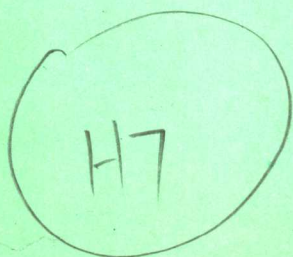


THEORY OF COMPUTATION

Derick Wood



THEORY OF COMPUTATION

Derick Wood

University of Waterloo



1817

HARPER & ROW, PUBLISHERS, New York
Cambridge, Philadelphia, San Francisco, Washington,
London, Mexico City, São Paulo, Singapore, Sydney

To Ma Li

Sponsoring Editor: John Willig
Project Editor: Thomas R. Farrell
Cover Design: 20/20 Services, Inc.
Text Art: RDL Artset, Ltd.
Production: Willie Lane
Printer and Binder: R. R. Donnelley & Sons Company

Theory of Computation

Copyright © 1987 by Harper & Row, Publishers, Inc.

All rights reserved. Printed in the United States of America. No part of this book may be used or reproduced in any manner whatsoever without written permission, except in the case of brief quotations embodied in critical articles and reviews. For information address Harper & Row, Publishers, Inc., 10 East 53d Street, New York, NY 10022.

Library of Congress Cataloging-in-Publication Data

Wood, Derick, 1940-
Theory of computation.

Bibliography: p.

Includes index.

1. Computational complexity. 2. Machine theory.
3. Electronic data processing—Mathematics. I. Title.
QA267.W66 1987 511 86-11976
ISBN 0-06-047208-1

86 87 88 89 9 8 7 6 5 4 3 2 1

Preface

This book is intended to be used as the basis of a one- or two-term introductory course in the theory of computation at the junior and senior levels. It concentrates on the fundamental models for languages and computation together with their properties. The models are finite automata, regular expressions, context-free grammars, pushdown automata, Turing machines, and transducers. The properties that are presented are family relationships (the relative power of the models), closure properties, decidability properties, and complexity properties. The book contains simple and elegant proofs of many results that are usually considered difficult. For example the proof that we give to show that every finite automaton has an equivalent regular expression is little known.

Throughout the text, algorithms are given in a PASCAL-like notation, since there is an emphasis on constructions and programming. In particular each of Chapters 0-7 has programming projects in addition to the more usual exercises. There is also an emphasis on practical applications throughout the text; for example, finite automata and pattern matching, regular expressions and text editing, extended context-free grammars and syntax diagrams, finite transducers and data compression. The text contains an abundance of worked examples to help the student understand the concepts as they are introduced.

Each chapter terminates with a summary of the material, its history, and a springboard; the last two items include references to the current literature. The springboard introduces a few topics for further investigation that can be used as projects for high-caliber students.

The text consists of four parts. Preliminaries are dealt with in Part I. Chapter 0 reviews sets, relations, graphs, functions, enumerability, and proof techniques. Chapter 1 introduces decision problems, formal languages, and their relationship.

Part II presents five models for languages and computation in Chapters 2-6: finite automata, regular expressions, context-free grammars, pushdown automata, and Turing machines. Chapters 2-6 are written to be order-independent to a large extent. In Chapter 7 these models are adapted for relations and functions. It presents finite transducers (from finite automata), translation grammars (from context-free grammars), pushdown transducers (from pushdown automata), and Turing transducers (from Turing machines). Turing transducers give rise to Turing-computable functions, the largest known class of computable functions. Finally, primitive recursive functions and recursive functions are introduced as an alternative method of specifying functions.

Part III covers the fundamental properties of the models. Chapter 8 explores the relative expressive power of the models, that is, family relationships. Chapter 9 investigates how instances of each model can be formed using a building block approach, that is, the closure properties of the associated language families. Decision problems for the models are studied in Chapter 10. Both basic decidable and undecidable problems are considered.

Part IV explores some further topics in a single chapter, Chapter 11. The topics are NP-completeness, general grammars, and parsing.

Lemmas and theorems are numbered sequentially within sections, in the same sequence. Examples are numbered sequentially within sections and continuations of the same example are indicated by using uppercase letters in addition. Exercises and programming projects are also numbered by section; programming projects are distinguished by an initial letter "P". Finally, an asterisk "*" and a double asterisk "**" indicate that a section or exercise is "difficult" and "more difficult", respectively; hence, the corresponding sections can be omitted on a first reading.

Many people have helped me to develop the present version of the text by giving their encouragement, their comments, or their methods of presenting material. These include Ray Bagley who reviewed the manuscript and made a large number of excellent suggestions; John Brzozowski, my colleague, who not only read and commented on an earlier version, but also by his teaching methods suggested many improvements; Patrick Dymond, who discussed the content of Chapters 6, 7, and 8; Nathan Friedman, a reviewer, who gave some excellent suggestions; Ed Gurari, another reviewer, who made many useful comments; Heikki Mannila, who was willing to participate in a discussion about a suggested approach at the drop of a hat; Areski Nait-Abdallah, who encouraged the germination of this text; Thomas Ottmann, who helped me in the development of Chapters 1 and 7; Jan van Leeuwen, who gave me some much needed positive response in the early days of the text; and Sheng Yu, who was willing to listen and comment on my mutterings. The many anonymous reviewers who waded through preliminary drafts have my sympathy as well as my thanks.

However, without the foundation of friendship and collaboration that I have enjoyed over many years with Hermann Maurer and Arto Salomaa I would not have been able to contemplate this book, let alone write it.

I am very grateful to Mary Chen, who typeset this book, for her concern, patience, and excellent work. She designed the layout, suggested type styles, and much more. I wish to thank Ian Allen, who was responsible for the macros that implemented her suggested layout. The excitement, continuous encouragement, and warmth of my editor, John Willig, my project editor, Thomas Farrell, and other staff members of Harper & Row, even when I fell behind schedule, was much appreciated.

Last, but not least, I tender thanks to the Lord of the Universe who enabled me to meet criticism and deadlines, to persevere, and to aim for excellence. Dei Gloria.

I am sure that errors remain to be found; however, they are mine and mine alone. I will be glad to hear of any errors that you find and any corrections or suggestions that you have.

Derick Wood

Course Outlines

This text contains much more than one term's worth of material; therefore, I give some suggested course outlines below. These reflect different emphases that an instructor may wish to give or that a syllabus requires. First, however, I wish to give some general guidelines to using this book.

The text has an unusual arrangement of material, in that a variety of models are presented in Part II and their properties are discussed mostly in Part III. This provides the basis of a course in which only models are emphasized; perhaps in a first course. It also allows the student to see similar results, concerning properties of models, side by side, making it easier for the student to compare and contrast proof techniques. One effect of this separation is that complete coverage of a standard topic such as finite automata and regular expressions will need not only material from Chapters 2 and 3, but also snippets from Chapters 8, 9, and 10 and also, possibly, from Chapter 7.

The text also contains an extensive review of discrete mathematics in Chapter 0. Experience has shown that students have little prior exposure to or understanding of graphs, relations, closure, and proof techniques. This will vary tremendously from university to university, therefore this is omitted from the course outlines given below. It is anticipated that students begin with Chapter 1, which introduces the basic questions of the theory of computation, the basics of language theory, and also sketches an important connection between languages and decision problems.

In a thirteen-week course at the University of Waterloo we have been able to cover finite automata, regular expressions, context-free grammars, Turing machines, closure properties, family relationships, and decidability properties. More recently we covered only finite automata, regular expressions, Turing machines, and their properties together with an introduction to context-free grammars. A sample thirteen-week schedule might be as follows:

Problems, decision problems, computation, and languages: 1.1-1.4

(includes a review of 0.4 and 0.5)

Finite automata and regular expressions:

2.1-2.3 (excluding 2.3.1), 2.6.1-2.6.2, 3.1, 3.2

8.1 (for nonregular languages)

9.1, 9.3 (for regular languages), 10.1.1

Turing machines:

6.1-6.5, 10.1.3, 10.2.3

Context-free grammars and pushdown automata:

4.1, 4.2 (excluding 4.2.5), 4.3, 5.1, 5.2

8.1 (for non-context-free languages)

9.1, 9.2, 9.3 (excluding *CF* substitution)

10.1.2, 10.2.2, 10.3.2

In a quartermester system a restricted version of this coverage could be given. However, it is probably better to concentrate in depth on either finite automata, regular expressions, and Turing machines, or finite automata, regular expressions, context-free grammars, and pushdown automata. It is important to realize that the second approach does not allow for undecidability results to be presented.

If two quartermesters are available, then the first could be devoted to finite automata, regular expressions, Turing machines, and undecidability. The second could then cover context-free grammars, pushdown automata, transductions, and their properties.

Contents

Preface	xiii
Course Outlines	xvii
PART I INTRODUCTION	1
Chapter 0 PRELIMINARIES	
0.1 Sequences, Sets, and Tuples	3
0.1.1 Sets and Their Specification	3
0.1.2 Set Operations	6
0.1.3 Multisets	8
0.1.4 Sequences	8
0.1.5 Tuples	9
0.2 Directed Graphs	9
0.2.1 Digraphs	10
0.2.2 Multiset Digraphs and Trees	14
0.3 Functions, Relations, and Closure	17
0.3.1 Functions	17
0.3.2 Big-O Notation	19
0.3.3 Relations	20
0.3.4 Closure	25
0.4 Cardinality, Countability, and Enumerability	27
0.5 Proof Methods and Techniques	28
0.5.1 Direct Proof	29
0.5.2 Proof by Contradiction	30
0.5.3 Proof by Induction	31
0.5.4 The Pigeonhole Principle	34
0.5.5 Counting	35

0.5.6 Diagonalization	36
0.6 Additional Remarks	38
0.6.1 Summary	38
0.6.2 Further Reading	38
Exercises	38
Programming Projects	49

Chapter 1 LANGUAGES AND COMPUTATION

1.1 Programming Problems and Computation	50
1.2 Alphabets, Words, and Languages	63
1.2.1 Alphabets and Words	64
1.2.2 Languages and Operations	68
1.2.3 Defining Languages	71
1.2.4 Definition Summaries	76
1.3 Languages, Problems, and Computation	77
1.4 Additional Remarks	80
1.4.1 Summary	80
1.4.2 History	80
1.5 Springboard	81
1.5.1 The λ -calculus	81
1.5.2 Markov Algorithms	81
1.5.3 Post Systems and Production Systems	82
1.5.4 Recursive Functions	83
1.5.5 Formal Semantics	84
1.5.6 Program Correctness	85
1.5.7 Program Construction	86
Exercises	86
Programming Projects	90

PART II MODELS 93

Chapter 2 FINITE AUTOMATA

2.1 States, State Diagrams, and Transitions	95
2.2 Deterministic Finite Automata	98
2.3 Nondeterministic Finite Automata	111
2.4 Minimization and Simplification	127
2.5 DFAs and Tries	133
2.6 λ -FAs and Lazy FAs	135
2.6.1 λ -FAs and λ -Transitions	135
2.6.2 Lazy Finite Automata	140
2.7 Additional Remarks	142
2.7.1 Summary	142
2.7.2 History	143
2.8 Springboard	143
2.8.1 The Syntactic Monoid	143

2.8.2 Life and Cellular Automata	144
2.8.3 Communication Protocols	144
2.8.4 Security	144
2.8.5 Tree Automata	145
2.8.6 Avoiding Spaghetti Programming	145
Exercises	147
Programming Projects	153

Chapter 3 REGULAR EXPRESSIONS

3.1 Motivation and Definition	155
3.2 Regular Expressions into Finite Automata	161
3.3 Extended Finite Automata into Regular Expressions	166
3.4 Extended Regular Expressions	175
3.5 Additional Remarks	179
3.5.1 Summary	179
3.5.2 History	179
3.6 Springboard	180
3.6.1 A Differential Calculus	180
3.6.2 Descriptive Complexity	180
3.6.3 VLSI	181
3.6.4 Path Expressions	181
Exercises	182
Programming Projects	185

Chapter 4 CONTEXT-FREE GRAMMARS

4.1 Basics of Context-Free Grammars	187
4.2 Simplifications	211
4.2.1 Redundant Symbols	212
4.2.2 Empty Productions	215
4.2.3 Unit Productions	220
4.2.4 Binary Form and Chomsky Normal Form	221
4.2.5 Greibach Normal Form and Two-Standard Normal Form	224
4.3 Linear and Regular Grammars	229
4.4 Extended Context-Free Grammars	232
4.5 Additional Remarks	236
4.5.1 Summary	236
4.5.2 History	237
4.6 Springboard	237
4.6.1 Recognition and Parsing	237
4.6.2 Ambiguity	238
4.6.3 W -grammars	238
4.6.4 Simplifications and Efficiency	239
4.6.5 Context-free Expressions	239
4.6.6 EOL Grammars and Languages	239
4.6.7 ALGOL-Like Languages	240

Exercises	240
Programming Projects	246

Chapter 5 PUSHDOWN AUTOMATA

5.1 Deterministic Pushdown Automata	248
5.2 Nondeterministic Pushdown Automata	259
5.3* Counter Automata	270
5.4* Two-Way Deterministic Pushdown Automata	271
5.5 Additional Remarks	273
5.5.1 Summary	273
5.5.2 History	274
5.6 Springboard	274
5.6.1 Data Structure Automata	274
5.6.2 Preset Pushdown Automata	274
5.6.3 Multihead and Multipushdown Automata	274
5.6.4 Finite-Buffer PDAs and Lazy PDAs	274
5.6.5 Small PDAs	275
Exercises	276
Programming Projects	279

Chapter 6 TURING MACHINES

6.1 The Turing Machine	280
6.2 Turing Machine Programming	295
6.3* Simplifications	301
6.4 Extensions	311
6.4.1 Two-Way Infinite Tape Turing Machines	312
6.4.2 Nondeterministic Turing Machines	316
6.4.3 Multihead Turing Machines	318
6.4.4 Multitape Turing Machines	320
6.5 Universal Turing Machines	320
6.6 Resource-Bounded Turing Machines	326
6.7 Additional Remarks	331
6.7.1 Summary	331
6.7.2 History	331
6.8 Springboard	331
6.8.1 The Busy Beaver Game	331
6.8.2 The Turing Micros	332
6.8.3 Computable Functions	333
6.8.4 NP-Completeness	333
6.8.5 Simulation Efficiency	333
6.8.6 Recursively Enumerable Languages	334
6.8.7 Context-Sensitive Languages	334
Exercises	334
Programming Projects	337

Chapter 7 FUNCTIONS, RELATIONS, AND TRANSLATIONS

7.1 Introduction	338
7.2 Finite Transducers	339
7.3 Translation Grammars	348
7.4 Pushdown Transducers	352
7.5 Turing Machines and Computable Functions	355
7.6 Recursive Functions	359
7.7 Additional Remarks	364
7.7.1 Summary	364
7.7.2 History	364
7.8 Springboard	364
7.8.1 Lexical Analysis	364
7.8.2 Syntax-Directed Translations	364
7.8.3 File Compaction	365
7.8.4 Translation Expressions	365
7.8.5 String Similarity	365
7.8.6 Recursive Function Theory	365
Exercises	366
Programming Projects	368

PART III PROPERTIES 371**Chapter 8 FAMILY RELATIONSHIPS**

8.1 Finite, Regular, and Context-Free Languages	373
8.1.1 Introduction	373
8.1.2 Regular Languages	375
8.1.3* Unary Regular Languages	379
8.2 Context-Free and Tractable Languages	382
8.2.1 Deterministic Context-Free Languages	382
8.2.2* Tractable Languages	383
8.2.3 Context-Free Pumping Lemma	387
8.2.4* Parikh's Theorem	394
8.3* Tractable and Recursive Languages	397
8.4 Recursive and <i>DTM</i> Languages	403
8.5 Additional Remarks	406
8.5.1 Summary	406
8.5.2 History	406
8.6 Springboard	407
8.6.1 Hierarchy Results	407
8.6.2 Pumping Lemmas	407
8.6.3 Nonlanguages and Closure Properties	408
Exercises	408

Chapter 9 CLOSURE PROPERTIES

9.1 Boolean and Reversal Operations	412
---	-----

9.2	Mappings	420
9.2.1	Morphism and Substitution	420
9.2.2*	Inverse Morphism and Finite Transduction	429
9.3	Catenation, Quotient, and Star	434
9.4	Composition	435
9.5	Cylinders, Cones, and <i>AFLs</i>	438
9.6	Additional Remarks	440
9.6.1	Summary	440
9.6.2	History	440
9.7	Springboard	441
9.7.1	Closure Properties	441
9.7.2	Cones	441
	Exercises	441

Chapter 10 DECISION PROBLEMS

10.1	Decidability and Membership	446
10.1.1	Introduction	446
10.1.2	Finite Automata	447
10.1.3	Context-Free Grammars	447
10.1.4	Deterministic Turing Machines	448
10.2	Emptiness and Finiteness	450
10.2.1	Finite Automata	450
10.2.2	Context-Free Grammars	451
10.2.3	Deterministic Turing Machines	452
10.3	Containment, Equivalence, and Intersection	454
10.3.1	Finite Automata	454
10.3.2	Context-Free Grammars	455
10.4	Context-Free Ambiguity	462
10.5	Additional Remarks	462
10.5.1	Summary	462
10.5.2	History	462
10.6	Springboard	463
10.6.1	Post's Correspondence Problem	463
	Exercises	464

PART IV ONWARD 469

Chapter 11 FURTHER TOPICS

11.1	<i>NP</i> -Complete Problems	472
11.1.1	The Ordering $\propto$	472
11.1.2	An <i>NP</i> -Complete Problem	475
11.1.3	Satisfiability	476
11.1.4	Traveling Salesman	481
11.1.5	Subset Sum	488
11.1.6	Hierarchical Grammars	492

11.2 Rewriting Systems and Church's Thesis	497
11.2.1 Phrase Structure Grammars	497
11.2.2 Interactive Lindenmayer Grammars	500
11.3 Parsing	504
11.3.1 General <i>CFG</i> Parsing	504
11.3.2 Deterministic Top-Down Parsing	507
11.3.3 Deterministic Bottom-Up Parsing	515
11.4 Additional Remarks	523
11.4.1 Summary	523
11.4.2 History	523
11.5 Springboard	524
11.5.1 Dealing with <i>NP</i> -Completeness	524
11.5.2 <i>LALR</i> Grammars	525
11.5.3 Other Kinds of Completeness	525
Exercises	526
BIBLIOGRAPHY	531
INDEX	547

Part I

INTRODUCTION

