The Book Review Column[1]

by William Gasarch
Department of Computer Science
University of Maryland at College Park
College Park, MD, 20742
email: gasarch@cs.umd.edu

In this column we review the following books.

1. **Data Structures and Algorithms Using Python and C++**. by David M. Reed and John Zelle. Review by Richard Jankowski. This is a traditional undergraduate "CS2" textbook that uses the Python programming language to introduce students to the concept of data structures and algorithms. Students are expected to know basic programming in Python to be able to start working their way through this book.

2. **An Introduction to Data Structures and Algorithms**. by James A. Storer. Review by Adel El-Atawy This book is a good simple introduction to data structures and algorithms (clear from the title). It is more focused on algorithms than data structures and their design.

3. **Advanced Data Structures** by Peter Brass. Review by Richard Jankowski. This is a graduate-level textbook providing comprehensive treatment of modern data structures. The book provides examples in standard C, and is compact while maintaining mathematical rigor.

4. **The Burrows-Wheeler Transform: Data Compression, Suffix Arrays, and Pattern Matching** by Donald Adjeroh, Timothy Bell and Amar Mukherjee. Review by Shoshana Neuburger. The Burrows-Wheeler transform is a very recent data compression method. This book gives a careful explanation of it and its many applications, which go beyond data compression.

5. **Curve and Surface Reconstruction: Algorithms with Mathematical Analysis** by Tamal K. Dey. Review by Matthew J. Sottile. Reconstructing curves and surfaces from sets of points is a common activity in medical imaging, computer graphics, and engineering. This text discusses algorithms for performing these reconstructions for both two-dim curves and three-dim surfaces.

6. **Concentration of Measure for the Analysis of Randomized Algorithms** by Devdatt P. Dubhashi and Alessandro Panconesi. Review by Aravind Srinivasan. Lets say you have a distribution. You pick an element using it. How close will it be to its mean? This is a hard problem! This book is about how to deal with this problem in the context of randomized algorithms.

7. **The Modern Algebra of Information Retrieval** by Sandor Dominich. Review by John S. Griffin. This book treats retrieval methods (major proven models and ranking techniques) and IR in general in a unified manner within the one formal framework of modern algebra, namely abstract algebraic structures (primarily lattices).

8. **Multiagent Systems: Algorithmic, Game-Theoretic, and Logical Foundations** by Y. Shoham and K. Leyton-Brown. Review by Haris Aziz. Network economics, game theory, learning, electronic commerce and logical theories have all received interest of theoretical computer scientists. The book is a holistic effort to combine all these strands under the unified umbrella of multiagent systems and introduce their algorithmic, game theoretic and logical foundations.

9. **The Political Mapping of Cyberspace** by Jeremy W. Crampton. Review by Rajesh Natarajan. What is the influence of WWW and Internet on human behavior in particular and society in general? Post the dot-com bust, the topics of these studies have shown a gradual shift from the Internet or information itself as the unit of analysis to the study of the Internet as an important constituent of human society. This book continues this tradition with an illuminating, innovative and philosophical study of the *spatial politics of cyberspace*. The goal, according to the author is to investigate how we find our place in the world with cyberspace being the domain.

10. **The Princeton Companion to Mathematics** by Timothy Gowers, June Barrow-Green and Imre Leader. Review by Haris Aziz. This book aims to give a broad outlook and snapshots of important concepts, ideas, intellectual movements and figures in mathematics. It succeeds!

11. **Computer Viruses and Malware** by John Aycock. Review by Michael Sanford. The title says it all.

12. **Formal Correctness of Security Protocols** by Giampaolo Bella. Review by Yannis C. Stamatiou. How do we prove protocols secure? We first need a model. This book discusses both models and proofs of security in those models.

13. **Computability of Julia Sets** by Mark Braverman and Michael Yampolsky. Review:by Wesley Calvert. Dynamic systems can produce rather strange sets. Are these sets (approx) computable? This book investigates this issue.

Review of[2]
Data Structures and Algorithms Using Python and C++
by David M. Reed and John Zelle
Franklin, Beedle and Associates 2009
568 pp., $88.00, Softcover

Review by
Richard Jankowski `rjankowski@acm.org`

1 Introduction

Data Structures and Algorithms Using Python and C++, by David M. Reed and John Zelle is a traditional undergraduate "CS2" textbook that uses the Python programming language to introduce students to the concept of data structures and algorithms. Students are expected to know basic programming in Python to be able to start working their way through this book.

The book is essentially broken into three major sections. The first section, encompassing chapters 1 - 7, introduce the student to the basic data structures using the Python programming language. After the basics are outlined, second section of the book introduces the student to the basics of C++ programming in chapters 8 - 12. Chapters 13 - 15 cover the final major section, with treatment of the more advanced topics while using Python as a the main teaching language.

2 Summary

Chapter One, *Abstraction and Analysis* introduces the students to concepts such as functional abstraction, pre- and post-condition assertions, and algorithm analysis and efficiency. Algorithm analysis concepts are given by comparing various search methodologies, and the student is given a informal introduction into the concept before coverage of formal analysis with Big-O and Theta notation.

Chapter Two, *Data Abstraction*, covers the fundamental concepts of data abstraction and object-oriented programming using Python. Coverage of encapsulation, polymorphism, and inheritance leads the student into many examples of implementing ADTs and objects with real-world examples.

Chapter Three, *Container Classes*, introduces the student to using a container class to manage sequential objects. One of Python's strengths is the List object, and the text makes good use of using the built-in List data structure as a tool for manipulating sequential data. Concepts such as sorting data and comparing data elements are covered, and the algorithmic efficiency of sorting is covered. The chapter closes with an introduction of Python's Dictionary data structure.

Chapter Four, *Linked Structures and Iterators*, introduces the student to implementing linked structures using Python's various data structures. The Python memory model is covered, giving students insight into variable references and the concept of using Python's references as pointers to other objects. Introducing Python's references at this stage of the book will serve the student later when C++ pointers are later covered in Chapter Ten. Iterators are also introduced, with coverage of using generators within Python to save the state of a traversal of a larger data structure.

Chapter Five, *Stacks and Queues*, is an introductory chapter that covers the ADTs used to implement stacks and queues. Basics for both data structures are covered in detail, with such examples as expression manipulation using Polish prefix notation for the stack data structure, and using the queue data structure to determine if a string is a palindrome.

Chapter Six, *Recursion*, introduces the student to the power of developing recursive functions in Python. As would be expected, this chapter contains much more rigor than previous chapters as recursive functions are outlined. Mergesort is covered with algorithmic analysis, and the chapter concludes with the Tower of Hanoi puzzle.

Chapter Seven, *Trees*, introduces the student to the applications and basics of implementing tree data structures. After an introduction of basic tree terminology, extensive coverage follows of binary search trees including implementation, tree traversal, and run-time analysis.

Chapter Eight, *C++ Introduction for Python Programmers*, is where the focus of the book switches from teaching basic algorithms and data structures in Python to introducing the student to C++. This chapter is aggressive in that it covers virtually all of the basics of C++ in just under 60 pages. The student should be able to leave this chapter being able to understand the basic syntax of C++ and write simple functional programs.

Chapter Nine, *C++ Classes*, takes the students into the syntax and structure of classes in C++. After coverage of the basic concepts, such as constructors, the text covers the usage of the string class, file input and output using ifstream and ofstream, and operator overloading.

Chapter Ten, *C++ Dynamic Memory*, introduces the student to the memory model used by C++ and the associated concept of pointers. Dynamic arrays are covered, along with dynamic memory classes. The chapter closes with memory errors - such as memory leaks.

Chapter Eleven, *C++ Linked Structures*, covers the concepts of creating linked structures in C++. Linked lists are covered in detail, highlighting the differences in implementation between C++ and Python.

Chapter Twelve, *C++ Templates*, introduces the C++ templates, including template functions and classes, such as the Standard Template Library vector class.

Chapter Thirteen, *Heaps, Balanced Trees, and Hash Tables*, covers the more advanced data structures. Code examples are given in Python with implementation assignments for the student in C++. Priority queues and heaps are introduced, with implementation examples with heapsort. Balanced trees and hashtables are also covered, with analysis of each topic.

Chapter Fourteen, *Graphs*, introduces the student to the fundamentals of graphs, such as edges and vertices, and then covers the graph data structures of adjacency matrix and adjacency list. Shortest path algorithms and depth first algorithms are covered, with the chapter closing on the topic of minimum spanning trees.

Chapter Fifteen, *Algorithm Techniques*, formalizes the topics previously covered as a way to introduce algorithms as a problem solving tool. Divide and conquer algorithms are covered, with treatment of analyzing recursive functions and the quicksort algorithm. Greedy algorithms, with an example of Huffman codes is outlined, followed by dynamic programming and optimization problems. The chapter closes with a basic explanation of NP-Complete problems.

3 Opinion

Data Structures and Algorithms Using Python and C++, is a well written introduction to data structures and algorithms for a student with basic Python programing experience. The authors did a great job of using Python's clarity as a tool to illustrate some of the more complex topics of data structures, while introducing the student to the power of C++ for some of the more advanced topics.

This book is not only suited to those in the classroom learning data structures and algorithms in a formal setting, but also those new Python programmers who may have picked up the language informally and are looking to hone their skills. The exposure to C++ is well taught, the authors themselves state that C++ is an excellent complementary language for Python programmers, ad this book uses both languages equally well.

Students working through this book should should get a solid grasp on the basics of data structures, algorithms, and the C++ programming language.

Review of[3] of
An Introduction to Data Structures and Algorithms
Author of book: James A. Storer

[3]

Birhauser, 624 pages, $54.97 on Amazon on Jan 11, 2010

Review by Adel El-Atawy aelatawy@cs.depaul.edu

1 Introduction

This book can be classified as a good simple introduction to data structures and algorithms (clear from the title). It is more focused on algorithms than data structures and their design. The book serves well as an introduction to an undergraduate who needs a book to introduce her to more formal thinking about algorithms than a layman programmer would know. It includes basic algorithm design (e.g., greedy, dynamic, ...), basic structures (e.g., lists, trees, heaps, hash tables, strings, sets and graphs), and did not go much further than the title in the topic of NP-completeness. The book covers some of the related algorithms to the mentioned data structures along with some general ones as DFT, polynomial, and set operations. The book is also supplemented with an introduction to parallel computation models.

The author in his preface mentioned that the book is designed for a college-level student, and is augmented with exercises and examples to help them go through the topic. We see that we have about 400 exercises/problems in all for the student to tackle. A good number to occupy a student for a whole semester (and more), and can be quite handy for instructors as well.

2 Summary

Here, we will have an overview of the book contents along with local comments on each chapter. Overall, the book is composed of 13 chapters (no parts are given, neither numbered sections), and a single appendix. The chapter style is intended to fit as a reference to go side by side with course lectures. Starting from the page style, and formatting to topic distribution over the book. Most chapters will be almost completely composed of examples, each nicely written to fit in a page using lots of page breaks. The core ideas and concepts will be introduced in - many cases - a few lines, and then explained along within the provided examples. At the end of each chapter we will find a long list of exercises and final chapter notes (the only place with references) that can be highly beneficial to the reader.

Chapter 1: RAM Model. A brief introduction of memory, what is an algorithm, asymptotic complexity, and how we will access memory locations is given. However, the name of the chapter can be a shock to some starting-to-check-the-area student. Having a more friendly name like "Introduction to algorithms", or "what is an algorithm?" can be much more soothing. Moreover, the author chose to discuss the problem of input representation (i.e., should we measure the size of the problem by input value or its size in bits) in the first page of the first chapter. This would force many readers (especially students) to take a bad first impression.

Chapter 2: Lists. Lists as a data structure are introduced then common uses from stacks, queues, and common uses of simple lists. Implementation in arrays and via pointer are described with their basic operations, and good set of simple applications.

Chapter 3: Induction and Recursion. A 2 page introduction to what is recursion and induction then the rest of the chapter (35-pages) are examples and exercises. Within the examples many ideas are presented like "proof by induction" and "elimination of recursion". The examples themselves

included some must-know problems like binary search, prefix expression evaluation, exponentiation, and Hanoi towers.
Chapter 4: Trees. Started with some definitions (leaf, root, sibling, depth, etc), then basic operations were discussed (e.g., traversal, delete, deletemin, split, and merge). The discussion then switched to balanced trees, and amortized analysis (was mentioned by name in chapter 2). Examples of rotation operation was given, but no specific balanced or self-adjusting tree structure was explained. However, splay trees were mentioned, but not by any means covered. Amortized analysis is introduced via the accounting method (same method was used with the multi-pop operation mentioned in chapter 2).
Chapter 5: Algorithm Design. Basic algorithm design techniques were introduced and many examples supported the demonstration. Divide and Conquer, and dynamic programming were the main techniques. Many examples were given as Merge Sort, Strassen's algorithm, i^{th} order statistics for D&C and optimal matrix multiplication, and the knapsack problem for the dynamic programming technique. After that a quicker pass over randomized algorithms followed by greedy algorithm and a 1-page introduction to simulated annealing.

There was a typo in an expression in the main theorem to solve recurrence relations (the master theorem). It is written as $T(n) = O(n^{log_c(d)})$ instead of $T(n) = O(n^{log_d(c)})$, but it was fixed in the proof in the same page.
Chapter 6: Hashing. A very good chapter on the topic. It describes the main ideas and concepts of basic hashing techniques. Then it digs into proving bounds on the collision probability, empty buckets, and size of maximum bucket in the hash table. Using Chernoff bounds, most of the proofs are presented in a simple to understand format. An example of dynamically sized hashing table is shown. Universal hashing, twin hashing and Bloom filters (one page each) are introduced at the end of the chapter, followed by 11 exercise questions (among the shortest exercise sections in the book).

Three of the proofs presented managed to prove an upper bound of $1/n^2$ on the probability of largest bucket size exceeding some limit for different cases. However, it was written afterwards (in all three of them) that the bound is as high as $1/n$.

There was a section named "the Constant e", which disrupted the flow of the chapter. It could have been moved to the appendix without much trouble. The same goes for another section named "Chernoff Bound".
Chapter 7: Heaps. Starts with introducing complete k-ary trees and full trees, then shows how to build a tree using array implemented trees. Normally, the chapter includes the heap sort. However, getting into proving the $O(n \log(n)$ lower bound on sorting algorithms was quite out of context in this chapter. The chapter ends with a few exercises and very short final notes. It was a very short chapter that did not cover but the basics of the standard heap implementation.
Chapter 8: Balanced Trees. The chapter includes three examples of balanced trees: 2-3 trees, red-black trees and AVL trees. Each of these flavors are introduced along with the pseudo-code for their basic operations. Detailed examples are shown for some instances of insertions, deletions, and splitting operations. Some brief analysis is also included for complexity and height guarantees provided by these structures.
Chapter 9: Sets Over a Small Universe. Motivation and examples to these structures are given by three well-known problems: on-the-fly array initialization, in-place permutation and bucket sorting. However, that was not clear in the first glance. Union and Find operations were explained over different implementations: bit-vector, linked list, tree and tree with path compression. The chapter

could have been more readable if augmented with some drawn examples of the presented operations.
Chapter 10: Graphs. This is one of the largest chapters in the book. It starts with a list of definitions of graph terms, from vertex to isomorphic graphs. Then a necessary discussion of common representations and traversal techniques are presented with examples and pseudo-code. The chapter then lists a list of the famous and commonly used problems in graph theory (a page for each problem): depth/breadth first spanning trees, strongly and bi-connected components for directed and undirected graphs, min weight spanning trees (with Kruskal and Prim's algorithms), topological sorting of directed graphs, Euler paths, single source minimum cost path (with Dijkstra's algorithm), and multi-source minimum cost paths, and transitive closures. A considerable part is given afterwards for the maximum flow problem with several algorithms and necessary related concepts as augmenting paths, residual and level graphs, etc. Also, the matching problem was given some attention including matching in bipartite graphs, and the stable marriage problem. The chapter ends with some discussion about NP-completeness with emphasis on graph-related problems. The chapter ends with a very good collection of challenging 57 problems, and very good final notes.
Chapter 11: Strings. The second largest chapter in the book along with the previous one. However, regarding the relative popularity of graphs over string matching algorithms, we can see that this chapter is quite elaborate on the topic of strings, string matchings and representations. Starting the chapter with several algorithms for string matching (simple, Knuth -Morris-Pratt, Boyer-Moore, Karp-Rabin, etc and several versions of the Shift-And variants as anything-but, and minimum-mismatches, etc). Regular expressions are introduced to the reader, followed by techniques for building pattern diagrams, and pattern matching. The third part of the chapter is allocated to Tries. The discussion quickly lead to data compression (a dear topic for the author, wrote several books on compression). Compact, suffix and sliding window tries conclude the discussion on tries. The author then finished with arithmetic encoding, and Burrow-Wheeler transform. After the first few sections, I think, it becomes more involved than what an undergraduate student will be interested in. However, it is a very valuable chapter for a stronger motivated reader.
Chapter 12: Discrete Fourier Transform. The need for DFT transforms are motivated followed by a detailed discussion of all the needed operations (theoretical and practical reasons) to perform FFT with a fair discussion of the ideas behind each. Then DCT follows naturally. The chapter ends with a one page description of each of the JPEG and MPEG standards. This chapter is one of the best explanations I have seen in this area within the the size limits of a single 50-page chapter.
Chapter 13: Parallel computation. While the author mentioned that the chapter on parallel computing is given as a very brief overview of the area, it includes the basics needed to start investigating the area seriously. It starts with a few simple examples, like Max, Merge sort, and matrix multiplication where different memory access models are introduced (EREW, CRCW, and CREW). Then a few examples are given in more detail for the EREW model (e.g., data distribution and sorting) followed by a section on connectivity networks of processors as the hypercube, the butterfly, and the mesh networks. Each of these networks were explained with several example problems (mostly the same examples for the sake of comparison). The chapter concludes with a section on hardware implementation and actual real IZATION of such parallel architectures.

Appendix: Common Sums. A quick handy summary of several famous sums (arithmetic, geometric, harmonic, etc) and sequences and how to obtain sum expressions and bounds.

3 Conclusion and Comments

The book, in my humble opinion, has its readability highly affected by the formatting style. Lacking section numbering as well as any kind of hierarchy that can be deduced by title point sizes and styles can be very confusing sometimes. The author reason was to make it look more friendly and in a lecture style, but I think this way they missed the goal, and added confusion instead. However, the writing style of the author is very enjoyable which compensated for the book formatting.

The end notes at each chapter were very interesting to read for many reasons. They link the chapter topic with other ideas and important concepts, and also they were the only place cited the references at the end of the book. However, with hundreds of references that are unnumbered, it becomes hard to search for each one when needed (they were sorted by authors names).

The book is suitable for undergraduates as well as graduate students who are not going to focus their studies on the theoretical aspects of computer science (otherwise, a book with deeper theoretical focus is recommended). The plan for undergraduates as described in the Preface, I think, is quite sparse and light weight. By covering around five/six chapters (1-4, parts of 5,6,12,13), the student would not have enough span of the area. I guess students will need some of the extra information provided by chapters 7,8,10 and some additional coverage of NP-completeness (a brief discussion is in chapter 10).

In conclusion, the book is extremely useful as a side help for a starting student, not as the main textbook or reference. For a student who is familiar with the territory, this book is a very good choice. Equipped with a large selection of solved examples and even a larger set of unsolved exercises, this book renders itself a useful tool for an algorithm student (as well as the instructor). Will be best beside a more reference-style textbook on the subject (like those by Cormen-Leiserson-Rivest-Stein, Aho-Hopcroft-Ullman). It will surely give the student a wide exposure on different problems and in all it is a nice book to read.

Review of[4]
Advanced Data Structures
by Peter Brass
Cambridge University Press 2008
472 pp., $75.00, Hardcover

Review by
Richard Jankowski rjankowski@acm.org

1 Introduction

Advanced Data Structures by Peter Brass is a graduate-level textbook providing comprehensive treatment to the topic of modern data structures. The book provides examples in standard C, and is compact while maintaining mathematical rigor. Much of the material is presented in the format of formal theorems with implementation code and figures used to reinforce and illustrate the concepts. In addition to its intended purpose as a textbook, *Advanced Data Structures* would also serve well as a reference or supplemental resource for a data structures course.

[4]

2 Summary

Chapter 1 opens with a gentle review of the more common data structures familiar to most programmers, including stacks and queues. While most likely already mastered by the intended audience of *Advanced Data Structures*, the focus of this chapter is to lay the groundwork for later material that utilizes these fundamental structures.

Chapters 2 through 4 provides extensive coverage of the tree data structures. Search trees, balanced search trees, and tree structures for sets of intervals are covered extensively in these chapters. The author provides the background and theorems pertaining to trees, including tree operations such as finding, inserting, and deleting nodes, as well as tree traversal. Ample coverage is provided on building optimal search trees, and Brass closes chapter two with sections on converting a tree to a list, and efficiently removing the tree data structure from memory.

Chapter three's coverage of balanced search trees, spanning almost a hundred pages, is the largest chapter of the book. Height-balanced and weight-balanced trees are covered as are B-Trees and red-black trees. Brass closes the treatment of tree data structures with chapter four's coverage of trees for sets of intervals. This chapter includes topics such as interval and segment trees.

Chapter 5 covers the topic of heaps, with the treatment of topics such as balanced search trees as heaps as well as array-based, skew, and Fibonacci heaps.

Chapter 6 covers the topic of Union-Find and other structures for the partitions of a set of data elements. This chapter covers topics such as merging classes of a partition as well as list splitting, including operations on the covered structures.

Chapter 7, entitled Data Structure Transformations, covers the subject of making the previously explored data structures dynamic in the sense that the underlying data can be changed after the structure is created, as well as making the structure persistent, or allow the ability to query old versions of the persistent structure.

Coverage of data structures for strings is the topic of chapter 8, including dictionary structures that allow for errors in queries and suffix trees and suffix arrays.

The author covers hash tables in chapter 9, including the topics of collision resolution and hash trees.

3 Opinion

Advanced Data Structures is a very well-written resource on data structures. The book is compact without sacrificing clarity and rigor. Extensive code samples exist to allow a programmer to go from conceptualization to implementation in a short period of time, and elegant and well-designed diagrams and other visual representation of the structures solidify the concepts in an easy-to-understand form.

This book does assume a certain degree of mathematical maturity, however the writing is clear and succinct enough that readers with an interest in the subject will be rewarded with an enjoyable tour through the mathematics and implementation of advanced data structure design. This book would not be out of the range of most mature computer science undergraduate students, and should be readable by most graduate students.

I personally liked the use of standard C in the text. C is a language understood by many, and the code is clear enough to allow implementation in another language if necessary. I think this book is well suited as a main or supplemental text in a graduate-level data structures course,

not to mention an invaluable desk reference for those interested in or implementing the advanced structures outlined in this book. This book was a joy to review, and deserves a place on my bookshelf.

Review of[5] of
The Burrows-Wheeler Transform:
Data Compression, Suffix Arrays, and Pattern Matching
by Donald Adjeroh, Timothy Bell and Amar Mukherjee
Springer, 2008
352 pages, Hardcover

Review by
Shoshana Neuburger shoshana@sci.brooklyn.cuny.edu
Graduate Center, CUNY, New York, NY, 10016

1 Introduction

David Wheeler and Mike Burrows introduced the Burrows-Wheeler Transform (BWT) as a practical method of data compression in 1994. The elegant simplicity of the BWT has captivated researchers for the past decade. The BWT is simple yet powerful. Other lossless compression techniques are a lot more cumbersome, making it difficult to assess their efficiency. Lossless compression is the preferred method of conserving space in text files. Audio and video files are typically compressed with lossy compression mechanisms since some content can be eliminated without noticeably compromising the quality of compressed data.

The Burrows-Wheeler Transform presents an innovative approach to data compression. The transform is simply a permutation of the original data. This unique permutation makes it easier to process the data. Since the Burrows-Wheeler Transform is easily reversed, it lies at the basis of lossless compression techniques. The nature of the transform allows the transformed data to readily be compacted by a variety of compression techniques. Among them are run-length and move-to-front encoding.

The Burrows-Wheeler transform works with one block of text at a time. The transform sorts the cyclic shifts of the text lexicographically. Its output consists of a string, L, composed of the last character in each of the sorted rotations, along with a pointer to the position of the last character in this permutation. The reverse transform reconstructs the original text by a series of sort and count operations. The transform exploits the property that several occurrences of a character will appear in the same order in both the original and permuted data.

This book exposes the reader to many applications of the Burrows-Wheeler Transform beyond data compression. The book begins by explaining how the transform works. Then the reader is introduced to the various compression techniques that can be applied to the transformed text to yield a concise representation of the original data. Related techniques in pattern matching algorithms are presented as well.

Only recently has the BWT spurred so many results. This book invites the reader to explore a new topic. The material is easily internalized in the textbook's clearly defined style and concise explanations.

2 Summary

Chapter 1

The text begins with an example of the Burrows-Wheeler Transform that is easy to follow. The inverse operation that reconstructs the text is also presented. With the use of pointers, a block of text is sorted without incurring additional storage space. Retrieving the original data from the transform consists of a series of sort and count operations, which can also be done in constant space using pointers. A transform does not alter data, but it does present a new perspective on existing information. The transform does not change the size of the file. However, it simplifies compression of a file since like characters are clustered together. The sorted permutation of text characters brings together characters that precede identical substrings. Repetition in the original text becomes noticeable in the permuted data. The chapter concludes with a brief history of data compression.

Chapter 2

After an exposure to the basic function of the Burrows-Wheeler Transform, the reader is presented with a practical implementation of the transform. With a few tricks, encoding and decoding is performed in time and space proportional to the input size. The text includes step-by-step examples with pseudo code that is easy to follow.

Chapter 3

Once we have a basic understanding of how the BWT works, we can consider its usage in data compression. Among the techniques used to concisely encode the output of a BWT are run-length encoding, move-to-front encoding, inversion frequencies, distance coding, frequency counting methods, and wavelet trees. Run-length encoding, which is one of the simplest approaches, yields the best compression of Burrows-Wheeler transformed data. In addition to this benefit, run-length encoding is also a very fast technique.

When space is a consideration, we seek to represent symbols in the fewest bits possible, using *entropy coding*. *Entropy* is a lower bound on the space required to represent a string. In entropy encoding, the representation of a symbol is based on the probability of that symbol occurring. The probabilities can either be estimated adaptively, "learning" during encoding, or non-adaptively, based on historical data.

Since the BWT works on entire blocks of text, the block size affects compression performance. There is a balance between larger block size and limited cache size, to allow faster access to Random Access Memory. The BZIP2 program is a compression system based on the BWT.

Chapter 4

The suffix tree is a compact, linear representation of the suffixes of a string. The suffix array is a permutation of the string that represents the lexicographical order of its suffixes. The suffix array and the suffix tree represent the same information in different forms. There exists a close relationship between the BWT and the suffix array of a string.

This chapter covers linear-time suffix tree and suffix array construction algorithms, as well as implementation issues to consider. In its obvious form, the suffix tree represents quadratic information, and thus occupies quadratic space relative to the size of the input string. Yet, its linear representation can be obtained in linear time. The most remarkable results are Ukkonen's online construction algorithm and Farach's alphabet-independent, recursive construction algorithm. The suffix array requires less space than the suffix tree. The first direct suffix tree construction algorithm is due to Manber and Myers, which relies on the *doubling* technique of Karp, Miller, and Rosenberg to construct the data structure in $O(n \log n)$ time, where n is the size of the input. More recently, several linear time algorithms have been developed using the divide and conquer approach.

Chapter 5

This chapter examines the theoretical performance of the BWT, in terms of time and space complexity. The output is also analyzed through empirical evaluation and with assumptions about the input. The compression performance is described in terms of entropy. The relationship with other compression schemes is also explored. Data compression by Prediction by Partial Matching (PPM) and Ziv-Lempel coding are compared to the compressed BWT.

Chapter 6

This chapter presents many variations, extensions, and generalizations of the BWT that have been considered. For each modification considered, empirical performance data is included whenever possible.

Chapter 7

The pattern matching problem is that of locating all occurrences of a *pattern* string in a larger string called the *text*. Pattern matching and data compression are intimately related, and at the same time can work against each other. Compression removes redundancies in a text, which also destroys the text's natural structure, thereby complicating information retrieval. The *decompress-then-search approach* involves restoring the original data so that pattern matching proceeds as usual. *Fully-compressed pattern matching*, searching among compressed data, is preferred since decompression can be time and space consuming. Many compression mechanisms use different representations for a substring depending on its context, making fully-compressed pattern matching an impossibility.

Efficient exact pattern matching algorithms are reviewed. Among them are the Knuth-Morris-Pratt automaton, the Boyer-Moore algorithm, and the Karp-Rabin randomized algorithm. Algorithms that solve variants of pattern matching, such as multiple pattern matching and pattern matching with don't cares, are also presented. The BWT provides a framework for efficient pattern matching directly on compressed data. Among the methods are binary search, adapted Boyer-Moore, and the suffix array. The *FM-index* combines the BWT and the suffix array to obtain a

compressed suffix array. It is a *f*ull text index in *m*inute space. A comparison of the performance of the BWT pattern matching methods is presented.

Chapter 8

The text concludes with detailed applications of the Burrows-Wheeler Transform to demonstrate that the BWT can be used to effectively address problems that at first glance seem unrelated to the transform. The compressed suffix tree occupies $O(\log n)$ bits rather than the usual $O(n \log n)$ bits, for an input string of size n. Compressed suffix trees and compressed suffix arrays lead to compressed full-text indexing.

3 Opinion

Although familiarity with algorithms and their analysis is assumed, the reader is not expected to have any prior exposure to pattern matching algorithms. I can recommend this text to a wide variety of readers. This book is well suited for a researcher familiar with the field of pattern matching algorithms who seeks to understand the Burrows-Wheeler Transform and its many applications to the existing framework of algorithms. Yet, it is also suitable as an introduction to pattern matching algorithms since the text describes the classical pattern matching algorithms and data structures as they are encountered.

This would be a wonderful course textbook since its coverage is comprehensive and its explanations do not assume that the reader has a great deal of background. Because this was not the author's original intent, the instructor would have to design his own exercises.

The clear exposition of the text and its well-defined structure facilitate a smooth traversal of its contents. Many computer scientists can gain from reading this book. Since the textbook touches a variety of topics, the curious reader might seek greater detail than that provided by this text. Each chapter ends with a *Further Reading* section which refers the reader to sources that discuss each specific topic at greater length. This section is a wonderful resource to researchers who would like to delve further into the topics and extend the work described by the text. There is plenty of room for innovation since there remain many unexplored avenues around the Burrows-Wheeler Transform.

Review of[6]

Curve and Surface Reconstruction: Algorithms with Mathematical Analysis

Author of book: Tamal K. Dey

Cambridge University Press, 214 pages, Hardcover

Review by

Matthew J. Sottile (matt@cs.galois.edu)

1 Introduction

Reconstructing curves and surfaces from sets of points is a common activity in medical imaging, computer graphics, and engineering. Given a physical object, we often seek to build a digital

[6]

representation of it from a set of discrete samples. This digital representation can then be visualized, analyzed and manipulated on the computer. To create this digital representation from samples collected from the physical object, we must employ algorithms to reconstruct the shape of the object. This text discusses algorithms for performing these reconstructions for both two-dimensional curves and three-dimensional surfaces.

The text presents these algorithms in the context of the mathematical basis upon which they are derived and their properties proven. The mathematical content of the text is drawn from the areas of topology and geometry. In addition to basic reconstruction algorithms, the text also addresses issues that appear in practical contexts, such as reconstructions in the presence of noisy measurements and undersampled regions. A large portion of the text focuses on algorithms based on the Voronoi and Delaunay diagrams commonly used in computational geometry. Additional discussion is provided for methods based on implicit functions and Morse theory.

2 Chapter overview

The book is structured in such a way that necessary mathematical concepts are introduced before using them to both derive and prove properties about algorithms.

Chapter 1: Basics. This chapter introduces concepts in topology and geometry that are used throughout the book. These include topological spaces and maps, manifolds, complexes (such as triangulations and cells), sampling and features, Delaunay triangulations and Voronoi diagrams. While reading the text, I found this chapter one that was referred to frequently to refresh my memory on terminology and definitions. The mathematical foundations are used throughout the text to prove that the reconstructions reliably reconstruct the surfaces from which the sampled points were taken.

Chapter 2: Curve Reconstruction. This chapter gives the reader a first taste of reconstruction algorithms by discussing two algorithms (`CRUST` and `NN-CRUST`) for reconstructing 2D curves from sampled points. The chapter starts by presenting some results related to sampling that are used to establish the correctness of both reconstruction algorithms. The algorithms are based on primitive algorithms from computational geometry for computing the Delaunay triangulation and Voronoi diagram of the sampled points.

Chapter 3: Surface Samples. Chapter 3 focuses on the mathematical basis for 3D surface reconstructions. Concepts include surface smoothness, surface normals, the use of Voronoi cells to approximate normals, and the properties of edge and triangle normals. The chapter also discusses the relationship of the topology of a surface and a set of sampled points of that surface to the restricted Voronoi and Delaunay diagrams.

Chapter 4: Surface Reconstruction. This chapter returns to reconstruction algorithms, establishing the concepts of poles and cocones necessary to derive an algorithm for extracting a manifold surface from a set of sampled points. The algorithm (`COCONE`) is based on computing the Voronoi diagram of the set of sampled points, and two additional algorithms defined in this chapter – `COCONETRIANGLES` and `EXTRACTMANIFOLD`. `COCONETRIANGLES` is used to determine which Voronoi edges can be used to build a set of cocone triangles which are then used to extract the manifold. First, a method for pruning the set of cocone triangles is discussed, which is a necessary step between building the set of cocone triangles and extracting the manifold. Next, the `EXTRACTMANIFOLD` algorithm is discussed based on a final algorithm, `SURFTRIANGLES`, which is used to build the set of triangles representing the reconstructed surface.

Chapter 5: Undersampling. This chapter discusses the topic of undersampling and how it can be detected and compensated for during reconstructions. Topics covered include boundaries and sampled points, the concept of flat sample points and flatness analysis. These concepts are then used to build algorithms to detect flat boundary sample points (`ISFLAT`), compute the set of boundary sample points (`BOUNDARY`), and a revisited version of the cocone algorithm (`BOUNDCOCONE`) that takes into account boundary sample points when performing the reconstruction. Simple visual examples are provided to show the effect these changes have on the reconstructions created by the algorithms in chapter 4 versus chapter 5.

Chapter 6: Watertight Reconstructions. As the name of the chapter hints, reconstructions based on the algorithms up to this point may contain holes. It is important for some applications that these holes be eliminated. This chapter presents the `POWERCRUST` and `TIGHTCOCONE` algorithms. The concept of power diagrams (a weighted generalization of the Voronoi diagram) is introduced, along with the notion of inner and outer poles needed to define the power crust as a mathematical construct. The relationship of the power crust to the surface is analyzed. The `POWERCRUST` algorithm is presented based on an algorithm for labeling poles (`LABELPOLES`). The `TIGHTCOCONE` algorithm is then presented based on two other algorithms for marking the Delaunay triangulation of the point set and peeling the set of tetrahedra that are not considered part of the reconstructed surface. The figures in this chapter nicely illustrate these algorithms.

Chapter 7: Noisy Samples. This chapter discusses a topic that frequently arises when sampling physical objects: noise. The text gives a detailed treatment of the types of noise that can occur with respect to the surface to reconstruct (e.g.: tangential and normal scatter), and how one can analyze and compensate for this noise during reconstructions. This chapter doesn't directly address the reconstruction topic (this is left to chapter 8), and instead discusses algorithms for approximating normals at sampled points and approximating features in the presence of noise.

Chapter 8: Noise and Reconstruction. The focus of this chapter is deriving the necessary mathematical basis upon which the `ROBUSTCOCONE` algorithm is designed such that the reconstructed surface is homeomorphic to the actual surface in the presence of noisy samples. The chapter starts by laying out preliminary concepts related to sampling and the topological properties of Delaunay balls and unions of balls. The proximity of this union to the surface being reconstructed is analyzed to establish topological equivalence of the reconstruction. The chapter ends by defining the `ROBUSTCOCONE` algorithm and showing its use in two instances with noisy samples.

Chapter 9: Implicit Surface-Based Reconstruction. This chapter discusses the reconstruction problem using the concept of implicit functions. An example of when this would be useful is to avoid bumpy reconstructions due to noisy samples. The chapter introduces the concept of an implicit function, and discusses moving least squares (MLS) surfaces. Much of the chapter looks at the properties of the surfaces and defining the Adaptive MLS algorithm, with the later portion of the chapter discussing variants of MLS and other implicit reconstruction methods.

Chapter 10: Morse Theoretic Reconstructions. The final chapter of the book discusses Morse theory from differential topology and its application to reconstruction problems. Morse functions and flows are introduced, along with stable and unstable manifolds. The author then discusses the relationship of these concepts from Morse theory to the Voronoi and Delaunay diagrams employed throughout the previous chapters, such as how critical points of a function relate to both diagrams. The `SMRECON` and `WRAP` reconstruction algorithms are also introduced in this chapter.

3 Opinion

This text provides an understandable treatment of reconstruction algorithms. Readers with at least a basic understanding of topology and geometry should find the text readable and informative. Readers who have no experience at all with topology and computational geometry may require additional references to clarify concepts that are only touched upon briefly in this text. The only significant issue that I had with the text was that much of the content is based on the Delaunay triangulation and Voronoi diagram algorithms from computational geometry. The discussion of the Delaunay triangulation and Voronoi diagram provided in Chapter 1 seemed too basic for readers who may not already be familiar with them. The author recognized this and refers the reader to additional texts for this information, so this minor omission should not be considered a flaw of the text. Many texts (including those referred to by the author) can be used to fill in this gap should the reader be unfamiliar with these concepts and algorithms.

The text is well written, and presents the algorithms in a way that makes them quite understandable. Instead of presenting the algorithms as single, monolithic and complex methods, they are broken into parts that can be explained and mathematically analyzed in an order that makes clear how they are later composed into the larger task. For example, in the description of Chapter 4 above, we see that the algorithm for extracting a 3D surface from a set of points is actually presented as a set of smaller algorithms. This pattern of breaking algorithms into analyzable pieces aided in understanding how the overall reconstruction methods worked. Readers will likely benefit from skimming each chapter, reading the algorithms, and then revisiting the proofs and derivations to appreciate how they relate.

Review of[7]
Concentration of Measure for the Analysis of Randomized Algorithms
by Devdatt P. Dubhashi and Alessandro Panconesi
Cambridge University Press, 2009
196 pages, Hardcover

Review by
Aravind Srinivasan, srin@cs.umd.edu
Dept. of Computer Science, University of Maryland at College Park, USA

1 Introduction

The utility of probabilistic methods in the computational context is one of the fundamental discoveries of computer science over the last four decades. In the analysis (and design) of randomized algorithms or stochastic processes, the tool required by far is one showing that certain types of random variables remain "close enough" to their means – or medians, in a few cases – with high probability. In fact, many randomized algorithms can be designed using the following recipe: "assume that the relevant random variables stay close enough to their means, and prove that everything works out".

The trick, of course, is to be able to prove such "concentration of measure" results. While the Chernoff bounds are perhaps well-known to theoretical computer scientists, their independence assumptions often do not hold, and we need more sophisticated machinery to handle all manner of correlations. This book brings together essentially all of the relevant state-of-the-art in a manner that is particularly appealing to computer scientists and to those with discrete sensibilities.

A disclaimer: I have co-authored papers with both of the authors, but believe this review to be unbiased.

2 Summary

The book does a superb job of describing a collection of powerful methodologies in a unified manner; what is even more striking is that basic combinatorial and probabilistic language is used in bringing out the power of such approaches. Once a reader is armed with the primary topics covered here – e.g., kth moment, Chernoff-Hoeffding, negative dependence (a powerful characterization of the intuitive notion of "negative correlation"), Janson's inequality, bounded differences & Martingales, concentration for multivariate polynomials, Talagrand, and Log-Sobolev inequalities, he or she would be well on their way to the design and analysis of randomized algorithms. There are many illustrative exercises, and solid undergraduate- or graduate-classes can be designed substantially based on this book. The book would be of value to anyone working in theoretical computer science.

Chapter 1 deals with the moment-generating-function approach of Chernoff and Hoeffding, gives simplified bounds in various cases (e.g., when the relative deviation from the mean is "small" or "large") and draws connection to the relative entropy between distributions. Improved bounds are derived when the underlying summands have small variances. Chapter 2 develops applications of

the Chernoff-Hoeffding bounds, many of which may be familiar to theoretical computer scientists: e.g., load balancing, skip lists, and the Johnson-Lindenstrauss Lemma.

Chapter 3 takes the first step in extending the Chernoff-Hoeffding bounds to dependent settings. Perhaps the four major extensions presented here are for limited independence, Janson's inequality, negative dependence, and Markov dependence. The first two may be familiar to many. The third is an important extension, of which "balls and bins" is a prototypical example: intuitively, if one bin gets many balls, you expect other bins to receive less. The fourth is about Markov chains with a good eigenvalue gap, a very good example of which is "deterministic amplification" for languages in BPP.

While the Chernoff-Hoeffding approach is primarily useful for sums of random variables, Chapters 5–8 deal with "smooth" functions of many (almost) independent random variables: in particular, when most partial derivatives are small, one intuitively expects good tail bounds. These chapters develop this phenomenon in detail, via Martingales and the method of bounded differences. In particular, the usual measure-theoretic terminology is avoided, and the discussion is based entirely on discrete probability. As mentioned at the beginning of this section, this is one of the key contributions of the book.

Chapters 10–13 develop isoperimetric inequalities and Talagrand's powerful isoperimetric inequality (concentration around the median). Chapters 12–14 have their roots in information theory; Chapters 12 and 13 develop an approach to, and generalization of, Talagrand's inequality using *transportation-cost* inequalities (due to Marton and Dembo). Chapter 14 develops the Log-Sobolev inequalities, which lead to a simpler proof of Talagrand's inequality: I am still digesting this material.

Chapters 4 and 9 present very useful "interludes": on probabilistic recurrences (which are typical in randomized divide-and-conquer algorithms), and the very important work of Kim, Vu, Janson, and Ruciński on concentration – especially in the notorious upper-tail region – of multi-variate polynomials of (independent) random variables.

One would have quibbles with any book: here are mine. Perhaps the authors can address some of these in the next edition, and/or in a Web-based companion.

One high-level comment is that a warmup section on the linearity of expectation along with Markov's & Chebyshev's inequalities and their limitations, can motivate quantitative comparisons in much of the book. Second, while the Lovász Local Lemma is not of direct relevance to concentration inequalities, its proof is quite similar to that (due to Boppana and Spencer) for Janson's basic inequality, and the connection may be fruitful for the reader to know; also, the Local Lemma is of course a powerful probabilistic tool. A short companion on the Local Lemma, while discussing Janson's inequality in Chapter 3, would be a good addition.

Some additional comments:

1. Last line of page 4: the negative sign should be removed from the definition of relative entropy. (In the footnote on page 4, it should instead be noted that if q is uniform, then $H(p,q) = \log n - H(p)$.)

2. In Section 3.1, should not the functions f, g etc. be further restricted to be non-negative? (The proof of Proposition 3.1 handles arbitrary non-decreasing functions, but are the general results similarly valid in such a case?)

3. Page 41, first displayed equation: $e^{c^3/36}$ should be $e^{-c^3/36}$.

4. In the discussion of [4] in pages 44–45, the authors could add that [4] also gives the stronger bound, where "$(nk/t^2)^{k/2}$" is essentially replaced by "$((\mu k + k^2)/t^2)^{k/2}$". This is essential for the usual target of deviations of the form $\sqrt{\mu\lambda}$ for some ("small") λ.

5. Page 53: in the two tail bounds at the very end of the page, I believe the two occurrences of "n" should be replaced by "m". Also, I am sure the authors are aware of the following alternative simple proof, say for the running-time bound, which is well worth spelling out as a useful tool. An easy application of Bayes' Theorem and induction on i shows that if M_i denotes the number of edges remaining after i iterations, then $\mathbf{E}[M_i] \leq m \cdot (7/8)^i$. Markov's inequality applied to this, directly yields the tail-bound on $T(G)$ that is asked for here.

6. Page 123, line before Section 9.3: "$\Theta(n^{-3/8})$" should instead be "-$\Theta(n^{3/8})$".

7. A spell-check is needed: I found some typographical errors.

To summarize, the book has done a great job of synthesizing diverse and important material in a very accessible manner.

3 Opinion

This book essentially only requires discrete math, discrete probability, and "mathematical maturity" at the level of a senior undergraduate. Any student, researcher, or practitioner of computer science, electrical engineering, mathematics, operations research, and related fields, could benefit from this wonderful book. The book would also make for fruitful classes at the undergraduate- and graduate- levels. I highly recommend it.

Review of
The Modern Algebra of Information Retrieval [8]
Author: Sandor Dominich
Publisher: Springer-Verlag Berlin Heidelberg
ISBN: 978-3-540-77658-1, $89.94 (amazon.com)

Reviewer: John S. Griffin (johng.sst@gmail.com)

1 Overview

Very briefly, but very exactly, Information Retrieval means *finding relevant information in a store of information* (relevant being the key word here). A broader definition would be that Information retrieval (IR) is the science of searching documents for information in their contents and from their metadata. Searching relational databases and the World Wide Web also applies here. With the advent of the Internet and World Wide Web (Web for short), IR has acquired tremendous practical impact as well as theoretical importance. This book treats retrieval methods (major proven models and ranking techniques) and IR in general in a unified manner within the one formal framework of modern algebra, namely abstract algebraic structures (primarily lattices). In the simplest of terms

[8]

a lattice is a 'well-behaved' partially ordered set (also called a poset). It is well behaved because any two of its elements:

- have a unique supremum (the elements' least upper bound; called their join)
- have a unique infimum (greatest lower bound; called their meet)

This text attempts to demonstrate how lattices can be used in information retrieval. Exercises and problems are located at the end of every chapter except 1 and 7. These exercises are IR related and used to reinforce ideas discussed in each chapter. A section at the end of the book contains hints on solving these problems.

A detailed chapter by chapter discussion of contents follows.

2 Summary of Contents

Chapter 1 The book starts with an introduction by way of a quick condensed discussion of the main topics and ideas presented in this book.

Chapter 2 Thankfully this chapter is a review of mathematics necessary for the text. It presents the concepts, operations, and properties of mathematical logic (proposition, negation, conjunction, disjunction, implication,equivalence), set theory (sets, set operations), and relations theory (binary relations, functions, equivalence relations, posets) that are being applied in modern computerized (IR) and which are used in the modern algebra of IR.

Chapter 3 Concepts and properties that pertain to lattice theory (lattice, posed, duality, Hasse diagrams, atomicity, modularity, distributivity, complementation, orthocomplementation, orthomodularity, Boolean algebra, important lattices) that are applied in the theory of information retrieval (IR), in the development of IR systems, and in the major retrieval methods are discussed.

Chapter 4 This chapter introduces the basics of information retrieval technology (document, stoplist, term, power law, stemming, inverted tile structure weighting schemes, term-document matrix, architecture of retrieval system, architecture of a search engine, relevance effectiveness, measures and measurement, precision-recall graph method, search engine effectiveness measurement). These are presented, first, for practitioners, and the material will be useful for those interested in developing practical retrieval systems. However, this material may also be helpful for those with a theoretically bent as well as enabling a better understanding of the chapters that follow.

Chapter 5 This chapter describes the application of lattices in retrieval systems (Mooers, FaIR, BR-Explorer, Rajapakse-Denham, FooCA). The use of latices for visualization or navigation is not considered, nor are programming and implementation issues dealt with as these are outside the scope of the book. The goal of describing these systems is to present the way in which lattices have been used to represent document structures, term relationships, and term-document matrices in actual retrieval systems developed thus far.

Mathematical properties (with proofs) of the lattices applied are established. The principal finding is that they are not modular. Further, a method is given to transform a term-document matrix into a Galois (concept) lattice[9].

Chapters 6 The Boolean retrieval method is a very important one as it is widely used in database systems (e.g., Oracle, SQL) and World Wide Web search engines. In principle, it is a

[9]See the short discussion of the Galois Lattice at the end of this review.

simple method, but a1l the more important for that. This chapter describes the Boolean retrieval method (both formally and using an example) and the application of lattices in Boolean retrieval. An effective method is presented to answer Boolean queries in relational databases.

Chapter 7 This chapter presents tile notions and results (metric space, complete space, linear space, subspace, linear operator, Banach space, Hilbert space, Euclidean space, projection theorem, projector, lattice of subspaces) that are applied in Chapters 8 and 9 (on vector space retrieval and algebra-based retrieval methods). Every notion is illustrated with detailed intuitive or mathematical examples to promote better understanding of their meaning.

The Gram-Schmidt procedure for defining an orthonormal basis of a subspace of a linear space is described, the lattice of closed subspaces of a Hilbert space is defined, and the way in which this lattice expresses the underlying geometry of the space is shown.

Chapter 8 This chapter begins with the original as well as a more formal description of vector space retrieval (VSR). An example is also given to the reader to help exactly understand what the method means and how it operates.

Then, the widely used similarity measures are presented in both a compact and parameterized form (having in mind a computer programmer who refers writing a compact code for a1l cases) and in their usual forms.

This is followed by a description of the use of the notion of a projector in Hilbert space for the calculation of meaning and for the expression of compatibility of relevance assessments.

The second part of the chapter is concerned with the application of lattices in VSR. It is shown that retrieving documents means projection. After introducing the concept of equivalent queries, the proof that nonequivalent queries form a nondistribution lattice (called a query lattice) is given. It is also show that VSR may be viewed as a nonsubmodular lattice-lattice mapping from the query lattice to the Boolean algebra of documents. A parallel is drawn between the lattice-based view of quantum mechanics and the lattice-based view of IR introduced in this chapter, and that is discussed.

Chapter 9 This chapter explains how fuzzy algebras can be used to provide new or novel retrieval methods. After presenting the necessary elements of tensor algebra, it is show that when the formal framework of information retrieval is a linear space of terms, the scalar product of the space is not necessarily a similarity measurement–contrary to the widely held belief.

Next, the required notions and results from fuzzy set theory are presented and it is shown that the set of a11 fuzzy sets in (0; 1) is a fuzzy algebra. Documents and queries are elements of this algebra. By introducing the principle of invariance, latent semantic indexing, vector space retrieval, and generalized vector space retrieval acquire a correct formal framework with which they are consistent (as opposed to the linear space as a framework). Based on the notion of fuzzy algebra, the fuzzy entropy method and the fuzzy probability method are discussed, together with experimental results as to their relevance effectiveness.

Chapter 10 After reviewing the necessary notions and results from probability theory probability measure, event space, relative frequency, independent events, conditional probability, Bayes's theorem), probabilistic retrieval methods are presented (probability ranking principle, Bayes's decision rule, nonbinary method, language model) together with examples.

After discussing various formal frameworks for probabilistic retrieval, a framework using lattices (distributive lattice of logical implications) is proposed. The notion of a Bayesian network is defined as a special kind of distributive lattice, and the inference retrieval method is described as an application of Bayesian networks.

Chapter 11 This is by far the longest chapter of the book. After introducing the notion of a Web graph and discussing degree distribution, the basic methods using link structure analysis (impact factor, connectivity, mutual citation, PageRank, HITS, SALSA, associative-interaction) are presented together with clarifying examples for each. A connection between HITS and LSI is also shown.

Then, an aggregated method for Web retrieval based on lattices is presented that allows one to calculate the importance of pages, taking into account both their link importance (using link analysis) and their intrinsic importance (stemming from page content). Experimental evidence for the relevance effectiveness of this method is also given in terms of comparison with commercial search engines (with Google, Altavista, Yahoo!).

After introducing the notion of Web lattice and chain, Web ranking as a lattice-lattice function between a Web lattice and a chain are defined. It is shown that ranking is not submodular. Then, global ranking is defined as a lattice-lattice function (i.e., a mapping from the direct product of Web lattices to the chain (0; 1). It is also shown that global ranking is not submodular.

Based on the concept of global ranking, a method that allows computing the local Importance of a Web page at Web level is presented, taking into account the importance of the site the page belongs to, but without the need to consider the entire Web graph of a11 pages.

After proving that any tree as well as any document can be transformed into a lattice, it is shown that the DocBall model and Galois (concept) lattice representations of a document are equivalent to one another.

Based on these results as well as on the fact that the structure of any site is a lattice, a method for computing site importance is described.

3 Opinion

Although there is a plethora of information on the Internet concerning lattices and their applicability to IR, there are also opinions out there that they are, in a practical sense, difficult to implement. This may be born out by the fact that this book is not an easy read. It is extremely comprehensive on the topic, perhaps too much so. Witness the (admittedly small) sections on Tensor Analysis and Quantum Mechanics (really necessary?). This is a huge amount of information packed into a small space. On the plus side, a precise description of every method is given in detail. For every method, a complete example is provided to enhance understanding.

For those interested in investigating this topic, the book should be helpful for a wide range of readers from graduate students and educators to researchers and system developers coming from a variety of fields such as computer science, mathematics, information science, engineering, logics and physics. This reviewer does, however, recommend that the reader be fluent in the fundamental concepts of Set Theory. If not, then spend some additional time in chapter 2. Prior basic knowledge of Lattice Theory would also help make this an easier read.

Every chapter concludes with a *References and further reading* section. These references would make it easy for an instructor to assign research and in-class presentations by students (just a suggestion).

With these thoughts in mind, this reviewer would recommend this book for the above mentioned people but not for undergraduates. Just be prepared for a heavy dose of abstraction and theory.

The Galois Lattice Formal concept analysis takes as input a matrix specifying a set of objects and the properties of those objects, called attributes, and finds both all the *natural* clusters of attributes and all the *natural* clusters of objects in the input data, where

- a *natural* object cluster is the set of all objects that share a common subset of attributes, and
- a *natural* property cluster is the set of all attributes shared by one of the natural object clusters

Natural property clusters correspond one-for-one with natural object clusters, and a *concept* is a pair containing both a natural property cluster and its corresponding natural object cluster. The family of these concepts obeys the mathematical axioms defining a lattice, and is called a *concept lattice* or ***Galois lattice***.

Review of [10]
Multiagent Systems: Algorithmic, Game-Theoretic, and Logical Foundations
by Y. Shoham and K. Leyton-Brown
Cambridge University Press, 2008
480 pages, Hardcover

Review by
Haris Aziz (haris.aziz@warwick.ac.uk)
Computer Science Department, University of Warwick, Coventry, CV4 7AL

1 Introduction

Game theory and social choice theory have traditionally been in the domain of economics. However, with the growth of the internet, they are increasingly being used by computer scientists as tools to analyse distributed settings. Nisan [6] points out the the two way flow of ideas between economics and computer science. Not only are algorithmic challenges taking into account social choice and game-theoretic paradigms but various economic interactions such as voting, coalition formation and resource allocation are requiring deeper algorithmic study. The same trend has also been pointed out in [2] and [7].

With these trends, network economics, game theory, learning, electronic commerce and logical theories have all received interest of theoretical computer scientists. The book is a holistic effort to combine all these strands under the unified umbrella of multiagent systems and introduce their algorithmic, game theoretic and logical foundations. The authors refrain from giving a constrained definition and emphasize the open-ended and inter-disciplinary nature of multiagent systems. This makes sense since multiagent systems have connections with fields as diverse as micoeconomics, linguistics and analytic philosophy.

2 Summary

The book is composed of fourteen chapters and four appendices. Chapter 1 and 2 cover distributed problem solving. Chapter 1 concerns constraint satisfaction in a distributed environment such as sensor networks. Among other topics, the chapter gives an overview of domain pruning and heuristic search algorithms. Chapter 2 covers distributed optimization in which different approaches are introduced such as distributed dynamic programming, Markov decision problems, economics tools and coordination through social laws.

Since game theory is a key tool to analyse multiagent systems and the theme of the book includes foundations of multiagent systems, Chapters 3, 4, 5 and 6 are devoted to noncooperative game theory. Chapter 3 is an introduction to non-cooperative game theory with a helpful use of classical example such as *Battle of the Sexes*, *Prisoners Dilemma*, *Matching Pennies*, *Rock, Paper, Scissors* etc. Apart from Nash equilibrium, other solution concepts such maxmin and minmax strategies, Minimax regret, removal of dominated strategies, rationalizability, correlated equilibrium, trembling-hand perfect equilibrium and epsilon-Nash equilibrium are also introduced. Chapter 4 is an excellent survey for some one entering the field of algorithmic game theory. It introduces computation of Nash equlibria in different kind of games in normal forms. Recent complexity results by Papadimitriou, von Stengel and others are also covered. Chapter 5 addresses extenstive games with perfect and imperfect information. In Chapter 6, various other rich representations of games such as repeated games, stochastic games, Bayesian games and congestion games are discussed. Computing of equilibria is given special consideration.

The learning aspect of multiagent is covered in Chapter 7. Among other models, ficticious play, rational learning, targetted learning and evolutionary learning are introduced. Chapter 8 covers the communication aspects of multiagent systems. Topics addressed include signalling games, speech-act theory and applications. Game theoretic aspects of multiagent communication are also explored.

Chapter 9, 10, 11 and 12 cover protocols and solutions for groups of agents. Chapter 9 covers social choice theory and voting theory. Desirable properties of social choice functions are well-explained. There is a separate section on the axiomatic properties of ranking systems. Chapter 10 introduces the rapidly growing area of mechanism design in which mechanisms are designed while taking account that agents will try to act strategically to maximize their personal utilities. Chapter 11 provides separate explanation of auctions which are one of the most important tools in mechanism design and have also been used in sponsored search algorithms. The chapter also covers variations of combinatorial auctions which have attracted considerable interest recently [1]. Chapter 12 provides coverage of cooperative game theory which also has bearing on resource allocation. Properties of different cooperative game solutions are covered with a particularly good treatment of the Shapley value.

The book concludes with Chapter 13 and Chapter 14 on logical theories, another cornerstone of multiagent system design and analysis. Chapter 13 is on the logic of knowledge and belief whereas belief revision models and logic of intention are covered in Chapter 14. There are also appendices which provides introductions to probability theory (Appendix A), linear & integer programming (Appendix B), Markov decision problems (Appendix C) and mathematical logic (Appendix D).

3 Opinion

The book with its comprehensive and broad treatment promises to become the standard textbook on multiagent systems. It provides a useful and easy to read survey of a rapidly growing field.

Compared to other artificial intelligence and multiagent books, the book is unique in its coverage of algorithmic and game theoretic foundations of multiagents. One of the established texts in multiagent systems is *An Introduction to Multiagent Systems* by Wooldridge [9] but it is more focused on logic and process algebra with not as much coverage of algorithms and game theory. Another book on the market is *Multiagent systems: a modern approach to distributed artificial intelligence* by Gerhard Weiss [8]. However, the book by Weiss is more artificial intelligence practitioner oriented. The format of the book by Weiss is that of a handbook with chapters which are self contained and written by different authors. On the other hand, the book by Shoham and K. Leyton-Brown is more well-knit and flowing. In terms of its essence, it appears more similar in flavour to *Algorithmic Game Theory* [5] but much broader in its scope (with additional aspects of multiagent systems such as learning, communication and modal logics of knowledge and belief).

Although many recent results are included, the clear handling of the subject makes it suitable for an advanced undergraduate or graduate textbook. This seems to be the intention as a website (`http://www.masfoundations.org`) has been maintained to include teaching resources. The website also has links to an errata page. The book is suitable for self-study and it is an advantage that there are no superfluous examples or exercise questions to distract from the core text. The reader is assumed to have some mathematical maturity and a basic background in algorithms and complexity.

The book has many positive features. A commendable aspect of the book is that there are many concise and self contained proofs or proof ideas of important mathematical results such as Brouwer's fixed-point theorem, existence of Nash equilibria, Arrow's theorem, Minimax theorem, Folk theorem, Revelation principle and Revenue equivalence theorem. Computational complexity of computing Nash equilibria of a general-sum finite game is provided good coverage.

Another appealing factor of the book is clarity. The text is one of the clearest expositions of concepts in game theory. The keywords on the margins of the paragraphs further help in clarifying the logical flow of the ideas and topics. In many textbooks, the development of the field and references to key texts are not given attention. However, Shoham and Leyton-Brown have a note on the history and key references at the end of each chapter. This gives a better idea of the historical development of the field and also provides directions for deeper study. The index is also well-organized.

If there are any possible improvements, two spring to mind. Chapter 12 could have benefited from an even more extensive coverage of algorithmic cooperative game theory [3, 4]. Secondly, although the book assumes basic grounding in algorithms, it would also have been an added advantage to include an appendix on computational complexity where there are already appendices on basic concepts in probability theory and linear programming. This would make the complexity results in the book more accessible for readers with no computational complexity background such as pure economists.

In any case, the well-balanced treatment of multiagent systems will make the book useful to both theoretical computer scientists and the more applied artificial intelligence community. Moreover, the interdisciplinary nature of the subject makes it relevant not only to computer scientists but also to people from operations research and microeconomics (social choice and game theory in

particular). The book easily deserves to be on the shelf of any modern theoretical computer scientist.

References

[1] In P. Cramton, Y. Shoham, and R. Steinberg, editors, *Combinatorial Auctions*. The MIT Press, 2007.

[2] Y. Chevaleyre, U. Endriss, J. Lang, and N. Maudet. A short introduction to computational social choice. In *Proceedings of the 33rd Conference on Current Trends in Theory and Practice of Computer Science (SOFSEM-2007)*, volume 4362 of *LNCS*, pages 51–69. Springer-Verlag, January 2007.

[3] I. Curiel. *Cooperative Game Theory and Applications: Cooperative Games Arising from Combinatorial Optimization Problems*. Springer, 1997.

[4] X. Deng and Q. Fang. Algorithmic cooperative game theory. In *Pareto Optimality, Game Theory And Equilibria*, volume 17 of *Springer Optimization and Its Applications*, pages 159–185. Springer, 2008.

[5] N. Nisan, T. Roughgarden, E. Tardos, and V. V. Vazirani. *Algorithmic Game Theory*. Cambridge University Press, New York, NY, USA, 2007.

[6] N.Nisan. Introduction to mechanism design. In N. Nisan, T. Roughgarden, E. Tardos, and V. Vazirani, editors, *Algorithmic Game Theory*. Cambridge University Press.

[7] Y. Shoham. Computer science and game theory. *Commun. ACM*, 51(8):74–79, 2008.

[8] G. Weiss, editor. *Multiagent systems: a modern approach to distributed artificial intelligence*. MIT Press, Cambridge, MA, USA, 1999.

[9] M. Wooldridge. *An Introduction to Multi-agent Systems*. John Wiley & Sons, June.

Review of[11]
Book Title: The Political Mapping of Cyberspace
Author: Jeremy W. Crampton
Publisher: The University of Chicago Press, 2003

Reviewer: Rajesh Natarajan
Data Warehousing (Business Analytics Group)
Cognizant Technology Solutions India Pvt. Ltd.
Techno Complex, 5/535, Old Mahabalipuram Road
Okkiyam, Thoraipakkam, Chennai 600 096
Tamil Nadu, India.

The Internet, that vast network that connects millions of computers across the world, has become a *place* wherein we can undertake our day-to-day transactions like banking, purchasing and the like or simply *hang-out* with our friends on-line. The Internet has shrunk the world in terms of distances, made time and location differences a non-issue and in general is increasingly influencing the way we do our work. Many researchers have sought to study the cyberspace and the influence of WWW and Internet on human behavior in particular and society in general. Post the dot-com bust, the topics of these studies have shown a gradual shift from the Internet or information itself as the unit of analysis to the study of the Internet as an important constituent of human society.

The Political Mapping of Cyberspace by Jeremy W. Crampton continues this tradition with an illuminating, innovative and philosophical study of the *spatial politics of cyberspace*. The goal, according to the author is to investigate how we find our place in the world with cyberspace being the domain. The author seeks to extend some, while redefine other cartographic notions of space and being to the cyberspace. Cyberspace in the author's view is an area of geographic knowledge that sits equally between society and technology and a classic case of space that is produced and which in turn produces subjectivity. The thought-provoking discussions of the issues and the very original conceptualization of cyberspace are the main highlights of the book.

An introductory discussion of the political nature of mapping sets the context. Thus, we are informed about the reasons that make the political always spatial, how mapping becomes an important part of our engagement with space and consequently a part of our political concern. Not withstanding its non-physical nature, the cyberspace is an extension of our physical and mental space. Thus, the question *how we find our place in cyberspace* acquires much importance. Using Foucault's idea of problematization, the book attempts to address this and other related questions and thus open up the possibilities for a critical politics of space.

The first section sets the context for the entire book with a short but engaging discussion on the historical deployment of Internet mapping and its consequences, namely, the production of cyberspace. The Internet and important new developments in cartography, GIS and geography have fundamentally changed the way spatial data is accessed, analyzed and communicated (Crampton and Krygier, 2006). The author uses the term *distributed mapping* with reference to the developments in cartography, GIS and geography in order to capture its highly dispersed and multi-user nature. Internet maps are transient in nature in that they exist for a few minutes as opposed to traditional maps that last for years. In addition, an important change from the perspective of the user is that he/she is virtually playing the role of the cartographer with increasing levels of

interactivity. The author contends that such developments make distributed mapping a significant break rather than a continuation with traditional cartography.

An issue that acquires some significance is the political nature of mapping. The author rejects the Cartesian scientific world-view that casts maps as communicators of spatial locations. Instead, he conceptualizes mapping as a problematization that leads him to think about being as such including the being of maps. How does a map become political? This question is sought to be answered by integrating the work of Monmonier, who pointed out that it is in the nature of maps to lie and that not only is it easy to lie with maps, its essential. The critical politics of cartography is sought to be pursued in numerous ways as a problematization in Foucault's sense, as an ethics, as a struggle and finally as a question of technology. An interesting issue, that has not been considered here, but which brings out the political nature of mapping is that of Google Earth. Google's project of providing high resolution satellite imagery maps of earth ran into rough weather with many countries. The detailed high resolution satellite imagery reveals the locations of many hitherto classified locations. With such knowledge becoming public, there were fears as regard the possibilities of its use for terrorist and other nefarious activities.

Entry into the cyberspace is usually associated with some kind of authentication mechanism. The usual physical mode of authentication that we are familiar in our daily lives cannot be used in the case of cyberspace. The second part of the book deals with the twin subjects of authenticity and confession. Some well-known authentication mechanisms are discussed along with their possible implications. With further advances in technology and as the Internet gets integrated into everyday routine, technologies and mechanisms that authenticate the identity of a person in cyberspace may serve the dual purpose of authentication in the real world too. A good example is the use of the same password to authenticate a person both in cyberspace while involved in Internet banking and while using the ATM. An implicit danger in this case is the possibility of authorization, in the sense of permission to access, easily slipping to become authorization in the sense of governing ones being and identity. The book enlivens this discussion through usage of examples and analogies.

With increasingly nefarious and ingenious methods of obtaining access to networks, correct authentication of user's identity has become important. Physical separation has made it necessary to ascertain whether the entity on the other side seeking authentication is human or an automaton such as a software agent. Consequently, web sites that offer free services such as emails are now employing ingenious methods like asking the respondent to interpret a particular sequence of letter and alphabets in an image file as a authentication mechanism during registration. This presumably will require human interpretation and thus disqualify software agents that register with the nefarious intentions of flooding spam emails. The author can consider this and related issues in the future.

After discussing the role of authentication in producing an identity, the author examines the competing roles of confession and parrhesia (free speech) in the production of the self in cyberspace. This is done against the backdrop of blogging, an activity that both takes place in and produces community. The virtual nature of cyberspace encourages a kind of openness that is extremely difficult to imagine in normal face-to-face encounters. The sense of security that arises as a consequence of physical distance has led many community participants to be very trusting. This both helps in community building and also allows considerable scope for manipulation. The author stresses the fact that the need for authentication in cyberspace calls forth confession. Cyberspace is also inherently confessional in that we always already renounce our bodies entirely in order to free the inner true self. There is an interesting discussion on how confession can be achieved in cyberspace. This is followed by an engaging account of how resistance in cyberspace is a process that works against

the process of normalization in everyday life. The author uses the experiences of the science fiction writer Philip K. Dick to bring forth one interesting characteristic of cyberspace namely, the fact that it encourages Parrhesia (frank speech). However, in both these accounts, the relationship of confession and authentication to cyber-spatial mapping and politics is not immediately apparent.

The third part of the book deals with case studies in the production of cyberspace. Here, the previous discussions on philosophy, mapping, cyberspace and virtual communities are thoughtfully tied together using the examples of digital cyberspace in crime-mapping and the differential access to technology that leads to what is termed as the digital divide. Digital mapping and GIS have assumed an important role in the practice of security that is increasingly becoming political. The author uses the case study of crime-mapping to analyze the deployment of mapping and GIS. He has highlighted how a rationality of carto-security is constructed in which geo-surveillance is deployed as a response to dangerousness, and in which people are constructed as at-risk resources subject to normalization and management. Advances in geographical crime-mapping have made surveillance straightforward, more rigorous and effective yet non-intrusive. The resulting increase in geo-profiling and geography-based surveillance has kindled fears about the loss of privacy. The author discusses the privacy issue extensively by examining different viewpoints. One appealing perspective is that the loss of privacy is not due to technology per se but can be viewed as a key symptom of one of the fundamental social problems of our age namely, the growing power of large public and private institutions in relation to the individual citizen. The author examines related privacy issues and brings out the risks of security within a broad Foucauldian perspective.

The second case study is related to the geography of the digital divide. Digital-divide is viewed as resulting from unequal access to knowledge in the information society. This definition is wide encompassing and goes beyond just the technological namely, access to the Internet. The other equally important aspects that contribute to digital divide include the knowledge of how to use the tools and what to access. These arise due to differential access to relevant content, take-up of resources and deployment of skills. The author examines the antecedents of the digital divide at various levels, the global, the regional and the local levels. One way to address this digital divide is by optimally locating the cyber centers using GIS, such that access to the Internet is maximized. The author correctly concludes that the issues concerning the digital divide are more than mere technical issues. Reducing the differential access to resources and thus the digital divide would require more proactive and creative initiatives ones that could be based on GIS.

According to Foucault, there is no knowledge outside of power relations and no power relations that do not involve and create knowledge. However, Foucault's conceptualization of power has been viewed with an underlying negative connotation i.e. power as being bad, politically repressive and politically counterproductive. However, the author challenges the conventional view of Foucault's work and philosophy by bringing out the positivities of power in the production of subjectivity. However, the relationship with mapping in the cyberspace is not immediately apparent. The final section of conclusion is an interesting analysis of how mapping can be used as a medium that induces/produces pleasure. This is tied up to the childhood interest in place and space the pleasures of exploring and discovery, of finding ones place in the world with a map. To conclude, we can say that this book examines the numerous ways by which space, politics and mapping may be productively brought together in the context of cyberspace. The book's chief contribution is that it examines not just what could be revealed about the landscape through the process of mapping, but perhaps how the process itself reveals much more than that and the tremendous opportunities that open up.

Although the author, Crampton has primarily used the Foucauldian concepts of problematization, subjectification, and technologies of the self prominently, these are also contrasted and sometimes related to the ideas of other theorists like Martin Heidegger, Brian Harley, Paul Virilio and Mark Monmonier. This not only brings in diverse viewpoints but also elevates the discussions wherein the reader can himself/herself get a glimpse of the numerous possibilities. This book, though dealing with dense concepts, has been lucidly written. However, it is not immediately apparent who the target audience is. The book can serve as an introductory text on the politics of mapping with cyberspace as the context. However, the apparent diversity of topics and complexities of related issues calls for a more exhaustive treatment, perhaps in a future work by the author. Nevertheless, the rigor and the painstaking hard work of the author that shines through the lucid discussions, makes this work a possible authoritative source and an important reference for future critical cyberspace studies that deal with the interplay between cartography and politics.

References: Crampton, J. W. and Krygier J., (2006), *An Introduction to Critical Cartography*, ACME: An International E-Journal for Critical Geographies, 4 (1), pp. 11-33.

Review of [12]
The Princeton Companion to Mathematics
Timothy Gowers, June Barrow-Green and Imre Leader
Princeton University Press, 2008
1008 pages, Hardcover

Review by
Haris Aziz (haris.aziz@warwick.ac.uk)
Computer Science Department, University of Warwick, Coventry, CV4 7AL

1 Introduction

> *To most outsiders, modern mathematics is unknown territory. Its borders are protected by dense thickets of technical terms; its landscapes are a mass of indecipherable equations and incomprehensible concepts. Few realize that the world of modern mathematics is rich with vivid images and provocative ideas* - Ivars Peterson ('The Mathematical Tourist', preface, page xiii [4])

Having an interest in philosophy, I occasionally delve into my copy of the *Oxford Companion to Philosophy* [2]. With its easy to read style, one is able to get a broad outlook and snapshots of important concepts, ideas, intellectual movements and figures. I have always been on a look out for a comparable book for mathematics. Finally, it has finally been published.

2 Summary

The Princeton Companion to Mathematics has almost two-hundred entries which are divided into eight parts. Each entry is followed by suggestions for further reading.

Part I and II introduce mathematics and its development. Part I is an introduction about the essence of mathematics, its main branches, mathematical language and grammar, elementary logic and goals of mathematics. It covers basic mathematical definitions in number systems, algebraic structures, functions and analysis which is a good way of making a novice comfortable with formal Mathematik. Part II discusses the development and origin of modern mathematics. This includes development of number systems, geometry, abstract algebra, algorithms, rigor in mathematical analysis and the idea of a proof. The crisis in the foundations of mathematics is also covered.

Part III, IV, and V are the meat of the book and explain various fields and concepts. Part III consists of more than a hundred entries on various concepts in mathematics. The range of entries includes *The Axiom of Choice*, *expanders*, *the heat equation*, *knot polynomials*, *L-Functions*, *measure theory*, *quantum groups*, *Simplex algorithm*, *topological spaces* and *Zermelo-Fraenkel Axioms*. Part IV provides overview surveys on twenty six major branches of mathematics. The entries are longer than the entries of other parts. Among other branches, algebraic and arithmetic geometry, computational and analytic number theory, algebraic and differential topology, representation theory, geometric and combinatorial group theory, partial differential equations, general relativity, numerical analysis, set theory, logic and model theory, stochastic processes and probabilistic models of critical phenomena are covered. There is a well organized essay on each branch. For example, the branch of 'Extremal and Probabilistic Combinatorics' covers extremal graph theory, Ramsey theory, set systems, combinatorial number theory, discrete geometry, key extremal combinatorics tools, random structures and probabilistic constructions.

In Part V, thirty-eight important theorems and well-known problems are described. Sample entries include *fixed point theorems*, *the four colour problem*, *Gödel's theorem*, *The Weil Conjectures*, *Hilbert's Nullstellensatz*, *Halting Problem* and the fundamental theorems of arithmetic and algebra. Part VI contains biographical notes on almost hundred eminent mathematicians with highlights of their contributions. The mathematicians range from ancient Greeks to twentieth century figures. For history lovers, this is fascinating stuff.

Part VII is on the applications of mathematics. It covers a wide variety of fields such as chemistry, biology, network design, traffic in networks, algorithm design, image processing, information theory, cryptography, economics, finance, medical statistics and even philosophy, music and art. The last part is on final perspectives on mathematics which includes topics like problem solving, reason for mathematics and experimental mathematics. The book ends with a particularly enjoyable and insightful entry on advice to young mathematicians. The advisers include luminaries such as Michael Atiyah and Béla Bollobós.

3 Opinion

The Princeton Companion to Mathematics meets its aim of not only capturing the big picture of mathematics with its interconnected fields, but it also gives insight into key mathematical concepts, results, contributions and the history. It is an excellent advocate for mathematics. It will be of interest to people ranging from recreational mathematicians to veteran researchers. The treatment of different areas seems to be fair. I could not pinpoint any area which was not represented. Compared to reference encyclopedias, the companion is thematically organized, unified in its structure and is pleasurable to read. In terms of treatment of the material, it does service to its intended purpose of being a companion. Therefore, it should not be compared to much more comprehensive reference books such as *Encyclopedia of Mathematics* by Hazewinkel [1]. However, one aspect of

the book which disappointing is that it is a bit too heavy to be a handy companion. In this regard, the thick glossy paper and sturdy binding has not helped.

The entries have all been written by the leaders in their area. It is also evident that the editors have ensured that every article is written in a very accessible way. This enables readers to ease into and appreciate concepts which they are unfamiliar with. The overall book is in great hands: Timothy Gowers is a Fields Medalist and Imre Leader is a well-known professor at Cambridge. The book paints a clear picture of the historical achievements and figures. Here, the expertise of Barrow-Green(who is a mathematics historian) seems to come to the fore. There are portraits of only a few mathematicians. It would have been nice to have portraits of many of the other personalities covered just so that one can put a face to a famous name.

The treatment of the subject is particularly fresh and progressive. I was immediately struck by the number of contributions by theoretical computer scientists such as Avi Wigderson, Madhu Sudan, Oded Goldreich and Jon Kleinberg. There are various entries on algorithm design, computational complexity and computability theory. Among the branches of mathematics, 'computational complexity' has a separate section with coverage of *P versus NP* question, reducibility, lower bounds, circuit complexity, proof complexity and randomized computation. Even in Part I, where the goals of mathematics are discussed, the importance of finding explicit proofs, constructive arguments and algorithms is outlined. The close interplay between mathematics and computer science of course requires no introduction to the theoretical computer science community [3].

The Princeton Companion to Mathematics is truly a labor of love by the best in the business. Although not handy enough to carry around, it is a readable and enjoyable reference.

References

[1] M. Hazewinkel. *Encyclopaedia of Mathematics*. Kluwer, 1994.

[2] T. Honderich. *The Oxford Companion to Philosophy (2nd edition)*. Oxford University Press, 2005.

[3] D. E. Knuth. Computer science and its relation to mathematics. *The American Mathematical Monthly*, 81(4):323–343, 1974.

[4] I. Peterson. *The Mathematical Tourist: Snapshots of Modern Mathematics*. Freeman and Company, New York, 1988.

Review of
Computer Viruses and Malware [13]
Author: John Aycock

Reviewer: Michael Sanford (michael.sanford@dsu.edu)

1 Overview

In the world of security, information is extremely valuable. Universities are the primary source of security related information with the exception of viruses and malware. For some reason, university administrators feel that teaching students about viruses and malware only to creates a larger problem. What they fail to realize is that the more people that understand the problem the more likely good solutions will be discovered. Associate Professor John Aycock and the University of Calgary in Canada took a more liberal approach to the problem: get the information to the students as early as possible. Consequently, Dr. Aycock set out to create a text book about viruses and malware. This review is about that book.

This book is organized into 11 chapters. Each chapter is broken down into subchapters with specific information. It is laid out similarly to an outline. What I found unique with this book is the notes at the end of each chapter. All notes that are numbered 1-99 relate to additional material for that chapter. Notes starting at 100 are citations and references to related material. I found this type of distinction quite useful.

This book is a concept book. Dr. Aycock points out in the preface that this book is not about implementation details, but on concepts. He also makes suggests that if the reader is using this book as a text book, the reader should supplement it with information regarding the latest malware issues.

2 Summary of Contents

2.1 Chapter 1: We've Got Problems

Dr. Aycock sets up the book here by defining some terms and getting the reader comfortable with what will be presented in the rest of the book. Like any good researcher he points out that there is no such thing as absolute security. He points out that with any security plan, there tradeoffs between security and usability. Also discussed in this chapter is the cost of malware. The book uses the term malware rather loosely. Malware, according to the book, encompasses not only traditional malware, but viruses and spam.

2.2 Chapter 2: Definitions And Timeline

Chapter 2 delves into definitions and generalities. Dr. Aycock discusses various types of malware in general terms: logic bomb, Trojan horse, back door, virus, worm, rabbit, spyware, adware, hybrids of malware, and zombies. This chapter is just to get the reader comfortable with the terms and get a general sense of what each type of malware is and its properties. Towards the end of the chapter

he talks about how names get associated to various types of malware and the purported authors of said malware.

2.3 Chapter 3: Viruses

Chapter 3 is where the meat of the book starts. This chapter is devoted to viruses. I was impressed with the amount of detail Aycock put into this chapter. He starts out with the basic file infector and moves on to macro viruses. He spends a good portion of the book discussing how a virus protects itself from detection using polymorphic and encryption techniques. There is a fair amount of detail with regard to detection and Aycock handles it well.

2.4 Chapter 4: Anti-Virus Techniques

Anti-Virus Techniques is the longest chapter in the book. The reason is simple: detection and disinfection. Detection and disinfection are the two main goals for most people dealing with viruses. There are several techniques that Aycock explores. He starts the chapter with static methods of detection. He discusses the following algorithms: Aho-Corasick, Veldman, and Wu-Manber. He discusses specifics of each algorithm without getting bogged down in too fine of details. He gives enough information about each algorithm to understand what is being done. He continues this section with how to improve performance of static scanners. Just like anything else in computer science, there is "no free lunch" and optimization of virus scanners is no exception. He brings to the forefront various optimization techniques but does not leave out the costs of those optimizations. At the end of this section, he touches on static heuristics and integrity checkers.

The next section is all about dynamic methods of detection. The only method discussed here is emulation. This is a pretty weighty section in terms of amount of information devoted to a single detection method. He describes the five parts to an emulator: CPU emulation, memory emulation, hardware and operating system emulation, emulation controller, and extra analysis. Also, as with static detection, Aycock discusses optimizations in the form of emulator optimizations. This is a small sections as the optimizations are fairly general.

The final section to this chapter is about disinfection. While this is an important topic, most of the information is fairly general in nature. Since this book is architecture agnostic, it is difficult to give specific solutions, so I do not fault him for that.

2.5 Chapter 5: Anti-Anti-Virus Techniques

This was probably the most interesting chapter to me. As a former software engineer, I find the techniques described in this chapter very interesting. I have seen some of the techniques he describes used as deterrents to reverse-engineering. There are some interesting parallels between anti-anti-virus techniques and java obfuscators. However, anti-anti-virus techniques as described by Aycock are much more sophisticated. For example, combining data instructions so that decompiling and reading the code is as difficult as possible. Also, the virus may contain encrypted instructions which are dynamically unencrypted as needed.

2.6 Chapter 6: Weaknesses Exploited

Up to this point, Aycock has been discussing issues that most people are aware of, but may not necessarily understand the technical issues. Furthermore, the previous topics have been primarily about willful harm. This chapter breaks from previous chapters and brings forth the topic of weaknesses. These weaknesses range from simple buffer overflows to stack smashing. This chapter will be a real eye opener for software developers. This kind of information needs to be taught side by side when learning a new language. This chapter has a specific string format attack for the C/C++ language as an example and should be studied by all software developers.

2.7 Chapter 7: Worms

This is an important chapter in that there is a distinct difference between worms and viruses. Aycock delves into the distinction. Furthermore, he discusses the history of the worm and in particular the Internet Worm of 1988. This is actually a relatively short chapter. However, besides the section on the Internet worm of 1988 the section on how worms find their targets is interesting as well.

2.8 Chapter 8: Deworming

Worms are very different from viruses and the techniques for getting rid of viruses are very different from the techniques used to get rid of worms. The chapter is organized in the following categories: defense, capture and containment, and automatic countermeasures. Defense starts out with the normal basics for securing a system: user education, patching, and limiting services to only those necessary. Aycock brings the point that anti-virus software is not just for viruses, it can be used for worms as well. This chapter is pretty thorough as a general discussion goes. It does go into some detail with regard to memory scanning and firewalls. Most organizations when facing a threat of this type just pull the internet plug and deal with it internally. While this will work for some organizations, Aycock discusses the use of throttling to help curb the spread of a worm. For companies that really can not pull the plug, throttling outgoing connections might be a good alternative.

2.9 Chapter 9: "Applications"

This chapter was somewhat different in the fact that it talks about the application of malware instead of the malware itself. Aycock starts out talking about malware that fixes bad malware. For example, in 2001 the Cheese worm was released. The purpose of this worm was to clean up after the Lion (li0n) worm that hit Linux machines. He calls these types of malware Benevolent Malware. He then moves on to the issue of spam, information warfare, worms for sale, and finally to cyber terrorism. A whole book could be devoted to this chapter, consequently it is just a brief discussion on each topic.

2.10 Chapter 10 and 11

The final two chapters are pretty much tying up loose ends. They discuss who malware writers are and what might possibly be their motivation for writing malware. In addition to the motivation of malware writers, Aycock gives us some advice on what we can do to protect ourselves.

3 Opinion

I liked this book. I was expecting it to be fairly dry. However, I was engaged by the writing style as well as the material presented. Aycock's style gets right down to the meat of the chapter pretty quickly. He does not spend too much time on fluff. You get the feeling that he is writing the book to disseminate information, not to see his words in print. The book is very usable for an upper division undergraduate degree classes. For graduate level use, I would definitely supplement it with something more technical. Overall, Computer Viruses and Malware was a good book. I am a person that is more into the technical issues so it was not as technical as I would have liked it. But as introduction to the topic, it was quite good.

Review of[14]

Formal Correctness of Security Protocols
Author of Book: Giampaolo Bella
Springer-Verlag, $64.00, 274 pages, Hardcover

Author of Review: Yannis C. Stamatiou
Dept. of Mathematics, University of Ioannina, Greece

1 Introduction

Obtaining formal security proofs for security protocols is, usually, a notoriously difficult task. The main reason is that security protocols are *dynamic* and *strictly ordered* sequences of computational steps that, also, involve complex interactions among several involved (malicious included) parties. This dynamic and complex nature of security protocols frequently leaves open the door for subtle problems that eventually render the protocols insecure and useless, after thorough and in depth scrutiny by researchers. And what is worse, is that uncovering these subtleties may be a long and time consuming process that, while in progress, allows the target protocol to be used in practical applications, even though it may be flawed. Thus, what is needed, first, is a way to unambiguously describe a security protocol so as to let all researchers understand the same things about the protocol, without any possible misunderstandings that may delay the security analysis. In addition, the description would, most desirably, be one that can be mechanized and, thus, allow speeding the analysis up through the use of a computer. The last to sentences describe, exactly, the two goals of the author in this book.

Over the past decades, various formalisms have been proposed and some of them have been successfully applied in proving certain security protocols secure or insecure. These formalisms usually involve the following steps: (i) The target protocol is cast into a set of statements using the rules set forth by the formalism, and (ii) stating a set of properties that, if proven within the formalism's expressive and deductive power, lead to a formally derived conclusion about the security of the target protocol. The author's approach is based on the *Inductive Method*, that was proposed by Paulson in 1998. However, this method is extended by the author (such extensions were among the main the targets of the author's PhD research) so as to include dynamic and temporal aspects of security protocols. Below, we will state briefly the contents of each chapter and discuss their

[14]

usefulness to the practitioner and theoretician alike in in handling the security aspects of real-life protocols.

2 Summary of Chapters

Chapter 1 provides a concise but very informative introduction to the book's aims as well as its contents so as the reader can, in a few minutes, evaluate if and how the book can be useful to him/her. Most importantly, this introductory chapter contains an very brief introduction to the inductive method, which lies in the heart of the author's approach.

Chapter 2 reviews some formalisms that have been proposed in the past for describing and analyzing security protocols. At the same time, the author comments on the difficulties that arise in conducting the analysis as well as interpreting the analysis results.

Chapter 3 forms the core of the book, by describing the basic Inductive Method in detail. The author's presentation is intuitive pedagogically appealing: the author introduces the modeling power of the Inductive Method focusing on each component,separately, of a typical security protocol (Normal and Compromised Agents, Cryptographic Keys, Messages, Events, Traces of Events, Threats Models, and the operations allowed within the formalism) and concludes with a simple example that demonstrates, however, convincingly the power of the formalism. The chapter, also, presents *Isabelle*, which is a generic, freely available, theorem proving software platform that allows the specification of formalisms and provides the environment for the execution of deductive reasoning, within the formalism. It also comes with a rich repository of proof files. It installs easily on Unix/Linux as well as Windows equipped with the Cygwin (Linux-like) environment. This combination of formalism/computer analysis (i.e. Inductive Reasoning/Isabelle) is, I believe, the main strength of the book: for each target protocol in the book, the author provides freely the corresponding files that contain the formal description (much like a program) of the protocol so as to be ready for execution within the Isabelle environment. The reader may learn a lot from these description files and adept them so as to meet his/her own analysis needs. With regard to this chapter I only have a minor negative remark: Figure 3.1 has too small fonts and is not easy to read.

Chapter 4 actually forms a continuation of Chapter 3 and relates the Inductive Reasoning with some of the principal goals that need to be established within a formal security analysis: the author proposes seven such goals and provides explanatory examples of their usefulness in establishing the security of the target protocol. The discussion also focuses on how well the the formal models grasp the main features of the target protocol and, thus, are reliable in deciding whether the protocol is secure or not.

Chapter 5 proposes the principle of *goal availability* as an important focus of a security analysis task. According to this principle, a security protocol should be accompanied by a set of assumptions (about the protocol) that are formally stated and can be formally verified by all the involved interacting (through the protocol) participants. The intuition behind this principle is that each security protocol should be enhanced with clearly stated properties about itself that are clearly verifiable by agents executing the protocol. Thus, the security protocol's security is manifested through its own workings and operation. Again, the discussion is augmented by a good and illuminating example.

Chapter 6 extends the Inductive Reasoning framework through the incorporation of dynamic and temporal features of security protocols: protocol step sequencing and timestamping. These

two elements are very crucial in determining the security properties of a protocol since, frequently, protocols fail either because some steps can sometimes be executed in the wrong order or because some messages are outdated (e.g. when nonces are not used) and, thus, can lead to successful attacks.

Chapter 7 provides the first thoroughly worked out example of the application of the theory described in the previous chapters. The author focuses on the Kerberos IV, a well known service and user authentication protocol. This is a sufficiently complex protocol for one to use in order to unveil the power of a formalism, such as the one proposed by the book author, yet it is intuitively appealing and compactly described through a number of clearly defined interacting steps. The author presents formal descriptions for these steps and shows how to derive security properties using the formalisms deductive power.

In **Chapter 8** the author defines the concept of *protocol message knowledge* by the involved interacting agents. This concept is in clear contrast with the concept of *belief*, that may lead to confusion, used in other reasoning formalisms. Inductive Reasoning is powerful enough to be able to express the fact that agents *know* the contents of the exchanged messages and not, simply, that some messages have been exchanged. This is important in, for instance, non-repudiation protocols in which it is important to be able to prove that an agent *knows* (is aware of) the contents of a message (he/she has created and sent, under his/her consent) and, thus, he/she may be held responsible for consequences that may ensue.

Chapter 9 focuses on Kerberos V, the newest version of Kerberos that was in used during the writing of the book (finished in early 2006, according to the preface). Kerberos V differs from Kerberos IV mainly in that it dispenses with double encryptions. The author uses Inductive Reasoning in order to prove that double encryptions do not really increase security, even though at first glance they seem to do so. In this sense, Kerberos V is "lighter" than Kerberos IV and achieves the same security goals. This achievement of formal analysis is particularly pleasing since it demonstrates the power of the formalism to prove equivalent security behavior under different assumptions (single or double encryptions).

Chapter 10 is a particularly useful and important treatment of smart cards and their role in the security of protocols. The importance stems from the wide use, nowadays, of hardware security tokens with relatively powerful processing capabilities and the ability to securely store information (e.g. keys), such as embedded devices and Java-based smart cards. The author demonstrate how Inductive Reasoning can handle hardware security tokens and extends the threat model by the ability of the attacker to make copies or steal the security token.

In order to demonstrate the power of the theory in Chapter 10, the author sets forth in **Chapter 11** to analyze a popular smartcard security protocol: the well known *Shoup-Rubin* protocol. The author formally proves that most of the security goals are achieved by the protocol while he demonstrates that the protocol lacks goal availability (as defined in Chapter 5). What is more, is that the analysis is able to demonstrate a fix to this problem. The analysis is very detailed but highly instructive, if one painstakingly follows the author's presentation.

In **Chapter 12** the author provides a final extension to the Inductive Reasoning framework to encompass the security requirement of *accountability*. Facets of accountability are, as the author reasonably argues, e-mail and non-repudiation. Accountability-related protocols are very important in today's Internet society since many interactions and agreements are performed remotely, without any previous acquaintance of the interacting parties (e.g. e-auctions or e-commerce). Thus, due to lack of trust, protocols are needed in order to, at least, be able to prove accountability of a

misbehaving party.

As application of the theory of Chapter 12 (modeling accountability) in **Chapter 13** the author models and analyzes two protocols, one related to e-mail exchange and one to non-repudiation. The former was proposed by Abadi, Glew, Horne, and Pinkas (certified email) and the latter by Zhou and Gollmann. These two protocols are representative in their categories and the analysis in this chapter can be easily extended and adapted in order to be applicable to general, accountability related protocols.

Finally, in **Chapter 14** the author discusses the usefulness and applicability of the approach proposed in the book, highlighting the most crucial points. A particularly pleasing and useful part of this chapter is the *Statistics* section. In this section the author gives an estimate, to the reader, about the difficulty and the required effort in modeling the protocols studied in the book, including the creation of the files containing the formal descriptions of the protocol, to be input in the Isabelle system. I liked particularly this section since, being more a practitioner in security, I would like to know the effort involved to model protocols of a variety complexity levels, such as the ones given in the book by the author. As it turns out, the effort is reasonable enough, given the insight that the modeling process gives about the modeled protocol as well as the ability to uncover potential security flaws in it.

The book also contains **four appendices** that contain the description of the key elements of four important and widely used protocols: Kerberos IV, Kerberos V, Shoup-Rubin, and Zhou-Gollmann. The appendices are very helpful since, in a single place, the gather the information given in the corresponding book chapters, about how to model the security aspects of these protocols.

3 Opinion

In summary, my opinion is that this is a great book in the field of computer security, for the practitioner and theoretician alike, since it provides an ideal mixture of theoretical results and applications of them in real protocol analysis scenarios. The book combines, in an ideal way, the features of a rigorous book and a "cookbook". The most remarkable feature of the book is that all formalisms and proofs described in it, are available freely in source form and are executable on a freely available program, Isabelle. This gives to experienced as well as beginner readers the opportunity to save a lot of effort in preparing and executing their own protocol description and analysis files without hassling much with familiarizing themselves with every aspect of the Inductive Reasoning formalism: they may, simply, select and tailor to their needs fragments of the files available from the author. However, I would like to make the following negative point: the book would be greatly benefited if each chapter was accompanied by some exercises that would also invite the reader to create and run his/her own analysis on the exercises' targets (either whole protocols or fragments thereof). Perhaps this issue could be handled in a subsequent edition of the book.

In conclusion, I would strongly recommend this book to people involved in formally proving properties about security protocols as well as students making their first steps in studying such protocols, since it excellently balances rigor with application and provides a great wealth of analysis files ready to be put in action with small adaptations and modifications.

Review of

Computability of Julia Sets [15]
Author: Mark Braverman and Michael Yampolsky
Publisher: Springer, 2009. 151pp, 31 figures.
978–3–540–68546–3, $89.95
Volume 23 in Algorithms and Computation in Mathematics

Reviewer: Wesley Calvert (wesley.calvert@murraystate.edu)

1 Overview

The study of dynamical systems has at its core, as has often been noted, a very computational feel. In complex dynamics, one considers the one point compactification $\hat{\mathbb{C}}$ of the compact plane (one can think of the complex plane itself without too much loss; one can also think of Riemann manifolds more generally), and a rational map $f : \hat{\mathbb{C}} \to \hat{\mathbb{C}}$, and studies the sequence $\mathcal{F} = \{f, f \circ f, f \circ f \circ f, \dots\}$. We will write $\mathcal{F}(x)$ for the sequence of points $\{f(x), f \circ f(x), f \circ f \circ f(x) \dots\}$, technically called the *orbit* of x under f. The inherently recursive nature of the subject, as well as the distinguished history of computer simulation in this area connects the subject closely with computation.

Some of the most recognizable icons of the subject are the *Julia sets*, the enigmatic, often paisley-like images produced by identifying regions in which $\mathcal{F}$ exhibits *sensitive dependence on initial conditions* — that is, small differences in x lead to large differences in the behavior of the sequence $\mathcal{F}(x)$. We denote the Julia set for f by J_f. The complement of the Julia set is called the *Fatou set*. A natural question is whether there is an algorithm to determine whether a given point is in the Julia set or the Fatou set. Obviously, such a determination would require full-precision complex arithmetic, and so is not literally possible.

The book under review approaches this question by treating it approximately: Given f, is there a Turing machine M such that, for any n, if we give M a program which computes the digits of the coordinates of a point, then M will output 1 if the point is within 2^{-n} of the Julia set, 0 if it is at least 2^{-n+1} away from the Julia set, and either 1 or 0 in every other case? In effect, is there an algorithm whose input is a degree of precision and whose output is a plot (of appropriate granularity), coloring pixels black or white, as appropriate? (This approach differs markedly from the earlier approach of Blum, Shub, and Smale, which calls nearly all Julia sets non-computable, but is attractive from different perspectives; see [2])

Thus, a slightly informal restatement of the central problem of the book is this:
Question: Given a rational self-map of $\hat{\mathbb{C}}$, along with a point p and a radius r, can one determine whether the dynamics of the map become chaotic within r of p?

2 Summary of Contents

Braverman and Yampolsky, sometimes with others, have shown in recent years (beginning with Braverman's 2004 master's thesis at the University of Toronto) that both positive and negative answers are possible, depending on the map (Zhong [5] also had an important early result). In short, if the dynamics of the map are sufficiently "well-behaved," then the answer is yes. If not,

[15]

then the answer depends on the computability of a certain real parameter. The present brief monograph documents these results.

One can feel the book trying to be self-contained, and it is, to a degree. Certainly the chapter introducing computable analysis will provide the needed background in this area for most readers. The background material on dynamics is a good deal harder — a reader with a background in complex analysis at the level of a first graduate course — say, of Ahlfors [1] or Greene and Krantz [3] will find the introductory material here rough going, and the reader without it may find it well-nigh impenetrable. However, if one is willing to take quite a lot for granted, the main line of the argument may be followed by making frequent reference (necessary in any case) to an outside resource on dynamics, like [4].

The principal positive results are these:

Theorem Let $f : \hat{\mathbb{C}} \to \hat{\mathbb{C}}$ be a rational function.

1. If f has the property that every critical point of f is either periodic or converges to an attracting cycle (i.e. f is *hyperbolic*), then J_f is computable in polynomial time.

2. If f has no regions isomorphic to a disk or annulus on which f restricts to an irrational rotation (i.e. no *rotation domains*), then J_f is computable.

3. The parameterized family $\mathcal{J} = \{(z, c) : z \in J_{z^2+c}\}$ is computable.

In short, the first two items mean that dynamically "well-behaved" systems have computable Julia sets, and the third means that this computability is, in some cases, robust to small changes in the parameters. On the negative side, the Julia set J_{z^2+c} is computable if and only if it either has no rotation domains containing indifferent fixed points, or if the *conformal radius* (a natural geometric parameter of such a region) is computable from the parameter c. Examples are given to show that the non-computable case is possible. The authors also show, in a similar way, how to construct computable Julia sets of arbitrarily high complexity and a noncomputable Julia set which is not locally connected. One open number-theoretic conjecture is given, with proofs that it would imply (again by the same method as the noncomputability result) that if f has only algebraic parameters then J_f is computable, and that the set of noncomputable Julia sets is small (of null Hausdorff dimension).

3 Opinion

The subject of the book is timely and important. There is a growing literature recently on computability of various aspects of dynamical systems. The questions posed and answered in the present book are natural and the approach well-suited to produce enlightening results.

With the aforementioned caution on the background expected of the reader — one must either have some rather serious background in analysis (and, in places, transcendental number theory) or be prepared to work hard — the book is quite readable. Arguments are clearly written, often in both outline and full versions. Except for the details of the dynamics, the authors are quite careful to explain the importance of the details in engaging, even lively prose.

The book is also generous with a varied set of open questions, including matters of complexity, transcendental number theory with implications for the subject, and relations between geometry and computability on Julia sets. It has the potential to inspire considerable future work in this intriguing field.

References

[1] L. Ahlfors, *Complex Analysis*, 3ed, McGraw-Hill, 1979.

[2] L. Blum, F. Cucker, M. Shub, and S. Smale, *Complexity and Real Computation*, Springer, 1998.

[3] R. E. Greene and S. G. Krantz, *Function Theory of One Complex Variable*, 2ed, Volume 40 in Graduate Studies in Mathematics, American Mathematical Society, 2002.

[4] J. Milnor, *Dynamics in One Complex Variable*, 3ed, Volume 160 in Annals of Mathematics Studies, Princeton University Press, 2006.

[5] N. Zhong, *Recursively enumerable subsets of R^q in two computing models. Blum-Shub-Smale machine and Turing machine*, Theoret. Comput. Sci. 197 (1998), no. 1-2, 79–94.

Data Structures and Algorithms : This course is purely designed to focus on data structures and algorithms. I am strongly believing that, data structures and algorithm are not a technology. using correct data structures and writing efficient algorithm is a skill, So I am going to focus on to teach you how the apply this skill in Real time application which means how to apply this knowledge while choosing data structure in real application.Â To make this work more easy, programming languages provides a concept called "Array" Data Structures. Thatâ€™s what we are going to discuss in this lecturer in Data Structures and Algorithms course. 1D Array. 07:01.Â But both the element canâ€™t go to the same place. This is known as collisions.