## Contents (detailed)

### Preface

0.1 To the student ................................................. 21  
0.1.1 Is the effort worth it? .................................. 21  
0.2 To potential instructors ................................. 22  
0.3 Acknowledgements ...................................... 24

### Preliminaries

0 Introduction .................................................. 29  
0.1 Integer multiplication: an example of an algorithm ... 30  
0.2 Extended Example: A faster way to multiply (optional) 33  
0.3 Algorithms beyond arithmetic ............................ 37  
0.4 On the importance of negative results. .................. 38  
0.5 Roadmap to the rest of this book ....................... 39  
0.5.1 Dependencies between chapters ...................... 40  
0.6 Exercises ................................................... 42  
0.7 Bibliographical notes .................................... 44

### Mathematical Background

1 This chapter: a reader’s manual .......................... 47  
1.1 A quick overview of mathematical prerequisites ...... 48  
1.2 Reading mathematical texts .............................. 49  
1.3 Definitions .................................................. 50  
1.3.2 Assertions: Theorems, lemmas, claims ............ 50  
1.3.3 Proofs ................................................... 50  
1.4 Basic discrete math objects .............................. 51  
1.4.1 Sets ..................................................... 51  
1.4.2 Special sets ........................................... 52  
1.4.3 Functions .............................................. 54  
1.4.4 Graphs .................................................. 56  
1.4.5 Logic operators and quantifiers .................... 59  
1.4.6 Quantifiers for summations and products ........ 60  
1.4.7 Parsing formulas: bound and free variables ...... 60  
1.4.8 Asymptotics and Big-O notation .................. 62
### 1.4.9 Some “rules of thumb” for Big-O notation

- 1.5 Proofs
  - 1.5.1 Proofs and programs
  - 1.5.2 Proof writing style
  - 1.5.3 Patterns in proofs

- 1.6 Extended example: Topological Sorting
  - 1.6.1 Mathematical induction
  - 1.6.2 Proving the result by induction
  - 1.6.3 Minimality and uniqueness

- 1.7 This book: notation and conventions
  - 1.7.1 Variable name conventions
  - 1.7.2 Some idioms

- 1.8 Exercises

- 1.9 Bibliographical notes

### 2 Computation and Representation

- 2.1 Defining representations
  - 2.1.1 Representing natural numbers
  - 2.1.2 Meaning of representations (discussion)

- 2.2 Representations beyond natural numbers
  - 2.2.1 Representing (potentially negative) integers
  - 2.2.2 Two’s complement representation (optional)
  - 2.2.3 Rational numbers, and representing pairs of strings

- 2.3 Representing real numbers
  - 2.3.1 Can we represent reals exactly?

- 2.4 Representing objects beyond numbers
  - 2.4.1 Finite representations
  - 2.4.2 Prefix-free encoding
  - 2.4.3 Making representations prefix-free
  - 2.4.4 “Proof by Python” (optional)
  - 2.4.5 Representing letters and text
  - 2.4.6 Representing vectors, matrices, images
  - 2.4.7 Representing graphs
  - 2.4.8 Representing lists and nested lists
  - 2.4.9 Notation

- 2.5 Defining computational tasks as mathematical functions
  - 2.5.1 Distinguish functions from programs!

- 2.6 Exercises

- 2.7 Bibliographical notes

### I Finite computation

### 3 Defining computation
# Table of Contents

## 3.1 Defining computation .................................. 122
## 3.2 Computing using AND, OR, and NOT ..................... 124
  3.2.1 Some properties of AND and OR .................... 126
  3.2.2 Extended example: Computing XOR from AND, OR, and NOT ............. 126
  3.2.3 Informally defining “basic operations” and “algorithms” .......... 129
## 3.3 Boolean Circuits ........................................ 130
  3.3.1 Boolean circuits: a formal definition ................. 131
  3.3.2 Equivalence of circuits and straight-line programs .......... 134
## 3.4 Physical implementations of computing devices (di-
genresion) ................................................ 138
  3.4.1 Transistors ........................................... 138
  3.4.2 Logical gates from transistors ....................... 139
  3.4.3 Biological computing .................................. 139
  3.4.4 Cellular automata and the game of life ............ 139
  3.4.5 Neural networks ...................................... 140
  3.4.6 A computer made from marbles and pipes .......... 140
## 3.5 The NAND function ..................................... 141
  3.5.1 NAND Circuits ....................................... 142
  3.5.2 More examples of NAND circuits (optional) .......... 144
  3.5.3 The NAND-CIRC Programming language .......... 145
## 3.6 Equivalence of all these models ......................... 147
  3.6.1 Circuits with other gate sets ....................... 148
  3.6.2 Specification vs. implementation (again) .......... 149
## 3.7 Exercises .................................................. 151
## 3.8 Biographical notes ...................................... 153

## 4 Syntactic sugar, and computing every function .......... 155
  4.1 Some examples of syntactic sugar ....................... 156
    4.1.1 User-defined procedures ......................... 156
    4.1.2 Proof by Python (optional) ...................... 159
    4.1.3 Conditional statements ......................... 161
  4.2 Extended example: Addition and Multiplication (op-
tional) ................................................ 162
  4.3 The LOOKUP function ................................... 164
    4.3.1 Constructing a NAND-CIRC program for LOOKUP ............ 164
  4.4 Computing every function ................................ 166
    4.4.1 Proof of NAND’s Universality ..................... 167
    4.4.2 Improving by a factor of \( n \) (optional) .......... 169
  4.5 Computing every function: An alternative proof .......... 170
  4.6 The class \( \text{SIZE}(T) \) .................................. 172
  4.7 Exercises .................................................. 175
4.8 Bibliographical notes .................................. 179

5 Code as data, data as code 181
5.1 Representing programs as strings ................. 182
5.2 Counting programs, and lower bounds on the size of NAND-CIRC programs .......... 184
  5.2.1 Size hierarchy theorem (optional) .......... 186
5.3 The tuples representation .......................... 188
  5.3.1 From tuples to strings ...................... 189
5.4 A NAND-CIRC interpreter in NAND-CIRC ........ 190
  5.4.1 Efficient universal programs .............. 191
  5.4.2 A NAND-CIRC interpreter in “pseudocode” .... 192
  5.4.3 A NAND interpreter in Python .......... 194
  5.4.4 Constructing the NAND-CIRC interpreter in NAND-CIRC .................. 194
5.5 A Python interpreter in NAND-CIRC (discussion) ... 197
5.6 The physical extended Church-Turing thesis (discussion) .......... 199
  5.6.1 Attempts at refuting the PECTT ............ 201
5.7 Recap of Part I: Finite Computation .............. 206
5.8 Exercises ............................................ 207
5.9 Bibliographical notes ................................ 209

II Uniform computation 211

6 Loops and infinity 213
6.1 Turing Machines ...................................... 216
  6.1.1 Extended example: A Turing machine for palindromes ............... 217
  6.1.2 Turing machines: a formal definition ........ 219
  6.1.3 Computable functions .......................... 220
  6.1.4 Infinite loops and partial functions .......... 222
6.2 Turing machines as programming languages .......... 223
  6.2.1 The NAND-TM Programming language ......... 224
  6.2.2 Sneak peak: NAND-TM vs Turing machines .... 227
  6.2.3 Examples .......................................... 228
6.3 Equivalence of Turing machines and NAND-TM programs .......... 230
  6.3.1 Specification vs implementation (again) .... 233
6.4 NAND-TM syntactic sugar .......................... 234
  6.4.1 “GOTO” and inner loops ..................... 235
6.5 Uniformity, and NAND vs NAND-TM (discussion) .... 237
6.6 Exercises ............................................ 238
6.7 Bibliographical notes ................................ 241
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>7 Equivalent models of computation</td>
<td>243</td>
</tr>
<tr>
<td>7.1 RAM machines and NAND-RAM</td>
<td>244</td>
</tr>
<tr>
<td>7.2 The gory details (optional)</td>
<td>248</td>
</tr>
<tr>
<td>7.2.1 Indexed access in NAND-TM</td>
<td>248</td>
</tr>
<tr>
<td>7.2.2 Two dimensional arrays in NAND-TM</td>
<td>250</td>
</tr>
<tr>
<td>7.2.3 All the rest</td>
<td>250</td>
</tr>
<tr>
<td>7.3 Turing equivalence (discussion)</td>
<td>251</td>
</tr>
<tr>
<td>7.3.1 The “Best of both worlds” paradigm</td>
<td>252</td>
</tr>
<tr>
<td>7.3.2 Let’s talk about abstractions.</td>
<td>253</td>
</tr>
<tr>
<td>7.3.3 Turing completeness and equivalence, a formal definition (optional)</td>
<td>255</td>
</tr>
<tr>
<td>7.4 Cellular automata</td>
<td>256</td>
</tr>
<tr>
<td>7.4.1 One dimensional cellular automata are Turing complete</td>
<td>257</td>
</tr>
<tr>
<td>7.4.2 Configurations of Turing machines and the next-step function</td>
<td>258</td>
</tr>
<tr>
<td>7.5 Lambda calculus and functional programming languages</td>
<td>261</td>
</tr>
<tr>
<td>7.5.1 Applying functions to functions</td>
<td>262</td>
</tr>
<tr>
<td>7.5.2 Obtaining multi-argument functions via Currying</td>
<td>263</td>
</tr>
<tr>
<td>7.5.3 Formal description of the λ calculus.</td>
<td>263</td>
</tr>
<tr>
<td>7.5.4 Infinite loops in the λ calculus</td>
<td>266</td>
</tr>
<tr>
<td>7.6 The “Enhanced” λ calculus</td>
<td>266</td>
</tr>
<tr>
<td>7.6.1 Computing a function in the enhanced λ calculus</td>
<td>269</td>
</tr>
<tr>
<td>7.6.2 Enhanced λ calculus is Turing-complete</td>
<td>270</td>
</tr>
<tr>
<td>7.7 From enhanced to pure λ calculus</td>
<td>273</td>
</tr>
<tr>
<td>7.7.1 List processing</td>
<td>274</td>
</tr>
<tr>
<td>7.7.2 The Y combinator, or recursion without recursion</td>
<td>275</td>
</tr>
<tr>
<td>7.8 The Church-Turing Thesis (discussion)</td>
<td>278</td>
</tr>
<tr>
<td>7.8.1 Different models of computation</td>
<td>279</td>
</tr>
<tr>
<td>7.9 Exercises</td>
<td>280</td>
</tr>
<tr>
<td>7.10 Bibliographical notes</td>
<td>284</td>
</tr>
<tr>
<td>8 Universality and uncomputability</td>
<td>287</td>
</tr>
<tr>
<td>8.1 Universality or a meta-circular evaluator</td>
<td>288</td>
</tr>
<tr>
<td>8.1.1 Proving the existence of a universal Turing Machine</td>
<td>289</td>
</tr>
<tr>
<td>8.1.2 Implications of universality (discussion)</td>
<td>292</td>
</tr>
<tr>
<td>8.2 Is every function computable?</td>
<td>293</td>
</tr>
<tr>
<td>8.3 The Halting problem</td>
<td>295</td>
</tr>
<tr>
<td>8.3.1 Is the Halting problem really hard? (discussion)</td>
<td>297</td>
</tr>
<tr>
<td>8.3.2 A direct proof of the uncomputability of HALT (optional)</td>
<td>298</td>
</tr>
<tr>
<td>8.4 Reductions</td>
<td>300</td>
</tr>
</tbody>
</table>
10.5.2 Step 2: Reducing mixed statements to integer statements ........................................ 372
10.6 Exercises .................................................. 374
10.7 Bibliographical notes ........................................ 377

III Efficient algorithms ........................................ 379

11 Efficient computation ........................................ 381
  11.1 Problems on graphs ........................................ 383
    11.1.1 Finding the shortest path in a graph .................. 384
    11.1.2 Finding the longest path in a graph ................... 386
    11.1.3 Finding the minimum cut in a graph .................. 386
    11.1.4 Min-Cut Max-Flow and Linear programming .......... 387
    11.1.5 Finding the maximum cut in a graph ................. 389
    11.1.6 A note on convexity ................................... 389
  11.2 Beyond graphs ............................................ 391
    11.2.1 SAT .................................................. 391
    11.2.2 Solving linear equations ............................... 392
    11.2.3 Solving quadratic equations ............................ 393
  11.3 More advanced examples ................................. 393
    11.3.1 Determinant of a matrix .............................. 393
    11.3.2 Permanent of a matrix ................................ 395
    11.3.3 Finding a zero-sum equilibrium ....................... 395
    11.3.4 Finding a Nash equilibrium ............................ 396
    11.3.5 Primality testing ..................................... 396
    11.3.6 Integer factoring ..................................... 397
  11.4 Our current knowledge .................................... 397
  11.5 Exercises .................................................. 398
  11.6 Bibliographical notes ...................................... 398
  11.7 Further explorations ...................................... 399

12 Modeling running time ...................................... 401
  12.1 Formally defining running time ......................... 402
    12.1.1 Polynomial and Exponential Time ...................... 404
  12.2 Modeling running time using RAM Machines / NAND-RAM ......................... 406
  12.3 Extended Church-Turing Thesis (discussion) ............ 411
  12.4 Efficient universal machine: a NAND-RAM interpreter in NAND-RAM ......................... 412
    12.4.1 Timed Universal Turing Machine ....................... 414
  12.5 The time hierarchy theorem ............................. 415
  12.6 Non uniform computation ................................ 419
    12.6.1 Oblivious NAND-TM programs ......................... 420
19.1.1 An alternative view: random coins as an “extra input” 526
19.1.2 Success amplification of two-sided error algorithms 528
19.2 BPP and NP completeness 530
19.3 The power of randomization 531
19.3.1 Solving BPP in exponential time 531
19.3.2 Simulating randomized algorithms by circuits 532
19.4 Derandomization 533
19.4.1 Pseudorandom generators 534
19.4.2 From existence to constructivity 536
19.4.3 Usefulness of pseudorandom generators 537
19.5 P = NP and BPP vs P 538
19.6 Non-constructive existence of pseudorandom generators (advanced, optional) 541
19.7 Exercises 544
19.8 Bibliographical notes 544

V Advanced topics 545

20 Cryptography 547
20.1 Classical cryptosystems 548
20.2 Defining encryption 550
20.3 Defining security of encryption 551
20.4 Perfect secrecy 553
20.4.1 Example: Perfect secrecy in the battlefield 554
20.4.2 Constructing perfectly secret encryption 555
20.5 Necessity of long keys 556
20.6 Computational secrecy 558
20.6.1 Stream ciphers or the “derandomized one-time pad” 560
20.7 Computational secrecy and NP 563
20.8 Public key cryptography 565
20.8.1 Defining public key encryption 567
20.8.2 Diffie-Hellman key exchange 568
20.9 Other security notions 570
20.10 Magic 570
20.10.1 Zero knowledge proofs 571
20.10.2 Fully homomorphic encryption 571
20.10.3 Multiparty secure computation 572
20.11 Exercises 573
20.12 Bibliographical notes 573

21 Proofs and algorithms 575
21.1 Lecture summary ........................................ 575
21.2 Exercises ............................................... 575
21.3 Bibliographical notes ................................... 575
21.4 Further explorations ................................... 575
21.5 Acknowledgements ..................................... 575

22 Quantum computing ......................................... 577
22.1 The double slit experiment ............................... 578
22.2 Quantum amplitudes ................................... 578
  22.2.1 Linear algebra quick review ......................... 581
22.3 Bell’s Inequality .......................................... 582
22.4 Quantum weirdness ...................................... 584
22.5 Quantum computing and computation - an executive
  summary ................................................... 584
22.6 Quantum systems ........................................ 587
  22.6.1 Quantum amplitudes ............................... 588
  22.6.2 Quantum systems: an executive summary ........ 589
22.7 Analysis of Bell’s Inequality (optional) ............... 589
22.8 Quantum computation ................................... 592
  22.8.1 Quantum circuits .................................. 592
  22.8.2 QNAND-CIRC programs (optional) ............... 595
  22.8.3 Uniform computation .............................. 596
22.9 Physically realizing quantum computation ............. 597
22.10 Shor’s Algorithm: Hearing the shape of prime factors 598
  22.10.1 Period finding .................................... 598
  22.10.2 Shor’s Algorithm: A bird’s eye view .......... 599
22.11 Quantum Fourier Transform (advanced, optional) ... 602
  22.11.1 Quantum Fourier Transform over the Boolean
        Cube: Simon’s Algorithm ............................ 603
  22.11.2 From Fourier to Period finding: Simon’s Algo-
        rithm (advanced, optional) ....................... 605
  22.11.3 From Simon to Shor (advanced, optional) ....... 605
22.12 Exercises ............................................... 607
22.13 Bibliographical notes ................................... 607

VI Appendices .................................................. 609