Preface

> The familiar essayist didn't speak to the millions; he spoke to *one* reader, as if the two of them were sitting side by side in front of a crackling fire with their cravats loosened, their favorite stimulants at hand, and a long evening of conversation stretching before them. His viewpoint was subjective, his frame of reference concrete, his style digressive, his eccentricities conspicuous, and his laughter usually at his own expense. And though he wrote about himself, he also wrote about a *subject*, something with which he was so familiar, and about which he was often so enthusiastic, that his words were suffused with a lover's intimacy.
>
> Anne Fadiman, *At Large and At Small*

> It is not incumbent upon you to finish the work,
> yet neither are you free to desist from it.
>
> Rabbi Tarfon

The sciences that awe and inspire us deal with fundamentals. Biology tries to understand the nature of life, from its cellular machinery to the gorgeous variety of organisms. Physics seeks the laws of nature on every scale from the subatomic to the cosmic. These questions are among the things that make life worth living. Pursuing them is one of the best things humans do.

The theory of computation is no less fundamental. It tries to understand why, and how, some problems are easy while others are hard. This isn't a question of how fast our computers are, any more than astronomy is the study of telescopes. It is a question about the *mathematical structures* of problems, and how these structures help us solve problems or frustrate our attempts to do so. This leads us, in turn, to questions about the nature of mathematical proof, and even of intelligence and creativity.

Computer science can trace its roots back to Euclid. It emerged through the struggle to build a foundation for mathematics in the early 20th century, and flowered with the advent of electronic computers, driven partly by the cryptographic efforts of World War II. Since then, it has grown into a rich field, full of deep ideas and compelling questions. Today it stands beside other sciences as one of the lenses we use to look at the world. Anyone who truly wants to understand how the world works can no more ignore computation than they can ignore relativity or evolution.

Computer science is also one of the most flexible and dynamic sciences. New subfields like quantum computation and phase transitions have produced exciting collaborations between computer scientists, physicists, and mathematicians. When physicists ask what rules govern a quantum system, computer scientists ask what it can compute. When physicists describe the phase transition that turns water to ice, computer scientists ask whether a similar transition turns problems from easy to hard.

This book was born in 2005 when one of us was approached by a publisher to write a book explaining computational complexity to physicists. The tale grew in the telling, until we decided—with some hubris—to explain it to everyone, including computer scientists. A large part of our motivation was to write the book we would have liked to read. We fell in love with the theory of computation because of the beauty and power of its ideas, but many textbooks bury these ideas under a mountain of formalism. We have not hesitated to present material that is technically difficult when it's appropriate. But at every turn we have tried to draw a clear distinction between deep ideas on the one hand and technical details on the other—just as you would when talking to a friend.

Overall, we have endeavored to write our book with the accessibility of Martin Gardner, the playfulness of Douglas Hofstadter, and the lyricism of Vladimir Nabokov. We have almost certainly failed on all three counts. Nevertheless, we hope that the reader will share with us some of the joy and passion we feel for our adopted field. If we have reflected, however dimly, some of the radiance that drew us to this subject, we are content.

We are grateful to many people for their feedback on the draft, and for their guidance on the topics in the book: Scott Aaronson, Heiko Bauke, Paul Chapman, Andrew Childs, Aaron Clauset, Varsha Dani, Josep Díaz, Owen Densmore, Irit Dinur, Ehud Friedgut, Tom Hayes, Robert Hearn, Stefan Helmreich, Reuben Hersh, Shiva Kasiviswanathan, Brian Karrer, David Kempe, Greg Kuperberg, Cormac McCarthy, Sarah Miracle, John Moore, Michel Morvan, Larry Nazareth, Mark Olah, Jim Propp, Dana Randall, Sasha Razborov, Omer Reingold, Paul Rendell, Sara Robinson, Jean-Baptiste Rouquier, Alex Russell, Jared Saia, Nicolas Schabanel, Cosma Shalizi, Thérèse Smith, Darko Stefanović, John Tromp, Vijay Vazirani, Robin Whitty, Lance Williams, Damien Woods, Jon Yard, Lenka Zdeborová, Yaojia Zhu, and Katharina Zweig.

For additional comments, we are also grateful to Lee Altenberg, László Babai, Nick Baxter, Marcus Calhoun-Lopez, Timothy Chow, Nathan Collins, Will Courtney, Zheng Cui, Wim van Dam, Aaron Denney, Hang Dinh, Taylor Dupuy, Bryan Eastin, Charles Efferson, Leigh Fanning, Steve Flammia, Matthew Fricke, Benjamin Gordon, Stephen Guerin, Samuel Gutierrez, Russell Hanson, Neal Holtschulte, Peter Høyer, Luan Jun, Valentine Kabanets, Richard Kenyon, Jeffrey Knockel, Leonid Kontorovich, Maurizio Leo, Phil Lewis, Scott Levy, Chien-Chi Lo, Jun Luan, Shuang Luan, Jon Machta, Jonathan Mandeville, Bodo Manthey, Sebastian Mingramm, Brian Nelson, ThanhVu Nguyen, Katherine Nystrom, Olumuyiwa Oluwasanmi, Boleszek Osinski, John Patchett, Robin Pemantle, Yuval Peres, Carlos Riofrio, Tyler Rush, Navin Rustagi, George Helmy Saad, Amin Saberi, Gary Sandine, Samantha Schwartz, Oleg Semenov, David Sherrington, Satomi Sugaya, Bert Tanner, Amitabh Trehan, Yamel Torres, David Wilson, Chris Wood, Ben Yackley, Yiming Yang, Sheng-Yang Wang, and Zhan Zhang. We apologize for any omissions from this list.

We express our heartfelt thanks to the Santa Fe Institute, without whose hospitality we would have been unable to complete this work. In particular, the SFI library staff—Margaret Alexander, Tim Taylor, and Joy LeCuyer—fulfilled literally hundreds of requests for articles and books.

We are grateful to our editor Sönke Adlung for his patience, and to Alison Lees for her careful copyediting. Mark Newman gave us invaluable help with the LaTeX Memoir class, in which this book is typeset, along with insights and moral support from his own book-writing experiences. And throughout the process, Alex Russell shaped our sense of the field, separating the wheat from the chaff and helping us to decide which topics and results to present to the reader. Some fabulous monsters didn't make it onto the ark, but many of those that did are here because he urged us to take them on board.

Finally, we dedicate this book to Tracy Conrad and Doro Frederking. Our partners, our loves, they have made everything possible.

Cristopher Moore and Stephan Mertens
Santa Fe and Magdeburg, 2011

How to read this book

Outside a dog a book is a man's best friend.
Inside a dog it's too dark to read.

Groucho Marx

We recommend reading Chapters 1–7 in linear order, and then picking and choosing from later chapters and sections as you like. Even the advanced chapters have sections that are accessible to nearly everyone.

For the most part, the only mathematics we assume is linear algebra and some occasional calculus. We use Fourier analysis and complex numbers in several places, especially Chapter 11 for the PCP Theorem and Chapter 15 on quantum computing. Mathematical techniques that we use throughout the book, such as asymptotic notation and discrete probability, are discussed in the Appendix. We assume some minimal familiarity with programming, such as the meaning of **for** and **while** loops.

Scattered throughout the text you will find Exercises. These are meant to be easy, and to help you check whether you are following the discussion at that point. The Problems at the end of each chapter delve more deeply into the subject, providing examples and fleshing out arguments that we sketch in the main text. We have been generous with hints and guideposts in order to make even the more demanding problems doable.

Every once in a while, you will see a quill symbol in the margin—yes, like that one there. This refers to a note in the Notes section at the end of the chapter, where you can find details, historical discussion, and references to the literature.

A note to the instructor

We have found that Chapters 1–8, with selections from Chapters 9–11, form a good text for an introductory graduate course on computational complexity. We and others have successfully used later chapters as texts or supplementary material for more specialized courses, such as Chapters 12 and 13 for a course on Markov chains, Chapter 14 for phase transitions, and Chapter 15 for quantum computing. Some old-fashioned topics, like formal languages and automata, are missing from our book, and this is by design.

The Turing machine has a special place in the history of computation, and we discuss it along with the λ-calculus and partial recursive functions in Chapter 7. But we decided early on to write about computation as if the Church-Turing thesis were true—in other words, that we are free to use whatever model of computation makes it easiest to convey the key ideas. Accordingly, we describe algorithms at a "software" level, as programs written in the reader's favorite programming language. This lets us draw on the reader's experience and intuition that programs need time and memory to run. Where necessary, such as in our discussion of LOGSPACE in Chapter 8, we drill down into the hardware and discuss details such as our model of memory access.

Please share with us your experiences with the book, as well as any mistakes or deficiencies you find. We will maintain errata at nature-of-computation.org. We can also provide a solution manual on request, which currently contains solutions for over half the problems.