Undergraduate Topics in Computer Science

‘Undergraduate Topics in Computer Science’ (UTiCS) delivers high-quality instructional content for undergraduates studying in all areas of computing and information science. From core foundational and theoretical material to final-year topics and applications, UTiCS books take a fresh, concise, and modern approach and are ideal for self-study or for a one- or two-semester course. The texts are all authored by established experts in their fields, reviewed by an international advisory board, and contain numerous examples and problems, many of which include fully worked solutions.

The UTiCS concept relies on high-quality, concise books in softback format, andgenerally a maximum of 275–300 pages. For undergraduate textbooks that arelikely to be longer, more expository, Springer continues to offer the highly regardedTexts in Computer Science series, to which we refer potential authors.

More information about this series at http://www.springer.com/series/7592

Gerard O'Regan

Mathematics in Computing

An Accessible Guide to Historical, Foundational and Application Contexts

Second Edition

Springer

Gerard O'Regan
SQC Consulting
Mallow, Cork, Ireland

ISSN 1863-7310 ISSN 2197-1781 (electronic)
Undergraduate Topics in Computer Science
ISBN 978-3-030-34208-1 ISBN 978-3-030-34209-8 (eBook)
https://doi.org/10.1007/978-3-030-34209-8

This Springer imprint is published by the registered company Springer Nature Switzerland AG
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

To

My dear aunts Mrs. Rita Lowry and Mrs. Kitty Butler

and

In memory of my late uncle Moses Fenton.

Preface

Overview

The objective of this book is to give the reader a flavour of mathematics used in the computing field. We emphasize the applicability of mathematics rather than the study of mathematics for its own sake, and the goal is that the reader will appreciate the rich applications of mathematics to the computing field. This includes applications to the foundations of computing; to error detection and correcting codes with finite field theory; to the field of cryptography with the results of number theory; to the modelling of telecommunication networks with graph theory; to the application of discrete mathematics and proof techniques to the software correctness field (especially safety-critical systems using formal methods and model checking); to language theory and semantics; and to computability and decidability.

Organization and Features

The first chapter introduces analog and digital computers, and the von Neumann architecture which is the fundamental architecture underlying a digital computer. Chapter 2 discusses the foundations of computing, and we describe the binary number system and the step reckoner calculating machine that were invented by Leibniz. Babbage designed the difference engine as a machine to evaluate polynomials, and his analytic engine provided the vision of a modern computer. Boole was an English mathematician who made important contributions to mathematics and logic, and his symbolic logic is the foundation for digital computing.

Chapter 3 provides an introduction to fundamental building blocks in mathematics including sets, relations and functions. A set is a collection of well-defined objects and it may be finite or infinite. A relation between two sets A and B indicates a relationship between members of the two sets and is a subset of the Cartesian product of the two sets. A function is a special type of relation such that for each element in A there is at most one element in the co-domain B. Functions may be partial or total and injective, surjective or bijective.

Chapter 4 presents a short introduction to algorithms, where an algorithm is a well-defined procedure for solving a problem. It consists of a sequence of steps that takes a set of values as input and produces a set of values as output. An algorithm is an exact specification of how to solve the problem, and it explicitly defines the procedure so that a computer program may implement the solution in some programming language.

Chapter 5 presents the fundamentals of number theory and discusses prime number theory and the greatest common divisor and least common multiple of two numbers.

Chapter 6 discusses algebra and we discuss simple and simultaneous equations, including the method of elimination and the method of substitution to solve simultaneous equations. We show how quadratic equations may be solved by factorization, completing the square or using the quadratic formula. We present the laws of logarithms and indices. We discuss various structures in abstract algebra, including monoids, groups, rings, integral domains, fields and vector spaces.

Chapter 7 discusses sequences and series and permutations and combinations. Arithmetic and geometric sequences and series are discussed, and we discuss applications of geometric sequences and series to the calculation of compound interest and annuities.

Chapter 8 discusses mathematical induction and recursion. Induction is a common proof technique in mathematics, and there are two parts to a proof by induction (the base case and the inductive step). We discuss strong and weak induction, and we discuss how recursion is used to define sets, sequences and functions. This leads us to structural induction, which is used to prove properties of recursively defined structures.

Chapter 9 discusses graph theory where a graph $G = (V, E)$ consists of vertices and edges. It is a practical branch of mathematics that deals with the arrangements of vertices and edges between them, and it has been applied to practical problems such as the modelling of computer networks, determining the shortest driving route between two cities and the travelling salesman problem.

Chapter 10 discusses cryptography, which is an important application of number theory. The codebreaking work done at Bletchley Park in England during the Second World War is discussed, and the fundamentals of cryptography, including private- and public-key cryptosystems, are discussed.

Chapter 11 presents coding theory and is concerned with error detection and error correction codes. The underlying mathematics includes abstract mathematics such as group theory, rings, fields and vector spaces.

Chapter 12 discusses language theory and includes a discussion on grammars, parse trees and derivations from a grammar. The important area of programming language semantics is discussed, including axiomatic, denotational and operational semantics.

Chapter 13 discusses computability and decidability. The Church–Turing thesis states that anything that is computable is computable by a Turing machine. Church and Turing showed that mathematics is not decidable. In other words, there is no mechanical procedure (i.e. algorithm) to determine whether an arbitrary mathematical proposition is true or false, and so the only way is to determine the truth or falsity of a statement is try to solve the problem.

Chapter 14 discusses matrices including 2×2 and general $n \times m$ matrices. Various operations such as the addition and multiplication of matrices are considered, and the determinant and inverse of a square matrix are discussed. The application of matrices to solving a set of linear equations using Gaussian elimination is considered.

Chapter 15 presents a short history of logic, and we discuss Greek contributions to syllogistic logic, stoic logic, fallacies and paradoxes. Boole's symbolic logic and its application to digital computing are discussed, and we consider Frege's work on predicate logic.

Chapter 16 provides an introduction to propositional and predicate logic. Propositional logic may be used to encode simple arguments that are expressed in natural language and to determine their validity. The nature of mathematical proof is discussed, and we present proof by truth tables, semantic tableaux and natural deduction. Predicate logic allows complex facts about the world to be represented, and new facts may be determined via deductive reasoning. Predicate calculus includes predicates, variables and quantifiers, and a predicate is a characteristic or property that the subject of a statement can have.

Chapter 17 presents some advanced topics in logic including fuzzy logic, temporal logic, intuitionistic logic, undefined values, theorem provers and the applications of logic to AI. Fuzzy logic is an extension of classical logic that acts as a mathematical model for vagueness. Temporal logic is concerned with the expression of properties that have time dependencies, and it allows temporal properties about the past, present and future to be expressed. Intuitionism was a controversial theory on the foundations of mathematics based on a rejection of the law of the excluded middle, and an insistence on constructive existence. We discuss three approaches to deal with undefined values, including the logic of partial functions; Dijkstra's approach with his cand and cor operators; and Parnas's approach which preserves a classical two-valued logic.

Chapter 18 discusses the nature of proof and theorem proving, and we discuss automated and interactive theorem provers. We discuss the nature of mathematical proof and formal mathematical proof. Chapter 19 discusses software engineering and the mathematics to support software engineering.

Chapter 20 discusses software reliability and dependability, and covers topics such as software reliability and software reliability models; the Cleanroom methodology, system availability, safety and security-critical systems; and dependability engineering.

Chapter 21 discusses formal methods, which consist of a set of mathematical techniques to rigorously specify and derive a program from its specification. Formal methods may be employed to rigorously state the requirements of the proposed

system; they may be employed to derive a program from its mathematical specification; and they may provide a rigorous proof that the implemented program satisfies its specification. They have been mainly applied to the safety-critical field.

Chapter 22 presents the Z specification language, which is one of the most widely used formal methods. It was developed at Oxford University in the U.K.

Chapter 23 discusses automata theory, including finite-state machines, pushdown automata and Turing machines. Finite-state machines are abstract machines that are in only one state at a time, and the input symbol causes a transition from the current state to the next state. Pushdown automata have greater computational power, and they contain extra memory in the form of a stack from which symbols may be pushed or popped. The Turing machine is the most powerful model for computation, and this theoretical machine is equivalent to an actual computer in the sense that it can compute exactly the same set of functions.

Chapter 24 discusses model checking which is an automated technique such that given a finite-state model of a system and a formal property, then it systematically checks whether the property is true or false in a given state in the model. It is an effective technique to identify potential design errors, and it increases the confidence in the correctness of the system design.

Chapter 25 discusses probability and statistics and includes a discussion on discrete and continuous random variables, probability distributions, sample spaces, sampling, the abuse of statistics, variance and standard deviation and hypothesis testing. The application of probability to the software reliability field is discussed.

Chapter 26 discusses complex numbers and quaternions. Complex numbers are of the form $a + bi$ where a and b are real numbers, and $i^2 = -1$. Quaternions are a generalization of complex numbers to quadruples that satisfy the quaternion formula $i^2 = j^2 = k^2 = -1$.

Chapter 27 provides a very short introduction to calculus and provides a high-level overview of limits, continuity, differentiation, integration, numerical analysis, Fourier series, Laplace transforms and differential equations.

Chapter 28 is the concluding chapter in which we summarize the journey that we have travelled in this book.

Audience

The audience of this book includes computer science students who wish to obtain an overview of mathematics used in computing, and mathematicians who wish to get an overview of how mathematics is applied in the computing field. The book will also be of interest to the motivated general reader.

Acknowledgements

I am deeply indebted to friends and family who supported my efforts in this endeavour. My thanks to the team at Springer for suggesting this new edition and for their professional work. A special thanks to my aunts (Mrs. Rita Lowry and Mrs. Kitty Butler) who are always a pleasure to visit in Co.Tipperary and Co.Cork, and who have clearly shown that it is possible to be over 90 and yet to have the energy and sense of fun of teenagers.

Cork, Ireland Gerard O'Regan

Contents

1 What Is a Computer? 1
1.1 Introduction 1
1.2 Analog Computers 2
1.3 Digital Computers 4
1.3.1 Vacuum Tubes 4
1.3.2 Transistors 5
1.3.3 Integrated Circuits 5
1.3.4 Microprocessors 7
1.4 von Neumann Architecture 8
1.5 Hardware and Software 9
1.6 Review Questions 10
1.7 Summary 10
References 11

2 Foundations of Computing 13
2.1 Introduction 13
2.2 Step Reckoner Calculating Machine 14
2.3 Binary Numbers 15
2.4 The Difference Engine 17
2.5 The Analytic Engine—Vision of a Computer 19
2.5.1 Applications of Analytic Engine 21
2.6 Boole's Symbolic Logic 22
2.6.1 Switching Circuits and Boolean Algebra 25
2.7 Application of Symbolic Logic to Digital Computing 27
2.8 Review Questions 28
2.9 Summary 28
References 29

3 Overview of Mathematics in Computing 31
3.1 Introduction 31
3.2 Set Theory 32
3.2.1 Set-Theoretical Operations 35
3.2.2 Properties of Set-Theoretical Operations 37

3.2.3 Russell's Paradox . . . 38
3.2.4 Computer Representation of Sets . . . 40
3.3 Relations . . . 41
3.3.1 Reflexive, Symmetric and Transitive Relations . . . 42
3.3.2 Composition of Relations . . . 44
3.3.3 Binary Relations . . . 46
3.3.4 Applications of Relations to Databases . . . 47
3.4 Functions . . . 49
3.5 Application of Functions to Functional Programming . . . 53
3.5.1 Miranda . . . 54
3.6 Number Theory . . . 56
3.7 Automata Theory . . . 57
3.8 Graph Theory . . . 58
3.9 Computability and Decidability . . . 58
3.10 Review Questions . . . 59
3.11 Summary . . . 60
References . . . 60

4 Introduction to Algorithms . . . 61
4.1 Introduction . . . 61
4.2 Early Algorithms . . . 62
4.2.1 Greatest Common Divisors (GCD) . . . 63
4.2.2 Euclid's Greatest Common Divisor Algorithm . . . 63
4.2.3 Sieve of Eratosthenes Algorithm . . . 65
4.2.4 Early Cipher Algorithms . . . 66
4.3 Sorting Algorithms . . . 68
4.4 Binary Trees and Graph Theory . . . 71
4.5 Modern Cryptographic Algorithms . . . 72
4.6 Computational Complexity . . . 73
4.7 Review Questions . . . 74
4.8 Summary . . . 74
References . . . 75

5 Number Theory . . . 77
5.1 Introduction . . . 77
5.2 Elementary Number Theory . . . 79
5.3 Prime Number Theory . . . 84
5.3.1 Greatest Common Divisors (GCD) . . . 86
5.3.2 Least Common Multiple (LCM) . . . 87
5.3.3 Euclid's Algorithm . . . 88
5.3.4 Distribution of Primes . . . 89
5.4 Theory of Congruences . . . 92
5.5 Binary System and Computer Representation of Numbers . . . 95
5.6 Review Questions . . . 96

5.7 Summary . . . 97
References . . . 98

6 Algebra . . . 99
6.1 Introduction . . . 99
6.2 Simple and Simultaneous Equations . . . 100
6.3 Quadratic Equations . . . 103
6.4 Indices and Logarithms . . . 106
6.5 Horner's Method for Polynomials . . . 107
6.6 Abstract Algebra . . . 108
6.6.1 Monoids and Groups . . . 108
6.6.2 Rings . . . 110
6.6.3 Fields . . . 111
6.6.4 Vector Spaces . . . 112
6.7 Review Questions . . . 114
6.8 Summary . . . 114

7 Sequences, Series and Permutations and Combinations . . . 117
7.1 Introduction . . . 117
7.2 Sequences and Series . . . 118
7.3 Arithmetic and Geometric Sequences . . . 119
7.4 Arithmetic and Geometric Series . . . 120
7.5 Simple and Compound Interest . . . 121
7.6 Time Value of Money and Annuities . . . 123
7.7 Permutations and Combinations . . . 124
7.8 Review Questions . . . 128
7.9 Summary . . . 129

8 Mathematical Induction and Recursion . . . 131
8.1 Introduction . . . 131
8.2 Strong Induction . . . 134
8.3 Recursion . . . 136
8.4 Structural Induction . . . 138
8.5 Review Questions . . . 139
8.6 Summary . . . 139
Reference . . . 140

9 Graph Theory . . . 141
9.1 Introduction . . . 141
9.2 Undirected Graphs . . . 143
9.2.1 Hamiltonian Paths . . . 147
9.3 Trees . . . 148
9.3.1 Binary Trees . . . 149
9.4 Graph Algorithms . . . 150
9.5 Graph Colouring and Four-Colour Problem . . . 150

9.6 Review Questions . . . 152
9.7 Summary . . . 152
Reference . . . 153

10 Cryptography . . . 155
10.1 Introduction . . . 155
10.2 Breaking the Enigma Codes . . . 157
10.3 Cryptographic Systems . . . 160
10.4 Symmetric-Key Systems . . . 161
10.5 Public-Key Systems . . . 166
10.5.1 RSA Public-Key Cryptosystem . . . 168
10.5.2 Digital Signatures . . . 169
10.6 Review Questions . . . 169
10.7 Summary . . . 170
References . . . 170

11 Coding Theory . . . 171
11.1 Introduction . . . 171
11.2 Mathematical Foundations . . . 172
11.3 Simple Channel Code . . . 173
11.4 Block Codes . . . 174
11.4.1 Error Detection and Correction . . . 176
11.5 Linear Block Codes . . . 177
11.5.1 Parity Check Matrix . . . 180
11.5.2 Binary Hamming Code . . . 180
11.5.3 Binary Parity Check Code . . . 182
11.6 Miscellaneous Codes in Use . . . 182
11.7 Review Questions . . . 182
11.8 Summary . . . 183
References . . . 183

12 Language Theory and Semantics . . . 185
12.1 Introduction . . . 185
12.2 Alphabets and Words . . . 186
12.3 Grammars . . . 187
12.3.1 Backus–Naur Form . . . 190
12.3.2 Parse Trees and Derivations . . . 191
12.4 Programming Language Semantics . . . 193
12.4.1 Axiomatic Semantics . . . 193
12.4.2 Operational Semantics . . . 195
12.4.3 Denotational Semantics . . . 196
12.5 Lambda Calculus . . . 197
12.6 Lattices and Order . . . 199

12.6.1 Partially Ordered Sets . . . 199
12.6.2 Lattices . . . 201
12.6.3 Complete Partial Orders . . . 203
12.6.4 Recursion . . . 204
12.7 Review Questions . . . 206
12.8 Summary . . . 206
References . . . 206

13 Computability and Decidability . . . 209
13.1 Introduction . . . 209
13.2 Logicism and Formalism . . . 210
13.3 Decidability . . . 213
13.4 Computability . . . 215
13.5 Computational Complexity . . . 218
13.6 Review Questions . . . 219
13.7 Summary . . . 219
Reference . . . 220

14 Matrix Theory . . . 221
14.1 Introduction . . . 221
14.2 Two × Two Matrices . . . 223
14.3 Matrix Operations . . . 225
14.4 Determinants . . . 228
14.5 Eigenvectors and Values . . . 230
14.6 Gaussian Elimination . . . 230
14.7 Review Questions . . . 232
14.8 Summary . . . 232
Reference . . . 233

15 A Short History of Logic . . . 235
15.1 Introduction . . . 235
15.2 Syllogistic Logic . . . 236
15.3 Paradoxes and Fallacies . . . 238
15.4 Stoic Logic . . . 240
15.5 Boole's Symbolic Logic . . . 241
15.5.1 Switching Circuits and Boolean Algebra . . . 242
15.6 Frege . . . 243
15.7 Review Questions . . . 244
15.8 Summary . . . 245
References . . . 245

16 Propositional and Predicate Logic . . . 247
16.1 Introduction . . . 247
16.2 Propositional Logic . . . 248
16.2.1 Truth Tables . . . 250

16.2.2 Properties of Propositional Calculus . . . 252
16.2.3 Proof in Propositional Calculus . . . 253
16.2.4 Semantic Tableaux in Propositional Logic . . . 256
16.2.5 Natural Deduction . . . 258
16.2.6 Sketch of Formalization of Propositional Calculus . . . 259
16.2.7 Applications of Propositional Calculus . . . 261
16.2.8 Limitations of Propositional Calculus . . . 262
16.3 Predicate Calculus . . . 263
16.3.1 Sketch of Formalization of Predicate Calculus . . . 265
16.3.2 Interpretation and Valuation Functions . . . 267
16.3.3 Properties of Predicate Calculus . . . 268
16.3.4 Applications of Predicate Calculus . . . 268
16.3.5 Semantic Tableaux in Predicate Calculus . . . 269
16.4 Review Questions . . . 271
16.5 Summary . . . 272
References . . . 273

17 Advanced Topics in Logic . . . 275
17.1 Introduction . . . 275
17.2 Fuzzy Logic . . . 276
17.3 Temporal Logic . . . 277
17.4 Intuitionistic Logic . . . 279
17.5 Undefined Values . . . 281
17.5.1 Logic of Partial Functions . . . 281
17.5.2 Parnas Logic . . . 283
17.5.3 Dijkstra and Undefinedness . . . 284
17.6 Logic and AI . . . 286
17.7 Review Questions . . . 290
17.8 Summary . . . 290
References . . . 291

18 The Nature of Theorem Proving . . . 293
18.1 Introduction . . . 293
18.2 Early Automation of Proof . . . 296
18.3 Interactive Theorem Provers . . . 298
18.4 A Selection of Theorem Provers . . . 300
18.5 Review Questions . . . 300
18.6 Summary . . . 300
References . . . 302

19 Software Engineering Mathematics 303
19.1 Introduction 303
19.2 What Is Software Engineering? 306
19.3 Early Software Engineering Mathematics 311
19.4 Mathematics in Software Engineering 314
19.5 Software Inspections and Testing 315
19.6 Process Maturity Models 316
19.7 Review Questions 317
19.8 Summary 317
References 318

20 Software Reliability and Dependability 319
20.1 Introduction 319
20.2 Software Reliability 320
20.2.1 Software Reliability and Defects 321
20.2.2 Cleanroom Methodology 323
20.2.3 Software Reliability Models 324
20.3 Dependability 327
20.4 Computer Security 329
20.5 System Availability 330
20.6 Safety-Critical Systems 330
20.7 Review Questions 331
20.8 Summary 331
References 332

21 Overview of Formal Methods 333
21.1 Introduction 333
21.2 Why Should We Use Formal Methods? 335
21.3 Industrial Applications of Formal Methods 337
21.4 Industrial Tools for Formal Methods 338
21.5 Approaches to Formal Methods 339
21.5.1 Model-Oriented Approach 339
21.5.2 Axiomatic Approach 341
21.6 Proof and Formal Methods 341
21.7 Mathematics in Software Engineering 342
21.8 The Vienna Development Method 343
21.9 VDM♣, the Irish School of VDM 344
21.10 The *Z* Specification Language 345
21.11 The *B*-Method 346
21.12 Predicate Transformers and Weakest Preconditions 347
21.13 The Process Calculi 348
21.14 Finite-State Machines 349
21.15 The Parnas Way 350
21.16 Model Checking 350

21.17 Usability of Formal Methods . . . 351
21.18 Review Questions . . . 352
21.19 Summary . . . 353
References . . . 354

22 Z Formal Specification Language . . . 355
22.1 Introduction . . . 355
22.2 Sets . . . 358
22.3 Relations . . . 359
22.4 Functions . . . 361
22.5 Sequences . . . 362
22.6 Bags . . . 363
22.7 Schemas and Schema Composition . . . 364
22.8 Reification and Decomposition . . . 367
22.9 Proof in Z . . . 368
22.10 Industrial Applications of *Z* . . . 369
22.11 Review Questions . . . 370
22.12 Summary . . . 370
Reference . . . 371

23 Automata Theory . . . 373
23.1 Introduction . . . 373
23.2 Finite-State Machines . . . 374
23.3 Pushdown Automata . . . 377
23.4 Turing Machines . . . 379
23.5 Review Questions . . . 381
23.6 Summary . . . 382
Reference . . . 382

24 Model Checking . . . 383
24.1 Introduction . . . 383
24.2 Modelling Concurrent Systems . . . 387
24.3 Linear Temporal Logic . . . 388
24.4 Computational Tree Logic . . . 389
24.5 Tools for Model Checking . . . 390
24.6 Industrial Applications of Model Checking . . . 390
24.7 Review Questions . . . 391
24.8 Summary . . . 391
References . . . 392

25 Probability and Statistics . . . 393
25.1 Introduction . . . 393
25.2 Probability Theory . . . 394
25.2.1 Laws of Probability . . . 395
25.2.2 Random Variables . . . 396

25.3 Statistics . . . 400
25.3.1 Abuse of Statistics . . . 400
25.3.2 Statistical Sampling . . . 401
25.3.3 Averages in a Sample . . . 402
25.3.4 Variance and Standard Deviation . . . 403
25.3.5 Bell-Shaped (Normal) Distribution . . . 403
25.3.6 Frequency Tables, Histograms and Pie Charts . . . 406
25.3.7 Hypothesis Testing . . . 407
25.4 Review Questions . . . 409
25.5 Summary . . . 409
Reference . . . 410

26 Complex Numbers and Quaternions . . . 411
26.1 Introduction . . . 411
26.2 Complex Numbers . . . 412
26.3 Quaternions . . . 417
26.3.1 Quaternion Algebra . . . 418
26.3.2 Quaternions and Rotations . . . 422
26.4 Review Questions . . . 423
26.5 Summary . . . 424

27 Calculus . . . 425
27.1 Introduction . . . 425
27.2 Differentiation . . . 429
27.2.1 Rules of Differentiation . . . 431
27.3 Integration . . . 432
27.3.1 Definite Integrals . . . 434
27.3.2 Fundamental Theorems of Integral Calculus . . . 437
27.4 Numerical Analysis . . . 437
27.5 Fourier Series . . . 439
27.6 The Laplace Transform . . . 441
27.7 Differential Equations . . . 442
27.8 Review Questions . . . 443
27.9 Summary . . . 444
Reference . . . 444

28 Epilogue . . . 445

Glossary . . . 449

Bibliography . . . 453

Index . . . 455

List of Figures

Fig. 1.1 Vannevar Bush with the differential analyser . . . 3
Fig. 1.2 Replica of transistor. Public domain . . . 6
Fig. 1.3 von Neumann architecture . . . 9
Fig. 1.4 Fetch/execute cycle . . . 9
Fig. 2.1 Replica of step reckoner at Technische Sammlungen Museum, Dresden . . . 15
Fig. 2.2 Decimal to binary conversion . . . 16
Fig. 2.3 Charles Babbage . . . 18
Fig. 2.4 Difference engine no. 2. Photo public domain . . . 20
Fig. 2.5 Lady Ada Lovelace . . . 21
Fig. 2.6 George Boole . . . 23
Fig. 2.7 Binary AND operation . . . 25
Fig. 2.8 Binary OR operation . . . 26
Fig. 2.9 NOT operation . . . 26
Fig. 2.10 Half adder . . . 26
Fig. 2.11 Claude Shannon . . . 27
Fig. 3.1 Bertrand Russell . . . 39
Fig. 3.2 Reflexive relation . . . 43
Fig. 3.3 Symmetric relation . . . 43
Fig. 3.4 Transitive relation . . . 43
Fig. 3.5 Partitions of A . . . 44
Fig. 3.6 Composition of relations S o R . . . 45
Fig. 3.7 Edgar Codd . . . 48
Fig. 3.8 PART relation . . . 48
Fig. 3.9 Domain and range of a partial function . . . 50
Fig. 3.10 Injective and surjective functions . . . 52
Fig. 3.11 Bijective function (one to one and onto) . . . 52
Fig. 4.1 Euclid of Alexandria . . . 64
Fig. 4.2 Primes between 1 and 50 . . . 66
Fig. 4.3 Caesar Cipher . . . 67
Fig. 4.4 Insertion sort example . . . 69
Fig. 4.5 Merge sort example . . . 70
Fig. 4.6 Sorted binary tree . . . 71
Fig. 5.1 Pierre de Fermat . . . 78

Fig. 5.2 Pythagorean triples . . . 79
Fig. 5.3 Square numbers . . . 79
Fig. 5.4 Rectangular numbers . . . 80
Fig. 5.5 Triangular numbers . . . 80
Fig. 5.6 Marin Mersenne . . . 81
Fig. 5.7 Leonard Euler . . . 91
Fig. 6.1 Graphical solution to simultaneous equations . . . 102
Fig. 6.2 Graphical solution to quadratic equation . . . 105
Fig. 9.1 Königsberg seven bridges problem . . . 142
Fig. 9.2 Königsberg graph . . . 143
Fig. 9.3 Undirected graph . . . 143
Fig. 9.4 Directed graph . . . 143
Fig. 9.5 Adjacency matrix . . . 145
Fig. 9.6 Incidence matrix . . . 145
Fig. 9.7 Travelling salesman problem . . . 148
Fig. 9.8 Binary tree . . . 150
Fig. 9.9 Determining the chromatic colour of G . . . 151
Fig. 9.10 Chromatic colouring of G . . . 152
Fig. 10.1 The Enigma machine . . . 157
Fig. 10.2 Bletchley Park . . . 158
Fig. 10.3 Alan Turing . . . 159
Fig. 10.4 Replica of Bombe . . . 159
Fig. 10.5 Symmetric-key cryptosystem . . . 161
Fig. 10.6 Public-key cryptosystem . . . 166
Fig. 11.1 Basic digital communication . . . 172
Fig. 11.2 Encoding and decoding of an (n,k) block . . . 175
Fig. 11.3 Error-correcting capability sphere . . . 177
Fig. 11.4 Generator matrix . . . 179
Fig. 11.5 Generation of codewords . . . 179
Fig. 11.6 Identity matrix $(k \times k)$ . . . 180
Fig. 11.7 Hamming code B (7, 4, 3) generator matrix . . . 181
Fig. 12.1 Noam Chomsky. public domain . . . 189
Fig. 12.2 Parse tree $5 \times 3 + 1$ . . . 192
Fig. 12.3 Parse tree $5 \times 3 + 1$ . . . 192
Fig. 12.4 Denotational semantics . . . 197
Fig. 12.5 Pictorial representation of a partial order . . . 200
Fig. 12.6 Pictorial representation of a complete lattice . . . 203
Fig. 13.1 David Hilbert . . . 211
Fig. 13.2 Kurt Gödel . . . 214
Fig. 13.3 Alonzo Church . . . 215
Fig. 14.1 Example of a 4×4 square matrix . . . 222
Fig. 14.2 Multiplication of two matrices . . . 226
Fig. 14.3 Identity matrix I_n . . . 227
Fig. 14.4 Transpose of a matrix . . . 227

Fig. 14.5 Determining the (i, j) minor of A 228
Fig. 15.1 Zeno of Citium 241
Fig. 15.2 Gottlob Frege 244
Fig. 16.1 Gerhard Gentzen 259
Fig. 17.1 Conjunction and disjunction operators 282
Fig. 17.2 Implication and equivalence operators 282
Fig. 17.3 Negation 282
Fig. 17.4 Finding index in array 284
Fig. 17.5 Edsger Dijkstra. Courtesy of Brian Randell 285
Fig. 17.6 John McCarthy. Courtesy of John McCarthy 287
Fig. 18.1 Idea of automated theorem proving 295
Fig. 19.1 David Parnas 307
Fig. 19.2 Waterfall lifecycle model (V-model) 308
Fig. 19.3 SPIRAL lifecycle model 309
Fig. 19.4 Standish group report—estimation accuracy 310
Fig. 19.5 Robert Floyd 311
Fig. 19.6 Branch assertions in flowcharts 312
Fig. 19.7 Assignment assertions in flowcharts 312
Fig. 19.8 C. A. R. Hoare 313
Fig. 19.9 Watts Humphrey. Courtesy of Watts Humphrey 316
Fig. 21.1 Deterministic finite-state machine 350
Fig. 22.1 Specification of positive square root 356
Fig. 22.2 Specification of a library system 357
Fig. 22.3 Specification of borrow operation 358
Fig. 22.4 Specification of vending machine using bags 364
Fig. 22.5 Schema inclusion 365
Fig. 22.6 Merging schemas $(S_1 \vee S_2)$ 365
Fig. 22.7 Schema composition 367
Fig. 22.8 Refinement commuting diagram 368
Fig. 23.1 Finite-state machine with output 375
Fig. 23.2 Deterministic FSM 376
Fig. 23.3 Non-deterministic finite-state machine 376
Fig. 23.4 Components of pushdown automata 378
Fig. 23.5 Transition in pushdown automata 378
Fig. 23.6 Transition function for pushdown automata M 379
Fig. 23.7 Turing machine 380
Fig. 23.8 Transition on Turing machine 381
Fig. 24.1 Concept of model checking 385
Fig. 24.2 Model checking 385
Fig. 24.3 Simple transition system 387
Fig. 24.4 LTL operators 389
Fig. 25.1 Carl Friedrich Gauss 404
Fig. 25.2 Standard unit normal bell curve (Gaussian distribution) 404
Fig. 25.3 Histogram test results 406

Fig. 25.4 Pie chart test results 407
Fig. 26.1 Argand diagram 412
Fig. 26.2 Interpretation of complex conjugate 414
Fig. 26.3 Interpretation of Euler's formula 415
Fig. 26.4 William Rowan Hamilton 417
Fig. 26.5 Plaque at Broom's Bridge 418
Fig. 26.6 Quaternions and rotations 423
Fig. 27.1 Limit of a function 426
Fig. 27.2 Derivative as a tangent to curve 426
Fig. 27.3 Interpretation of mean value theorem 427
Fig. 27.4 Interpretation of intermediate value theorem 428
Fig. 27.5 Isaac newton 430
Fig. 27.6 Wilhelm Gottfried Leibniz 430
Fig. 27.7 Local minima and maxima 432
Fig. 27.8 Area under the curve 434
Fig. 27.9 Area under the curve—lower sum 435
Fig. 27.10 Bisection method 438