CONTENTS

1 Introduction 1

1.1 Introduction and motivation 2

1.2 How the book is organized 3

1.3 Basic concepts and notation 5
- 1.3.1 Cells, fields, and pointers 5
- 1.3.2 Variables and assignments 7
- 1.3.3 Notation for algorithms 8
- 1.3.4 Mathematical notation 11

1.4 Mathematical background 12

2 Stacks and queues 19

2.1 Introduction and motivation 20

2.2 Notation 22

2.3 Quicksort—A sorting method using stacks 23

2.4 Representations of stacks and queues 28
- 2.4.1 Linked representations 28
- 2.4.2 Sequentially allocated representations 29
  - 2.4.2.1 Managing a single stack in a fixed region 30
  - 2.4.2.2 The coexistence of two stacks 30
  - 2.4.2.3 Representing queues using modular arithmetic 31

2.5 Garwick's technique for managing three or more stacks 32
- 2.5.1 Degradation of performance in Garwick allocation as memory becomes saturated 36
- 2.5.2 Choosing a reallocation policy 37
- 2.5.3 Other applications and observations 39

3 Trees

3.1 Introduction and motivation 44

3.2 Definitions and basic concepts 45

3.3 Two quantitative facts about binary trees 53
- 3.3.1 The relation between leaves and internal nodes in full binary trees 53
- 3.3.2 The number of binary trees with n nodes 54

3.4 Linked tree representations 58
3.4.1 Linked binary trees 58
3.4.2 Traversal orders for binary trees 59
3.4.3 Threaded trees 60
3.4.4 Linked blocks of nonuniform length 64

3.5 Representations in contiguous storage 65
3.5.1 Contiguous representation of complete binary trees 66
3.5.2 Sequential enumerations 67
3.5.2.1 Preorder 67
3.5.2.2 Postorder 72
3.5.2.3 Level order 73

3.6 Binary tree traversal algorithms 74
3.6.1 Stack traversals 75
3.6.2 Link inversion traversals 75
3.6.3 Robson traversal 77
3.6.4 Lindstrom scanning 80
3.6.5 Siklóssy traversal 82

3.7 Additional tree structures and their applications 84
3.7.1 Heaps, heapsort, and priority queues 84
3.7.2 Minimum spanning trees 92
3.7.2.1 Efficient union of disjoint sets 97
3.7.3 Binary search trees 99
3.7.3.1 Static binary search trees 105
3.7.3.2 Dynamic binary search trees 107
3.7.4 *B*-trees and 2-3 trees 118
3.7.5 Tries 122

3.8 Bibliographic notes 124

4 Tables 131

4.1 Introduction and motivation 132

4.2 Searching sequential tables 133
4.2.1 Most frequent to least frequent order 134
4.2.2 Ordered table search 136

4.3 Hash tables 141
4.3.1 Chaining 143
4.3.2 The use of buckets 144
4.3.3 Open addressing 145
4.3.3.1 Pseudo random probing 147
4.3.3.2 Quadratic residue search 147
4.3.3.3 Double hashing 148
4.3.4 Ordered hashing 149
4.3.5 Using pass bits 153
4.3.6 Binary tree hashing 155

4.3.7 Optimal preloading 157
4.3.8 Deletions 157
4.3.9 Rehashing *in situ* 157
4.3.10 Hashing functions 159
4.3.10.1 The division method 160
4.3.10.2 The multiplicative method 161
4.3.11 Choosing a hashing technique 163
4.4 Decision tables 164
4.4.1 Conversion to computer programs 172
4.4.1.1 Barnard's technique 172
4.4.1.2 Decision-tree techniques 173
4.5 Symbol tables 177
4.6 Bibliographic notes 181

5 Lists 185

5.1 Introduction and motivation 186
5.2 Definitions and basic concepts 188
5.3 List representation techniques 191
5.3.1 Sequentially allocated lists 191
5.3.2 Linked allocation for lists 192
5.3.2.1 One-way linked allocation 192
5.3.2.2 Symmetrically linked allocation 195
5.3.2.3 Circular lists 199
5.3.2.4 Threaded lists 201
5.3.2.5 Orthogonal lists 204
5.3.2.6 List headers 206
5.3.3 List representations in associative memory 206
5.3.3.1 Hash linking 209
5.4 Storage reclamation 210
5.4.1 Marking algorithms 212
5.4.1.1 Marking with pointer stacks 213
5.4.1.2 Marking with link-inversion and tag bits 214
5.4.1.3 Wegbreit marking with a bit stack 215
5.4.1.4 Lindstrom marking without stacks or tag bits 217
5.4.2 Collection 218
5.4.3 Incremental garbage collection 219
5.4.3.1 Reference counter methods 220
5.4.3.2 Minsky's algorithm for interleaved collection 225
5.4.3.3 Parallel garbage collection 226
5.4.4 Garbage collection in multi-level storage environments 230
5.5 Compacting and coexistence regimes 232

5.6 Copying cyclic lists in bounded workspace 235
5.7 Moving cyclic lists in bounded workspace 240
5.8 Optimal storage for garbage collection 241
5.9 Bibliographic notes 243

6 Multilinked structures 247

6.1 Introduction and motivation 248
6.2 Dynamic storage allocation techniques 249
6.2.1 Sequential fit methods 251
6.2.1.1 First fit 254
6.2.1.2 Best fit 255
6.2.1.3 Optimal fit 255
6.2.1.4 Worst fit 256
6.2.1.5 Variations in ring ordering 256
6.2.2 Buddy-system methods 257
6.2.3 Segregated-storage methods 263
6.2.3.1 Tree-indexed segregated storage 263
6.2.3.2 Fast-fit segregated storage 264
6.2.4 Comparison of the methods 267
6.3 Management algorithms for multilinked structures 274
6.3.1 Traversal 276
6.3.1.1 Schorr–Waite–Deutsch traversal 277
6.3.1.2 Thorelli traversal 277
6.3.2 Compacting 280
6.3.3 Copying 282
6.3.4 Equality testing 283
6.4 Bibliographic notes 284

7 Strings 287

7.1 Introduction and motivation 288
7.2 Representations of strings in contiguous storage 288
7.2.1 Fixed-length character codes 289
7.2.2 Packed strings 289
7.2.3 Unpacked strings 289
7.2.4 Radix-*N* representations 290
7.2.5 Karlgren's representation 290
7.2.6 Huffman codes 295
7.2.7 Composing Huffman codes with Karlgren's representation 302
7.2.8 Encoding blanks and compressing decks 302
7.3 String conversions 304
7.3.1 Encrypting 304

7.3.2 Fast code conversions 316
7.3.3 Matula's test for number conversion 317

7.4 Representations for variable-length strings 318
7.4.1 Representations using linked blocks of uniform size 319
7.4.2 Berztiss' representation 320
7.4.3 Madnick's comparison 322
7.4.4 Examples of string representations used in real systems 324

7.5 Examples of operations on strings 325
7.5.1 Macroexpansion 325
7.5.2 Fast string-pattern matching 329
7.5.2.1 The Knuth–Morris–Pratt algorithm 330
7.5.2.2 The Boyer-Moore algorithm 335
7.5.2.3 Harrison's substring test using hash signatures 339

7.6 Bibliographic notes 341

8 Arrays 347

8.1 Introduction and motivation 348

8.2 Basic concepts and notation 349

8.3 Storage allocation functions 350
8.3.1 Lexicographic storage allocation 351
8.3.2 Triangular and tetrahedral storage allocation functions 356
8.3.3 Cubic and diagonal shell functions 359
8.3.4 Sacrificing storage utilization efficiency to gain extensibility 361

8.4 Linked allocation 362
8.4.1 Techniques using vectors of pointers and linked lists 363
8.4.2 Techniques using orthogonal lists 369

8.5 Hashed allocation techniques 371

8.6 Comparison of representation techniques 373

8.7 Bibliographic notes 373

9 Files 377

9.1 Introduction and motivation 378

9.2 Properties of physical storage media 382
9.2.1 Tapes 382
9.2.2 Drums and disks 384

9.3 File organization techniques 388
9.3.1 Sequential files 388
9.3.1.1 Sorting, merging, and reporting 388
9.3.2 Direct access files 392
9.3.3 The indexed sequential access method 393
9.3.3.1 Handling overflows 396

9.3.3.2 Deletions and file maintenance 396
9.3.3.3 Random access by interpolation 398
9.3.4 Files indexed by multiway trees 398
9.3.4.1 B-tree indexes 400
9.3.5 Random access with hashing methods 400
9.3.6 Multi-attribute file techniques 402
9.3.6.1 Inverted files 403
9.4 Bibliographic notes 405

References and bibliography 409

Index 435