1. Record Nr. UNINA9910254981403321

Autore	Reus Bernhard
Titolo	Limits of Computation : From a Programming Perspective / / by Bernhard Reus
Pubbl/distr/stampa	Cham : , : Springer International Publishing : , : Imprint : Springer, , 2016
ISBN	3-319-27889-4
Edizione	[1st ed. 2016.]
Descrizione fisica	1 online resource (XVIII, 348 p. 80 illus.)
Collana	Undergraduate Topics in Computer Science, , 1863-7310
Disciplina	001.642
Soggetti	Algorithms Algorithm Analysis and Problem Complexity Mathematics of Algorithmic Complexity
Lingua di pubblicazione	Inglese
Formato	Materiale a stampa
Livello bibliografico	Monografia
Note generali	Includes index.
Nota di contenuto	Intro -- Foreword -- Preface -- For Tutors -- Acknowledgements -- Contents -- 1 Limits? What Limits? -- 1.1 Physical Limits of Computation -- 1.1.1 Fundamental Engineering Constraints to Semiconductor Manufacturing and Scaling -- 1.1.2 Fundamental Limits to Energy Efficiency -- 1.1.3 Fundamental Physical Constraints on Computing in General -- 1.2 The Limits Addressed -- 1.2.1 Computability Overview -- 1.2.2 Complexity Overview -- References -- Part I Computability -- 2 Problems and Effective Procedures -- 2.1 On Computability -- 2.1.1 Historical Remarks -- 2.1.2 Effective Procedures -- 2.2 Sets, Relations and Functions -- 2.2.1 Sets -- 2.2.2 Relations -- 2.2.3 Functions -- 2.2.4 Partial Functions -- 2.2.5 Total Functions -- 2.3 Problems -- 2.3.1 Computing Solutions to Problems -- References -- 3 The WHILE-Language -- 3.1 The Data Type of Binary Trees -- 3.2 WHILE-Syntax -- 3.2.1 Expressions -- 3.2.2 Commands -- 3.2.3 Programs -- 3.2.4 A Grammar for WHILE -- 3.2.5 Layout Conventions and Brackets -- 3.3 Encoding Data Types as Trees -- 3.3.1 Boolean Values -- 3.3.2 Lists and Pairs -- 3.3.3 Natural Numbers -- 3.3.4 Finite Words -- 3.4 Sample Programs -- 3.4.1 Addition -- 3.4.2 List Reversal -- 3.4.3 Tail Recursion -- 3.4.4 Analysis of Algorithms -- References -- 4 Semantics of WHILE -- 4.1 Stores -- 4.2 Semantics of Programs -- 4.3 Semantics of Commands -- 4.4 Semantics of Expressions --

References -- 5 Extensions of WHILE -- 5.1 Equality -- 5.2 Literals -- 5.2.1 Number Literals -- 5.2.2 Boolean Literals -- 5.3 Adding Atoms -- 5.4 List Constructor -- 5.5 Macro Calls -- 5.6 Switch Statement -- References -- 6 Programs as Data Objects -- 6.1 Interpreters Formally -- 6.2 Abstract Syntax Trees -- 6.3 Encoding of WHILE-ASTs in mathbbD -- Reference -- 7 A Self-interpreter for WHILE -- 7.1 A Self-interpreter for WHILE -Programs with One Variable.
7.1.1 General Tree Traversal for ASTs -- 7.1.2 The STEP Macro -- 7.2 A Self-interpreter for WHILE -- 7.2.1 Store Manipulation Macros -- References -- 8 An Undecidable (Non-computable) Problem -- 8.1 WHILE-Computability and Decidability -- 8.2 The Halting Problem for WHILE -- 8.3 Diagonalisation and the Barber ``Paradox'' -- 8.4 Proof of the Undecidability of the Halting Problem -- References -- 9 More Undecidable Problems -- 9.1 Semi-decidability of the Halting Problem -- 9.2 Rice's Theorem -- 9.3 The Tiling Problem -- 9.4 Problem Reduction -- 9.5 Other (Famous) Undecidable Problems -- 9.6 Dealing with Undecidable Problems -- 9.7 A Fast-Growing Non-computable Function -- References -- 10 Self-referencing Programs -- 10.1 The S-m-n Theorem -- 10.2 Kleene's Recursion Theorem -- 10.3 Recursion Elimination -- References -- 11 The Church-Turing Thesis -- 11.1 The Thesis -- 11.2 Semantic Framework for Machine-Like Models -- 11.3 Turing Machines TM -- 11.4 GOTO-Language -- 11.5 Register Machines RAM and SRAM -- 11.6 Counter Machines CM -- 11.7 Cellular Automata -- 11.7.1 2D: Game of Life -- 11.7.2 1D: Rule 110 -- 11.8 Robustness of Computability -- 11.8.1 The Crucial Role of Compilers -- 11.8.2 Equivalence of Models -- References -- Part II Complexity -- 12 Measuring Time Usage -- 12.1 Unit-Cost Time Measure -- 12.2 Time Measure for WHILE -- 12.3 Comparing Programming Languages Considering Time -- References -- 13 Complexity Classes -- 13.1 Runtime Bounds -- 13.2 Time Complexity Classes -- 13.3 Lifting Simulation Properties to Complexity Classes -- 13.4 Big-O and Little-o -- References -- 14 Robustness of P -- 14.1 Extended Church--Turing Thesis -- 14.2 Invariance or Cook's Thesis -- 14.2.1 Non-sequential Models -- 14.2.2 Evidence for Cook's Thesis -- 14.2.3 Linear Time -- 14.3 Cobham--Edmonds Thesis -- References -- 15 Hierarchy Theorems.
15.1 Linear Time Hierarchy Theorems -- 15.2 Beyond Linear Time -- 15.3 Gaps in the Hierarchy -- References -- 16 Famous Problems in P -- 16.1 Decision Versus Optimisation Problems -- 16.2 Predecessor Problem -- 16.3 Membership Test for a Context Free Language -- 16.4 Primality Test -- 16.5 Graph Problems -- 16.5.1 Reachability in a Graph -- 16.5.2 Shortest Paths in a Graph -- 16.5.3 Maximal Matchings -- 16.5.4 Min-Cut and Max-Flow -- 16.5.5 The Seven Bridges of Konigsberg -- 16.6 Linear Programming -- References -- 17 Common Problems Not Known to Be in P -- 17.1 The Travelling Salesman Problem (TSP) -- 17.2 The Graph Colouring Problem -- 17.3 Max-Cut Problem -- 17.4 The 0-1 Knapsack Problem -- 17.5 Integer Programming Problem -- 17.6 Does Not Being in P Matter? -- References -- 18 The One-Million-Dollar Question -- 18.1 The Complexity Class NP -- 18.2 Nondeterministic Programs -- 18.2.1 Time Measure of Nondeterministic Programs -- 18.2.2 Some Basic Facts About NP -- 18.3 Robustness of NP -- 18.4 Problems in NP -- 18.5 The Biggest Open Problem in (Theoretical) Computer Science -- References -- 19 How Hard Is a Problem? -- 19.1 Reminder: Effective Reductions -- 19.2 Polynomial Time Reduction -- 19.3 Hard Problems -- References -- 20 Complete Problems -- 20.1 A First NP-complete Problem -- 20.2 More NP-complete Problems -- 20.3 Puzzles and Games -- 20.3.1 Chess -- 20.3.2 Sudoku -- 20.3.3 Tile-Matching

Games -- 20.4 Database Queries -- 20.5 Policy Based Routing -- 20.6 ``Limbo'' Problems -- 20.7 Complete Problems in Other Classes -- 20.7.1 P-complete -- 20.7.2 RE-complete -- References -- 21 How to Solve NP-Complete Problems -- 21.1 Exact Algorithms -- 21.2 Approximation Algorithms -- 21.3 Parallelism -- 21.4 Randomization -- 21.4.1 The Class RP -- 21.4.2 Probabilistic Algorithms -- 21.5 Solving the Travelling Salesman Problem -- 21.5.1 Exact Solutions. 21.5.2 Approximative Solutions -- 21.6 When Bad Complexity is Good News -- References -- 22 Molecular Computing -- 22.1 The Beginnings of DNA Computing -- 22.2 DNA Computing Potential -- 22.3 DNA Computing Challenges -- 22.4 Abstract Models of Molecular Computation -- 22.4.1 Chemical Reaction Networks (CRN) -- 22.4.2 CRNs as Effective Procedures -- 22.4.3 Are CRNs Equivalent to Other Notions of Computation? -- 22.4.4 Time Complexity for CRNs -- 22.4.5 Implementing CRNs -- References -- 23 Quantum Computing -- 23.1 Molecular Electronics -- 23.2 The Mathematics of Quantum Mechanics -- 23.3 Quantum Computability and Complexity -- 23.4 Quantum Algorithms -- 23.4.1 Shor's Algorithm -- 23.4.2 Grover's Algorithm -- 23.5 Building Quantum Computers -- 23.6 Quantum Computing Challenges -- 23.7 To Boldly Go -- References -- Further Reading-Computabilityand Complexity Textbooks -- Glossary -- Index.

Sommario/riassunto

This textbook discusses the most fundamental and puzzling questions about the foundations of computing. In 23 lecture-sized chapters it provides an exciting tour through the most important results in the field of computability and time complexity, including the Halting Problem, Rice's Theorem, Kleene's Recursion Theorem, the Church-Turing Thesis, Hierarchy Theorems, and Cook-Levin's Theorem. Each chapter contains classroom-tested material, including examples and exercises. Links between adjacent chapters provide a coherent narrative. Fundamental results are explained lucidly by means of programs written in a simple, high-level imperative programming language, which only requires basic mathematical knowledge. Throughout the book, the impact of the presented results on the entire field of computer science is emphasised. Examples range from program analysis to networking, from database programming to popular games and puzzles. Numerous biographical footnotes about the famous scientists who developed the subject are also included. "Limits of Computation" offers a thorough, yet accessible, introduction to computability and complexity for the computer science student of the 21st century.