The Book Review Column [1]

by **Nicholas Tran** (ntran@scu.edu)
Department of Mathematics & Computer Science, Santa Clara University

1 Notable new releases

When Least Is Best: How Mathematicians Discovered Many Clever Ways to Make Things as Small (or as Large) as Possible (2021 ed.) by Paul J. Nahin updates a historical but highly technical account of "what many brilliant mathematicians have done in the subject of extrema over the last two dozen centuries."

Number Theory and Combinatorics by Bruce Landman et al. (Eds.) is a collection of 20 articles by leading experts dedicated to the life and memory of the late mathematician Ron Graham.

Mathematical Analysis of Machine Learning Algorithms by Tong Zhang presents a systematic treatment of the mathematical techniques for analyzing supervised learning and sequential decision making problems.

2 This column

Jonathan Katz enthusiastically recommends *Beyond the Worst-Case Analysis of Algorithms* by Tim Roughgarden, a collection of 29 surveys of different approaches to explaining empirically excellent performance of algorithms with poor worst-case guarantees, but he cautions that the book is not written with undergraduates in mind.

The textbook *Connecting Discrete Mathematics and Computer Science* by David Liben-Nowell seeks to demonstrate through examples the relevance of discrete mathematics to computer science. David Luginbuhl says that it succeeds admirably in his review. The author has kindly made freely available for personal use a pre-publication PDF version of this book on his website `https://cs.carleton.edu/faculty/dln/book`.

Using science fiction stories as case studies to explore ethical issues in technology is a novel approach taken by *Computing and Technology Ethics: Engaging through Science Fiction* by Emanuelle Burton, Judy Goldsmith, Nicholas Mattei, Cory Siler, and Sara-Jo Swiatek. In his review Brian Patrick Green calls this engaging book "a tremendous gift to the scholarly community teaching technology ethics and especially computing ethics," and he suggests adding a more careful discussion of moral relativism to emphasize the absolute necessity of good ethical decision-making.

Finally, Bill Gasarch has a retrospective advice for the current owner of Twitter (now X) in his review of *Should You Believe Wikipedia? Online Communities and the Construction of Knowledge* by Amy S. Bruckman. He says that this "intelligent" book on online communities is especially timely but goes too easy on the challenges plaguing communication on these platforms.

[1] ©2023 Nicholas Tran

3 How to contribute

Please contact me to write a review! Either choose from the books listed on the next page, or propose your own. In either case, the publisher will send you a free copy of the book. Guidelines and a LaTeX template can be found at `https://algoplexity.com/~ntran`.

Pearls of ALGORITHM ENGINEERING
Paolo Ferragina
AUTOMATA THEORY
An Algorithmic Approach
TOPICS IN ALGORITHMIC GRAPH THEORY
Introduction to Proofs and Proof Strategies
Shay Fuchs
NUMBER THEORY AND COMBINATORICS
COMPUTABILITY AND COMPLEXITY
HUBIE CHEN
Algorithms and Data Structures
WHEN LEAST IS BEST
Mathematical Foundations of Data Science
200 Problems on Languages, Automata, & Computation
Variational Bayesian Learning Theory
Introduction to Probability for Computing
Mor Harchol-Balter
Hardness of Approximation Between P and NP
Sequential and Parallel Algorithms and Data Structures
The Basic Toolbox
Computational Bayesian Statistics
An Introduction
Information-theoretic Cryptography
Himanshu Tyagi AND Shun Watanabe
MATHEMATICAL ANALYSIS OF MACHINE LEARNING ALGORITHMS
TONG ZHANG

BOOKS THAT NEED REVIEWERS FOR THE SIGACT NEWS COLUMN

Algorithms, Complexity, & Computability

1. Rubinstein, A. (2019). *Hardness of Approximation between P and NP* (ACM Books). Morgan & Claypool.
2. Knebl, H. (2020). *Algorithms and Data Structures: Foundations and Probabilistic Methods for Design and Analysis.* Springer.
3. Murlak, F., Niwiński D., & Rytter, W. (2023). *200 Problems on Languages, Automata, and Computation.* Cambridge University Press.
4. Chen, H. (2023). *Computability and Complexity.* MIT Press.
5. Ferragina. P. (2023). *Pearls of Algorithm Engineering.* Cambridge University Press.
6. Esparza, J., & Blondin, M. (2023). *Automata Theory: An Algorithmic Approach.* The MIT Press.

Miscellaneous Computer Science & Mathematics

1. Nahin, P. (2021). *When Least Is Best: How Mathematicians Discovered Many Clever Ways to Make Things as Small (or as Large) as Possible.* Princeton University Press.

Data Science

1. Amaral Turkman, M., Paulino, C., & Müller, P. (2019). *Computational Bayesian Statistics: An Introduction* (Institute of Mathematical Statistics Textbooks). Cambridge University Press.
2. Nakajima, S., Watanabe, K., & Sugiyama, M. (2019). *Variational Bayesian Learning Theory.* Cambridge University Press.
3. Hrycej, T., Bermeitinger, B., Cetto, M., & Handschuh, S. (2023). *Mathematical Foundations of Data Science.* Springer.
4. Zhang, T. (2023) *Mathematical Analysis of Machine Learning Algorithms.* Cambridge University Press.

Discrete Mathematics and Computing

1. Harchol-Balter, M. (2023). *Introduction to Probability for Computing.* Cambridge University Press.
2. Fuchs, S. (2023). *Introduction to Proofs and Proof Strategies.* Cambridge University Press.

Cryptography and Security

1. Tyagi, H., & Watanabe, S. (2023). *Information-Theoretic Security.* Cambridge University Press.

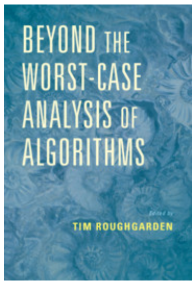
Combinatorics and Graph Theory

1. Beineke, L., Golumbic, M., & Wilson, R. (Eds.). (2021). *Topics in Algorithmic Graph Theory* (Encyclopedia of Mathematics and its Applications). Cambridge University Press.
2. Landman, B., Luca, F., Nathanson, M., Nešetřil, J., & Robertson, A. (Eds.). (2022). *Number Theory and Combinatorics: A Collection in Honor of the Mathematics of Ronald Graham.* De Gruyter.

Programming etc.

1. Sanders, P., Mehlhorn, K., Dietzfelbinger, M., & Dementiev, R. (2019). *Sequential and Parallel Algorithms and Data Structures: The Basic Toolbox.* Springer.

Review of [1]



Beyond the Worst-Case Analysis of Algorithms
edited by **Tim Roughgarden**

Cambridge University Press, 2021
USD 68.99, Hardcover, 704 pages

Reviewed by **Jonathan Katz**
Dept. of Computer Science, University of Maryland

1 Overview

When we traditionally evaluate algorithms, we do so based on their performance on *all possible inputs*. That is, we expect an algorithm for some problem to solve that problem for every possible input, and we measure the efficiency of the algorithm by looking at its performance on all inputs of a given size. This *worst-case* approach is the perspective taught to students when they are first introduced to programming, throughout undergraduate algorithms courses, and in most of complexity theory. And it is the viewpoint typically taken even when considering more "advanced" algorithms like randomized algorithms (which should solve the problem with high probability for any input), approximation algorithms (which should approximate the solution for any input), or online algorithms (which must be competitive with the best offline algorithm for all possible inputs).

Yet there are several examples of algorithms that have very good performance in practice even though their worst-case guarantees are quite poor. (Some examples are described below.) Is it possible to develop theoretical frameworks "beyond worst-case analysis" that can justify the use of such algorithms or help develop improved algorithms with superior real-world performance?

The present volume, edited by Tim Roughgarden, contains 29 surveys of different approaches for addressing exactly this question. Each survey is written by an expert (or experts) in a particular domain, and taken together they provide an approachable and thorough introduction to the diverse techniques for going beyond worst-case analyses of algorithms. Most of the surveys also contain extensive references for additional reading.

The Introduction, written by Tim Roughgarden, provides a robust defense (in case one is needed!) of the value of going beyond worst-case analyses. In particular, Roughgarden provides four examples of instances where worst-case analysis fails to adequately explain real-world performance:

1. In the context of linear programming, the simplex method does well in practice but may run for exponential time on certain inputs. But the inputs on which it runs for exponential time are "pathological"—indeed, even finding a sequence of inputs that forces the algorithm to run for exponential time was an open research question in the 1970s. In contrast, the ellipsoid method has a polynomial-time bound on its worst-case running time but performs

[1] ©2023 Jonathan Katz

much worse than the simplex method in practice. What theoretical approach[2] can be used to justify using the simplex method instead of the ellipsoid method?

2. A worst-case analysis of online algorithms (which must operate in an "online" fashion as inputs arrive, with incomplete knowledge of future inputs) for page caching shows that on certain input sequences the "least recently used" (LRU) policy can result in a 100% cache-miss rate. Yet the LRU policy is the one favored by systems architects, due to the empirical observation that it performs well on most real-world access sequences. How can the practical effectiveness of the LRU-based algorithm be accounted for?

3. Clustering (under most objective functions that people care about in practice) is NP-hard. Yet in practice there are several algorithms that output good clusterings when run on real-world data. How can this discrepancy be explained?

4. The real-world success of neural networks for solving supervised learning problems is hard to ignore. But it is difficult to provide theoretical explanations of both the observed convergence time of gradient-descent methods, as well as the effectiveness of overparameterized neural networks in correctly labeling unobserved data. What guidance can theorists offer here?

I hope the reader will enjoy coming up with their own solutions the above conundrums before reading some possible answers below!

2 Summary of the Book

The book itself, consisting of the remaining 29 chapters, is divided into six parts, four which can be said to cover "techniques" and two containing surveys focusing on particular "applications." A non-exhaustive list of the chapters in the different parts follows.

Part One. The first part of the book can roughly be said to deal with algorithmic analyses that still follow a worst-case approach, but with generalized ways of measuring efficiency (beyond merely expressing it as a function of the input size). This includes a chapter on *parameterized algorithms*, where the idea is to measure the running time of algorithms based on some parameter k of the input (and not only the input size), a chapter on *instance-optimal algorithms* that (up to a constant multiplicative factor) perform better than any other algorithm on every input, and a chapter on *resource augmentation* where, roughly, the performance of an algorithm using some amount of resources is compared to an optimal algorithm using fewer resources.

Part Two. The next part of the book can be viewed as focusing on meaningful ways to constrain the set of inputs for which an algorithm is analyzed (i.e., either the algorithm does not need to be correct for inputs outside some set of interest, or the running time of the algorithm for those inputs is not considered); this allows for excluding "pathological" inputs on which an algorithm might do poorly. This part has three chapters. The first two constrain the input space by restricting attention to *stable* inputs, under different notions of stability—one based on properties of the input

[2]Of course, one might argue that no theoretical justification is needed if we have empirical evidence that the simplex method is better. This is certainly unsatisfying to a theorist who wants to "explain" the observed behavior. More generally, taking that stance provides no assurance to a user that the inputs on which they run the simplex algorithm will not be exactly those pathological inputs that lead to a long running time, and it gives no guidance to researchers for developing improvements to the simplex method.

instances themselves, and the other based on properties of the solutions to those instances. The third chapter restricts attention to algorithms that are tailored for *sparse instances* in the context of streaming algorithms, compressed sensing, and linear algebra.

Part Three. The chapters in the third part of the book consider what might be called "average-case analysis," where there is some distribution on inputs and algorithms are measured by their expected performance rather than their worst-case performance. This part includes a chapter on *distributional analysis*, where a simple distribution over inputs is assumed, as well as a chapter on *planted models*, where an input is chosen from some distribution but that input can then be modified in a (restricted) adversarial fashion. A counterpart to planted models considered in another chapter is the *random-order model*, where an adversary first specifies an input and then that input is randomly shuffled before being given to an algorithm. (That model makes the most sense in the context of online algorithms.) This part concludes with a chapter on *self-improving algorithms* that do not know the input distribution in advance, but are instead supposed to "learn" how to perform well for the (unknown) input distribution as they are presented with more and more input samples drawn from that distribution.

Part Four. The next part of the book focuses on *smoothed analysis.* Roughly, smoothed analysis considers a worst-case selection of the input, but assumes the input is randomly perturbed before it is given to an algorithm. This is a compelling framework in settings where input instances may not be exact, e.g., they may be based on real numbers obtained from physical measurements that anyway have some inherent imprecision or uncertainty. While the three chapters here could also fit into the previous part, the field of smoothed analysis has become important enough to justify giving these chapters their own dedicated section of the book.

Part Five. The final two parts of the book look at various applications that can benefit from analytical approaches to algorithms that go beyond a worst-case analysis. Part Five considers applications in the areas of machine learning and statistics. The first two chapters look at solving machine-learning and statistical problems in the presence of (adversarial) *noise.* Later chapters examine conditions under which stochastic gradient descent and overparameterized models have provable guarantees. Other chapters focus on *nearest-neighbor search and classification*, computing *low-rank tensor decompositions*, and *instance-optimal algorithms for distribution testing/learning.* While several of these chapters could have fit in previous parts based on the techniques being used, it makes sense to put them together in Part Five since they all related to learning.

Part Six. The final section consists of a "grab bag" of applications of some of the techniques from earlier parts. There is a chapter on alternatives to worst-case competitive analysis of online algorithms, an important chapter on SAT-solving algorithms (which work well in practice even as they solve instances of an NP-complete problem), a chapter on auctions from the perspective of algorithmic game theory, and one on algorithms for restricted classes of graphs suitable for modeling social networks.

2.1 Some Solutions to the Motivating Problems

Based on the above outline, the reader may already be able to guess some possible solutions to the motivating problems discussed at the beginning of this review. The book proposes the following (of course, other solutions may exist!):

1. Arguably one of the greatest successes of smoothed analysis was its application to analysis of the running time of the simplex method. (Spielman and Teng received the Gödel prize in 2008 for that result.) Roughly, it can be shown that the simplex method has polynomial smoothed complexity with respect to Gaussian perturbations of the input. This can explain why the simplex algorithm runs quickly on inputs encountered in the real world.

2. The LRU page-caching algorithm can be profitably analyzed using a parameterized approach where performance is measured as a function of the *locality* of the input sequence. (Locality measures how many distinct pages can be requested in a given number of consecutive requests.) Already in Chapter 1, it is shown that the LRU algorithm is essentially optimal when measured that way.

3. Although clustering is NP-hard, the hardest instances are essentially those that don't cluster well in the first place; for such instances, clustering is not an appropriate goal, anyway. As Roughgarden puts it in Chapter 1, "clustering is hard only when it doesn't matter." Conversely, then, clustering algorithms tend to perform well when they are applied to problems for which clustering makes sense as a solution concept.

4. Approaches for explaining the real-world performance of supervised learning algorithms are explored in Part Five. Roughly, gradient descent can be shown to perform well for inputs satisfying a natural criterion that typically holds in practice. Understanding the success of neural networks is currently an active area of research, but some plausible explanations are discussed in the chapter on overparameterized models.

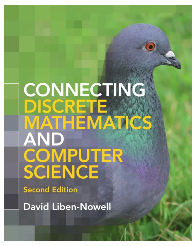
3 Evaluation

I enjoyed reading this book, and enthusiastically recommend it for graduate students or researchers looking for an introduction to this active area of research. Overall, the quality of the chapters is high, the topics covered are compelling, and good pointers to the literature are given. While the book does suffer from some of the drawbacks of most edited volumes (e.g., inconsistent notation, different styles, a relative lack of integration between different chapters), it is clear that some effort was made to alleviate these issues to the extent possible.

I don't work in the field of algorithms, but I do teach undergraduate algorithms from time to time. In reading the book, it struck me that a good chunk of the material here could be taught at the undergraduate level, and might spark interest in students (and even instructors) bored by the well-trodden algorithms and applications that constitute a typical course in algorithms. While the current book could be used by a instructor (with significant effort) to develop undergraduate lecture materials or as part of a graduate course, it would be a challenge to do so because the book is simply not written with that goal in mind. (And I would not recommend assigning it to your average undergraduate student, though perhaps a sharp, theoretically inclined student could make some headway.) Perhaps the next generation of algorithms textbooks will incorporate some of the material covered in this book in a more-accessible manner suitable for undergraduates.

Review of [1]

Connecting Discrete Mathematics and Computer Science
by **David Liben-Nowell**



Cambridge University Press, 2022
Hardcover/eBook, 674 pages, 2nd ed., $74.99

Reviewed by **David Luginbuhl**
(`dluginbu@samford.edu`)
Dept. of Mathematics and Computer Science, Samford University

1 Overview

For those of us who teach discrete math in a computer science program, one of the challenges is making the topics relevant to computer science. I often find students struggle with this: "Why am I taking another math class, and why this one in particular?" David Liben-Nowell, in *Connecting Discrete Mathematics and Computer Science*, agrees: "Computer science students taking a class like this one sometimes don't see why this material has anything to do with computer science — particularly if you enjoy CS because you enjoy programming" (p. 2).

As the title of his text suggests, Liben-Nowell intends to tackle relevancy head-on. It covers most topics one would expect in a course in discrete math – logic, proofs, counting, and graphs and trees, to name a few – but it also weaves in content, examples, and additional features that can assist an instructor in establishing a "connection" to computer science.

2 Summary of Contents

Each chapter has several common features that are designed to help students see relevancy to computer science or reinforce the current topic:

- A *Why You Might Care* section introduces each chapter and provides hooks to computing topics that will help motivate computer science students.
- There are smaller print notes sometimes labeled *Problem-solving tip* or *Taking it further* embedded throughout the text that provide clarification or additional useful information. Some of these also allow interested students to dig deeper into the concept being covered.
- The *Computer Science Connections* at the end of each section provide immediate relevance and help keep the topic grounded in computer science applications. I could see assigning these as mini-research projects, where students explore the topic in further detail and either write a short paper or provide a brief presentation to the rest of the class (depending on class size, of course). I will highlight some of these in the *Chapter Highlights* below.

[1]©2023 David Luginbuhl

- The summaries in the last section of each chapter (*Chapter at a Glance*), organized by subsection, provide a valuable study tool and a way to query students in class to see if they understand what they've been reading.

Below I provide a summary and some thoughts on each of the twelve chapters (an intro and closing chapter, and ten devoted to various discrete math topics).

3 Chapter Highlights

Chapter 1 *On the Point of this Book* – I am impressed by the readability of this first chapter. In three pages, Liben-Nowell does a good job of describing the purpose of the book, previewing the features discussed above, showing the interconnection of the chapters, and ending with "three very reasonable ways to think about this book" ("mathematical foundations," "practice," and "application"). It is not long – only three pages. I would probably assign it with a brief reflection essay as homework, asking students what they anticipate most about this course having read this chapter.

Chapter 2 *Basic Data Types* – A popular starting point for discrete math is logic, which make sense – a solid logical foundation is necessary for thinking through all the other topics in this broad area. The challenge with logic is that it usually involves new terminology, new notation, and formality that students may be uncomfortable with.

Liben-Nowell takes a different approach: he starts with data types. The advantage is that this is familiar territory for students in a standard computer science curriculum, so it might be seen as a more gentle introduction to the world of discrete math for computer science students.

The chapter starts with scalar types (Boolean, integers, reals, and associated operations). It then moves to sets, then ordered collections (such as sequences and matrices), then to functions.

For this chapter, *Computer Science Connections* include representation of integers and reals in the machine, algorithms for computing square root, and MapReduce (for processing collections).

Chapter 3 *Logic* – This chapter includes sections on propositional logic, predicate logic, and nested quantifiers. There is also a section on "extensions" to propositional logic, where we are introduced to concepts like tautologies and satisfiability, as well as circuits and normal forms.

The *Computer Science Connections* in this chapter include fuzzy logic in natural language computing and short-circuit evaluation of logical connectives in programming languages.

Chapter 4 *Proofs* – The *Why You Might Care* section of this chapter provides a solid connection to computer science, explaining, for example, that proofs are useful for:

- demonstrating one algorithm is more efficient than another
- finding bugs in programs through program proving
- understanding the concept of computability (e.g., the halting problem)

The chapter begins with "an extended exploration of error-correcting codes" as an example for showing how to apply various types of proof techniques. This provides an appropriate application area (certainly relevant to computer science) to introduce the techniques that will be elaborated on in the remainder of the chapter. For instructors who would rather not cover the additional material on error-correcting codes, this section is fairly self-contained, and there are not many back-references to it in the rest of the chapter.

There are numerous examples in the following sections demonstrating the various proof techniques, many drawn from propositional logic. The final section on errors is very helpful and provides exercises where students have to determine whether or not a proof is valid.

Computer Science Connections for proofs include the Four-Color Theorem and some of the controversy surrounding using a computer to prove a mathematical result and "Some Famous Bugs in CS" including the Pentium bug and the Therac-25 tragedy.

Chapter 5 ***Mathematical Induction*** – Liben-Nowell begins this chapter on induction by drawing a connection to (not surprisingly) recursion. Technical sections include an introduction to both weak and strong induction.

I like his informal introduction to proof by induction: "to prove a statement $P(n)$ is true for all nonnegative integers n, we can prove that P 'starts being true' (the *base case*) and that P 'never stops being true' (the *inductive case*)" (p. 217). He uses the analogy of dominoes falling as another informal way to explain induction. This would be a terrific way to introduce the concept of induction, using a physical demonstration.

The final section is on *Recursively Defined Structures and Structural Induction.* Examples here include linked lists and binary trees, so the connection to computer science is direct and should be easily picked up by students.

Computer Science Connections for this chapter include using loop invariants for program proving and structural induction on parse trees.

Chapter 6 ***Analysis of Algorithms*** – This chapter is a comprehensive introduction to algorithmics, including a detailed discussion of big Oh, big Omega, and big Theta, analysis of searching and sorting algorithms, and recurrence relations for analyzing recursive algorithms.

Computer Science Connections for this chapter on algorithms include discussions of Moore's Law (to illustrate exponential growth) and why we count *operations* in algorithm analysis rather than wall clock time.

Chapter 7 ***Number Theory*** – With the rise in importance of cybersecurity, it is only natural to see more emphasis on number theory. Liben-Nowell addresses this at the beginning of the chapter: "more so than any other chapter of the book, the technical material in this chapter leads directly to a single absolutely crucial (and ubiquitous) modern application of computer science: *cryptography* ..." (p. 328).

The chapter provides a relatively deep dive into the theory behind the RSA encryption algorithm, but I think it is also possible to avoid some of the longer in-depth proofs in this chapter and still cover the critical concepts.

It starts with modular arithmetic, then moves to prime numbers (and what it means to be relatively prime). Following this is a discussion of multiplicative inverses in modular arithmetic. The chapter culminates in a section on cryptography, and in particular, the RSA encryption algorithm. The treatment of RSA is quite comprehensive and includes proofs of correctness.

Computer Science Connections include another example of error-correcting codes (this time with Reed-Solomon) and an exposition on the Diffie-Hellman key exchange protocol.

For this chapter, more than some of the others, it might be good to start at the end, with the *Chapter at a Glance* section. It provides a helpful TL;DR explanation that ties everything together. From there, the instructor (and students) could back up to the beginning of the chapter and work forward.

Chapter 8 *Relations* – This chapter is a straightforward introduction to relations, covering inverse relations, composition, properties such as reflexivity and transitivity, and equivalence relations and partial and total orders. There is a subsection on asymptotics, which will be relevant if an instructor chooses to cover the immediately preceding algorithms chapter.

Computer Science Connections for this chapter cover deterministic finite automata (where each state represents an equivalence class) and the Painter's Algorithm used for hidden-surface removal in computer graphics (as an example of building a partial order).

Chapter 9 *Counting* – The chapter on counting covers the usual ground: counting rules (Sum, Product, Inclusion-Exclusion, and Generalized Product), rules for transforming from a set that is easier to count (Mapping and Division), and combinations and permutations.

Computer Science Connections in this chapter include switching from IP addresses to IPv6 addresses (how many more addresses are made possible by moving from a 32-bit sequence to a 128-bit sequence?) and the Enigma machine (alphabetic permutations).

Chapter 10 *Probability* – This is a detailed treatment of probability and randomness. The *Why You Might Care* section describes some relatable connections to computer science: randomized algorithms for solving certain kinds of problems, hashing, and creating realism in graphics with randomness. In fact, Liben-Nowell uses hashing as a "Running Example" (he provides a brief introduction to hashing in the first section, and comes back to it throughout the rest of chapter).

The next section introduces basic concepts and definitions (outcomes, probability, events) and provides some standard examples (cards, coin flips, words on a page). The use of tree diagrams to explain probability is helpful here, with examples that involve hashing and the Monty Hall problem.

Subsequent sections cover independence, conditional probability, random variables, and expectation. These sections are detailed and quite thorough. If students in your program are not going to be otherwise exposed to probability, this chapter would provide a good introduction to the topic.

One *Computer Science Connection* in this chapter that I think is particularly motivating for computer science students involves using randomness to fuzz data for privacy protection.

Chapter 11 *Graphs and Trees* – The last technical chapter of the text introduces graph theory, covering directed and undirected graphs, connectivity, isomorphisms, trees traversals, and Dijsktra's and Kruskal's algorithms. There are plenty of relevant examples throughout the chapter.

I thought the discussion of tree traversals was especially good, and the examples that demonstrate the difference between pre-, post-, and inorder traversals should be very easy for students to follow.

The *Computer Science Connections* in this section are particularly apropos, including graph drawing in computer graphics, the way some programming languages use reachability of nodes for garbage collection algorithms, and the use of random walks in page rank algorithms.

Chapter 12 *Looking Forward* – This final chapter touches on the impact of computer science to society and the importance of understanding how our choices as computing professionals can affect those around us. One way to effectively make use of this small (2.5 pages) concluding chapter is for an instructor to look for links to societal impact throughout the book (some are provided in this chapter) and emphasize them throughout the course so they can be reinforced here at the end.

4 Summary

Liben-Nowell delivers on his promise to "Connect Discrete Math with Computer Science." There are plenty of computing-related examples peppered throughout each chapter, and the *Computer Science Connections* at the end of each section make this even more explicit. Starting out with data types helps establish computer science relevancy as well.

Beyond that, the summaries and key terms provided at the end of each chapter will be useful for students in assuring they know what they need to understand.

Each section ends with many exercises at various levels of difficulty.

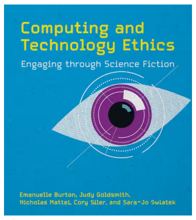
As with most discrete math texts, it will be necessary to pick and choose, both in terms of subjects ("Do I want to cover Analysis of Algorithms in this course?") and depth ("Do students need to see a detailed proof of this particular theorem?"). One note about depth: as I have indicated above, the book's treatment of some subjects (number theory, for example) is quite extensive and detailed. Students who have already taken at least a couple of math courses will have an easier time engaging the content in those sections. Still, there is plenty to select from in a book this large and comprehensive. It should definitely be in the mix of candidates if you are looking for a new text in this area.

Review of [1]

Computing and Technology Ethics:
Engaging through Science Fiction
by
Emanuelle Burton, Judy Goldsmith, Nicholas Mattei,
Cory Siler, and **Sara-Jo Swiatek**

The MIT Press, 2023
$85, hardcover, $36.99 electronic, 486 pages

Reviewed by **Brian Patrick Green** (`bpgreen@scu.edu`)
Markkula Center for Applied Ethics, Santa Clara University


Computing and Technology Ethics
Engaging through Science Fiction
Emanuelle Burton, Judy Goldsmith, Nicholas Mattei, Cory Siler, and Sara-Jo Swiatek

1 Overview

Writing a book is difficult. Writing a really good book is exceedingly difficult; and writing a perfect book is impossible. I am happy to say that, in the context of the three previous categories, *Computing and Technology Ethics: Engaging through Science Fiction* is a really good book, and the authors obviously put a tremendous amount of thought and effort into writing it.

By way of preface, for over a decade I have been wishing that someone would write a book that used short science fiction stories with ethical themes to stoke the imaginations of students and prime them for ethical discussions. Like engineering, ethics is a fundamentally imaginative activity. Envisioning the social and ethical impacts of technologies and their potential risks can range in difficulty from easy to impossible; envisioning good outcomes and how to achieve them takes imagination; and both of the above require experience as well: experience of what is possible, both good and bad, and how good and bad come about.

This matter of experience is where science fiction comes in. Science fiction stories provide us with the “experience” of what a technology might do to a society and the people within it. Related to ethics, this “virtual experience” then, hopefully, provokes within us our imagination which can perhaps analogically relate the ideas and technologies in the stories to our current or future ideas and technologies.

Whether this actually results in more imaginative students who then use that imagination to make better ethical choices, is, of course, a matter of empirical investigation, the conclusions of which (if there are any) remain unknown to me. But inspiration by science fiction stories of particular behaviors and technologies has actually been demonstrated, one recent example being Facebook/Meta’s desire to create a virtual reality “metaverse” similar to that depicted in the Neal Stephenson novel *Snow Crash* [1]. In the novel, the metaverse is not a utopia (in fact, it is one desired vehicle for world domination by evil schemers), and yet Facebook/Meta was inspired enough by this to name the very thing they were doing – indeed their company itself – after the metaverse. If science fiction stories can inspire this sort of action in the real world, perhaps they might inspire better actions too.

It is within this context, then, that we come to the contents of the book.

[1] ©2023 Brian Patrick Green

2 Summary of Contents

Computing and Technology Ethics is an approximately 480-page book consisting of two main parts: about 310 pages of textbook and about 160 pages of science fiction stories (the remaining pages are index, acknowledgments, internal cover pages, etc.). Part One consists of six chapters. The first is explanatory and entitled "Why Ethics? Why Science Fiction?" It is a brief chapter and answers exactly what it asks, and more, such as "Basic/Perennial Problems in Ethics" and "Life after Ethics Class." The authors thankfully prove themselves to be quite reasonable in their answers, descriptions, and explanations, which is always a relief when reading an ethics text. I was surprised that the section "How Have Recent Advances in Technology Changed the Conditions for Ethics?" did not mention Hans Jonas, whose magnum opus *The Imperative of Responsibility* is precisely on this question [2], but no critical reader is likely to find a book that says exactly what they want it to, unless they wrote it, and in any case Jonas is mentioned in other places.

The other chapters include "Ethical Frameworks," "Managing Knowledge," "Personhood and Privacy," "Technology and Society," and "Professional Ethics." I will not go into all of them in detail, but suffice it to say that there are not only the "standards" of ethics education (e.g., deontology, virtue ethics, utilitarianism, etc.), but also the main ideas from technology ethics and computing ethics, such as how to think about knowledge, the connections between knowledge, privacy, and identity, the impacts of technology on society, and the role of computing professionals in society. As one example, Chapter 6 on Professional Ethics is admirable for connecting codes of ethics (IEEE, ACM, and SECEPP) to fundamental ethical concepts (pp. 290–296).

Part Two, about a third of the book, is the anthology or "story bank" and it is a wonderful resource indeed. The explanation near the start which describes how stories are useful for teaching ethics is spot-on, and the "story points" chart on pages 320–321 is immensely useful for connecting the stories to the text. The twelve stories themselves are wonderful and thought-provoking, exactly what you would want from a collection of science fiction stories in an ethics textbook. I won't say too much about them other than that they cover a wide range of topics and the study questions for them are typically quite good. The stories also have an excellent set of study guides for instructors available on the MIT Press website.

Computing and Technology Ethics is quite good in its engagement of diversity, equity, and inclusion, considering (as just a brief selection) Ubuntu and Yoruba communitarianism (pp. 63–64), gender identity (pp. 173–175), and the risks of state power in policing and surveillance (pp. 254–256), and with excellent diversity of stories. It also has fascinating insights and anecdotes sprinkled throughout, including, for example (in just a few pages of Chapter 3), a section on the nature of wisdom (pp. 109–111), a sidebar on the relevance of information overload and decision fatigue for design (p. 115), and a discussion of Vannevar Bush's Memex (p. 117).

The good points of this book go on and on, and I can't list them all: indeed, that is what the book itself is for. So at the risk of being unfair, I will move on to the very few less-than-stellar points. But know that in the book itself the balance is nothing like in this review: overall the book is 99.9% stellar.

3 Considerations

I like this book a lot, and I would certainly use it for teaching a course on software ethics for engineers at either the undergraduate or graduate level. However, it does have a few considerations worth noting.

One is a structural question, and not necessarily a criticism, just a wondering about a design choice on the part of the authors. The book could have been structured with the twelve stories interspersed throughout the text and then the ethical material surrounding each story, rather than with all of the stories placed at the end. This would have undoubtedly been a difficult task, making the stories drive the textbook rather than holding them as a resource at the end, but I wonder if it would have made the book overall of more readable and of more interest to students, with a science fiction story to draw the reader forward every few tens of pages. This interspersing of stories and ethical thoughts would have been a very challenging task, and certainly a course instructor could assign stories as they see fit, so this is not a critique as much as a contemplation. Additionally, Part One does have "Story Point"s throughout the text indicating points of connection to the stories. So, the authors' chosen structure might be the best compromise for a book of this nature.

Secondly, 300+ pages broken into only six chapters is demanding of the teacher and the student, forcing the teacher to break up some chapters (e.g., Chapters 2 and 5) because they are simply too long.

Third, at $85, the price of the hardcover book is fairly high. I can understand why this would be the case, given the costs of printing and the necessities of paying royalties to the short-story authors, and I do not begrudge authors receiving a just compensation for their work. But it is left to students to pay the price, so it is a consideration. Thankfully, this consideration is one which just points to the comparative inexpensiveness of the electronic version, which is only $36.99.

4 Opportunities for Improvement

Now for a few improvements I would make, were I given the task.

First, as someone who knows a bit about Thomas Aquinas' virtue ethics, I did find it a bit odd that he was only presented in the context of deontology (pp. 35–36), with natural law theory presented as merely deontological. While "law" would appear deontological and duty-based on first look, Aquinas's natural law theory is typically understood as a form of virtue ethics based on Aristotelian and Biblical sources. The confusion might be because human nature and conditions govern us, but in a way that make certain dispositions of human capacities beneficial or malefic: virtuous or vicious. The "law" in this case serves as a framework for human virtue. Aquinas is not so easily classified, though I do appreciate his presence in the book.

Second, I do have to disagree with the statement on page 319 (building from page 318) that "...there are no right answers!" when it comes to the study questions before each story, or the resolutions of "stories" in general. While the questions themselves are quite open to many answers, they certainly have wrong answers and therefore also answers that are more or less right. Rather than saying "there are no right answers," I think it would be preferable to say that there are many "right answers" possible. I understand the desire to promote discussion by saying "there are no right answers," but in the context of an ethics textbook it raises questions about the ontological status of ethical judgments and of moral relativism, which is something that an ethics textbook should present, but primarily for the sake of refuting. When someone designs a bridge, we don't say "there are no right answers." There are clearly better and worse designs, and many right answers are possible, just as many wrong answers are possible.

For engineers and technologists, the "there are no right answers" idea – even if contextualized and specified to literary criticism – in an ethics book is dangerous. For those being taught to seek ethically right answers, suddenly telling them that right answers are not possible in the literary field – the very fields that this book is connecting – risks confusion and discouragement; perhaps so much so that they might no longer take the subject matter seriously. In ethics, we need to be able to say that better and worse answers are real. Stories – fictional or real – can have better and worse resolutions too. Fictional ones are not real, and so we can learn from them and think about how to make them better from a more clinical and creative perspective. But if these were real stories from the real world, then we should attempt to make the better resolutions come about and not the worse ones, especially if one is a technologist with the agency to be able to make such decisions.

The purpose of an ethics textbook is not only to equip the student for making their own ethical decisions but also to make clear to them that good ethical decision-making is *absolutely necessary.* Ethics is not optional. It is not a "fake" subject with no right answers (an unfortunate student belief sometimes heard, typically in the first week of the quarter). Just because ethics is difficult does not mean that it is not real, and ethical decisions can be very real in their effects, especially when dealing with powerful new technologies. The book does briefly dismiss moral relativism on page 31; however, it does not have an extensive discussion of relativism – a pernicious and pervasive ideology all-too-common among undergraduate students – which I believe is an oversight.

In sum, the book could use a section on moral relativism and more careful language about "no right answers" in literary criticism potentially causing confusion about "no right answers" in ethics.

Lastly, I have a recommendation for another resource that the authors might find to be useful for potential future editions, or in general. Jo-Ann Archibald Q'um Q'um Xiiem is a scholar of indigenous "storywork" and of the meaning that is found in stories. The edited volume *Decolonizing Research: Indigenous Storywork as Methodology* (especially the first 38 pages and chapter 13) and Archibald's 2008 book *Indigenous Storywork: Educating the Heart, Mind, Body, and Spirit* both consider deeply the role of stories in human identity and action [3, 4]. As noted, *Computing and Technology Ethics* is quite good on diversity, equity, and inclusion, but if the authors wanted to go even deeper into the meaning and role of stories, including their cultural and ethical relevance, Archibald's thought would be an invaluable addition to their book.

5 Conclusion

Computing and Technology Ethics is a tremendous gift to the scholarly community teaching technology ethics and especially computing ethics. I used to teach a course on software ethics, and were I to do so again I would use this book. I heartily recommend it to those who teach that subject now, or related subjects.

References

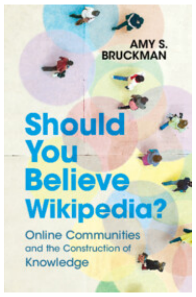
[1] Neal Stephenson. *Snow Crash.* New York: Del Rey, 1992.

[2] Hans Jonas. *The Imperative of Responsibility: In Search of an Ethics for the Technological Age.* Chicago: University of Chicago Press, 1984.

[3] Jo-Ann Archibald Q'um Q'um Xiiem, Jenny Bol Jun Lee-Morgan, and Jason De Santolo, eds. *Decolonizing Research: Indigenous Storywork as Methodology.* London: Bloomsbury, 2019.

[4] Jo-Ann Archibald Q'um Q'um Xiiem. *Indigenous Storywork: Educating the Heart, Mind, Body, and Spirit.* Vancouver, BC: UBC Press, 2008.

Review of [1]


AMY S. BRUCKMAN
Should You Believe Wikipedia?
Online Communities and the Construction of Knowledge

Should You Believe Wikipedia?
Online Communities and the Construction of Knowledge
by **Amy S. Bruckman**

Cambridge University Press, 2022
270 pages, $59.99 Hardcover, $19.99 Kindle, $19.99 Paperback

Reviewed by **William Gasarch**
(gasarch@umd.edu)

1 Introduction

Elon Musk could have saved himself $44 billion dollars if he had (1) bought *Should You Believe Wikipedia? Online Communities and the Construction of Knowledge*, (2) learned how hard it is to manage an online community, and (3) decided that buying Twitter was not worth the trouble.

The book is about online communities: how they evolve, thrive, or collapse, and how they are moderated. Despite the title, Wikipedia is only a small part of the book.

2 Summary of Contents

The book has 8 chapters. I will describe several adjacent chapters that are on similar themes all together.

2.1 Chapters 1, 2, 3, 4: Online Communities

Chapter 1: Are Online "Communities" Really Communities?
Chapter 2: What Can Online Collaboration Accomplish?
Chapter 3: Should You Believe Wikipedia?
Chapter 4: How Does the Internet Change How We Think?

1. Are Online "Communities" Really Communities? Chapter 1 gives a resounding yes. The chapter also gets into the more basic question of *What is a Community?*.

2. When does group think set in? On some sites people are scared to propose an unpopular opinion.

3. How are the rules of discourse established? (This is discussed throughout the book.) Reddit is particularly interesting, since each group decides for themselves.

[1] ©2023 William Gasarch

4. How many are contributing and how many are *only* lurking? An interesting contrast: on technical groups there tend to be more lurkers, whereas on medical groups people tend to talk about their problems.

5. What is the best way to get unpaid people to contribute to a project? This is obviously relevant to Wikipedia. The main factor is that people choose what they want to work on and hence will do a good job.

6. Should you trust Wikipedia? The book says yes—errors are corrected quickly, even on controversial sites like the vaccine hesitancy (which should not be controversial), and people are writing about what they know and care about. I think the book is too optimistic about Wikipedia. My issue is *not* that some of the websites are incorrect. I mostly look up science, where it is pretty good (though sometimes incomplete). My issue is that making a change to a page is hard. There are people who police Wikipedia, which is needed, but they also block perfectly good changes. I also have an issue with which topics are covered and by whom. Around 90% of Wikipedia writers are white, and around 85% are male.

7. How to best moderate an online community? (This is discussed in many chapters.) We all know what an *echo chamber* is: people who reinforce each other's viewpoints. This book points out that this can be *positive*. If there is a group discussing feminism, then this can be productive if they all agree that women are people too. The author does point out two downsides: (1) if someone wants an honest discussion of whether women really are paid less than men, that would not be welcome; and (2) if the people in the echo chamber are objectively wrong (e.g., Holocaust denial), then this can be very harmful. The author discusses various ways that different sites handle the issue of moderation.

3 Chapters 5, 6, 7: Behaviour Online

Chapter 5: How Do People Express Identity Online and Why?
Chapter 6: What Is Bad Behaviour Online and What Can We Do About It?
Chapter 7: How Do Business Models Shape Online Communities?

Chapter 5 is about how we present ourselves online. We have all heard stories about people who claim to be a different gender. But people are dishonest about other aspects of their lives as well. And some of how we present ourselves is not dishonest in that it may be how we see ourselves. Or, as George Costanza says, *it is not a lie if you believe it.*

Chapters 6 and 7 are an intelligent unbiased discussion of the enormous tension between free speech and civil order. What regulates speech on line? Laws, social norms, technology, and markets. All of these are discussed and are complicated. Technology can help some; however, having an AI determine what is hate speech or otherwise objectionable is a hard problem. The biggest problem may be that hate speech can be profitable. No real answers are given (sorry Elon), but the issues are all laid out nicely.

4 Chapter 8: How Can We Help the Internet to Bring Out the Best in Us All?

This chapter is a summary of the book and a plea to do better. It reminded me of Rodney King's famous quote: *People, I just want to say, can't we all get along?*

Alas, if only it was that easy.

5 My Opinion

If I was to tweet one word to describe this book, it would be *intelligent*. Much of what the author says we sort-of know, but it's really good to get it written down clearly in one place. COVID-misinformation and the Jan 6 insurrection make the issues raised more relevant.

The only drawback is that the book is a little too positive. The author is not angry enough about the problems with online communication.