cost Paths Algorithm . . . 223
12.5.2 Minimum Weight Matchings . . . 225
12.5.3 Improving the Algorithm . . . 225

**13 Geometry . . . 229**
13.1 Geometric Techniques . . . 229
13.1.1 Complex Numbers . . . 229
13.1.2 Points and Lines . . . 231
13.1.3 Polygon Area . . . 234
13.1.4 Distance Functions . . . 236
13.2 Sweep Line Algorithms . . . 238
13.2.1 Intersection Points . . . 238
13.2.2 Closest Pair Problem . . . 239
13.2.3 Convex Hull Problem . . . 241

**14 String Algorithms** . . . 243
14.1 Basic Topics . . . 243
14.1.1 Trie Structure . . . 244
14.1.2 Dynamic Programming . . . 245
14.2 String Hashing . . . 246
14.2.1 Polynomial Hashing . . . 247
14.2.2 Applications . . . 247
14.2.3 Collisions and Parameters . . . 248
14.3 Z-Algorithm . . . 250
14.3.1 Constructing the Z-Array . . . 250
14.3.2 Applications . . . 252
14.4 Suffix Arrays . . . 253
14.4.1 Prefix Doubling Method . . . 253
14.4.2 Finding Patterns . . . 254
14.4.3 LCP Arrays . . . 255
14.5 String Automata . . . 256
14.5.1 Regular Languages . . . 257
14.5.2 Pattern Matching Automata . . . 258
14.5.3 Suffix Automata . . . 260

**15 Additional Topics** . . . 263
15.1 Square Root Techniques . . . 263
15.1.1 Data Structures . . . 264
15.1.2 Subalgorithms . . . 265
15.1.3 Integer Partitions . . . 267
15.1.4 Mo's Algorithm . . . 268
15.2 Segment Trees Revisited . . . 269
15.2.1 Lazy Propagation . . . 270
15.2.2 Dynamic Trees . . . 273
15.2.3 Data Structures in Nodes . . . 275
15.2.4 Two-Dimensional Trees . . . 276
15.3 Treaps . . . 277
15.3.1 Splitting and Merging . . . 277
15.3.2 Implementation . . . 280
15.3.3 Additional Techniques . . . 281
15.4 Dynamic Programming Optimization . . . 282
15.4.1 Convex Hull Trick . . . 282
15.4.2 Divide and Conquer Optimization . . . 284
15.4.3 Knuth's Optimization . . . 285
15.5 Backtracking Techniques . . . 287
15.5.1 Pruning the Search Tree . . . 287
15.5.2 Heuristic Functions . . . 289

15.6 Miscellaneous . . . . . 291
15.6.1 Meet in the Middle . . . . . 291
15.6.2 Counting Subsets . . . . . 292
15.6.3 Parallel Binary Search . . . . . 294
15.6.4 Dynamic Connectivity . . . . . 295

**Appendix: Mathematical Background** . . . . . 297

**References** . . . . . 303

**Index** . . . . . 305