Review of the book

"Theoretical Computer Science"

Introduction to Automata, Computability, Algorithmics,

Randomization, Communication and Cryptography

by Juraj Hromkovič

Springer, 2010

ISBN: 978-3-642-05729-8

Michael Daniel Samson
Nanyang Technological University, Singapore

November 10, 2014

1 The review in a nutshell

This is a review of the text "Theoretical Computer Science" by J. Hromkovič. As the subtitle of the text indicates, it is a broadly-scoped introductory text to theoretical computer science, mainly discussing the Turing machine model, with topics in computability theory and complexity theory, an overview of algorithmics for hard problems, finally choosing as examples some applications in network communication and public-key cryptography. In its three-hundred-page length, the text makes up for a lack of detail in parts with a focussed narrative, and a selection of illustrative applications.

The approach taken in the text is acknowledged in the Preface, as the text is meant to entice computer science majors that are disincentivized by both the amount of mathematics and the lack of application in theoretical computer science. Compared with other texts on theoretical computer science I have seen, the text is compact yet dense and on point with what are the core concepts that have shaped the direction in which computer science as a discipline has grown. There is still a fair amount of mathematics involved, but mostly explored only to the point that their use is highlighted.

The breadth of the work invites the interested reader to dig deeper into the topics covered, while maintaining a narrative which sets up the roughly chronological development of landmark ideas, such as Turing machines, the Church-Turing thesis and P versus NP within the context of algorithmic models, while addressing their broader significance. On the other hand, the text sacrifices details dealing with mathematical rigor in some of the less-significant aspects related to these contexts, choosing to focus on the main narrative and the selected applications. There are also a small but appropriate set of examples—anyone wanting to cut their teeth on the featured tools will likely want to continue with a more dedicated textbook.

2 Book summary

The Introduction in the first chapter provides a refreshingly informal yet precise outline of its goals: primarily, it is meant to influence the thinking of the computer science student to appreciate the devel-

opment of the science from its early formalism via automata and Turing machines, and to understanding the current limits of computation and some of the steps taken in addressing the problems that arise, in an effort to bridge theoretical and practical computation. This appreciation is supported well by the consistent narrative of the text.

The subtitle comprehensively lists the coverage of the book. The next five chapters cover the major topics of theoretical computer science:

- the second chapter establishes the basic ideas and measures used in computation theory, such as the concepts of languages and Kolmogorov complexity;
- the third chapter develops the mechanism of finite automata and introduces the important concept of nondeterministic computation;
- the fourth chapter covers Turing machines as a generalization of automata, highlighting the Church-Turing thesis;
- the fifth chapter discusses computability theory, defining decidable languages and outlining reduction techniques, leaning on the undecidability of the halting problem and the concept of the universal language;
- the sixth chapter discusses complexity theory, covering complexity measures and classes, elaborating on the crux of the P versus NP problem.

The final three chapters leverage the concepts in the previous chapters to develop an appreciation of the underlying difficulties that arise from practical computations:

- the seventh chapter introduces approaches to complexity-wise hard problems, which are untenable to solve in the general case, wherein weakening requirements or exploring reasonable (typical) subproblems are discussed as alternatives, and potentially exponential-time approaches such as local search and simulated annealing for optimization problems are outlined;
- the eighth chapter discusses some principles in designing randomized algorithms, and demonstrates their efficacy on primacy testing and polynomial testing;
- the last chapter discusses some topics in communication and cryptography, such as public-key encryption systems, interactive proof systems and Beneš networks.

Contrasting with other theoretical computer science books, which also devote some space to discussing general algorithm design techniques, the text stays on point with its focus on theoretical aspects, choosing application examples that highlight the difficulties on theoretical grounds, such as NP-hardness and established one-way functions.

3 Does the book successfully make theoretical CS more palatable for CS students?

The book delivers on its promise to cover the fundamentals of theoretical computer science with as encouraging a narrative as is established in the seeds of the early computational models, through to the development of complexity classes, all in the effort of developing a general framework for handling computational problems. It also tries to motivate the student to appreciate some of the consequences of the theories in appropriate applications. It is uncertain, though, that the math-averse student will feel less daunted by the mathematics found within the text.

Despite its efforts in making the mathematics required more palatable for its intended audience, like other theoretical computer science books, the text still relies on the power and precision of mathematical formalism to define its concepts—it does make the effort to clarify as much as it can through well-chosen examples, and the portions are paced well enough to allow time to take in new ideas and theorems.

One browsing the book off-hand may not see much difference in the density of mathematical expressions used from any other textbook, but the text couches its mathematics with some amount of preamble, and usually explains the common usage and significance to some satisfaction. One may be tempted to suggest moving the mathematical details to appendices, to declutter the main text of equations, but it would be counterproductive to hide the mechanisms of the mathematical framework when it is the text emphasizes the power of that framework.

However, it is noticeable that, where the text does not have to endorse the power of the mechanisms in the models, it can afford not to go too deep mathematically—but where the might of the mathematics is necessary to establish a point, such as in the proof of the existence of a language that is not recursively enumerable, the text establishes the scaffolding of mathematical concepts necessary to determine that that language exists, even if much of that scaffolding is not used elsewhere in the text. This is more a result of the fact that computer science relies on many mathematical principles, as broadly as those from set theory and probability theory to as (relative, in the text) tangential as those from number theory and group theory, as part of its theoretical toolkit.

To this end, overall, the text is a laudable effort to minimize the amount of formal mathematics in a theoretical computer science book.

4 Recommendations for readers

I heartily recommend the book to computer science students that are interested in the theoretical frameworks of the science. It is relatively lightweight compared other theoretical computer science books I have seen—this makes the text more of a companion tome to more detailed texts, or even to details that can be investigated using online sources. It serves as a good starting point, and a decent survey for those that are interested in the topics, or those interested in the context of topics such as automata and Turing machines, the halting problem, the basics of Kolmogorov complexity, complexity classes and P versus NP, languages in the computer-scientific context, even public-key cryptography and randomized algorithms.

Unlike other theoretical computer science books, it does not have a dedicated portion for algorithm design, and is not meant to be used for that purpose. It also does not cover lambda calculus, parallel computation, Petri nets, genetic algorithms, and so on, which can be found in some other theoretical computer science books.

5 Some comments on the book

I would like to point out that the proof of Theorem 5.20 can use the clarification that

$$w_i \notin L_{\text{diag}} \Leftrightarrow d_{ii} = 1 \Leftrightarrow w_i \in L(M_i),$$

although this is inferrable from the last line of the proof. Also, the figures in Section 9.6 are not appropriately numbered.

The reviewer is a Ph.D. holder in Mathematics assisting in classes at Nanyang Technological University, Singapore, and is a programming enthusiast with little formal training.