

An Introduction to Computational Complexity

Draft version June 12, 2008

Michael Soltys

© 2007 by Michael Soltys

Preface

The intended audience for this book are graduate students in computer science and mathematics who want to quickly familiarize themselves with computational complexity. As such, this book aims more at depth than breadth. There are many excellent comprehensive guides to computational complexity (in particular classics such as [Pap94] and [Sip06]). Here, on the other hand, the reader will find major results of complexity presented with a minimum of background.

The highlights are as follows. In chapter 1 we present two applications of the crossing sequences method: we show that a single tape Turing machine requires $\Omega(n^2)$ many steps to decide the language of palindromes (which can also be seen as an application of rudimentary Kolmogorov complexity), and that languages decidable with $o(\log \log n)$ space are in fact regular.

In chapter 2 we use the self-reducibility of SAT to show that tally sets cannot be NP-complete unless $P = NP$, and we prove the Karp-Lipton theorem (if $NP \subseteq P/\text{poly}$, then PH collapses to its second level). We also introduce the so called “padding technique,” and use it to prove Ladner’s theorem (if $P \neq NP$, then there are languages in $NP - P$ which are not NP-complete).

In chapter 3 we deal with space complexity, and present Savitch’s theorem as well as the inductive counting technique, and prove the Immerman-Szelepcsényi theorem (i.e., NL is closed under complementation). Chapter 3 ends with interactive proof systems and a proof of $IP = PSPACE$.

In chapter 4 the Hierarchy Theorems are presented, and we prove the somewhat technical result stating that “ $P^A \neq NP^A$ with probability 1” (with respect to a random oracle A). The Polytime Hierarchy is examined in detail, and we prove some properties of Alternating Turing machines. We end chapter 4 with an application of the so called “Bennett’s trick”: Nepomnjaschij’s theorem.

In chapter 5 we have Shannon’s (easy) lower bound for circuits, and we introduce the probabilistic method in order to give two different proofs

showing that $\text{PARITY} \notin \text{AC}^0$. We end with the definition of computation with advice.

In chapter 6 we prove Haken's lower bound for resolution refutations of the pigeonhole principle. We also give an alternative lower bound for resolution based on the idea of interpolation and Razborov's lower bound for monotone circuits computing CLIQUE .

The last chapter, chapter 7, starts with three examples of randomized algorithms, we introduce the notion of amplification, and present Sipser's theorem ($\text{BPP} \subseteq \Sigma_2^p$) and finish with Toda's theorem ($\text{PH} \subseteq \text{P}^{\text{PP}}$).

In the Appendix (chapter 8) we collect miscellanea: we have a section with different NP-complete problems; some number theory, mostly background for the Rabin-Miller algorithm (algorithm 7.1.2 on page 106); a section on the RSA public key encryption; the Isolation Lemma (used in the proof of Toda's theorem); and a section on Berkowitz's algorithm (an NC^2 algorithm for computing the determinant of a matrix).

Many standard results, not mentioned in the above outline, are sprinkled throughout the text. On the other hand, there are many complexity classes that do not make an appearance; a comprehensive list of all complexity classes can be found in the Complexity Zoo¹. The tapestry of complexity classes is truly overwhelming; *"But the man who sets himself the task of singling out the thread of order from the tapestry will by the decision alone have taken charge of the world"*².

This book contains one semester worth of material, that is, it is intended for a ten to twelve week course, that meets for two to three one hour lectures a week.

Notation

The symbol Σ will have several (but all standard) meanings. It will denote a finite alphabet of symbols, such as the binary alphabet $\Sigma = \{0, 1\}$, and Σ^* denotes the set of all finite strings over the alphabet Σ (i.e., Kleene's³ star of Σ). We shall use Σ_i to denote alternations; for example, Σ_i^p denotes the set of relations expressible by a polytime predicate with i alternations of quantifiers in front of it, where the first quantifier is \exists ; see section 4.3.

¹wiki.caltech.edu/wiki/Complexity_Zoo

²Cormac McCarthy, "Blood Meridian Or the Evening Redness in the West", Vintage Books, first Vintage edition, May 1992, pg. 199.

³After the mathematician Stephen Cole Kleene. Σ^i often denotes the set of all strings of length i over the alphabet Σ , so Σ^* can be defined as $\bigcup_{i=0}^{\infty} \Sigma^i$, where $\Sigma^0 = \{\varepsilon\}$, and where ε is the (unique) string of length 0.

$\vec{x} = x_1, x_2, \dots, x_n$, i.e., a vector of variables, while \bar{x} denotes the same as $\neg x$, i.e., a negation of a Boolean variable. We use $[n]$ to denote the subset of natural number (\mathbb{N}) consisting of $\{1, 2, \dots, n\}$. We use “:=” to denote a definition.

We use $A \subseteq B$ to denote that A is a subset of B , and possibly $A = B$. We use $A \subset B$ to denote that A is a *proper* subset of B , i.e., $A \subset B$ iff $A \subseteq B$ and $A \neq B$. Also, $A - B$ will denote set difference: all those elements in A which are not in B . We use $\wedge, \vee, \neg, \rightarrow, \leftrightarrow$ to denote Boolean and, or, not, implies, equivalent, respectively. Sometime we use the more expressive names, **And**, **Or**, **Not**, **Xor**. We use **T**, **F** (as well as $1, 0$) to denote the Boolean constants true and false, respectively, and we use $\Gamma \models \alpha$ to denote that the set of formulas Γ logically implies the formula α .

We use the standard *big-Oh notation*, $g(n) \in O(f(n))$ if there exist constants c, n_0 such that for all $n \geq n_0$, $g(n) \leq cf(n)$, and the *little-oh notation*, $g(n) \in o(f(n))$, which denotes that $\lim_{n \rightarrow \infty} g(n)/f(n) = 0$. We also say that $g(n) \in \Omega(f(n))$ if there exist constants c, n_0 such that for all $n \geq n_0$, $g(n) \geq cf(n)$. Finally, we say that $g(n) \in \Theta(f(n))$ if it is the case that both $g(n) \in O(f(n))$ and $g(n) \in \Omega(f(n))$.

We shall need a little bit of number theory: let \mathbb{Z}_n be $\{0, 1, 2, \dots, (n-1)\}$, the set of numbers modulo n , and let \mathbb{Z}_n^* denote the subset of \mathbb{Z}_n consisting of those numbers co-prime with n . We shall write $x =_n y$ and $x = y \pmod{n}$ to denote that $n \mid (x - y)$, i.e., that n divides $(x - y)$. Let $(n)_b$ be the binary representation of the integer n ; for example $(5)_b = 101$

Our logarithm function, $\log x$, is assumed to be in base 2, i.e., $\log x := \log_2 x$. Using standard notation, we let $\ln x := \log_e x$.

Sans-serif fonts will be used to denote complexity classes, for example NP, and SMALL CAPS FONTS will be used to denote languages (i.e., problems), for example MINFORMULA.

Acknowledgments

First of all, it will be clear to anyone reading this book that it relies heavily on many other books and papers (each chapter ends with a Notes section that points the reader to the appropriate sources). I am greatly indebted to all those authors.

This book came about as the result of the author teaching a graduate complexity course at McMaster University (Hamilton, Canada) in the years 2002–2006, and also during a visit at the Algorithmics Research Group of the Jagiellonian University (Kraków, Poland) in the summer 2007, while

giving a *Ulam Seminar* at the University of Colorado at Boulder in the Fall 2007, and in February 2008, during the IX Escuela de Verano de Ciencias Informáticas, Universidad Nacional de Río Cuarto, Argentina.

I am grateful to all the students who took this course, and in particular, I am deeply grateful to the following proof readers: Lech Duraj, Grzegorz Gutowski, Grzegorz Herman (for reading the *entire* book), Jan Jeżabek, Ryan Lortie, Bartosz Walczak, and Craig Wilson.

I am very grateful to Prof. Jan Mycielski for comments and corrections, especially for a very careful proofreading of section 7.1.2, and to Prof. Hugo Ryckeboer.

Contents

Preface	iii
1 Turing machines	1
1.1 Definition	1
1.2 Basic properties	4
1.3 Crossing sequences	5
1.3.1 A lower bound for palindromes	5
1.3.2 Little space is no space at all	8
1.4 Answers to selected exercises	10
1.5 Notes	11
2 P and NP	13
2.1 Introduction	13
2.2 Reductions and completeness	16
2.3 Self-reducibility of satisfiability	20
2.4 Padding argument	22
2.5 Answers to selected exercises	24
2.6 Notes	26
3 Space	27
3.1 Basic definitions and results	27
3.2 The inductive counting technique	31
3.3 Interactive Proof Systems	33
3.3.1 $IP \subseteq PSPACE$	34
3.3.2 $PSPACE \subseteq IP$	35
3.4 Answers to selected exercises	38
3.5 Notes	39

4	Diagonalization and Relativization	41
4.1	Hierarchy Theorems	41
4.2	Oracles and Relativization	46
4.2.1	A random oracle separating P and NP	48
4.3	Polynomial time hierarchy	51
4.4	More on Alternating TMs	55
4.5	Bennett's Trick	56
4.6	Answers to selected exercises	57
4.7	Notes	58
5	Circuits	59
5.1	Basic results and definitions	59
5.2	Shannon's lower bound	64
5.3	The probabilistic method	67
5.3.1	First proof of Parity not in AC^0	67
5.3.2	Second proof of Parity not in AC^0	75
5.4	Computation with advice	79
5.5	Answers to selected exercises	80
5.6	Notes	81
6	Proof Systems	83
6.1	Introduction	83
6.2	Resolution	85
6.2.1	DPLL and DP	87
6.3	A lower bound for resolution	92
6.4	Automatizability and interpolation	95
6.5	Answers to selected exercises	101
6.6	Notes	101
7	Randomized Classes	103
7.1	Three examples of randomized algorithms	103
7.1.1	Perfect matching	103
7.1.2	Primality testing	106
7.1.3	Pattern matching	109
7.2	Basic Randomized Classes	110
7.3	The Chernoff Bound and Amplification	113
7.4	More on BPP	115
7.5	Toda's Theorem	118
7.6	Answers to selected exercises	123
7.7	Notes	125

8 Appendix	127
8.1 NP-complete problems	127
8.2 A little number theory	131
8.3 RSA	134
8.4 The Isolation Lemma	136
8.5 Berkowitz's algorithm	138
8.5.1 Clow Sequences	142
8.6 Answers to selected exercises	149
8.7 Notes	149
 Bibliography	 150
 Index	 154