PREFACE

This book grew out of a text I prepared eight years ago, along similar lines but with a more limited scope [53]. That earlier edition was based on notes for a course I gave in the Department of Computer Science of the Technion in Israel, in 1986. The course was intended for both graduates and undergraduates who were looking for an area in which to specialize. This largely determined the coverage of the course, and consequently, of the text as well. The present version keeps to the same intent, but contains much more material, so that it could be used as a research reference as well.

Frequently, "analysis of algorithms" is synonymous with determining the "complexity class" of an algorithm. This has the objective of finding whether in *all* cases the resource requirements of the algorithm can be bounded by some specified function of the size of the problem. It naturally boils down to the consideration of some extreme, specifically crafted problem instances. But there is more one can say to characterize the cost of an algorithm. It was probably Knuth's series titled *The Art of Computer Programming*, which started out in 1968, that gave this view its first wide exposure: the operation of algorithms was shown to be associated with probabilistic concepts and processes.

Random elements can enter algorithms in two ways. On the one hand, we find "probabilistic algorithms," which choose part of their actions on the basis of random numbers, that are part of the algorithm specification (normally pseudorandom numbers are used). Numerous algorithms of this class were recently developed, some showing amazing capabilities (primality testing algorithms provide a good early example). On the other hand we find the operation of deterministic algorithms on input data over which some probability measure can be assumed. While these sources of randomness are entirely different, the required analyses turn out typically to be of the same nature in both cases. Both varieties of algorithms can be found among those we analyze in detail. Chapter 1 shows that the second type of algorithmic analysis brings up methodological and conceptual problems that the first need not entail. At the time of writing there is no coherent accepted theory or even complete taxonomy, comparable to standard complexity theory, for these vexing issues.

A recurrent issue in the analysis is the required precision of the results. This is frequently a matter of personal taste. Complexity issues are too often disposed of with the magic adjectives 'polynomial' and 'exponential.' Within classificatory complexity theory this may be all one needs, but when algorithms are viewed as engineering tools, this is unfortunate—we should do better, and find the functions hidden in such phrases. An example may be in order: Recently, a mathematician better left unnamed here, chuckled about "the amusing habit of computer scientists [to refer to] algorithms with running times like $O(n^{120})$ and $O(e^{0.0001n})$ as fast and slow, respectively."[1] In fact, even comparing humdrum functions such as $O(n^2)$ and $O(n^3)$ is of limited practical use—when a specific range of n is considered—unless we can find information about the multiplicative coefficients hidden there. This book is designed to show the way and provide effective tools.

Prerequisites

The probabilistic analysis of algorithms, as a discipline, draws on a number of branches of mathematics: combinatorics, probability theory, graph theory, real and complex analysis, and occasionally algebra, number theory, computation theory, operational calculus and more. It is unreasonable to expect the students to have more than a cursory knowledge of the techniques we use, so much of the text introduces and explores these methods. Arranging the material for the book, so it could be conveniently used both as a text and as a reference, posed a problem that was solved by departing from the order of the class presentation, collecting most of the methodological material in the first part.

The prerequisites that *are* assumed are basic courses in discrete mathematics, calculus (including a smattering of differential equations and complex functions), linear algebra, probability theory, data structures, and graph algorithms; all these are required courses in the department. The last two are assumed to impart to the students some algorithmic literacy.

The emphasis throughout is on the analytic and probabilistic aspects, rather than the algorithmic ones. Good texts for the latter exist and are burgeoning; I mention several in chapter 1. Should the text have a chapter on probability theory? or an introduction to the subject, or at least an appendix? The question led to a long debate, and I finally decided against it; this subject has very good books available, at every level of depth and sophistication. Indeed, I am not even aware of all the good sources that exist now, and the bibliography section lists only those recently used in my work, as well as for the preparation of this text. What I *did* include is chapter 7, with a few topics from probability theory that are outside the basic curricula, and that I have found useful.

Outline

Chapter 1 introduces and establishes the context of the book, and surveys sources and various aids for analysis. Part I of the book (chapters 2–7) is methodological. Chapters 2 and 3 present generating functions and their uses in combinatorics and probability. Chapter 4 discusses the structure of permutations. Chapter 5 deals with integral transforms. These are helpful in asymptotic analysis, which is the subject of chapter 6, by far the largest.

---

[1] Papadimitriou pleads an eloquent case in [80, p. 6*ff*] for the validity of polynomiality as the criterion for tractability (of problems, rather than specific algorithms). To underline the meaninglessness of the above quip(?), note that a polynomial such as n^{120}, even for $n = 2$, and expressed in picoseconds, far exceeds the lifetime of the universe as we know it—rather less than 10^{30}ps (bearing in mind that this is written in Texas,

Following the discussion of tools, a few sample analyses are collected in the last three chapters. This proved to be a good opportunity to present a few additional analytical techniques. The choice of the particular algorithms to analyze in detail was a nontrivial issue. The first few are easy, relatively simple ones which are used mainly to introduce the approach. Beyond that I was guided by personal preferences, and to some extent by requests of students in the class, inasmuch as they agreed with the types of analysis I was interested in or able to display. The last reservation hints at some of the difficulty in teaching this subject: for many interesting analyses, the sheer technical complexity (as distinct from their mathematical profundity) is such that presenting them in class is an unjustifiable harassment of the students (and the teacher too, no doubt). Several analyses in which I was involved could serve as illustrations of the problem (e.g., Fayolle *et al.* [149] – which is presented in part, or Hofri and Konheim [187]). Finally, three chapters were prepared: Chapter 8 on the analysis of resource requirements of algorithms, in the style of Knuth – on searching and sorting. Chapter 9 brings analyses of communications protocols, which lead to very different mathematical structures. Chapter 10 shows analyses of relative efficiency of suboptimal algorithms; it focuses on the bin-packing problem as a typical example.

The exercises are mostly extensions of the text. Some are used to avoid giving much space to technicalities that tax the endurance of the reader and in which students should gain some proficiency in performing on their own. Many complements are given there, to introduce further results and technical material. Answers and many detailed solutions carry this intent further. Occasionally, a large exercise presents a complete research problem, broken down into steps, with hints on the way to proceed.

Given the course (or book) prerequisites, the text is fairly self-contained; I added, however, pointers to further reading whenever it seemed indicated. There was no effort to keep the level of detail in the presentations uniform, and earlier sections tend to be more detailed. In the later sections a larger part of the derivation devolves to exercises.

The material I used in class was, in order of presentation: chapter 1, portions of sections 2.1, 2.2, 7.1, 7.4, selections from chapters 8, 3, 5, and a little from chapter 9. The teaching assistant, beside doing some of the exercises in class, also covered examples from sections 6.1 and 6.2.

The current volume can be used for a few other courses. Here are two examples: Combinatorics for Computer Science (chapters 2–4, selections from section 6.1 and chapter 8); Asymptotic Methods (selections from chapters 2, 3, 5, most of chapter 6, sections 7.4, 9.1, selections from chapter 10).

A good case could be made for including in the book material, tutorials on the various software packages available for symbolic computations. I use these tools often, and *I strongly urge the reader to use them as well.* The investment of effort needed to know such packages and master their use is anything but trivial, but it will be repaid many times over. Ultimately, it is the plurality of these packages that stopped me. There are several popular products, and while the principles of most appear similar, there are simply too many differences in detail to make any general material useful. Devoting a section to each did not seem warranted, especially since this field is a moving target: new versions and products appear every few months. Such a chapter would have been obsolete shortly after publication, if not before. Since the users of the text are likely to be computer-literate, I hope that giving them this encouragement and providing some pointers in the book about possible applications will provide all the necessary impetus.

Notation

References are cited by author and the corresponding ordinal in the bibliography section at the back of the volume, as in the previous example (Fayolle *et al.* [149]). I make there no claim for either completeness or historical fairness; these are simply the sources I found useful in preparing the text.

Equations are numbered separately in each major section (2.2, 7.3, etc.). Equation (7) in section 3.2 will be denoted as (7) throughout 3.2, and by (3.2-7) elsewhere; similar notation applies to exercises. The notation (A-1.7) and (C-3) refers to equations number (1.7) and (3) in appendices A and C, respectively. The mark □ indicates the end of a proof, a theorem which is not followed by a proof, or of an example.

The use of symbols follows a few ground rules: random variables and processes are denoted by capital italics, the values they realize by lowercase ones, and probability distribution-, density- and mass-functions by F, f, and p, respectively, with suitable subscripts. Various types of generating functions are denoted by forms of the letter g, or by the name of the "generated" series or variable. The symbols $\Re$ and $\Im$ are for the real and imaginary components of a complex number. Integer indices are usually denoted by i, j, k, l, m, n (shades of FORTRAN?). When a summation index may be nonobvious, as for example, when writing $\Sigma_{i\neq k}$, we want to specify which of i and k varies, or that they both do, I follow Comtet's usage and place a dot under the dummy index. The two possibilities just considered would then be denoted by $\Sigma_{\underset{\cdot}{i}\neq k}$ and $\Sigma_{\underset{\cdot}{i}\neq \underset{\cdot}{k}}$, respectively. There are a few exceptions – most of which result from my attempt, when following an analysis developed elsewhere, to adhere to the notation used there, to facilitate reference to the original paper. Symbols for operators and special functions are collected in the Notation Index, in the back of the volume. Acronyms are expanded in the general index. Starred(*) exercises are those for which no satisfactory solution has yet been worked out.

Acknowledgments

The decision to write the original version was taken following a conversation with Philippe Flajolet; his work over the years inspired much of the material on asymptotic methods in the present book. The following colleagues and students read portions of the text or provided comments and corrections that improved the presentation: David Aldous, Eyal Bardavid, Edward A. Bender, Arnon and Shahar Boneh, John Burke, John Bruno, Luan Cong, Michael Drmota, Guy Fayolle, Aura Ganz, Oded Goldreich, Yaakov Kogan, Zehava Koren, Shay Leshkovitz, George Lueker, Omid Madani, Johann Makowsky, Osnat Meir, Hadas Shachnai, Moshe Sidi, Wojciech Szpankowski, and anonymous reviewers. Their contributions are gratefully acknowledged.

I retain however sole claim to any errors and misrepresentations. Help in further reducing these is avidly solicited. It will be very welcome, and acknowledged.

The sources of problems and exercises are given when known. Many however got first into my notes with no intention of using them in a book, and without a trace where they come from; my apologies to all who find among the exercises their unacknowledged creations. Please let me know so I can rectify this omission in the future.

The general index was prepared with a (slightly modified) version of Bentley and Kernighan's Indexing Tools [115]. Having started to use it about a year ago, it led me to an unexpected discovery: a detailed index is a great help to the *author* as well!

The software I used to typeset the book is the GTROFF suite of programs obtained from the GNU project, a reimplementation of the *troff* system originally distributed with UNIX. They have done a great job.

Houston, Texas, April 1995. M. H.

,

Contents

List of Figures

List of Tables