
Computational Complexity: A Modern Approach

Draft of a book: Dated February 2006
Comments welcome!

Sanjeev Arora
Princeton University
arora@cs.princeton.edu

**Not to be reproduced or distributed without the author's
permission**

*Some chapters are more finished than others. References and attributions
are very preliminary and I apologize in advance for any omissions (but
hope you will nevertheless point them out to me).*

About this book

Computational complexity theory has developed rapidly in the past three decades. The list of surprising and fundamental results proved in the 1990s alone could fill a book: these include new probabilistic definitions of classical complexity classes ($IP = PSPACE$ and the PCP Theorems) and their implications for the field of approximation algorithms; Shor's algorithm to factor integers using a quantum computer; an understanding of why current approaches to the famous P versus NP will not be successful; a theory of derandomization and pseudorandomness based upon computational hardness; and beautiful constructions of pseudorandom objects such as extractors and expanders.

This book is a graduate level text that aims to describe such recent achievements of complexity theory in the context of the classical results. Though it is arguably self-contained, it will be most appreciated by those who have some familiarity with the theory of computation (e.g., from Sipser's book *Theory of Computation*). It is divided into three parts: (Part A) **Basic complexity classes:** This part covers broadly the same ground — but more quickly— as Papadimitriou's text from the early 1990s. (Part B) **Lowerbounds for concrete computational models:** This concerns lowerbounds on resources required to solve algorithmic tasks on concrete models such as circuits, decision trees, etc. Such models may seem at first sight very different from the Turing machine used in Part A, but looking deeper one finds interesting interconnections. (Part C) **Advanced topics:** This constitutes the latter half of the book and is largely devoted to developments since the late 1980s.

Any successful text in this area must simultaneously cater to many audiences, and the book is carefully designed with that goal. The book chapters can largely be read in isolation except for the 150 pages in Part A. The appendix outlines mathematical ideas that may be useful for following the text. The use of examples and exercises throughout the book is intended to make it suitable both for selfstudy and as a text.

While writing the book I am keeping the following classes of readers in mind:

- *Physicists, mathematicians, and other scientists.* This group has become increasingly interested in computational complexity theory, especially because of high-profile results such as Shor's algorithm and the recent deterministic test for primality. This intellectually sophisticated group will generally be interested in the 150-page synopsis of basic complexity theory in Part A. Progressing on to Parts B and C, they can read individual chapters and find almost everything they need to understand current research.
- *Computer scientists (e.g., algorithms designers) who do not work in complexity theory per se.* They may use the book for selfstudy or even to teach a graduate course as a way of learning the latest results. Such a course would probably include many topics from Part A and then a sprinkling from the rest of the book. I plan to include —either in the book or on my website— detailed course-plans they can follow (e.g., if they plan to teach an advanced result such as the PCP Theorem, they may wish to prepare the students by teaching other results first).
- *All those —professors or students— who do research in complexity theory or plan to do so.* They will largely know Part A and will use the book for Parts B and C. They would typically use Part C in a seminar or reading course. The coverage of advanced topics here is detailed enough to make this possible. For example, teaching all results related to the PCP Theorems in this book takes me at least 12 lectures of 1.5 hours each (though I will include sample teaching plans for introducing this topic in fewer lectures) and the chapter on Derandomization and Extractors takes me another 4 to 5 lectures.

Contents

1	The computational model —and why it doesn't matter	15
1.1	Encodings and Languages: Some conventions	16
1.2	The Turing machine	17
1.2.1	Nondeterministic TMs	19
1.3	Turing machine: a universal computational model	20
1.3.1	The Church-Turing Thesis and its strong form.	21
I	Basic Complexity Classes	23
2	P, NP and all that	25
2.0.2	The philosophical importance of P	27
2.0.3	The philosophical importance of NP	27
2.1	Reducibility and NP-completeness	28
2.1.1	The Cook-Levin Theorem: Computation is Local . . .	30
2.1.2	The web of reductions	32
2.2	coNP	32
2.3	EXPTIME and NEXPTIME	33
2.4	Decision versus search	34
2.5	List of some problems in P and NP	35
2.6	More thoughts about P, NP, and all that	36
2.6.1	Why 3SAT?	36
2.6.2	What if $P = NP$?	36
2.6.3	In praise of reductions	37
3	Space complexity	41
3.1	PSPACE completeness	44
3.1.1	The essence of PSPACE: optimum strategies for game- playing.	47

3.2	NL completeness	48
3.3	Two suprising algorithms	51
4	Diagonalization	57
4.1	Time Hierarchy Theorem	58
4.2	Space Hierarchy Theorem	59
4.3	Nondeterministic Time Hierarchy Theorem	59
4.4	Can diagonalization resolve P vs NP?	61
5	Alternating TM's and Polynomial Hierarchy	67
5.0.1	Complete problems for levels of PH	70
5.1	Alternating Turing machines	71
5.1.1	Unlimited number of alternations?	72
6	Circuits	75
6.0.2	Turing machines that take advice	78
6.1	Karp-Lipton Theorem	79
6.2	Circuit lowerbounds	81
6.3	Finer gradations among circuit classes	82
6.3.1	Parallel computation and NC	82
6.3.2	P-completeness	84
6.4	Circuits of exponential size	85
7	Randomized Computation	89
7.1	RP, coRP, BPP, ZPP	91
7.1.1	The robustness of our definitions	92
7.1.2	Alternative definitions	93
7.1.3	Error reduction and recycling of random bits	94
7.2	Some examples of PTMs	95
7.3	$BPP \subseteq P/poly$	99
7.4	BPP is in PH	99
7.5	State of our knowledge about BPP	101
7.5.1	Complete problems for BPP?	101
7.5.2	Does BPTIME have a hierarchy theorem?	101
7.6	Randomized reductions	102
7.7	Randomized space-bounded computation	103
7.7.1	Universal traversal sequences	104

8	Complexity classes having to do with counting	107
8.1	#P and #P-completeness	109
8.1.1	Permanent and Valiant's Theorem	110
8.1.2	Approximate solutions to #P problems	115
8.2	Today's Theorem: $\text{PH} \subseteq \text{P}^{\#\text{SAT}}$	116
8.2.1	Step 1: Randomized reduction from PH to $\oplus\text{SAT}$	117
8.2.2	Step 2: Making the reduction deterministic	120
8.3	Open Problems	122
9	Cryptography	125
9.1	Hard-on-average problems and one-way functions	127
9.1.1	Discussion of the definition of one-way function	129
9.1.2	Random self-reducibility	130
9.2	What is a random-enough string?	131
9.2.1	Blum-Micali and Yao definitions	132
9.2.2	Equivalence of the two definitions	134
9.3	One-way functions and pseudorandom number generators	136
9.3.1	Goldreich-Levin hardcore bit	137
9.3.2	Pseudorandom number generation	140
9.4	Applications	141
9.4.1	Pseudorandom functions	141
9.4.2	Private-key encryption: definition of security	142
9.4.3	Derandomization	143
9.4.4	Tossing coins over the phone and bit commitment	144
9.4.5	Secure multiparty computations	145
9.4.6	Lowerbounds for machine learning	145
9.5	Recent developments	146
10	Interactive proofs	149
10.1	Warmup: Interactive proofs with a deterministic verifier	150
10.2	IP	151
10.2.1	Zero Knowledge and Cryptography	153
10.2.2	Public coins and AM	154
10.2.3	Some properties of IP and AM	156
10.3	IP = PSPACE	157
10.3.1	Arithmetization	158
10.3.2	Interactive protocol for $\#\text{SAT}_L$	159
10.3.3	Protocol for TQBF: proof of Theorem 10.4	161
10.4	Interactive proof for the Permanent	162
10.5	The power of the prover	165

10.6	Program Checking	166
10.6.1	Languages that have checkers	167
10.7	Multiprover interactive proofs (MIP)	169
II	Lowerbounds for Concrete Computational Models	173
11	Yao’s MinMax Lemma	177
12	Decision Trees	179
12.1	Certificate Complexity	182
12.2	Randomized Decision Trees	184
12.3	Lowerbounds on Randomized Complexity	185
12.4	Some techniques for decision tree lowerbounds	186
12.5	Comparison trees and sorting lowerbounds	188
12.6	Yao’s MinMax Lemma	188
13	Communication Complexity	191
13.1	Definition	192
13.2	Lowerbound methods	193
13.2.1	Fooling set	193
13.2.2	The tiling lowerbound	194
13.2.3	Rank lowerbound	196
13.2.4	Discrepancy	196
	A technique for upperbounding the discrepancy	198
13.2.5	Comparison of the lowerbound methods	199
13.3	Multiparty communication complexity	200
	Discrepancy-based lowerbound	201
13.4	Probabilistic Communication Complexity	202
13.5	Overview of other communication models	203
13.6	Applications of communication complexity	204
14	Circuit lowerbounds	207
14.1	AC^0 and Håstad Switching Lemma	207
14.1.1	Proof of Håstad Switching Lemma	211
14.2	Circuits With “Counters”: ACC	213
14.3	Lowerbounds for monotone circuits	217
14.4	Circuit complexity: The frontier	222
14.4.1	Circuit lowerbounds using diagonalization	222
14.4.2	Status of ACC versus P	223

14.4.3	Linear Circuits With Logarithmic Depth	224
14.4.4	Branching Programs	225
14.5	Approaches using communication complexity	226
14.5.1	Connection to ACC^0 Circuits	226
14.5.2	Connection to Linear Size Logarithmic Depth Circuits	227
14.5.3	Connection to branching programs	228
14.5.4	Karchmer-Wigderson communication games and depth lowerbounds	228
15	Algebraic computation models	233
15.1	Algebraic Computation Trees	234
15.2	Algebraic circuits	239
15.3	The Blum-Shub-Smale Model	240
15.3.1	Complexity Classes over the Complex Numbers	241
15.3.2	Hilbert's Nullstellensatz	242
15.3.3	Decidability Questions: Mandelbrot Set	243
III	Advanced topics	245
16	Average Case Complexity: Levin's Theory	247
16.1	Distributional Problems	248
16.1.1	Formalizations of "real-life distributions."	249
16.2	DistNP and its complete problems	250
16.2.1	Polynomial-Time on Average	250
16.2.2	Reductions	252
16.2.3	Proofs using the simpler definitions	255
16.3	Existence of Complete Problems	257
16.4	Polynomial-Time Samplability	258
17	Amplification of Hardness	261
17.1	Hardness and how it is amplified with Yao's Lemma	262
17.2	Proof of Impagliazzo's Lemma	264
17.3	Hard functions from worst-case assumptions	267
18	Derandomization and Extractors	271
18.1	Pseudorandom Generators and Derandomization	273
18.1.1	Hardness and Derandomization	275
18.2	Proof of Theorem 18.2: Nisan-Wigderson Construction	276
18.2.1	The Pseudorandom Generator	277

18.3	Weak Random Sources and Extractors	280
18.3.1	An extractor based upon Nisan-Wigderson	283
18.3.2	Extractors based upon hash functions	287
18.3.3	Extractors based upon random walks on expanders	287
18.4	Applications of Extractors	288
18.4.1	Graph constructions	288
18.4.2	Running randomized algorithms using weak random sources	288
18.4.3	Recycling random bits	290
18.4.4	Pseudorandom generators for spacebounded computation	291
18.5	Derandomization requires circuit lowerbounds	296
18.5.1	Proof of Lemma 18.22: Easy witness method	299
18.6	A logspace algorithm for undirected connectivity	301
18.7	Technical Details	305
19	The PCP Theorems	311
19.1	Probabilistic checking of membership proofs	312
19.2	Relations between PCP and standard complexity classes	314
19.2.1	Two PCP Theorems	315
19.3	Approximation algorithms and hardness of approximation	316
19.4	Proof of PCP Theorem	318
19.4.1	ALG-SAT \in sPCP($\log n, 1, \log^{O(1)} n$)	321
	Low degree test	322
	Reconstruction procedure	324
	Proof of Theorem 19.6	326
19.4.2	ALG-SAT \in PCP($\log^c n, 1$): Constant-bit verifier	327
	Linear function reconstruction procedure	328
	Verifying the concatenation property of encodings	329
	Converting a circuit to a system of quadratic equations	329
	Verifying satisfiability of quadratic system by an encoded assignment	330
	Proof of Theorem (19.9)	331
19.4.3	NP \subseteq PCP($\log n, 1$): sketch	331
19.5	Low Degree Test: Correctness	332
19.5.1	The case $m = 3$	332
19.5.2	The Bootstrapping	335
19.6	The Fourier transform technique	339
19.6.1	The discrete Fourier transform	340
	The connection to PCPs: High level view	341

19.6.2	Analysis of the linearity test over $GF(2)$	342
19.6.3	Coordinate functions, Long code and its testing	343
19.6.4	Proof of Håstad's 3-bit PCP Theorem	345
19.7	Other PCP Theorems: A survey	351
19.8	More advanced techniques for constructing PCPs	353
19.9	Technical details: ALG-SAT is NP-complete	353
20	Hardness of approximating NP-hard optimization problems	361
20.1	MAX-SNP problems	361
20.2	CLIQUE	361
20.3	COLORING	361
20.4	SET-COVER	361
21	Quantum Computation	363
21.1	Quantum physics	363
21.2	Quantum superpositions	365
21.3	Classical computation using reversible gates	367
21.4	Quantum gates	368
21.4.1	Universal quantum gates	370
21.5	BQP	371
21.6	Factoring integers using a quantum computer	372
21.6.1	Phase estimation: the first few bits	374
21.6.2	Better phase estimation using structure of U	377
21.6.3	Uniform superpositions of eigenvectors of U	378
21.6.4	Uniform superposition suffices	379
21.7	Quantum Computing: a Tour de horizon	379
21.8	Technical details	379
22	Logic in complexity theory	383
22.1	Logical definitions of complexity classes	384
22.1.1	Fagin's definition of NP	384
22.1.2	MAX-SNP	386
22.2	Proof complexity as an approach to NP versus coNP	386
22.2.1	Resolution	386
22.2.2	Frege Systems	386
22.2.3	Polynomial calculus	386
22.3	Is $P \neq NP$ unprovable?	386

23 The polynomial method	387
23.1 Robust interpolation of univariate polynomials	387
23.1.1 Berlekamp-Welch Procedure: $\rho < 1/2$	388
23.1.2 Sudan's list-decoding: $\rho > 1/2$	388
23.2 Self-correction for multivariate polynomials	390
23.3 Local Self-correction for multivariate polynomials	390
23.4 Application1: From worst-case to average-case hardness . . .	390
23.4.1 An approach to hardness amplification using polynomial self-correction	391
24 Why are circuit lowerbounds so difficult?	393
24.1 Formal Complexity Measures	393
24.2 Natural Properties	395
24.3 Limitations of Natural Proofs	398
24.4 My personal view	400
A Appendix	403
A.1 Probability theory	403
A.1.1 The averaging argument	403
A.1.2 Deviation upperbounds	404
A.1.3 Sampling methods	406
A.1.4 The random subsum principle	407
A.2 Polynomials	407
A.2.1 Representing data by multivariate polynomials	409
A.3 Eigenvalues and expanders	410