Algorithms on Strings, Trees, and Sequences

COMPUTER SCIENCE AND COMPUTATIONAL BIOLOGY

Dan Gusfield
University of California, Davis

CAMBRIDGE UNIVERSITY PRESS
Contents

Preface xiii

I Exact String Matching: The Fundamental String Problem 1

1 Exact Matching: Fundamental Preprocessing and First Algorithms 5

1.1 The naive method 5
1.2 The preprocessing approach 6
1.3 Fundamental preprocessing of the pattern 7
1.4 Fundamental preprocessing in linear time 8
1.5 The simplest linear-time exact matching algorithm 10
1.6 Exercises 11

2 Exact Matching: Classical Comparison-Based Methods 16

2.1 Introduction 16
2.2 The Boyer-Moore Algorithm 16
2.3 The Knuth-Morris-Pratt algorithm 23
2.4 Real-time string matching 27
2.5 Exercises 29

3 Exact Matching: A Deeper Look at Classical Methods 35

3.1 A Boyer-Moore variant with a “simple” linear time bound 35
3.2 Cole’s linear worst-case bound for Boyer–Moore 39
3.3 The original preprocessing for Knuth-Morris-Pratt 48
3.4 Exact matching with a set of patterns 52
3.5 Three applications of exact set matching 61
3.6 Regular expression pattern matching 65
3.7 Exercises 67

4 Seminumerical String Matching 70

4.1 Arithmetic versus comparison-based methods 70
4.2 The *Shift-And* method 70
4.3 The match-count problem and Fast Fourier Transform 73
4.4 Karp–Rabin fingerprint methods for exact match 77
4.5 Exercises 84
II Suffix Trees and Their Uses

5 Introduction to Suffix Trees

5.1 A short history
5.2 Basic definitions
5.3 A motivating example
5.4 A naive algorithm to build a suffix tree

6 Linear-Time Construction of Suffix Trees

6.1 Ukkonen’s linear-time suffix tree algorithm
6.2 Weiner’s linear-time suffix tree algorithm
6.3 McCreight’s suffix tree algorithm
6.4 Generalized suffix tree for a set of strings
6.5 Practical implementation issues
6.6 Exercises

7 First Applications of Suffix Trees

7.1 APL1: Exact string matching
7.2 APL2: Suffix trees and the exact set matching problem
7.3 APL3: The substring problem for a database of patterns
7.4 APL4: Longest common substring of two strings
7.5 APL5: Recognizing DNA contamination
7.6 APL6: Common substrings of more than two strings
7.7 APL7: Building a smaller directed graph for exact matching
7.8 APL8: A reverse role for suffix trees, and major space reduction
7.9 APL9: Space-efficient longest common substring algorithm
7.10 APL10: All-pairs suffix-prefix matching
7.11 Introduction to repetitive structures in molecular strings
7.12 APL11: Finding all maximal repetitive structures in linear time
7.13 APL12: Circular string linearization
7.14 APL13: Suffix arrays – more space reduction
7.15 APL14: Suffix trees in genome-scale projects
7.16 APL15: A Boyer–Moore approach to exact set matching
7.17 APL16: Ziv–Lempel data compression
7.18 APL17: Minimum length encoding of DNA
7.19 Additional applications
7.20 Exercises

8 Constant-Time Lowest Common Ancestor Retrieval

8.1 Introduction
8.2 The assumed machine model
8.3 Complete binary trees: a very simple case
8.4 How to solve \textit{lca} queries in \( B \)
8.5 First steps in mapping \( T \) to \( B \)
8.6 The mapping of \( T \) to \( B \)
8.7 The linear-time preprocessing of \( T \)
8.8 Answering an \textit{lca} query in constant time
8.9 The binary tree is only conceptual
## CONTENTS

8.10 For the purists: how to avoid bit-level operations 192
8.11 Exercises 193

9 More Applications of Suffix Trees 196

9.1 Longest common extension: a bridge to inexact matching 196
9.2 Finding all maximal palindromes in linear time 197
9.3 Exact matching with wild cards 199
9.4 The k-mismatch problem 200
9.5 Approximate palindromes and repeats 201
9.6 Faster methods for tandem repeats 202
9.7 A linear-time solution to the multiple common substring problem 205
9.8 Exercises 207

### III Inexact Matching, Sequence Alignment, Dynamic Programming 209

10 The Importance of (Sub)sequence Comparison in Molecular Biology 212

11 Core String Edits, Alignments, and Dynamic Programming 215

11.1 Introduction 215
11.2 The edit distance between two strings 215
11.3 Dynamic programming calculation of edit distance 217
11.4 Edit graphs 223
11.5 Weighted edit distance 224
11.6 String similarity 225
11.7 Local alignment: finding substrings of high similarity 230
11.8 Gaps 235
11.9 Exercises 245

12 Refining Core String Edits and Alignments 254

12.1 Computing alignments in only linear space 254
12.2 Faster algorithms when the number of differences are bounded 259
12.3 Exclusion methods: fast expected running time 270
12.4 Yet more suffix trees and more hybrid dynamic programming 279
12.5 A faster (combinatorial) algorithm for longest common subsequence 287
12.6 Convex gap weights 293
12.7 The Four-Russians speedup 302
12.8 Exercises 308

13 Extending the Core Problems 312

13.1 Parametric sequence alignment 312
13.2 Computing suboptimal alignments 321
13.3 Chaining diverse local alignments 325
13.4 Exercises 329

14 Multiple String Comparison – The Holy Grail 332

14.1 Why multiple string comparison? 332
14.2 Three “big-picture” biological uses for multiple string comparison 335
14.3 Family and superfamily representation 336
14.4 Multiple sequence comparison for structural inference 341
14.5 Introduction to computing multiple string alignments 342
14.6 Multiple alignment with the sum-of-pairs (SP) objective function 343
14.7 Multiple alignment with consensus objective functions 351
14.8 Multiple alignment to a (phylogenetic) tree 354
14.9 Comments on bounded-error approximations 358
14.10 Common multiple alignment methods 359
14.11 Exercises 366

15 Sequence Databases and Their Uses – The Mother Lode 370
15.1 Success stories of database search 370
15.2 The database industry 373
15.3 Algorithmic issues in database search 375
15.4 Real sequence database search 376
15.5 FASTA 377
15.6 BLAST 379
15.7 PAM: the first major amino acid substitution matrices 381
15.8 PROSITE 385
15.9 BLOCKS and BLOSUM 385
15.10 The BLOSUM substitution matrices 386
15.11 Additional considerations for database searching 387
15.12 Exercises 391

IV Currents, Cousins, and Cameos 393
16 Maps, Mapping, Sequencing, and Superstrings 395
16.1 A look at some DNA mapping and sequencing problems 395
16.2 Mapping and the genome project 395
16.3 Physical versus genetic maps 396
16.4 Physical mapping 398
16.5 Physical mapping: STS-content mapping and ordered clone libraries 398
16.6 Physical mapping: radiation-hybrid mapping 401
16.7 Physical mapping: fingerprinting for general map construction 406
16.8 Computing the tightest layout 407
16.9 Physical mapping: last comments 411
16.10 An introduction to map alignment 412
16.11 Large-scale sequencing and sequence assembly 415
16.12 Directed sequencing 415
16.13 Top-down, bottom-up sequencing: the picture using YACs 416
16.14 Shotgun DNA sequencing 420
16.15 Sequence assembly 420
16.16 Final comments on top-down, bottom-up sequencing 424
16.17 The shortest superstring problem 425
16.18 Sequencing by hybridization 437
16.19 Exercises 442
CONTENTS

17 Strings and Evolutionary Trees 447

17.1 Ultrametric trees and ultrametric distances 449
17.2 Additive-distance trees 456
17.3 Parsimony: character-based evolutionary reconstruction 458
17.4 The centrality of the ultrametric problem 466
17.5 Maximum parsimony, Steiner trees, and perfect phylogeny 470
17.6 Phylogenetic alignment, again 471
17.7 Connections between multiple alignment and tree construction 474
17.8 Exercises 475

18 Three Short Topics 480

18.1 Matching DNA to protein with frameshift errors 480
18.2 Gene prediction 482
18.3 Molecular computation: computing with (not about) DNA strings 485
18.4 Exercises 490

19 Models of Genome-Level Mutations 492

19.1 Introduction 492
19.2 Genome rearrangements with inversions 493
19.3 Signed inversions 498
19.4 Exercises 499

Epilogue – where next? 501
Bibliography 505
Glossary 524
Index 530