

edition to improve its usability. Please e-mail comments and corrections to me at sipserbook@math.mit.edu. All original suggestions will be acknowledged in the final edition. A world wide web site is being developed that will contain a list of corrections as well as other good stuff. The location for this site is

<http://www-math.mit.edu/~sipser/book.html> .

I for one am looking forward to examining the book in more detail for a rather self-serving reason: the table of contents maps almost to a \mathcal{T} the way I like to teach the course, and the preliminary edition of the book, at least, reaches exactly the point at which I run out of time during a semester.

Logout

From Bozeman, where in waist-deep snow high on Tollman Ridge near Lone Mountain in the Madison Range of the Rockies, the elusive elk keep just one patch of timber ahead and out of sight, while the Madison River valley shimmers, white and quiet, far below in the light blue haze of a cold January dawn...

Rocky Ross	
Computer Science Department	INTERNET: ross@cs.montana.edu
Montana State University	WWW: http://www.cs.montana.edu/~ross
Bozeman, MT 59717	PHONE: (406) 994-4804

Introduction to the Theory of Computation Michael Sipser

PWS Publishing Company, 1996
ISBN: 0-534-95250-X

Preface (Abridged)

This book is intended as an upper level undergraduate or introductory graduate text in computer science theory. It contains a mathematical treatment of the subject, designed around theorems and proofs. I have made some effort to accommodate students with little prior experience in proving theorems, though more experienced students will have an easier time.

My primary goal in presenting the material has been to make it clear and interesting. In so doing, I have emphasized intuition and the big picture in the subject over some lower level details.

For example, even though I present the method of proof by induction in Chapter 0 along with other mathematical preliminaries, it doesn't play an important role subsequently. Generally I do not present the usual induction proofs of the correctness of various constructions concerning automata. If presented clearly, these constructions convince and do not need further argument. An induction may confuse rather than enlighten because induction itself is a rather sophisticated technique that many find mysterious. Belaboring the obvious with an induction risks teaching students that mathematical proof is a formal manipulation instead of teaching them what is and what is not a cogent argument.

A second example occurs in Parts II and III, where I describe algorithms in prose instead of pseudocode. I don't spend much time programming Turing machines (or any other formal model). I believe that students today come with a programming background and find the Church-Turing thesis to be self-evident. Hence I don't present lengthly simulations of one model by another to establish their equivalence.

Besides giving extra intuition and suppressing some details, I give what might be called a classical presentation of the subject material. Most theorists will find the choice of material, terminology, and order of presentation consistent with that of other widely used textbooks. I have introduced original terminology in only a few places, when I found the standard terminology particularly obscure or confusing. For example I introduce the term *mapping reducibility* instead of *many-one reducibility*.

Practice through solving problems is essential to learning any mathematical subject. In this book, the problems are organized into two main categories called *Exercises* and *Problems*. The Exercises review definitions and concepts. The Problems require some ingenuity. Problems marked with a star are harder. I have tried to make both the Exercises and Problems interesting challenges.

Preliminary Edition

This edition of *Introduction to the Theory of Computation* is a preliminary edition of the final book, due out in late 1996. My intent in producing this edition is to test the treatment of most of the material with a widespread audience so that adjustments can be made in the final edition to improve its usability. Please e-mail comments and corrections to me at sipserbook@math.mit.edu. All original suggestions will be acknowledged in the final edition. A world wide web site is being developed that will contain a list of corrections as well as other good stuff. The location for this site is <http://www-math.mit.edu/~sipser/book.html>.

My current plans for the final edition include several additional chapters on complexity theory: Chapter 8 on space complexity; Chapter 9 on provable intractability; and Chapter 10 on advanced topics, including approximation algorithms, alternation, cryptography, and parallel computing. Chapter 6, currently on the recursion theorem, will be expanded to include other advanced topics in computability theory, including Turing reducibility, Kolmogorov complexity, and the decidability and undecidability of some logical theories.

Table of Contents

Preface

0. Introduction 1

Automata, Computability, and Complexity 1 / Complexity theory 2 / Computability theory 2 / Automata theory 3 / Mathematical Notions and Terminology 3 / Sets 3 / Sequences and tuples 5 / Functions and relations 7 / Graphs 9 / Strings and languages 13 / Summary of mathematical terms 14 / Definitions, Theorems, and Proofs 15 / Finding proofs 15 / Types of Proof 19 / Proof by construction 19 / Proof by contradiction 19 / Proof by induction 21 / Exercises and Problems 23

Part One: Automata and Languages 29

1. Regular Languages 29

Finite Automata 29 / Formal definition of a finite automaton 32 / Examples of finite automata 34 / Formal definition of computation 37 / Designing finite automata 38 / The

regular operations 41 / Nondeterminism 44 / Formal definition of a nondeterministic finite automaton 49 / Equivalence of NFAs and DFAs 50 / Regular Expressions 58 / Formal definition of a regular expression 60 / Equivalence with finite automata 61 / Nonregular Languages 71 / The pumping lemma for regular languages 71 / Exercises and Problems 75

2. Context-Free Languages 81

Context-free Grammars 81 / Formal definition of a context-free grammar 83 / Examples of context-free grammars 84 / Ambiguity 85 / Chomsky normal form 87 / Pushdown Automata 89 / Formal definition of a pushdown automaton 91 / Examples of pushdown automata 92 / Equivalence with context-free grammars 95 / Non-context-free Languages 102 / The pumping lemma for context-free languages 102 / Exercises and Problems 106

Part Two: Computability Theory 111

3. The Church-Turing Thesis 111

Turing Machines 111 / Formal definition of a Turing machine 114 / Examples of Turing machines 116 / Variants of Turing Machines 118 / Multitape Turing machines 118 / Non-deterministic Turing machines 120 / Enumerators 122 / Equivalence with other models 124 / The Definition of Algorithm 125 / Hilbert's problems 125 / Terminology for describing Turing machines 127 / Exercises and Problems 130

4. Decidability 133

Decidable Languages 134 / Decidable problems concerning regular languages 134 / Decidable problems concerning context-free languages 138 / The Halting Problem 140 / The diagonalization method 142 / The halting problem is undecidable 147 / A nonenumerable language 149 / Exercises and Problems 150

5. Reducibility 153

Undecidable Problems from Language Theory 154 / Reductions via computation histories 158 / A Simple Undecidable Problem 164 / Mapping Reducibility 171 / Computable functions 172 / Formal definition of mapping reducibility 172 / Exercises and Problems 176

6. The Recursion Theorem 179

Self-reference 180 / Terminology for the recursion theorem 183 / Applications 184 / Exercises and Problems 185 /

Part Three: Complexity Theory 189

7. Time Complexity 189

Time Complexity 189 / Measuring Complexity 189 / Big-O and small-o notation 190 / Analyzing algorithms 192 / Complexity relationships among models 195 / The Class P 198 / Polynomial time 198 / Examples of problems in P 200 / The Class NP 205 / Examples of problems in NP 208 / The P versus NP question 210 / NP-completeness 211 / Polynomial time reducibility 212 / Definition of NP-completeness 215 / The Cook-Levin Theorem 215 / Examples of NP-complete Problems 222 / Exercises and Problems 230