Contents

Preface xiii

I Preliminaries 1

1 Data Structures and Algorithms 3
   1.1 A Philosophy of Data Structures 4
      1.1.1 The Need for Data Structures 4
      1.1.2 Costs and Benefits 6
   1.2 Abstract Data Types and Data Structures 8
   1.3 Design Patterns 12
      1.3.1 Flyweight 13
      1.3.2 Visitor 13
      1.3.3 Composite 14
      1.3.4 Strategy 15
   1.4 Problems, Algorithms, and Programs 16
   1.5 Further Reading 18
   1.6 Exercises 20

2 Mathematical Preliminaries 23
   2.1 Sets and Relations 23
   2.2 Miscellaneous Notation 27
   2.3 Logarithms 29
   2.4 Summations and Recurrences 30
   2.5 Recursion 34
   2.6 Mathematical Proof Techniques 36
**2.6** Direct Proof 37
2.6.2 Proof by Contradiction 37
2.6.3 Proof by Mathematical Induction 38
2.7 Estimation 44
2.8 Further Reading 45
2.9 Exercises 46

3 **Algorithm Analysis** 53
3.1 Introduction 53
3.2 Best, Worst, and Average Cases 59
3.3 A Faster Computer, or a Faster Algorithm? 60
3.4 Asymptotic Analysis 63
3.4.1 Upper Bounds 63
3.4.2 Lower Bounds 65
3.4.3 Θ Notation 66
3.4.4 Simplifying Rules 67
3.4.5 Classifying Functions 68
3.5 Calculating the Running Time for a Program 69
3.6 Analyzing Problems 74
3.7 Common Misunderstandings 75
3.8 Multiple Parameters 77
3.9 Space Bounds 78
3.10 Speeding Up Your Programs 80
3.11 Empirical Analysis 83
3.12 Further Reading 84
3.13 Exercises 85
3.14 Projects 89

**II Fundamental Data Structures** 91

4 **Lists, Stacks, and Queues** 93
4.1 Lists 94
4.1.1 Array-Based List Implementation 97
4.1.2 Linked Lists 100
4.1.3 Comparison of List Implementations 108
## Contents

4.1.4 Element Implementations 111  
4.1.5 Doubly Linked Lists 112  
4.2 Stacks 117  
4.2.1 Array-Based Stacks 117  
4.2.2 Linked Stacks 120  
4.2.3 Comparison of Array-Based and Linked Stacks 121  
4.2.4 Implementing Recursion 121  
4.3 Queues 125  
4.3.1 Array-Based Queues 125  
4.3.2 Linked Queues 128  
4.3.3 Comparison of Array-Based and Linked Queues 131  
4.4 Dictionaries 131  
4.5 Further Reading 138  
4.6 Exercises 138  
4.7 Projects 141  

5 Binary Trees 145  
5.1 Definitions and Properties 145  
5.1.1 The Full Binary Tree Theorem 147  
5.1.2 A Binary Tree Node ADT 149  
5.2 Binary Tree Traversals 149  
5.3 Binary Tree Node Implementations 154  
5.3.1 Pointer-Based Node Implementations 154  
5.3.2 Space Requirements 160  
5.3.3 Array Implementation for Complete Binary Trees 161  
5.4 Binary Search Trees 163  
5.5 Heaps and Priority Queues 170  
5.6 Huffman Coding Trees 178  
5.6.1 Building Huffman Coding Trees 179  
5.6.2 Assigning and Using Huffman Codes 185  
5.6.3 Search in Huffman Trees 188  
5.7 Further Reading 188  
5.8 Exercises 189  
5.9 Projects 192  

6 Non-Binary Trees 195
### Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>6.1</td>
<td>General Tree Definitions and Terminology</td>
<td>195</td>
</tr>
<tr>
<td>6.1.1</td>
<td>An ADT for General Tree Nodes</td>
<td>196</td>
</tr>
<tr>
<td>6.1.2</td>
<td>General Tree Traversals</td>
<td>197</td>
</tr>
<tr>
<td>6.2</td>
<td>The Parent Pointer Implementation</td>
<td>199</td>
</tr>
<tr>
<td>6.3</td>
<td>General Tree Implementations</td>
<td>206</td>
</tr>
<tr>
<td>6.3.1</td>
<td>List of Children</td>
<td>206</td>
</tr>
<tr>
<td>6.3.2</td>
<td>The Left-Child/Right-Sibling Implementation</td>
<td>206</td>
</tr>
<tr>
<td>6.3.3</td>
<td>Dynamic Node Implementations</td>
<td>207</td>
</tr>
<tr>
<td>6.3.4</td>
<td>Dynamic “Left-Child/Right-Sibling” Implementation</td>
<td>210</td>
</tr>
<tr>
<td>6.4</td>
<td>$K$-ary Trees</td>
<td>210</td>
</tr>
<tr>
<td>6.5</td>
<td>Sequential Tree Implementations</td>
<td>212</td>
</tr>
<tr>
<td>6.6</td>
<td>Further Reading</td>
<td>215</td>
</tr>
<tr>
<td>6.7</td>
<td>Exercises</td>
<td>215</td>
</tr>
<tr>
<td>6.8</td>
<td>Projects</td>
<td>218</td>
</tr>
</tbody>
</table>

### III Sorting and Searching

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>Internal Sorting</td>
<td>221</td>
</tr>
<tr>
<td>7.1</td>
<td>Sorting Terminology and Notation</td>
<td>223</td>
</tr>
<tr>
<td>7.2</td>
<td>Three $\Theta(n^2)$ Sorting Algorithms</td>
<td>225</td>
</tr>
<tr>
<td>7.2.1</td>
<td>Insertion Sort</td>
<td>225</td>
</tr>
<tr>
<td>7.2.2</td>
<td>Bubble Sort</td>
<td>227</td>
</tr>
<tr>
<td>7.2.3</td>
<td>Selection Sort</td>
<td>229</td>
</tr>
<tr>
<td>7.2.4</td>
<td>The Cost of Exchange Sorting</td>
<td>230</td>
</tr>
<tr>
<td>7.3</td>
<td>Shellsort</td>
<td>231</td>
</tr>
<tr>
<td>7.4</td>
<td>Mergesort</td>
<td>233</td>
</tr>
<tr>
<td>7.5</td>
<td>Quicksort</td>
<td>236</td>
</tr>
<tr>
<td>7.6</td>
<td>Heapsort</td>
<td>243</td>
</tr>
<tr>
<td>7.7</td>
<td>Binsort and Radix Sort</td>
<td>244</td>
</tr>
<tr>
<td>7.8</td>
<td>An Empirical Comparison of Sorting Algorithms</td>
<td>251</td>
</tr>
<tr>
<td>7.9</td>
<td>Lower Bounds for Sorting</td>
<td>253</td>
</tr>
<tr>
<td>7.10</td>
<td>Further Reading</td>
<td>257</td>
</tr>
<tr>
<td>7.11</td>
<td>Exercises</td>
<td>257</td>
</tr>
<tr>
<td>7.12</td>
<td>Projects</td>
<td>261</td>
</tr>
</tbody>
</table>
8 File Processing and External Sorting  265
  8.1 Primary versus Secondary Storage  265
  8.2 Disk Drives  268
    8.2.1 Disk Drive Architecture  268
    8.2.2 Disk Access Costs  272
  8.3 Buffers and Buffer Pools  274
  8.4 The Programmer’s View of Files  282
  8.5 External Sorting  283
    8.5.1 Simple Approaches to External Sorting  285
    8.5.2 Replacement Selection  288
    8.5.3 Multiway Merging  290
  8.6 Further Reading  295
  8.7 Exercises  295
  8.8 Projects  299

9 Searching  301
  9.1 Searching Unsorted and Sorted Arrays  302
  9.2 Self-Organizing Lists  307
  9.3 Bit Vectors for Representing Sets  313
  9.4 Hashing  314
    9.4.1 Hash Functions  315
    9.4.2 Open Hashing  320
    9.4.3 Closed Hashing  321
    9.4.4 Analysis of Closed Hashing  331
    9.4.5 Deletion  334
  9.5 Further Reading  335
  9.6 Exercises  336
  9.7 Projects  338

10 Indexing  341
  10.1 Linear Indexing  343
  10.2 ISAM  346
  10.3 Tree-based Indexing  348
  10.4 2-3 Trees  350
  10.5 B-Trees  355
    10.5.1 B+-Trees  358
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>10.5.2 B-Tree Analysis</td>
<td>364</td>
</tr>
<tr>
<td>10.6 Further Reading</td>
<td>365</td>
</tr>
<tr>
<td>10.7 Exercises</td>
<td>365</td>
</tr>
<tr>
<td>10.8 Projects</td>
<td>367</td>
</tr>
<tr>
<td>IV Advanced Data Structures</td>
<td>369</td>
</tr>
<tr>
<td>11 Graphs</td>
<td>371</td>
</tr>
<tr>
<td>11.1 Terminology and Representations</td>
<td>372</td>
</tr>
<tr>
<td>11.2 Graph Implementations</td>
<td>376</td>
</tr>
<tr>
<td>11.3 Graph Traversals</td>
<td>380</td>
</tr>
<tr>
<td>11.3.1 Depth-First Search</td>
<td>383</td>
</tr>
<tr>
<td>11.3.2 Breadth-First Search</td>
<td>384</td>
</tr>
<tr>
<td>11.3.3 Topological Sort</td>
<td>384</td>
</tr>
<tr>
<td>11.4 Shortest-Paths Problems</td>
<td>388</td>
</tr>
<tr>
<td>11.4.1 Single-Source Shortest Paths</td>
<td>389</td>
</tr>
<tr>
<td>11.5 Minimum-Cost Spanning Trees</td>
<td>393</td>
</tr>
<tr>
<td>11.5.1 Prim’s Algorithm</td>
<td>393</td>
</tr>
<tr>
<td>11.5.2 Kruskal’s Algorithm</td>
<td>397</td>
</tr>
<tr>
<td>11.6 Further Reading</td>
<td>399</td>
</tr>
<tr>
<td>11.7 Exercises</td>
<td>399</td>
</tr>
<tr>
<td>11.8 Projects</td>
<td>402</td>
</tr>
<tr>
<td>12 Lists and Arrays Revisited</td>
<td>405</td>
</tr>
<tr>
<td>12.1 Multilists</td>
<td>405</td>
</tr>
<tr>
<td>12.2 Matrix Representations</td>
<td>408</td>
</tr>
<tr>
<td>12.3 Memory Management</td>
<td>412</td>
</tr>
<tr>
<td>12.3.1 Dynamic Storage Allocation</td>
<td>414</td>
</tr>
<tr>
<td>12.3.2 Failure Policies and Garbage Collection</td>
<td>421</td>
</tr>
<tr>
<td>12.4 Further Reading</td>
<td>425</td>
</tr>
<tr>
<td>12.5 Exercises</td>
<td>426</td>
</tr>
<tr>
<td>12.6 Projects</td>
<td>427</td>
</tr>
<tr>
<td>13 Advanced Tree Structures</td>
<td>429</td>
</tr>
<tr>
<td>13.1 Tries</td>
<td>429</td>
</tr>
</tbody>
</table>
## Contents

13.2 Balanced Trees 434
   13.2.1 The AVL Tree 435
   13.2.2 The Splay Tree 437
13.3 Spatial Data Structures 440
   13.3.1 The K-D Tree 442
   13.3.2 The PR quadtree 447
   13.3.3 Other Point Data Structures 451
   13.3.4 Other Spatial Data Structures 453
13.4 Further Reading 453
13.5 Exercises 454
13.6 Projects 455

V Theory of Algorithms 459

14 Analysis Techniques 461
   14.1 Summation Techniques 462
   14.2 Recurrence Relations 467
      14.2.1 Estimating Upper and Lower Bounds 467
      14.2.2 Expanding Recurrences 470
      14.2.3 Divide and Conquer Recurrences 472
      14.2.4 Average-Case Analysis of Quicksort 474
   14.3 Amortized Analysis 476
   14.4 Further Reading 479
   14.5 Exercises 479
   14.6 Projects 483

15 Lower Bounds 485
   15.1 Introduction to Lower Bounds Proofs 486
   15.2 Lower Bounds on Searching Lists 488
      15.2.1 Searching in Unsorted Lists 488
      15.2.2 Searching in Sorted Lists 490
   15.3 Finding the Maximum Value 491
   15.4 Adversarial Lower Bounds Proofs 493
   15.5 State Space Lower Bounds Proofs 496
   15.6 Finding the $i$th Best Element 499
| 15.7 | Optimal Sorting | 501 |
| 15.8 | Further Reading | 504 |
| 15.9 | Exercises | 504 |
| 15.10 | Projects | 507 |

16 Patterns of Algorithms 509

16.1 Dynamic Programming 509
16.1.1 The Knapsack Problem 511
16.1.2 All-Pairs Shortest Paths 513
16.2 Randomized Algorithms 515
16.2.1 Randomized algorithms for finding large values 515
16.2.2 Skip Lists 516
16.3 Numerical Algorithms 522
16.3.1 Exponentiation 523
16.3.2 Largest Common Factor 523
16.3.3 Matrix Multiplication 524
16.3.4 Random Numbers 526
16.3.5 The Fast Fourier Transform 527
16.4 Further Reading 532
16.5 Exercises 532
16.6 Projects 533

17 Limits to Computation 535

17.1 Reductions 536
17.2 Hard Problems 541
17.2.1 The Theory of \( \mathcal{NP} \)-Completeness 543
17.2.2 \( \mathcal{NP} \)-Completeness Proofs 547
17.2.3 Coping with \( \mathcal{NP} \)-Complete Problems 552
17.3 Impossible Problems 555
17.3.1 Uncountability 556
17.3.2 The Halting Problem Is Unsolvable 559
17.4 Further Reading 561
17.5 Exercises 562
17.6 Projects 564

Bibliography 567
<table>
<thead>
<tr>
<th>Contents</th>
<th>xi</th>
</tr>
</thead>
<tbody>
<tr>
<td>Index</td>
<td>573</td>
</tr>
</tbody>
</table>