

# Contents

<b>0</b>	<b>Prologue</b>	<b>7</b>
1	Books and Algorithms . . . . .	7
2	Enter Fibonacci . . . . .	8
3	Big- <i>O</i> notation . . . . .	11
4	Exercises . . . . .	14
<b>1</b>	<b>Algorithms with Numbers</b>	<b>17</b>
1	Basic arithmetic . . . . .	17
1.1	Addition . . . . .	17
1.2	Multiplication and division . . . . .	19
2	Modular arithmetic . . . . .	21
2.1	Modular addition and multiplication . . . . .	23
2.2	Modular exponentiation . . . . .	24
2.3	Euclid's algorithm for greatest common divisor . . . . .	25
2.4	An extension of Euclid's algorithm . . . . .	26
2.5	Modular division . . . . .	27
3	Primality testing . . . . .	28
3.1	Generating random primes . . . . .	32
4	Cryptography . . . . .	33
4.1	Private-Key Schemes: One-time Pad and AES . . . . .	35
4.2	RSA . . . . .	36
5	Universal hashing . . . . .	38
5.1	Hash tables . . . . .	38
5.2	Families of hash functions . . . . .	39
6	Exercises . . . . .	42
<b>2</b>	<b>Divide-and-conquer algorithms</b>	<b>49</b>
1	Multiplication . . . . .	49
2	Recurrence relations . . . . .	51
3	Mergesort . . . . .	54
4	Medians . . . . .	57
5	Matrix multiplication . . . . .	59
6	The fast Fourier transform . . . . .	61
6.1	An alternative representation of polynomials . . . . .	63
6.2	Evaluation by divide-and-conquer . . . . .	64

6.3	Interpolation . . . . .	67
6.4	A closer look at the fast Fourier transform . . . . .	70
7	Exercises . . . . .	75
<b>3</b>	<b>Decompositions of graphs</b>	<b>85</b>
1	Why graphs? . . . . .	85
1.1	How is a graph represented? . . . . .	86
2	Depth-first search in undirected graphs . . . . .	87
2.1	Exploring mazes . . . . .	87
2.2	Depth-first search . . . . .	89
2.3	Connectivity in undirected graphs . . . . .	90
2.4	Previsit and postvisit orderings . . . . .	91
3	Depth-first search in directed graphs . . . . .	92
3.1	Types of edges . . . . .	92
3.2	Directed acyclic graphs . . . . .	93
4	Strongly connected components . . . . .	95
4.1	Defining connectivity for directed graphs . . . . .	95
4.2	An efficient algorithm . . . . .	96
5	Exercises . . . . .	99
<b>4</b>	<b>Paths in graphs</b>	<b>109</b>
1	Distances . . . . .	109
2	Breadth-first search . . . . .	110
3	Lengths on edges . . . . .	112
4	Dijkstra's algorithm . . . . .	112
4.1	An adaptation of breadth-first search . . . . .	112
4.2	An alternative derivation . . . . .	117
4.3	Running time . . . . .	118
5	Priority queue implementations . . . . .	119
5.1	Array . . . . .	119
5.2	Binary heap . . . . .	119
5.3	$d$ -ary heap . . . . .	120
6	Shortest paths in the presence of negative edges . . . . .	122
6.1	Negative edges . . . . .	122
6.2	Negative cycles . . . . .	123
7	Shortest paths in dags . . . . .	124
8	Exercises . . . . .	126
<b>5</b>	<b>Greedy algorithms</b>	<b>133</b>
1	Minimum spanning trees . . . . .	133
1.1	A greedy approach . . . . .	134
1.2	The cut property . . . . .	135
1.3	Kruskal's algorithm . . . . .	136
1.4	A data structure for disjoint sets . . . . .	137
1.5	Prim's algorithm . . . . .	143
2	Huffman encoding . . . . .	145

3	Horn formulas . . . . .	149
4	Set cover . . . . .	150
5	Exercises . . . . .	153
<b>6</b>	<b>Dynamic Programming</b>	<b>161</b>
1	Shortest paths in dags, revisited . . . . .	161
2	Longest increasing subsequences . . . . .	162
3	Edit distance . . . . .	165
4	Knapsack . . . . .	171
5	Chain matrix multiplication . . . . .	174
6	Shortest Paths . . . . .	175
7	Independent sets in trees . . . . .	179
8	Exercises . . . . .	181
<b>7</b>	<b>Linear Programming and Reductions</b>	<b>191</b>
1	An introduction to linear programming . . . . .	191
1.1	Example: profit maximization . . . . .	192
1.2	Example: production planning . . . . .	195
1.3	Example: optimum bandwidth allocation . . . . .	197
1.4	Variants of linear programming . . . . .	199
2	Flows in networks . . . . .	201
2.1	Shipping oil . . . . .	201
2.2	Maximizing flow . . . . .	201
2.3	A closer look at the algorithm . . . . .	202
2.4	A certificate of optimality . . . . .	204
2.5	Efficiency . . . . .	205
3	Bipartite matching . . . . .	208
4	Duality . . . . .	209
5	Zero-sum games . . . . .	212
6	The Simplex algorithm . . . . .	215
6.1	Vertices and neighbors in $n$ -dimensional space . . . . .	216
6.2	The algorithm . . . . .	217
6.3	Loose ends . . . . .	220
6.4	The running time of simplex . . . . .	221
7	Postscript: circuit evaluation . . . . .	223
<b>8</b>	<b>NP-complete Problems</b>	<b>225</b>
1	Search Problems . . . . .	225
2	The Reductions . . . . .	239
<b>9</b>	<b>Coping with NP-completeness</b>	<b>255</b>
1	Intelligent exhaustive search . . . . .	256
1.1	Backtracking . . . . .	256
1.2	Branch-and-bound . . . . .	258
2	Approximation algorithms . . . . .	261
2.1	Vertex cover . . . . .	261

2.2	Clustering . . . . .	263
2.3	TSP . . . . .	265
2.4	Knapsack . . . . .	266
2.5	The approximability hierarchy . . . . .	267
3	Local Search Heuristics . . . . .	268
3.1	Traveling salesman, once more . . . . .	268
3.2	Graph partitioning . . . . .	271
3.3	Dealing with local optima . . . . .	273
<b>10</b>	<b>Quantum Algorithms</b>	<b>277</b>
1	Qubits, superposition, and measurement . . . . .	277
2	The plan . . . . .	280
3	The quantum Fourier transform . . . . .	282
4	Periodicity . . . . .	284
5	Factoring as periodicity . . . . .	286
6	The quantum algorithm for factoring . . . . .	287
7	Quantum circuits . . . . .	288
7.1	Elementary quantum gates . . . . .	288
7.2	The quantum Fourier transform circuit . . . . .	288