Contents

1 Algorithms 11
  1.1 Pretest 12
  1.2 Algorithms and correctness 15
    Bounded linear search 16  Binary search 24
    A linked list class 32  Selection sort 37
  1.3 Algorithms and efficiency 43
    A line-by-line analysis 44
    Big-oh and big-theta defined formally 52
    Constant factors 59  Recurrences 62
  1.4 Experiments 65
    Counting comparisons 65  Merge sort and quick sort 73
    Measuring running time 83
    Sorting algorithm summary 87
  1.5 Chapter summary 91

2 Data structures 93
  2.1 Abstract data types 94
    The ADT cannon 95  Lists 96
    Stacks and queues 98  Sets 101
    Maps 102  Bags 105
  2.2 Array-based data structures 111
    Random access 112  ArrayList 114
    Efficiency of an array 117  ArrayStack 119
  2.3 Linked data structures 121
    LinkedStack 123  Recursion in the node 124
2 CONTENTS

2.4 Data structures built from abstractions 129
   Multi-dimensional arrays 129
   Ring buffers 131  Linked trees 133
2.5 Data structures adapted from ADTs 141
   Lists adapted as stack and queues 141
   Lists adapted as maps 144  Maps adapted as bags 146
   Bags adapted as sets 147
2.6 ADTs and data structures in other languages 152
   ADTs in C 153  ADTs in SML 167
   ADTs in Python 175
2.7 Programming practices 181
   Assertions 181  Nested classes 182
   Iterators 186  Interfaces 197
2.8 The road ahead 200
2.9 Chapter summary 204

3 Case studies 207
3.1 Linear-time sorting algorithms 208
   Counting sort 209  Radix sort 213
   Bucket sort 218
3.2 Disjoint sets and array forests 223
   The disjoint set ADT 226  The array forest data structure 228
   Find and union strategies 232
3.3 Priority queues and heaps 243
   The priority queue ADT 244
   Brute force implementations 245
   The heap data structure 251
   Heap sort 260  HeapPriorityQueue 265
3.4 N-Sets and bit vectors 272
   The N-set ADT 275  The bit vector data structure 279
3.5 Skiplists 286
   Map data structure performance 286
   The skiplist data structure 290
   Skiplist performance 293
3.6 Chapter summary 302
## CONTENTS

### 4 Graphs 305

#### 4.1 Concepts 306

#### 4.2 Implementation 311
- Adjacency list and adjacency matrix 314
- Space and time efficiency 320
- Weighted graphs 324

#### 4.3 Traversal 327
- Breadth-first and depth-first 329
- Implementing traversal algorithms 334
- The complexity of traversal 340

#### 4.4 Minimum spanning trees 347
- The MST problem 349
- Kruskal’s algorithm 354
- Prim’s algorithm 361
- Performance 366

#### 4.5 Shortest paths 370
- The SSSP problem 371
- The Bellman-Ford algorithm 377
- Dijkstra’s algorithm 381
- Routing algorithms 386

#### 4.6 Chapter summary 392

### 5 Search trees 397

#### 5.1 Binary search trees 398
- Iterative implementation 402
- Recursive implementation 408
- Object-oriented recursive implementation 410
- Performance 415

#### 5.2 The balanced tree problem 419

#### 5.3 AVL trees 427
- Violations 428
- Implementation 433
- Performance 438

#### 5.4 Traditional red-black trees 441
- Insertion 447
- Deletion 453
- Performance 461

#### 5.5 Left-leaning red-black trees 464
- Insertion 466
- Deletion 470
- Performance 474

#### 5.6 B-trees 479
## CONTENTS

- Two-three trees 480
- Deriving B-trees 490
- Implementing B-trees 494
- Insertion 499
- Performance 507

### 5.7 Chapter summary 510

## 6 Dynamic programming 513

### 6.1 Overlapping subproblems 514
- Characterizing coin-changing 519
- Computing the best bag of coins 527

### 6.2 Three problems 531
- The knapsack problem 531
- The longest common subsequence problem 536
- The optimal matrix multiplication problem 539

### 6.3 Elements of dynamic programming 543

### 6.4 Three solutions 547
- A solution to the knapsack problem 547
- A solution to the the longest common subsequence problem 551
- A solution to the optimal matrix multiplication problem 554

### 6.5 Optimal binary search trees 563
- The optimal BST problem 565
- Building tables 571

### 6.6 Natural-breaks classification 576
- Recursive formulation 581
- A dynamic-programming algorithm 583

### 6.7 Chapter summary 587

## 7 Hash tables 591

### 7.1 Hash table basics 592

### 7.2 Separate chaining 601
- Design 601
- Implementation 604

### 7.3 Open addressing 608
- Basic implementation 610
- Deletion 615
- Alternate probe strategies 620

### 7.4 Hash functions 629
- Integer hash functions 632
- String hash functions 641
CONTENTS

7.5 Perfect hashing 647
   Universal hash functions 648
   Design of the table 652

7.6 Chapter summary 656

8 String processing 659

8.1 Sorting algorithms for strings 664
   String quick sort 666
   String bucket sort 671
   String radix sort 674

8.2 Tries 678
   Abstraction 679
   Implementation 682

8.3 Huffman encoding 691
   Text encoding schemes 692
   Building the key 699
   Optimality 702

8.4 Edit distance 708

8.5 Grammars 714
   Recursive descent parsing 717
   CKY parsing described 728
   CKY parsing implemented 734

8.6 Chapter summary 738

Index 741