

# Contents

Introduction	1
<b>Part 1. Basic model theory and computability</b>	<b>3</b>
Chapter 1. Propositional logic	5
1.1. Propositional language	5
1.1.1. Construction of formulas	5
1.1.2. Proof by induction	7
1.1.3. Decomposition of a formula	7
1.2. Semantics	9
1.2.1. Tautologies. Equivalent formulas	10
1.2.2. Logical consequence	11
1.2.3. Value of a formula and substitution	11
1.2.4. Complete systems of connectives	15
1.3. Normal forms	15
1.3.1. Disjunctive and conjunctive normal forms	15
1.3.2. Functions associated to formulas	16
1.3.3. Transformation methods	17
1.3.4. Clausal form	19
1.3.5. OBDD: Ordered Binary Decision Diagrams	20
1.4. Exercises	23
Chapter 2. Deduction systems	25
2.1. Examples of tableaux	25
2.2. Tableaux method	27
2.2.1. Trees	28
2.2.2. Construction of tableaux	29
2.2.3. Development and closure	30
2.3. Completeness theorem	31
2.3.1. Provable formulas	31
2.3.2. Soundness	31
2.3.3. Completeness	32
2.4. Natural deduction	33
2.5. Compactness theorem	36
2.6. Exercises	38
Chapter 3. First-order logic	41

3.1. First-order languages	41
3.1.1. Construction of terms	42
3.1.2. Construction of formulas	43
3.1.3. Free and bound variables	44
3.2. Semantics	45
3.2.1. Structures and languages	45
3.2.2. Structures and satisfaction of formulas	46
3.2.3. Valid and equivalent formulas	48
3.2.4. Substitution	50
3.3. Prenex formulas and Skolem forms	51
3.3.1. Prenex formulas and normal forms	51
3.3.2. Skolem forms	53
3.4. Exercises	55
Chapter 4. Completeness of first order logic	57
4.1. Natural deduction system	57
4.2. Soundness	59
4.3. Completeness	59
4.4. Examples of theories	61
4.4.1. Successor function	62
4.4.2. Discrete successor function	62
4.4.3. Graphs	63
4.4.4. Robinson's arithmetic	64
4.5. Exercises	65
Chapter 5. Models of computation	67
5.1. General model of computation	67
5.2. Turing machines	68
5.2.1. Deterministic Turing machines	68
5.2.2. Non-deterministic Turing machines	71
5.2.3. Machines with multiple tapes	74
5.2.4. Turing machine with oracles	76
5.2.5. Alternating machines	76
5.3. Other sequential models of computation	78
5.3.1. Finite automata	78
5.3.2. RAM: Random Access Machines	80
5.4. Exercises	82
Chapter 6. Recursion and decidability	83
6.1. Recursive functions	83
6.1.1. Primitive recursive functions	83
6.1.2. Construction of primitive recursive functions	84
6.1.3. Coding sequences	86
6.1.4. Non-primitive recursion	87
6.1.5. Recursive functions	88
6.1.6. Church's thesis	89

6.2. Recursion and computation	90
6.2.1. Recursive functions are Turing computable	90
6.2.2. Turing computable functions are recursive	91
6.2.3. Universal Turing machine	93
6.3. Recursive systems	95
6.3.1. Equivalence with the recursive functions	98
6.3.2. Implementation of recursive functions	98
6.4. Recursion and decidability	101
6.4.1. Recursive and recursively enumerable sets	101
6.4.2. The halting problem	103
6.4.3. Reductions	104
6.4.4. The s-m-n and Rice's theorems	105
6.5. Exercises	107
<b>Chapter 7. Incompleteness of Peano arithmetic</b>	<b>109</b>
7.1. Arithmetic theory and representable functions	109
7.1.1. Peano arithmetic	109
7.1.2. Arithmetical proofs	110
7.1.3. Non-standard models of arithmetic	111
7.1.4. Representable functions	111
7.2. Coding arithmetical proofs	114
7.2.1. Coding terms and formulas	115
7.2.2. Coding sequences of formulas	116
7.2.3. Arithmetical theorems are recursively enumerable	116
7.3. Undecidability and incompleteness	117
7.4. The validity problem over finite structures	118
7.5. The arithmetical hierarchy	120
7.6. Exercises	127
<b>Part 2. Descriptive Complexity</b>	<b>129</b>
<b>Chapter 8. Complexity: time and space</b>	<b>131</b>
8.1. Complexity classes	132
8.1.1. Complexity of decision problems	132
8.1.2. Examples of decision problems	135
8.1.3. Complexity of search problems	136
8.2. Hierarchies for time and space	139
8.2.1. Simulation of several tapes	139
8.2.2. The role of constants	140
8.2.3. Relations between classes	140
8.2.4. Hierarchies	141
8.2.5. Non-deterministic space classes	143
8.3. Other models and measures	147
8.3.1. Random access machines	147
8.3.2. Complexity on general structures	148
8.3.3. Average and smoothed complexities	148

8.3.4. Other measures	148
8.4. Exercises	150
Chapter 9. First-order definability	151
9.1. Explicit definitions	151
9.1.1. Definability on a class of structures	152
9.1.2. Partial elementary equivalence	153
9.1.3. Ehrenfeucht–Fraïssé games	154
9.1.4. Logical characterization of $r$ -equivalence	156
9.1.5. Applications of Ehrenfeucht–Fraïssé games	158
9.1.6. Relational structures	160
9.2. 0-1 Law	162
9.2.1. Łoś–Vaught’s test	163
9.2.2. The theory of almost all relational structures	163
9.2.3. Convergence and 0-1 law	164
9.3. Implicit definitions	166
9.3.1. Craig’s interpolation theorem	166
9.3.2. Beth’s definability theorem	168
9.3.3. Finite structures	169
9.4. Exercises	171
Chapter 10. Inductive definitions and second-order logic	175
10.1. Inductive definitions	175
10.1.1. Inductive systems	176
10.1.2. Fixed point logic	178
10.1.3. Substitution	179
10.1.4. Stage comparison	180
10.1.5. Fixed point hierarchy	182
10.1.6. Datalog	184
10.2. Second-order logic	186
10.2.1. Second-order formulas and interpretation	186
10.2.2. Inductive definitions and second-order logic	187
10.3. Exercises	189
Chapter 11. Time complexity : the classes P and NP	191
11.1. The class NP and NP-complete problems	191
11.1.1. Polynomial time reductions	192
11.1.2. NP-complete problems	192
11.1.3. Logical characterization of NP	199
11.1.4. Examples of definable NP problems	202
11.2. The logical characterizations of P and FP classes	204
11.2.1. P: the class of global inductive relations	204
11.2.2. FP: the class of global recursive functions	208
11.3. Exercises	211
Chapter 12. Models of parallel computations	213

12.1. The boolean circuits	213
12.2. Complexity in parallel models	217
12.2.1. Other circuits	219
12.3. Examples of NC problems	219
12.3.1. Matrix multiplication	219
12.3.2. Csanky's matrix inversion algorithm	220
12.4. The Parallel RAM	223
12.5. Exercises	226
<b>Chapter 13. Space complexity: the classes L, FL, NL and PSPACE</b>	<b>229</b>
13.1. The classes L, NL and FL	230
13.1.1. NL-complete problems	230
13.1.2. Alternation with space constraints	231
13.1.3. Logical characterization of NL	232
13.1.4. Logical characterization of FL	236
13.2. Sequential space and parallel time	241
13.2.1. The parallel time thesis	243
13.2.2. P-complete problems	243
13.3. The class PSPACE	244
13.3.1. The quantified boolean formulas and the class AP	244
13.3.2. Stochastic satisfiability	246
13.3.3. Problems with uncertainty	247
13.4. Exercises	249
<b>Chapter 14. Definability of optimization and counting problems</b>	<b>251</b>
14.1. Optimization problems	251
14.2. Counting problems	257
14.2.1. The counting classes	257
14.2.2. Definability of counting functions	259
14.3. Exercises	262
<b>Part 3. Approximation and classes beyond NP</b>	<b>263</b>
<b>Chapter 15. Probabilistic Classes</b>	<b>265</b>
15.1. Probabilistic decision classes	265
15.1.1. Error amplification	268
15.1.2. Comparing PP and #P	269
15.1.3. Other probabilistic classes	269
15.2. Examples of probabilistic algorithms	270
15.2.1. Random walk in an undirected graph	270
15.2.2. Perfect matching in a bipartite graph	272
15.2.3. Solovay–Strassen primality test	275
15.3. Exercises	277
<b>Chapter 16. Probabilistic verification</b>	<b>279</b>
16.1. Interactive proofs	280
16.2. Examples of interactive protocols	281

16.2.1. The problem of non-isomorphism of graphs	281
16.2.2. A protocol for the permanent	283
16.2.3. A protocol for QBF	285
16.3. Multiprover protocols	289
16.4. Probabilistic checkable proofs: PCP	290
16.4.1. A PCP( $n^3, 1$ ) for 3SAT	293
16.5. Exercises	297
Chapter 17. Approximation	299
17.1. Optimization problems	299
17.1.1. Examples of approximation algorithms	301
17.1.2. Approximation of optimization problems	304
17.1.3. Reductions and completeness	309
17.1.4. Non-approximability	311
17.2. Counting problems	313
17.2.1. Approximation of counting problems	314
17.2.2. Approximation of #DNF	315
17.3. Approximate verification	318
17.4. Exercises	320
Chapter 18. Classes beyond NP	323
18.1. The polynomial hierarchy	323
18.1.1. Second-order definability	323
18.1.2. Relative computability	324
18.1.3. Alternating machines	326
18.2. Generalizations of NP	327
18.2.1. The classes $\oplus P$ , PP and BPP	328
18.2.2. Universal hashing	331
18.2.3. The Valiant–Vazirani theorem	333
18.2.4. Toda’s theorem	334
18.3. Exponential time	338
18.4. Exercises	342
Bibliography	343
Index	349

## Introduction

The book presents some of the fundamental ideas of Logic for Computer Science. It introduces classical notions of mathematical logic to computer scientists and the new ideas brought by the theory of complexity. Given a problem, it is important to know if there is an algorithmic solution, i.e. if the problem can be solved by an automatic procedure based on a finite set of instructions. The classical notions of mathematical logic, such as decidability, completeness and incompleteness are used to answer this question. When an algorithmic solution exists, it is then important to know if there exists an efficient solution, i.e. an automatic procedure which uses few resources such as time or space. Complexity theory introduces notions such as NP-completeness, reductions, approximations which are used to answer this refined question. We present these concepts from the viewpoint of logic, called *descriptive complexity* and emphasize the roles of randomness and approximation. The presentation is divided into three parts:

**Part 1. Model theory and recursive functions.** We introduce the basic model theory of propositional, first order, inductive definitions and second order logic. Then, we define recursive functions, show their equivalence with Turing computable functions, prove the completeness of first order logic and the incompleteness theorems.

**Part 2. Descriptive complexity.** The functions computable in polynomial time are called effectively computable and can be defined as global recursive functions on classes of finite ordered structures. Decision problems of other complexity classes such as AC, NC, LOGSPACE, NLOGSPACE, NP, PH, have similar logical characterizations and allow to view complexity questions as definability problems in logic. This approach is called descriptive as it is independent of any model of computation.

**Part 3. Approximation.** Randomized algorithms in polynomial time define the class BPP for decision problems which generalizes the class P and the class IP (Interactive proofs) for verification problems which generalizes the class NP. Other models of verification include PCP (probabilistic checkable proofs) and property testing. Some optimization problems and counting problems can also be approximated according to their logical form. The PCP techniques allow one to prove non-approximability results.

The design of programming languages, query languages and specification languages follows many logical principles presented in the first part of the book. The second part explains the relationship between the definitions of algorithms, queries, properties of programs and their computational complexity. For many classical tasks such as the verification of programs, optimization problems or counting problems, it is essential to first study the complexity of these problems. When the degree of unfeasibility is too high, approximation techniques presented in the third part of the book can be useful.

This book is based on two books by the same authors, in French: *Logique et Fondements de l'Informatique*, published in 1993 and *Logique et Complexité*, published in 1996 (Editions Hermès, <http://www.editions-hermes.fr>). The first part is the translation of some of the chapters of the first book, whereas the second and third parts of the present book are translations of the second book with additional new subjects. We will maintain: <http://www.lri.fr/~mdr/logicbook/> for the corrections of some of the exercises, remarks and updates.

Most of the presented material appears in the books: *Computational complexity* by C. Papadimitriou [**Pap94**], *Descriptive Complexity* by N. Immerman [**Imm99**], and *Finite Model Theory* by H. D. Ebbinghaus and J. Flum [**EF91**].

We wish to thank all our colleagues who helped us on the French books, in particular Stéphane Boucheron, Miklós Sántha and Avy Sharell. In addition, we thank Mikaël Cozic, Miki Hermann, Isabelle Bril and our translator Elena Calude for their contribution on this English version. We bear the responsibility for the errors which remain in this book.