A PRACTICAL GUIDE TO DATA STRUCTURES AND ALGORITHMS USING JAVA

Sally and Ken Goldman
A PRACTICAL GUIDE TO DATA STRUCTURES AND ALGORITHMS USING JAVA
A PRACTICAL GUIDE TO DATA STRUCTURES AND ALGORITHMS USING JAVA

Sally Goldman
Kenneth Goldman
Washington University
Saint Louis, Missouri, U.S.A.
to Julie, Ben, and Mark
# Contents

Preface .................................................. xxiii
Acknowledgments .......................................... xxv
Authors .................................................. xxvii

## I  INTRODUCTION

### 1  Design Principles
1.1  Object-Oriented Design and This Book ......................... 4
1.2  Encapsulation ............................................ 5
1.3  Invariants and Representation Properties ....................... 6
1.4  Interfaces and Data Abstraction ................................. 7
1.5  Case Study on Conceptual Design: Historical Event Collection ............................................. 8
1.6  Case Study on Structural Design: Trees ........................ 10
1.7  Further Reading ........................................... 13

### 2  Selecting an Abstract Data Type
2.1  An Illustrative Example .................................. 16
2.2  Broad ADT groups .......................................... 19
2.3  Partition of a Set .......................................... 21
2.4  A Collection of Elements .................................. 21
2.5  Markers and Trackers ...................................... 23
2.6  Positioning and Finding Elements ............................ 24
2.6.1  Manually Positioned Collections ........................ 26
2.6.2  Algorithmically Positioned Collections .................. 26
2.7  Graphs .................................................. 28

### 3  How to Use This Book
3.1  Conventions .............................................. 35
3.2  Parts II and III Presentation Structure ......................... 36
3.2.1  ADT Chapters ......................................... 36
3.2.2  Data Structures ........................................ 38
3.2.3  Algorithms ............................................. 40
3.3  Appendices and CD ........................................ 41

## II  COLLECTION DATA STRUCTURES AND ALGORITHMS

### 4  Part II Organization

### 5  Foundations
5.1  Wrappers for Delegation ................................ 50
5.2  Objects Abstract Class .................................... 51
5.2.1  Singleton Classes: Empty and Deleted .................... 51
5.2.2  Object Equivalence .................................... 51
5.2.3  Object Comparison ..................................... 53
9 Positional Collection ADT 107
  9.1 Interface ......................................................... 107
  9.2 Positional Collection Locator Interface ..................... 109
  9.3 Terminology ..................................................... 110
  9.4 Competing ADTs ............................................... 110
  9.5 Selecting a Data Structure ................................... 111
    9.5.1 Tradeoffs among Array-Based Data Structures .......... 114
    9.5.2 Tradeoffs among List-Based Data Structures .......... 115
  9.6 Summary of Positional Collection Data Structures .......... 116
  9.7 Further Reading ............................................... 119

10 Abstract Positional Collection 121
  10.1 Abstract Positional Collection ............................ 121
  10.2 Internal Representation ..................................... 121
  10.3 Quick Method Reference ..................................... 122

11 Array Data Structure 125
  11.1 Internal Representation ..................................... 126
  11.2 Representation Properties .................................. 128
  11.3 Methods ....................................................... 128
    11.3.1 Constructors ............................................. 128
    11.3.2 Trivial Accessors ......................................... 129
    11.3.3 Representation Accessors ................................. 130
    11.3.4 Algorithmic Accessors .................................. 131
    11.3.5 Representation Mutators .................................. 132
    11.3.6 Content Mutators ......................................... 133
    11.3.7 Locator Initializers ..................................... 139
  11.4 Sorting Algorithms ........................................... 140
    11.4.1 Insertion Sort ........................................... 143
    11.4.2 Mergesort ................................................. 145
    11.4.3 Heap Sort .................................................. 147
    11.4.4 Tree Sort .................................................. 149
    11.4.5 Quicksort .................................................. 149
    11.4.6 Radix Sort ................................................. 155
    11.4.7 Bucket Sort ............................................... 157
  11.5 Selection and Median Finding ................................ 158
  11.6 Basic Marker Inner Class .................................... 160
  11.7 Marker Inner Class ........................................... 162
  11.8 Performance Analysis ......................................... 163
  11.9 Further Reading .............................................. 166
  11.10 Quick Method Reference .................................... 167

12 Circular Array Data Structure 171
  12.1 Internal Representation ..................................... 171
  12.2 Representation Properties .................................. 173
  12.3 Methods ....................................................... 173
    12.3.1 Constructors ............................................. 173
    12.3.2 Representation Accessors ................................ 174
    12.3.3 Representation Mutators .................................. 175
    12.3.4 Content Mutators ......................................... 176
  12.4 Performance Analysis ......................................... 181
  12.5 Quick Method Reference ..................................... 183
## 13 Dynamic Array and Dynamic Circular Array Data Structures

13.1 Dynamic Array .......................... 185
13.2 Internal Representation ............. 186
13.3 Representation Properties .......... 187
13.4 Methods ................................ 187
    13.4.1 Constructors ..................... 187
    13.4.2 Representation Mutators ....... 188
    13.4.3 Content Mutators ............... 189
13.5 Performance Analysis ................. 190
13.6 Dynamic Circular Array .............. 191

## 14 Tracked Array Data Structure

14.1 Internal Representation .............. 194
14.2 Representation Properties .......... 197
14.3 Node Inner Class ..................... 198
14.4 Tracked Array Methods ............... 198
    14.4.1 Constructors ................... 198
    14.4.2 Representation Accessors ...... 199
    14.4.3 Algorithmic Accessors ......... 199
    14.4.4 Representation Mutators ..... 200
    14.4.5 Content Mutators .............. 201
    14.4.6 Locator Initializers .......... 204
14.5 Wrappers for Sorting Tracked Arrays 205
14.6 Tracker Inner Class .................. 207
14.7 Performance Analysis ................. 213
14.8 Quick Method Reference .............. 214

## 15 Singly Linked List Data Structure

15.1 Internal Representation .............. 217
15.2 Representation Properties .......... 220
15.3 List Item Inner Class ............... 220
15.4 Singly Linked List Methods .......... 221
    15.4.1 Constructors and Factory Methods 221
    15.4.2 Representation Accessors ....... 222
    15.4.3 Algorithmic Accessors ........ 222
    15.4.4 Representation Mutators ..... 224
    15.4.5 Content Mutators .............. 224
    15.4.6 Locator Initializers .......... 231
15.5 Sorting Algorithms Revisited ....... 232
    15.5.1 Insertion Sort .................. 232
    15.5.2 Mergesort ...................... 233
    15.5.3 Heap Sort ...................... 235
    15.5.4 Tree Sort ...................... 236
    15.5.5 Quicksort ...................... 237
    15.5.6 Radix Sort ..................... 242
    15.5.7 Bucket Sort .................... 244
15.6 Selection and Median Finding ...... 244
15.7 Tracker Inner Class .................. 246
15.8 Performance Analysis ................. 251
15.9 Quick Method Reference .............. 254
16 Doubly Linked List Data Structure.................................................................257
16.1 Internal Representation ..............................................................................257
16.2 Representation Properties .........................................................................258
16.3 Doubly Linked List Item Inner Class ..........................................................259
16.4 Doubly Linked List Methods .......................................................................259
   16.4.1 Constructors and Factory Methods ......................................................259
   16.4.2 Representation Accessors ......................................................................260
   16.4.3 Algorithmic Accessors ..........................................................................260
   16.4.4 Representation Mutators .......................................................................261
16.5 Performance Analysis ..................................................................................261
16.6 Quick Method Reference ............................................................................263

17 Buffer ADT and Its Implementation .............................................................265
17.1 Internal Representation ...............................................................................265
17.2 Representation Properties ..........................................................................266
17.3 Methods .......................................................................................................266
   17.3.1 Constructors .........................................................................................266
   17.3.2 Trivial Accessors ....................................................................................267
   17.3.3 Algorithmic Accessors ..........................................................................268
   17.3.4 Content Mutators ..................................................................................268
   17.3.5 Locator Initializers ................................................................................269
17.4 Performance Analysis ..................................................................................270
17.5 Quick Method Reference ............................................................................270

18 Queue ADT and Implementation .................................................................271
18.1 Internal Representation ...............................................................................271
18.2 Methods .......................................................................................................271
18.3 Performance Analysis ..................................................................................272
18.4 Quick Method Reference ............................................................................273

19 Stack ADT and Implementation .....................................................................275
19.1 Internal Representation ...............................................................................275
19.2 Methods .......................................................................................................276
19.3 Performance Analysis ..................................................................................278
19.4 Quick Method Reference ............................................................................278

20 Set ADT .........................................................................................................279
20.1 Case Study: Airline Travel Agent ...............................................................279
20.2 Interface ......................................................................................................280
20.3 Terminology .................................................................................................280
20.4 Hasher Interface .........................................................................................281
20.5 Competing ADTs .........................................................................................282
20.6 Selecting a Data Structure .........................................................................283
20.7 Summary of Set Data Structures .................................................................286
20.8 Further Reading ..........................................................................................289

21 Direct Addressing Data Structure ...............................................................291
21.1 Internal Representation ...............................................................................291
21.2 Representation Properties ..........................................................................293
21.3 Default Direct Addressing Hasher Class ....................................................293
21.4 Methods ......................................................................................................293
22 Open Addressing Data Structure  
22.1 Internal Representation .......................... 305  
22.2 Representation Properties .......................... 308  
22.3 Default Open Addressing Hasher Class .................. 309  
22.4 Open Addressing Methods .......................... 310  
22.4.1 Constructors .................................. 310  
22.4.2 Trivial Accessors .................................. 311  
22.4.3 Representation Accessors ...................... 311  
22.4.4 Selecting a Hash Function ...................... 312  
22.4.5 Algorithmic Accessors ....................... 313  
22.4.6 Representation Mutators ....................... 314  
22.4.7 Content Mutators .......................... 316  
22.5 Performance Analysis .......................... 317  
22.6 Quick Method Reference .......................... 318  

23 Separate Chaining Data Structure  
23.1 Internal Representation .......................... 322  
23.2 Representation Properties .......................... 324  
23.3 Chain Item Inner Class .......................... 325  
23.4 Default Separate Chaining Hasher Class .............. 326  
23.5 Separate Chaining Methods .......................... 326  
23.5.1 Constructors .................................. 326  
23.5.2 Trivial Accessors .................................. 328  
23.5.3 Algorithmic Accessors ....................... 328  
23.5.4 Representation Mutators ....................... 329  
23.5.5 Content Mutators .......................... 331  
23.5.6 Locator Initializers .......................... 333  
23.6 Marker Inner Class .......................... 334  
23.7 Performance Analysis .......................... 339  
23.8 Quick Method Reference .......................... 340  

24 Priority Queue ADT  
24.1 Case Study: Huffman Compression .................. 343  
24.2 Interface .................................. 345  
24.3 Priority Queue Locator Interface .......................... 345  
24.4 Selecting a Data Structure .......................... 346  
24.5 Terminology .................................. 347  
24.6 Competing ADTs .................................. 347  
24.7 Summary of Priority Queue Data Structures ............. 348  
24.8 Further Reading .......................... 351
25 **Binary Heap Data Structure** 353
   25.1 Internal Representation .................................................. 353
   25.2 Representation Properties .................................................. 356
   25.3 Methods ................................................................. 356
      25.3.1 Constructors .......................................................... 356
      25.3.2 Trivial Accessors ....................................................... 357
      25.3.3 Representation Accessors ............................................ 357
      25.3.4 Algorithmic Accessors .............................................. 357
      25.3.5 Representation Mutators ............................................ 358
      25.3.6 Content Mutators .................................................... 361
      25.3.7 Locator Initializers ................................................ 366
   25.4 Locator Inner Class ...................................................... 367
   25.5 Performance Analysis .................................................... 368
   25.6 Quick Method Reference ................................................. 370

26 **Leftist Heap Data Structure** 373
   26.1 Internal Representation .................................................. 373
   26.2 Representation Properties ................................................ 375
   26.3 Leftist Heap Node Inner Class ......................................... 376
   26.4 Leftist Heap Methods ..................................................... 378
      26.4.1 Constructors .......................................................... 378
      26.4.2 Algorithmic Accessors .............................................. 379
      26.4.3 Content Mutators .................................................... 382
      26.4.4 Locator Initializers ................................................ 391
   26.5 Tracker Inner Class ...................................................... 391
   26.6 Performance Analysis .................................................... 393
   26.7 Quick Method Reference ................................................. 396

27 **Pairing Heap Data Structure** 399
   27.1 Internal Representation .................................................. 399
   27.2 Representation Properties ................................................ 402
   27.3 Heap Node Inner Class ................................................... 402
   27.4 Pairing Heap Methods .................................................... 405
      27.4.1 Constructors and Factory Methods ................................ 406
      27.4.2 Algorithmic Accessors .............................................. 406
      27.4.3 Representation Mutators ............................................ 407
      27.4.4 Content Mutators .................................................... 408
      27.4.5 Locator Initializers ................................................ 414
   27.5 Tracker Inner Class ...................................................... 415
   27.6 Performance Analysis .................................................... 418
   27.7 Quick Method Reference ................................................. 419

28 **Fibonacci Heap Data Structure** 423
   28.1 Internal Representation .................................................. 424
   28.2 Representation Properties ................................................ 426
   28.3 Fibonacci Heap Node Inner Class ..................................... 426
   28.4 Fibonacci Heap Methods ................................................ 428
      28.4.1 Constructors and Factory Methods ................................ 428
      28.4.2 Representation Mutators ............................................ 429
      28.4.3 Content Mutators .................................................... 433
   28.5 Performance Analysis .................................................... 437
   28.6 Quick Method Reference ................................................. 441
# Ordered Collection ADT

## 29.1 Case Study: Historical Event Collection (Range Queries)

## 29.2 Case Study: Linux Virtual Memory Map

## 29.3 Interface

## 29.4 Terminology

## 29.5 Competing ADTs

## 29.6 Selecting a Data Structure

## 29.7 Summary of Ordered Collection Data Structures

## 29.8 Further Reading

# Sorted Array Data Structure

## 30.1 Internal Representation

## 30.2 Representation Properties

## 30.3 Methods

### 30.3.1 Constructors

### 30.3.2 Trivial Accessors

### 30.3.3 Binary Search Algorithm

### 30.3.4 Algorithmic Accessors

### 30.3.5 Content Mutators

### 30.3.6 Utilities for the B-Tree and B+-Tree Classes

### 30.3.7 Locator Initializers

## 30.4 Performance Analysis

## 30.5 Quick Method Reference

# Abstract Search Tree Class

## 31.1 Internal Representation

## 31.2 Representation Properties

## 31.3 Abstract TreeNode Inner Class

## 31.4 Abstract Search Tree Class

## 31.5 Abstract Search Tree Methods

### 31.5.1 Constructors

### 31.5.2 Algorithmic Accessors

### 31.5.3 Content Mutators

## 31.6 Quick Method Reference

# Binary Search Tree Data Structure

## 32.1 Internal Representation

## 32.2 Representation Properties

## 32.3 BSTNode Inner Class

## 32.4 Binary Search Tree Methods

### 32.4.1 Constructors and Factory Methods

### 32.4.2 Algorithmic Accessors

### 32.4.3 Content Mutators

### 32.4.4 Locator Initializers

## 32.5 Tracker Inner Class

## 32.6 Performance Analysis

## 32.7 Quick Method Reference

# Balanced Binary Search Trees

## 33.1 Methods

## 33.2 Quick Method Reference
# Red-Black Tree Data Structure

34.1 Internal Representation .......................................................... 514
34.2 Representation Properties ....................................................... 515
34.3 RBNode Inner Class ................................................................. 515
34.4 Methods ................................................................................. 517
   34.4.1 Constructors and Factory Methods ................................... 517
   34.4.2 Content Mutators ........................................................... 517
34.5 Performance Analysis ............................................................... 528
34.6 Quick Method Reference .......................................................... 528

# Splay Tree Data Structure

35.1 Internal Representation ............................................................ 531
35.2 Methods ................................................................................. 532
   35.2.1 Constructors .................................................................. 532
   35.2.2 Representation Mutators ................................................... 533
   35.2.3 Algorithmic Accessors ..................................................... 533
   35.2.4 Content Mutators ........................................................... 537
   35.2.5 Locator Initializers .......................................................... 541
35.3 Performance Analysis ............................................................... 541
35.4 Quick Method Reference .......................................................... 543

# B-Tree Data Structure

36.1 Internal Representation ............................................................ 546
36.2 Representation Properties ....................................................... 549
36.3 B-Tree Node Inner Class .......................................................... 549
   36.3.1 B-Tree Node Methods ..................................................... 550
   36.3.2 B-Tree Node Representation Mutators ............................. 551
   36.3.3 B-Tree Node Content Mutators ........................................ 555
36.4 B-Tree Methods ....................................................................... 557
   36.4.1 Constructors and Factory Methods ................................. 557
   36.4.2 Algorithmic Accessors ..................................................... 558
   36.4.3 Representation Mutators ................................................... 562
   36.4.4 Content Mutators ........................................................... 563
   36.4.5 Locator Initializers .......................................................... 567
36.5 Marker Inner Class ................................................................. 568
36.6 Performance Analysis ............................................................... 570
36.7 Quick Method Reference .......................................................... 573

# B+-Tree Data Structure

37.1 Case Study: A Web Search Engine ............................................. 576
37.2 Internal Representation ............................................................ 577
37.3 Representation Properties ....................................................... 578
37.4 Leaf Node Inner Class ............................................................... 579
37.5 B+-Tree Methods ..................................................................... 583
   37.5.1 Constructors and Factory Methods ................................. 584
   37.5.2 Representation Accessors ................................................... 584
   37.5.3 Algorithmic Accessors ..................................................... 585
   37.5.4 Content Mutators ........................................................... 588
37.6 Performance Analysis ............................................................... 589
37.7 Quick Method Reference .......................................................... 591
<table>
<thead>
<tr>
<th>Chapter</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>38</td>
<td>Skip List Data Structure</td>
<td>593</td>
</tr>
<tr>
<td>38.1</td>
<td>Internal Representation</td>
<td>593</td>
</tr>
<tr>
<td>38.2</td>
<td>Representation Properties</td>
<td>596</td>
</tr>
<tr>
<td>38.3</td>
<td>Tower Inner Class</td>
<td>597</td>
</tr>
<tr>
<td>38.4</td>
<td>Skip List Methods</td>
<td>598</td>
</tr>
<tr>
<td>38.4.1</td>
<td>Constructors</td>
<td>598</td>
</tr>
<tr>
<td>38.4.2</td>
<td>Algorithmic Accessors</td>
<td>600</td>
</tr>
<tr>
<td>38.4.3</td>
<td>Representation Mutators</td>
<td>604</td>
</tr>
<tr>
<td>38.4.4</td>
<td>Content Mutators</td>
<td>605</td>
</tr>
<tr>
<td>38.4.5</td>
<td>Locator Initializers</td>
<td>610</td>
</tr>
<tr>
<td>38.5</td>
<td>Tracker Inner Class</td>
<td>610</td>
</tr>
<tr>
<td>38.6</td>
<td>Performance Analysis</td>
<td>613</td>
</tr>
<tr>
<td>38.7</td>
<td>Quick Method Reference</td>
<td>617</td>
</tr>
<tr>
<td>39</td>
<td>Digitized Ordered Collection ADT</td>
<td>619</td>
</tr>
<tr>
<td>39.1</td>
<td>Case Study: Packet Routing</td>
<td>619</td>
</tr>
<tr>
<td>39.2</td>
<td>Case Study: Inverted Index for Text Retrieval</td>
<td>620</td>
</tr>
<tr>
<td>39.3</td>
<td>Digitized Ordered Collection Interface</td>
<td>621</td>
</tr>
<tr>
<td>39.4</td>
<td>Selecting a Data Structure</td>
<td>622</td>
</tr>
<tr>
<td>39.5</td>
<td>Terminology</td>
<td>623</td>
</tr>
<tr>
<td>39.6</td>
<td>Competing ADTs</td>
<td>624</td>
</tr>
<tr>
<td>39.7</td>
<td>Summary of Digitized Ordered Collection Data Structures</td>
<td>625</td>
</tr>
<tr>
<td>39.8</td>
<td>Trie Variations</td>
<td>630</td>
</tr>
<tr>
<td>39.9</td>
<td>Suffix Trees</td>
<td>630</td>
</tr>
<tr>
<td>39.10</td>
<td>Indexing Tries</td>
<td>631</td>
</tr>
<tr>
<td>39.11</td>
<td>Further Reading</td>
<td>633</td>
</tr>
<tr>
<td>40</td>
<td>Trie Node Types</td>
<td>635</td>
</tr>
<tr>
<td>40.1</td>
<td>Trie Node Interface</td>
<td>635</td>
</tr>
<tr>
<td>40.2</td>
<td>Abstract Trie Node Class</td>
<td>635</td>
</tr>
<tr>
<td>40.3</td>
<td>Trie Leaf Node Interface</td>
<td>636</td>
</tr>
<tr>
<td>40.4</td>
<td>Abstract Trie Leaf Node Class</td>
<td>637</td>
</tr>
<tr>
<td>41</td>
<td>Trie Data Structure</td>
<td>639</td>
</tr>
<tr>
<td>41.1</td>
<td>Internal Representation</td>
<td>639</td>
</tr>
<tr>
<td>41.2</td>
<td>Representation Properties</td>
<td>642</td>
</tr>
<tr>
<td>41.3</td>
<td>Internal Node Inner Class</td>
<td>642</td>
</tr>
<tr>
<td>41.4</td>
<td>Leaf Node Inner Class</td>
<td>644</td>
</tr>
<tr>
<td>41.5</td>
<td>Search Data Inner Class</td>
<td>644</td>
</tr>
<tr>
<td>41.6</td>
<td>FindResult Enumerated Type</td>
<td>644</td>
</tr>
<tr>
<td>41.7</td>
<td>Trie Methods</td>
<td>650</td>
</tr>
<tr>
<td>41.7.1</td>
<td>Constructors and Factory Methods</td>
<td>650</td>
</tr>
<tr>
<td>41.7.2</td>
<td>Algorithmic Accessors</td>
<td>651</td>
</tr>
<tr>
<td>41.7.3</td>
<td>Content Mutators</td>
<td>657</td>
</tr>
<tr>
<td>41.7.4</td>
<td>Locator Initializers</td>
<td>661</td>
</tr>
<tr>
<td>41.8</td>
<td>Trie Tracker Inner Class</td>
<td>662</td>
</tr>
<tr>
<td>41.9</td>
<td>Performance Analysis</td>
<td>664</td>
</tr>
<tr>
<td>41.10</td>
<td>Quick Method Reference</td>
<td>667</td>
</tr>
</tbody>
</table>
### 42 Compact Trie Data Structure

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>42.1</td>
<td>Internal Representation</td>
<td>671</td>
</tr>
<tr>
<td>42.2</td>
<td>Representation Properties</td>
<td>672</td>
</tr>
<tr>
<td>42.3</td>
<td>Compact Trie Methods</td>
<td>672</td>
</tr>
<tr>
<td>42.3.1</td>
<td>Constructors and Factory Methods</td>
<td>673</td>
</tr>
<tr>
<td>42.3.2</td>
<td>Algorithmic Accessors</td>
<td>673</td>
</tr>
<tr>
<td>42.3.3</td>
<td>Content Mutators</td>
<td>675</td>
</tr>
<tr>
<td>42.4</td>
<td>Performance Analysis</td>
<td>679</td>
</tr>
<tr>
<td>42.5</td>
<td>Quick Method Reference</td>
<td>680</td>
</tr>
</tbody>
</table>

### 43 Compressed Trie Data Structure

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>43.1</td>
<td>Internal Representation</td>
<td>683</td>
</tr>
<tr>
<td>43.2</td>
<td>Representation Properties</td>
<td>685</td>
</tr>
<tr>
<td>43.3</td>
<td>Compressed Trie Node Interface</td>
<td>686</td>
</tr>
<tr>
<td>43.4</td>
<td>Internal Node Inner Class</td>
<td>686</td>
</tr>
<tr>
<td>43.5</td>
<td>Compressed Trie Leaf Node Inner Class</td>
<td>687</td>
</tr>
<tr>
<td>43.6</td>
<td>Compressed Trie Search Data Inner Class</td>
<td>687</td>
</tr>
<tr>
<td>43.7</td>
<td>Compressed Trie Methods</td>
<td>688</td>
</tr>
<tr>
<td>43.7.1</td>
<td>Constructors and Factory Methods</td>
<td>688</td>
</tr>
<tr>
<td>43.7.2</td>
<td>Algorithmic Accessors</td>
<td>689</td>
</tr>
<tr>
<td>43.7.3</td>
<td>Content Mutators</td>
<td>690</td>
</tr>
<tr>
<td>43.8</td>
<td>Performance Analysis</td>
<td>693</td>
</tr>
<tr>
<td>43.9</td>
<td>Quick Method Reference</td>
<td>694</td>
</tr>
</tbody>
</table>

### 44 Patricia Trie Data Structure

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>44.1</td>
<td>Internal Representation</td>
<td>697</td>
</tr>
<tr>
<td>44.2</td>
<td>Representation Properties</td>
<td>699</td>
</tr>
<tr>
<td>44.3</td>
<td>Patricia Trie Node Inner Class</td>
<td>700</td>
</tr>
<tr>
<td>44.4</td>
<td>Patricia Trie Search Data Inner Class</td>
<td>702</td>
</tr>
<tr>
<td>44.5</td>
<td>Patricia Trie Methods</td>
<td>705</td>
</tr>
<tr>
<td>44.5.1</td>
<td>Constructors</td>
<td>705</td>
</tr>
<tr>
<td>44.5.2</td>
<td>Content Mutators</td>
<td>706</td>
</tr>
<tr>
<td>44.6</td>
<td>Performance Analysis</td>
<td>717</td>
</tr>
<tr>
<td>44.7</td>
<td>Quick Method Reference</td>
<td>717</td>
</tr>
</tbody>
</table>

### 45 Ternary Search Trie Data Structure

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>45.1</td>
<td>Internal Representation</td>
<td>719</td>
</tr>
<tr>
<td>45.2</td>
<td>Representation Properties</td>
<td>721</td>
</tr>
<tr>
<td>45.3</td>
<td>Ternary Search Trie Internal Node Inner Class</td>
<td>722</td>
</tr>
<tr>
<td>45.4</td>
<td>Ternary Search Trie Search Data Inner Class</td>
<td>722</td>
</tr>
<tr>
<td>45.5</td>
<td>Ternary Search Trie Methods</td>
<td>723</td>
</tr>
<tr>
<td>45.5.1</td>
<td>Constructors and Factory Methods</td>
<td>723</td>
</tr>
<tr>
<td>45.5.2</td>
<td>Algorithmic Accessors</td>
<td>724</td>
</tr>
<tr>
<td>45.6</td>
<td>Performance Analysis</td>
<td>725</td>
</tr>
<tr>
<td>45.7</td>
<td>Quick Method Reference</td>
<td>725</td>
</tr>
</tbody>
</table>

### 46 Spatial Collection ADT

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>46.1</td>
<td>Case Study: Collision Detection in Video Games</td>
<td>729</td>
</tr>
<tr>
<td>46.2</td>
<td>Interface</td>
<td>730</td>
</tr>
<tr>
<td>46.3</td>
<td>Competing ADTs</td>
<td>731</td>
</tr>
<tr>
<td>46.4</td>
<td>Summary of Spatial Collection Data Structures</td>
<td>732</td>
</tr>
<tr>
<td>46.5</td>
<td>Further Reading</td>
<td>732</td>
</tr>
</tbody>
</table>
47 KD-Tree Data Structure 735
47.1 Internal Representation .................................................. 736
47.2 Representation Properties ............................................... 739
47.3 Alternating Comparator .................................................. 739
47.4 KDNode Inner Class ......................................................... 742
47.5 KDTreerImpl Class ......................................................... 747
47.6 KD-Tree Methods ........................................................... 750
47.7 Performance Analysis ..................................................... 752
47.8 Quick Method Reference .................................................. 754

48 Quad Tree Data Structure 757
48.1 Internal Representation ................................................... 757
48.2 Representation Properties ............................................... 760
48.3 Partitioning a Two-Dimensional Space ................................. 761
48.4 QTNode Inner Class ....................................................... 762
48.5 Box Inner Class ............................................................... 765
48.6 Quad Tree Methods ........................................................ 767
  48.6.1 Constructors and Factory Methods ............................... 767
  48.6.2 Representation Accessors ............................................ 768
  48.6.3 Algorithmic Accessors .............................................. 768
  48.6.4 Content Mutators ..................................................... 772
  48.6.5 Locator Initializers .................................................. 781
48.7 Performance Analysis ..................................................... 782
48.8 Quick Method Reference .................................................. 782

49 Tagged Collection ADTs 785
49.1 Tagged Element ............................................................. 786
  49.1.1 Mutable Tagged Element ......................................... 787
  49.1.2 Tagged Element Comparator .................................... 787
  49.1.3 Tagged Element Digitizer ....................................... 788
  49.1.4 Tagged Element XY Comparator ................................ 788
49.2 Tagged Collection Interface ............................................ 789
49.3 Tracked Tagged Interface ................................................. 791
49.4 Competing ADTs ............................................................ 791
49.5 Selecting a Tagged Collection ADT ................................... 792
49.6 Tagged Collection Wrapper .............................................. 793
49.7 Mapping ADT ................................................................. 797
  49.7.1 Direct Addressing Mapping ....................................... 799
  49.7.2 Open Addressing Mapping ....................................... 799
  49.7.3 Separate Chaining Mapping ..................................... 801
49.8 Tagged Priority Queue ADT .............................................. 801
  49.8.1 Tagged Priority Queue Wrapper ................................ 803
  49.8.2 Tagged Binary Heap ................................................ 806
  49.8.3 Tagged Leftist Heap .............................................. 806
  49.8.4 Tagged Pairing Heap .............................................. 807
  49.8.5 Tagged Fibonacci Heap .......................................... 807
49.9 Tagged Ordered Collection ADT ...................................... 807
  49.9.1 Tagged Ordered Collection Wrapper ............................... 811
  49.9.2 Tagged Sorted Array ............................................. 812
  49.9.3 Tagged Binary Search Tree ..................................... 813
  49.9.4 Tagged Splay Tree .................................................. 813
54 Adjacency Matrix Data Structure
54.1 Internal Representation ........................................ 883
54.2 Representation Properties .................................. 886
54.3 Methods ......................................................... 887
  54.3.1 Constructors .............................................. 887
  54.3.2 Trivial Accessors ....................................... 888
  54.3.3 Algorithmic Accessors ................................. 888
  54.3.4 Representation Mutators ............................ 889
  54.3.5 Content Mutators .................................. 890
54.4 Edge Iterators ................................................ 893
54.5 Incident Edge Iterator Inner Class ......................... 893
54.6 Adjacency Matrix Class .................................. 896
54.7 Performance Analysis ..................................... 897
54.8 Quick Method Reference .................................. 898

55 Adjacency List Data Structure ................................. 901
55.1 Internal Representation .................................. 902
55.2 Representation Properties ................................ 905
55.3 Methods ......................................................... 906
  55.3.1 Constructors .............................................. 906
  55.3.2 Trivial Accessors ....................................... 906
  55.3.3 Algorithmic Accessors ................................. 906
  55.3.4 Content Mutators .................................. 908
55.4 Edge Iterators ................................................ 910
55.5 Edge Iterator Inner Class .................................. 910
55.6 Adjacency List Class ...................................... 912
55.7 Performance Analysis ..................................... 912
55.8 Quick Method Reference .................................. 915

56 Weighted Graph ADT ........................................... 917
56.1 Case Study: Airline Travel Agent (Revisited) .......... 917
56.2 Case Study: Image Segmentation ......................... 919
56.3 Terminology ................................................... 921
56.4 Weighted Edge Interface ................................. 922
56.5 Simple Weighted Edge Class .......................... 922
56.6 Weighted Graph Interface ................................ 923
56.7 Selecting a Data Structure ................................ 923
56.8 Weighted Adjacency Matrix Data Structure ............ 924
56.9 Weighted Adjacency List Data Structure ............... 924

57 Abstract Weighted Graph and Weighted Graph Algorithms 925
57.1 Greedy Tree Builder .......................................... 926
57.2 Dijkstra’s Single-Source Shortest Path Algorithm ...... 931
57.3 Prim’s Minimum Spanning Tree Algorithm ............... 935
57.4 Kruskal’s Minimum Spanning Tree Algorithm .......... 938
57.5 Bellman-Ford’s Single-Source Shortest Path Algorithm 941
57.6 Shortest Path Matrix .......................................... 943
57.7 Floyd-Warshall’s All-Pairs Shortest Path Algorithm .... 945
57.8 Edmonds-Karp Maximum Flow Algorithm ............... 948
57.9 Further Reading ................................................. 954
57.10 Quick Method Reference .................................. 955
Preface

This handbook of data structures and algorithms is designed as a comprehensive resource for computer science students and practitioners. The book is, quite literally, the product of a marriage of theory and practice. As an alternative to the survey approach taken by traditional data structures and algorithms textbooks, this book builds on a theoretical foundation to offer a top-down application-centered approach and a systematic treatment of data structure design and their practical implementation.

The book serves three major purposes: guidance, implementation, and insight. Charts, decision trees, and text provide guidance through the large body of material presented. Unlike a textbook, it is not necessary to read the entire book to fully benefit from its contents. Our intention is that readers with a specific problem will follow the provided guidance and organizational tools to quickly identify the most appropriate data structure or algorithm for their problem. For example, readers seeking a data structure for an application are first guided to a suitable abstract data type (ADT), and then to the most appropriate implementation of that ADT. Trade-offs between competing data types and implementations motivate each decision in the context of the problem at hand.

Traditional textbooks generally gloss over the different possible variations of a given data structure type. For example, a typical textbook has a chapter on “hashing” that treats all of the various uses of hashing uniformly as one idea (for example, hash-based implementations of a set or mapping). However, in reality, implementing them all in terms of a single ADT would lead to inefficiencies for alternate uses. Consider an application that requires a mapping from each word in a text document to the positions at which it occurs. One could use Java’s `HashMap` to associate each word with a linked list of line numbers. However, each insertion to associate a new word with a line number would require using `get` (to discover that the word is not yet in the mapping), and then `put` (that duplicates most of the work performed by `get`). In this book, we explicitly include the `BucketMapping` interface to provide efficient support for such an application. By explicitly introducing separate interfaces and ADTs for important variations in usage, differences can be highlighted and understood.

The book includes complete implementations for a wide variety of important data structures and algorithms. Unlike most textbooks that sweep details under the rug to simplify the implementation for “ease of explanation,” we have taken the approach of providing complete object-oriented implementations within an extensible class hierarchy. Yet we have not done so at the expense of clarity. Because of the completeness of implementation, chapters on some topics are longer than one might see in a textbook covering a similar topic. However, the organization of the chapters simplifies navigation, and the detailed implementations provide design insights useful to practitioners. Our implementations follow standard Java programming conventions.

Parts II and III of the book cover a large number of data structures and algorithms. We include many abstract data types not provided in the standard Java libraries, but for those data types that are also present in the Java Collections classes, we have tried to remain consistent with the Java interfaces and semantics wherever possible. However, we have diverged in places where our design goals differ. One important departure from the Java Collections is our separation of the iterator concept into two types: markers and trackers. Unlike Java’s provided iterator implementations, markers and trackers support concurrent modification of data structures. In addition, the introduction of a tracker, which maintains the location of a particular object even if its location changes within the structure, is crucial for efficient implementations of even some standard algorithms, such as the use of a priority queue to implement Dijkstra’s shortest path algorithm. However, care must be taken in
many data structure implementations to efficiently support tracking, and our presentation includes a discussion of such design choices.

We integrate the presentation of algorithms with the ADTs that support them. In many cases the algorithms are implemented in terms of the ADT interface and included in an abstract implementation of the ADT. The advantage of such an approach is that the algorithm (in both its presentation and instantiation) is decoupled from the particular ADT implementation.

As thorough as we have tried to be, it would not be possible to cover all possible variations of each data structure. Therefore, explanations of each implementation are designed not only to assist readers in understanding the given implementations of data structures and algorithms, but also to support readers in customizing implementations to suit the requirements of particular applications. Making such modifications while preserving correctness and efficiency requires an understanding of not only how the code operates, but why the code is correct and what aspects of the implementation contribute to its efficiency. To this end, we have provided clearly identified explanations of correctness properties for each implementation, as well as correctness highlights that explain how each method depends upon and preserves these properties. For data structures, these properties often relate to an abstraction function that captures, without undue formalism, how the organization of the data structure maps to the user view of the abstraction. This aids understanding at the intuitive level and serves as a foundation for the methodology we use to reason about program correctness. In this way, if readers choose to modify the provided code, they will be able to check that their change preserves the correctness of the implementation. Similarly, we provide a clearly identified section in which time complexity analysis is provided for each data structure and algorithm. Readers interested in modifying a particular method can look in that section to understand how that method (and consequently their proposed change) influences the overall performance of the implementation. Space complexity issues are also discussed.

The format of the book is designed for easy reference. In Parts II and III, each major data type and its implementations are presented in a sequence of chapters beginning with the semantics of that data type, and followed by each implementation. Within each chapter, standardized section headings help the reader quickly locate the required information. The stylized format is designed to help readers with different needs find what they want to read, as well as what they want to skip. A reader planning to use, but not modify, a data structure implementation may decide to read the introductory explanations and then skim through the implementation while omitting the correctness highlights, and finally read the time complexity analysis at the end of the chapter.

The case studies presented throughout the book provide further examples of how various data structures and algorithms presented in Parts II and III can be applied. They also exemplify the process by which those particular data structures and algorithms were selected for the application. One appendix provides a brief overview of the major features of the Java programming language, another appendix reviews asymptotic notation and complexity analysis, and a third appendix briefly discusses the design patterns we illustrate in this book. Source code for the data types, including interfaces, implementations, algorithms, and sample test cases, is included on the accompanying CD. Complete documentation, in Javadoc format, is also provided.

Note to instructors: An introductory data structures and algorithms course could begin with Part I, with an emphasis on selecting abstract data types and implementations appropriate for applications. Then, based on the interests of the instructor and students, a selected subset of the ADTs could be covered in detail. It is not necessary to present every data structure for each ADT, but instead the comparison tables can be used to highlight the differences, and then students can concentrate on one or two representative implementations of each. For courses with a more applied focus, homework and projects might concentrate on empirical comparisons of the provided implementations, modifications based on the optimizations suggested in the chapters, and projects based on the case studies. A more theoretical course might cover the complexity analysis material in the appendix early in the course, and focus more on theoretical analysis of the algorithms and correctness proofs. Online educational materials will be made available at http://goldman.cse.wustl.edu in December 2007.
Acknowledgments

Successful completion of a large project like this is possible only in the context of an understanding family, engaging colleagues, and a supportive community. First and foremost, we thank our children, Mark, Ben, and Julie, for relinquishing our time for many long hours while we worked on this book. We thank our parents, Robert and Marilyn, and Lester and Judy, for all that they have done to support the busy schedules of our children and ourselves. We thank our colleagues Jeremy Buhler for guidance on the case study on biosequence comparison, Chris Gill for helpful discussions about design patterns, Patrick Crowley and Fred Kuhns for guidance on the case study on Linux virtual memory organization, and Tao Ju and Robert Piess for helpful discussion on spatial collection data structures. We thank two exceptional undergraduate students, Ben Birnbaum and Troy Ruths, who carefully read an early draft of large sections of the manuscript and provided valuable advice that helped shape the final form of the book.

The philosophy and methodology of this book are largely products of the combined influence of our former research advisors and teachers, most notably Nancy Lynch, Ron Rivest, Barbara Liskov, Paris Kanellakis, Stan Zdonick, Andy Van Dam, Jeff Vitter, Tom Miller, and many other dedicated educators. We thank them deeply.

We thank Tom Cormen, Charles Leiserson, Ron Rivest, and Cliff Stein for graciously sharing their bibliography database, which was checked by Denise Sergent and Ayorkor Mills-Tettey.

We acknowledge the professional staff at CRC Press and Taylor & Francis Group, LLC, including Sunil Nair, Tom Skipp, Ari Silver, Jill Jurgensen, Katy Smith, Grace Bishop, Kevin Craig, Theresa Del Forn, and Clare Brannigan. We thank Shashi Kumar, Nishith Arora, George Shibu, and Sanjay Kandwal for their support with the formatting. This book was formatted using \TeX, \LaTeX, and pdflatex. Code segments were formatted with java2latex.

We thank the National Science Foundation for supporting our research during this period under grants IDM-0329241, CCR-9988314, CISE-EI-0305954, and CISE-CCLI-0618266.
Authors

Sally Goldman earned an Sc.B. from Brown University in 1984. She completed her Ph.D. under the supervision of Dr. Ronald Rivest at MIT in 1990. Since 1990, she has been a professor in the Department of Computer Science and Engineering at Washington University. She is the Edwin H. Murty Professor of Engineering and the Associate Chair of the Department of Computer Science and Engineering. She regularly teaches an introductory data structures and algorithms course as well as an advanced algorithms course. She has many journal and conference publications in computational learning theory, machine learning, image retrieval and on-line algorithms.

Kenneth Goldman earned an Sc.B. from Brown University in 1984. He completed his Ph.D. under the supervision of Dr. Nancy Lynch at MIT in 1990. Since 1990, he has been a professor in the Department of Computer Science and Engineering at Washington University. He regularly teaches introductory computer science and design methodology courses. His journal and conference publications are primarily in the area of distributed systems, distributed algorithms, and programming environments.
Part I

INTRODUCTION
References

100 Appendix C


[12] Arvind Arasu, Junghoo Cho, Hector Garcia-Molina,


[56] Robert W. Floyd. Assigning meaning to programs. Mathematical aspects of computer sci


[71] Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides. Design Patterns: Elements of Reusable Object-Oriented Software. Addison-Wesley, 1995.


[76] Michael T. Goodrich and Roberto Tamassia. Data Structures and Algorithms in Java. John N o t h i n g


1977.


[105] Barbara Liskov and John Guttag. Program Development in Java: Abstraction, Specification, and Object-Oriented


Addison-Wesley, 1990.


