

University for providing technical assistance. Thanks also to Narain Gehani, AT&T Bell Laboratories, Tomasz Müldner, Arcadia University, and Ronald Prather, Trinity University, who reviewed early drafts of the manuscript. Special thanks go to Barbara and Art Friedman, our first publishers who nurtured the book through its early years.

Ellis Horowitz
Sartaj Sahni
Susan Anderson-Freed
June 2007

Contents

CHAPTER 1	BASIC CONCEPTS	1
1.1	Overview: System Life Cycle	1
1.2	Pointers and Dynamic Memory Allocation	4
1.2.1	Pointers	4
1.2.2	Dynamic Memory Allocation	5
1.2.3	Pointers Can Be Dangerous	7
1.3	Algorithm Specification	8
1.3.1	Introduction	8
1.3.2	Recursive Algorithms	14
1.4	Data Abstraction	18
1.5	Performance Analysis	22
1.5.1	Space Complexity	22
1.5.2	Time Complexity	25
1.5.3	Asymptotic Notation	33
1.5.4	Practical Complexities	41
1.6	Performance Measurement	44

1.6.1	Clocking	44
1.6.2	Generating Test Data	48
1.7	References And Selected Readings	50

CHAPTER 2 ARRAYS AND STRUCTURES 51

2.1	Arrays	51
2.1.1	The Abstract Data Type	51
2.1.2	Arrays in C	52
2.2	Dynamically Allocated Arrays	55
2.2.1	One-dimensional Arrays	55
2.2.2	Two-dimensional Arrays	56
2.3	Structures and Unions	59
2.3.1	Structures	59
2.3.2	Unions	62
2.3.3	Internal Representation of Structures	63
2.3.4	Self-referential Structures	63
2.4	Polynomials	64
2.4.1	The Abstract Data Type	64
2.4.2	Polynomial Representation	66
2.4.3	Polynomial Addition	69
2.5	Sparse Matrices	72
2.5.1	The Abstract Data Type	72
2.5.2	Sparse Matrix Representation	74
2.5.3	Transposing a Matrix	76
2.5.4	Matrix Multiplication	79
2.6	Representation of Multidimensional Arrays	85
2.7	Strings	87
2.7.1	The Abstract Data Type	87
2.7.2	Strings in C	87
2.7.3	Pattern Matching	90
2.8	References and Selected Readings	98
2.9	Additional Exercises	99

CHAPTER 3 STACKS AND QUEUES 107

3.1	Stacks	107
3.2	Stacks Using Dynamic Arrays	112
3.3	Queues	114
3.4	Circular Queues Using Dynamic Arrays	119
3.5	A Mazing Problem	120
3.6	Evaluation of Expressions	127
3.6.1	Expressions	127

3.6.2	Evaluating Postfix Expressions	129
3.6.3	Infix to Postfix	132
3.7	Multiple Stacks and Queues	139
3.7	Additional Exercises	142

CHAPTER 4 LINKED LISTS 145

4.1	Singly Linked Lists and Chains	145
4.2	Representing Chains in C	149
4.3	Linked Stacks and Queues	156
4.4	Polynomials	160
4.4.1	Polynomial Representation	160
4.4.2	Adding Polynomials	161
4.4.3	Erasing Polynomials	165
4.4.4	Circular List Representation of Polynomials	166
4.4.5	Summary	168
4.5	Additional List Operations	171
4.5.1	Operations for Chains	171
4.5.2	Operations for Circularly Linked Lists	172
4.6	Equivalence Classes	174
4.7	Sparse Matrices	178
4.7.1	Sparse Matrix Representation	178
4.7.2	Sparse Matrix Input	181
4.7.3	Sparse Matrix Output	184
4.7.4	Erasing a Sparse Matrix	185
4.8	Doubly Linked Lists	186

CHAPTER 5 TREES 191

5.1	Introduction	191
5.1.1	Terminology	191
5.1.2	Representation of Trees	194
5.2	Binary Trees	197
5.2.1	The Abstract Data Type	197
5.2.2	Properties of Binary Trees	199
5.2.3	Binary Tree Representations	202
5.3	Binary Tree Traversals	205
5.3.1	Inorder Traversal	206
5.3.2	Preorder Traversal	208
5.3.3	Postorder Traversal	208
5.3.4	Iterative Inorder Traversal	209
5.3.5	Level-Order Traversal	209
5.3.6	Traversal Without a Stack	210

5.4	Additional Binary Tree Operations	212
5.4.1	Copying Binary Trees	212
5.4.2	Testing Equality	212
5.4.3	The Satisfiability Problem	213
5.5	Threaded Binary Trees	216
5.5.1	Threads	216
5.5.2	Inorder Traversal of a Threaded Binary Tree	220
5.5.3	Inserting a Node into a Threaded Binary Tree	220
5.6	Heaps	222
5.6.1	Priority Queues	222
5.6.2	Definition of a Max Heap	224
5.6.3	Insertion into a Max Heap	225
5.6.4	Deletion from a Max Heap	228
5.7	Binary Search Trees	231
5.7.1	Definition	231
5.7.2	Searching a Binary Search Tree	232
5.7.3	Insertion into a Binary Search Tree	234
5.7.4	Deletion from a Binary Search Tree	235
5.7.5	Joining and Splitting Binary Search Trees	236
5.7.6	Height of a Binary Search Tree	237
5.8	Selection Trees	240
5.8.1	Introduction	240
5.8.2	Winner Trees	241
5.8.3	Loser Trees	243
5.9	Forests	244
5.9.1	Transforming a Forest into a Binary Tree	245
5.9.2	Forest Traversals	246
5.10	Representation of Disjoint Sets	247
5.10.1	Introduction	247
5.10.2	Union and Find Operations	248
5.10.3	Application to Equivalence Classes	256
5.11	Counting Binary Trees	259
5.11.1	Distinct Binary Trees	259
5.11.2	Stack Permutations	259
5.11.3	Matrix Multiplication	262
5.11.4	Number of Distinct Binary Trees	263
5.12	References and Selected Readings	264
CHAPTER 6	GRAPHS	265
6.1	The Graph Abstract Data Type	265
6.1.1	Introduction	265
6.1.2	Definitions	267

6.1.3	Graph Representations	272	
6.2	Elementary Graph Operations	279	
6.2.1	Depth First Search	279	
6.2.2	Breadth First Search	281	
6.2.3	Connected Components	283	
6.2.4	Spanning Trees	284	
6.2.5	Biconnected Components	286	
6.3	Minimum Cost Spanning Trees	292	
6.3.1	Kruskal's Algorithm	292	
6.3.2	Prim's Algorithm	296	
6.3.3	Sollin's Algorithm	297	
6.4	Shortest Paths and Transitive Closure	299	
6.4.1	Single Source/All Destination: Nonnegative Edge Costs	300	
6.4.2	Single Source/All Destination: General Weights	303	
6.4.3	All Pairs Shortest Paths	307	
6.4.4	Transitive Closure	310	
6.5	Activity Networks	315	
6.5.1	Activity-on-Vertex (AOV) Networks	315	
6.5.2	Activity-on-Edge (AOE) Networks	320	
6.6	References and Selected Readings	330	
6.7	Additional Exercises	331	

CHAPTER 7 SORTING 333

7.1	Motivation	333	
7.2	Insertion Sort	337	
7.3	Quick Sort	340	
7.4	How Fast Can We Sort?	343	
7.5	Merge Sort	346	
7.5.1	Merging	346	
7.5.2	Iterative Merge Sort	347	
7.5.3	Recursive Merge Sort	349	
7.6	Heap Sort	352	
7.7	Sorting on Several Keys	356	
7.8	List and Table Sorts	361	
7.9	Summary of Internal Sorting	370	
7.10	External Sorting	376	
7.10.1	Introduction	376	
7.10.2	k-way Merging	379	
7.10.3	Buffer Handling for Parallel Operation	381	
7.10.4	Run Generation	387	
7.10.5	Optimal Merging of Runs	388	
7.11	References and Selected Readings	394	

CHAPTER 8	HASHING	395
8.1	Introduction	395
8.2	Static Hashing	396
8.2.1	Hash Tables	396
8.2.2	Hash Functions	398
8.2.3	Overflow Handling	401
8.2.4	Theoretical Evaluation of Overflow Techniques	407
8.3	Dynamic Hashing	410
8.3.1	Motivation for Dynamic Hashing	410
8.3.2	Dynamic Hashing using Directories	411
8.3.3	Directoryless Dynamic Hashing	414
8.4	Bloom Filters	416
8.4.1	An Application -- Differential Files	16
8.4.2	Bloom Filter Design	418
8.5	References and selected Readings	420
CHAPTER 9	PRIORITY QUEUES	422
9.1	Single- and Double-Ended Priority Queues	422
9.2	Leftist Trees	424
9.2.1	Height-Biased Leftist Trees	424
9.2.2	Weight-Biased Leftist Trees	428
9.3	Binomial Heaps	433
9.3.1	Cost Amortization	433
9.3.2	Definition of Binomial Heaps	435
9.3.3	Insertion into a Binomial Heap	436
9.3.4	Melding Two Binomial Heaps	436
9.3.5	Deletion of Min Element	436
9.3.6	Analysis	439
9.4	Fibonacci Heaps	442
9.4.1	Definition	442
9.4.2	Deletion from an F-heap	442
9.4.3	Decrease Key	443
9.4.4	Cascading Cut	444
9.4.5	Analysis	445
9.4.6	Application to The Shortest Paths Problem	447
9.5	Pairing Heaps	450
9.5.1	Definition	450
9.5.2	Meld and Insert	451
9.5.3	Decrease Key	452
9.5.4	Delete Min	453
9.5.5	Arbitrary Delete	456
9.5.6	Implementation Considerations	457

9.5.7	Complexity	457
9.6	Symmetric Min-Max Heaps	458
9.6.1	Definition and Properties	458
9.6.2	SMMH Representation	460
9.6.3	Inserting into an SMMH	460
9.6.4	Deleting from an SMMH	465
9.7	Interval Heaps	469
9.7.1	Definition and Properties	469
9.7.2	Inserting into an Interval Heap	471
9.7.3	Deleting the Min Element	473
9.7.4	Initializing an Interval Heap	475
9.7.5	Complexity of Interval Heap Operations	475
9.7.6	The Complementary Range Search Problem	475
9.8	References and Selected Readings	478

CHAPTER 10 EFFICIENT BINARY SEARCH TREES 481

10.1	Optimal Binary Search Trees	481
10.2	AVL Trees	491
10.3	Red-Black Trees	506
10.3.1	Definition	506
10.3.2	Representation of a Red-Black Tree	508
10.3.3	Searching a Red-Black Tree	508
10.3.4	Inserting into a Red-Black Tree	508
10.3.5	Deletion from a Red-Black Tree	514
10.3.6	Joining Red-Black Trees	514
10.3.7	Splitting a Red-Black Tree	515
10.4	Splay Trees	518
10.4.1	Bottom-Up Splay Trees	518
10.4.2	Top-Down Splay Trees	524
10.5	References and Selected Readings	531

CHAPTER 11 MULTIWAY SEARCH TREES 532

11.1	m -way Search Trees	532
11.1.1	Definition and Properties	532
11.1.2	Searching an m -way Search Tree	534
11.2	B-Trees	535
11.2.1	Definition and Properties	535
11.2.2	Number of Elements in a B-Tree	536
11.2.3	Insertion into a B-tree	538
11.2.4	Deletion from a B-tree	542
11.3	B^+ -Trees	551

11.3.1	Definition	551
11.3.2	Searching a B ⁺ -Tree	552
11.3.3	Insertion into a B ⁺ -Tree	553
11.3.4	Deletion from a B ⁺ -Tree	554
11.4	References and Selected Readings	560

CHAPTER 12 DIGITAL SEARCH STRUCTURES 561

12.1	Digital Search Trees	561
12.1.1	Defintion	561
12.1.2	Search, Insert and Delete	562
12.2	Binary Tries and Patricia	563
12.2.1	Binary Tries	563
12.2.2	Compressed Binary Tries	564
12.2.3	Patricia	564
12.3	Multiway Tries	571
12.3.1	Definition	571
12.3.2	Searching a Trie	574
12.3.3	Sampling Strategies	574
12.3.4	Insertion into a Trie	577
12.3.5	Deletion from a Trie	577
12.3.6	Keys with Different Length	578
12.3.7	Height of a Trie	578
12.3.8	Space Required and Alternative Node Structures	579
12.3.9	Prefix Search and Applications	582
12.3.10	Compressed Tries	584
12.3.11	Compressed Tries with Skip Fields	587
12.3.12	Compressed Tries with Labeled Edges	588
12.3.13	Space Required by a Compressed Trie	592
12.4	Suffix Trees	593
12.4.1	Have You Seen this String?	593
12.4.2	The Suffix Tree Data Structure	594
12.4.3	Let's Find That Substring (Searching a Suffix Tree)	597
12.4.4	Other Nifty Things You Can Do with a Suffix Tree	598
12.5	Tries and Internet Packet Forwarding	600
12.5.1	IP Routing	600
12.5.2	1-Bit Tries	601
12.5.3	Fixed-Stride Tries	602
12.5.4	Variable-Stride Tries	605
12.6	References and Selected Readings	608

INDEX 610

