A COMPUTATIONAL INTRODUCTION TO NUMBER THEORY AND ALGEBRA

Second Edition

A COMPUTATIONAL INTRODUCTION TO NUMBER THEORY AND ALGEBRA

Second Edition

VICTOR SHOUP

CAMBRIDGE
UNIVERSITY PRESS

CAMBRIDGE UNIVERSITY PRESS
Cambridge, New York, Melbourne, Madrid, Cape Town,
Singapore, São Paulo, Delhi, Mexico City

Cambridge University Press
The Edinburgh Building, Cambridge CB2 8RU, UK

Published in the United States of America by Cambridge University Press, New York

www.cambridge.org
Information on this title: www.cambridge.org/9780521516440

First published 2009

A catalogue record for this publication is available from the British Library

ISBN 978-0-521-51644-0 Hardback

Contents

Preface

Number theory and algebra play an increasingly significant role in computing and communications, as evidenced by the striking applications of these subjects to such fields as cryptography and coding theory. My goal in writing this book was to provide an introduction to number theory and algebra, with an emphasis on algorithms and applications, that would be accessible to a broad audience. In particular, I wanted to write a book that would be appropriate for typical students in computer science or mathematics who have some amount of general mathematical *experience*, but without presuming too much specific mathematical *knowledge*.

Prerequisites. The mathematical prerequisites are minimal: no particular mathematical concepts beyond what is taught in a typical undergraduate calculus sequence are assumed.

The computer science prerequisites are also quite minimal: it is assumed that the reader is proficient in programming, and has had some exposure to the analysis of algorithms, essentially at the level of an undergraduate course on algorithms and data structures.

Even though it is mathematically quite self contained, the text does presuppose that the reader is comfortable with mathematical formalism and also has some experience in reading and writing mathematical proofs. Readers may have gained such experience in computer science courses such as algorithms, automata or complexity theory, or some type of "discrete mathematics for computer science students" course. They also may have gained such experience in undergraduate mathematics courses, such as abstract or linear algebra. The material in these mathematics courses may overlap with some of the material presented here; however, even if the reader already has had some exposure to this material, it nevertheless may be convenient to have all of the relevant topics easily accessible in one place; moreover, the emphasis and perspective here will no doubt be different from that in a traditional mathematical presentation of these subjects.

Structure of the text. All of the mathematics required beyond basic calculus is developed "from scratch." Moreover, the book generally alternates between "theory" and "applications": one or two chapters on a particular set of purely mathematical concepts are followed by one or two chapters on algorithms and applications; the mathematics provides the theoretical underpinnings for the applications, while the applications both motivate and illustrate the mathematics. Of course, this dichotomy between theory and applications is not perfectly maintained: the chapters that focus mainly on applications include the development of some of the mathematics that is specific to a particular application, and very occasionally, some of the chapters that focus mainly on mathematics include a discussion of related algorithmic ideas as well.

In developing the mathematics needed to discuss certain applications, I have tried to strike a reasonable balance between, on the one hand, presenting the absolute minimum required to understand and rigorously analyze the applications, and on the other hand, presenting a full-blown development of the relevant mathematics. In striking this balance, I wanted to be fairly economical and concise, while at the same time, I wanted to develop enough of the theory so as to present a fairly well-rounded account, giving the reader more of a feeling for the mathematical "big picture."

The mathematical material covered includes the basics of number theory (including unique factorization, congruences, the distribution of primes, and quadratic reciprocity) and of abstract algebra (including groups, rings, fields, and vector spaces). It also includes an introduction to discrete probability theory—this material is needed to properly treat the topics of probabilistic algorithms and cryptographic applications. The treatment of all these topics is more or less standard, except that the text only deals with commutative structures (i.e., abelian groups and commutative rings with unity)—this is all that is really needed for the purposes of this text, and the theory of these structures is much simpler and more transparent than that of more general, non-commutative structures.

The choice of topics covered in this book was motivated primarily by their applicability to computing and communications, especially to the specific areas of cryptography and coding theory. Thus, the book may be useful for reference or self-study by readers who want to learn about cryptography, or it could also be used as a textbook in a graduate or upper-division undergraduate course on (computational) number theory and algebra, perhaps geared towards computer science students.

Since this is an introduction, and not an encyclopedic reference for specialists, some topics simply could not be covered. One such, whose exclusion will undoubtedly be lamented by some, is the theory of lattices, along with algorithms for and applications of lattice basis reduction. Another omission is fast algorithms for integer and polynomial arithmetic — although some of the basic ideas of this topic are developed in the exercises, the main body of the text deals only with classical, quadratic-time algorithms for integer and polynomial arithmetic. However, there are more advanced texts that cover these topics perfectly well, and they should be readily accessible to students who have mastered the material in this book.

Note that while continued fractions are not discussed, the closely related problem of "rational reconstruction" is covered, along with a number of interesting applications (which could also be solved using continued fractions).

Guidelines for using the text.

- There are a few sections that are marked with a "($*$)," indicating that the material covered in that section is a bit technical, and is not needed elsewhere.
- There are many examples in the text, which form an integral part of the book, and should not be skipped.
- There are a number of exercises in the text that serve to reinforce, as well as to develop important applications and generalizations of, the material presented in the text.
- Some exercises are underlined. These develop important (but usually simple) facts, and should be viewed as an integral part of the book. It is highly recommended that the reader work these exercises, or at the very least, read and understand their statements.
- In solving exercises, the reader is free to use any *previously* stated results in the text, including those in previous exercises. However, except where otherwise noted, any result in a section marked with a "($*$)," or in §5.5, need not and should not be used outside the section in which it appears.
- There is a very brief "Preliminaries" chapter, which fixes a bit of notation and recalls a few standard facts. This should be skimmed over by the reader.
- There is an appendix that contains a few useful facts; where such a fact is used in the text, there is a reference such as "see §An," which refers to the item labeled "An" in the appendix.

The second edition. In preparing this second edition, in addition to correcting errors in the first edition, I have also made a number of other modifications (hopefully without introducing too many *new* errors). Many passages have been rewritten to improve the clarity of exposition, and many new exercises and examples have been added. Especially in the earlier chapters, the presentation is a bit more leisurely. Some material has been reorganized. Most notably, the chapter on probability now follows the chapters on groups and rings — this allows a number of examples and concepts in the probability chapter that depend on algebra to be more fully developed. Also, a number of topics have been moved forward in the text, so as to enliven the material with exciting applications as soon as possible; for example, the RSA cryptosystem is now described right after Euclid's algorithm is presented, and some basic results concerning quadratic residues are introduced right away, in the chapter on congruences. Finally, there are numerous changes in notation and terminology; for example, the notion of a *family* of objects is now used consistently throughout the book (e.g., a pairwise independent family of random variables, a linearly independent family of vectors, a pairwise relatively prime family of integers, etc.).

Feedback. I welcome comments on the book (suggestions for improvement, error reports, etc.) from readers. Please send your comments to

`victor@shoup.net`.

There is also a web site where further material and information relating to the book (including a list of errata and the latest electronic version of the book) may be found:

`www.shoup.net/ntb`.

Acknowledgments. I would like to thank a number of people who volunteered their time and energy in reviewing parts of the book at various stages: Joël Alwen, Siddhartha Annapureddy, John Black, Carl Bosley, Joshua Brody, Jan Camenisch, David Cash, Sherman Chow, Ronald Cramer, Marisa Debowsky, Alex Dent, Nelly Fazio, Rosario Gennaro, Mark Giesbrecht, Stuart Haber, Kristiyan Haralambiev, Gene Itkis, Charanjit Jutla, Jonathan Katz, Eike Kiltz, Alfred Menezes, Ilya Mironov, Phong Nguyen, Antonio Nicolosi, Roberto Oliveira, Leonid Reyzin, Louis Salvail, Berry Schoenmakers, Hovav Shacham, Yair Sovran, Panos Toulis, and Daniel Wichs. A very special thanks goes to George Stephanides, who translated the first edition of the book into Greek and reviewed the entire book in preparation for the second edition. I am also grateful to the National Science Foundation for their support provided under grants CCR-0310297 and CNS-0716690. Finally, thanks to David Tranah for all his help and advice, and to David and his colleagues at Cambridge University Press for their progressive attitudes regarding intellectual property and open access.

New York, June 2008 *Victor Shoup*

Preliminaries

We establish here some terminology, notation, and simple facts that will be used throughout the text.

Logarithms and exponentials

We write $\log x$ for the natural logarithm of x, and $\log_b x$ for the logarithm of x to the base b.

We write e^x for the usual exponential function, where $e \approx 2.71828$ is the base of the natural logarithm. We may also write $\exp[x]$ instead of e^x.

Sets and families

We use standard set-theoretic notation: $\emptyset$ denotes the empty set; $x \in A$ means that x is an element, or member, of the set A; for two sets A, B, $A \subseteq B$ means that A is a subset of B (with A possibly equal to B), and $A \subsetneq B$ means that A is a proper subset of B (i.e., $A \subseteq B$ but $A \neq B$). Further, $A \cup B$ denotes the union of A and B, $A \cap B$ the intersection of A and B, and $A \setminus B$ the set of all elements of A that are not in B. If A is a set with a finite number of elements, then we write $|A|$ for its **size**, or **cardinality**. We use standard notation for describing sets; for example, if we define the set $S := \{-2, -1, 0, 1, 2\}$, then $\{x^2 : x \in S\} = \{0, 1, 4\}$ and $\{x \in S : x \text{ is even}\} = \{-2, 0, 2\}$.

We write $S_1 \times \cdots \times S_n$ for the **Cartesian product** of sets $S_1, \ldots, S_n$, which is the set of all n-tuples $(a_1, \ldots, a_n)$, where $a_i \in S_i$ for $i = 1, \ldots, n$. We write $S^{\times n}$ for the Cartesian product of n copies of a set S, and for $x \in S$, we write $x^{\times n}$ for the element of $S^{\times n}$ consisting of n copies of x. (This notation is a bit non-standard, but we reserve the more standard notation S^n for other purposes, so as to avoid ambiguity.)

A **family** is a collection of objects, indexed by some set I, called an **index set**. If for each $i \in I$ we have an associated object x_i, the family of all such objects is denoted by $\{x_i\}_{i \in I}$. Unlike a set, a family may contain duplicates; that is, we may have $x_i = x_j$ for some pair of indices i, j with $i \neq j$. Note that while $\{x_i\}_{i \in I}$ denotes a family, $\{x_i : i \in I\}$ denotes the *set* whose members are the (distinct) x_i's. If the index set I has some natural order, then we may view the family $\{x_i\}_{i \in I}$ as being ordered in the same way; as a special case, a family indexed by a set of integers of the form $\{m, \ldots, n\}$ or $\{m, m+1, \ldots\}$ is a **sequence**, which we may write as $\{x_i\}_{i=m}^{n}$ or $\{x_i\}_{i=m}^{\infty}$. On occasion, if the choice of index set is not important, we may simply define a family by listing or describing its members, without explicitly describing an index set; for example, the phrase "the family of objects a, b, c" may be interpreted as "the family $\{x_i\}_{i=1}^{3}$, where $x_1 := a$, $x_2 := b$, and $x_3 := c$."

Unions and intersections may be generalized to arbitrary families of sets. For a family $\{S_i\}_{i \in I}$ of sets, the union is

$$\bigcup_{i \in I} S_i := \{x : x \in S_i \text{ for some } i \in I\},$$

and for $I \neq \emptyset$, the intersection is

$$\bigcap_{i \in I} S_i := \{x : x \in S_i \text{ for all } i \in I\}.$$

Note that if $I = \emptyset$, the union is by definition $\emptyset$, but the intersection is, in general, not well defined. However, in certain applications, one might define it by a special convention; for example, if all sets under consideration are subsets of some "ambient space," Ω, then the empty intersection is usually taken to be Ω.

Two sets A and B are called **disjoint** if $A \cap B = \emptyset$. A family $\{S_i\}_{i \in I}$ of sets is called **pairwise disjoint** if $S_i \cap S_j = \emptyset$ for all $i, j \in I$ with $i \neq j$. A pairwise disjoint family of non-empty sets whose union is S is called a **partition** of S; equivalently, $\{S_i\}_{i \in I}$ is a partition of a set S if each S_i is a non-empty subset of S, and each element of S belongs to exactly one S_i.

Numbers

We use standard notation for various sets of numbers:

$$\begin{aligned}
\mathbb{Z} &:= \text{the set of integers} = \{\ldots, -2, -1, 0, 1, 2, \ldots\},\\
\mathbb{Q} &:= \text{the set of rational numbers} = \{a/b : a, b \in \mathbb{Z}, b \neq 0\},\\
\mathbb{R} &:= \text{the set of real numbers},\\
\mathbb{C} &:= \text{the set of complex numbers}.
\end{aligned}$$

We sometimes use the symbols ∞ and $-\infty$ in simple arithmetic expressions involving real numbers. The interpretation given to such expressions should be obvious: for example, for every $x \in \mathbb{R}$, we have $-\infty < x < \infty$, $x + \infty = \infty$, $x - \infty = -\infty$, $\infty + \infty = \infty$, and $(-\infty) + (-\infty) = -\infty$. Expressions such as $x \cdot (\pm\infty)$ also make sense, provided $x \neq 0$. However, the expressions $\infty - \infty$ and $0 \cdot \infty$ have no sensible interpretation.

We use standard notation for specifying intervals of real numbers: for $a, b \in \mathbb{R}$ with $a \leq b$,

$$[a, b] := \{x \in \mathbb{R} : a \leq x \leq b\}, \qquad (a, b) := \{x \in \mathbb{R} : a < x < b\},$$
$$[a, b) := \{x \in \mathbb{R} : a \leq x < b\}, \qquad (a, b] := \{x \in \mathbb{R} : a < x \leq b\}.$$

As usual, this notation is extended to allow $a = -\infty$ for the intervals $(a, b]$ and (a, b), and $b = \infty$ for the intervals $[a, b)$ and (a, b).

Functions

We write $f : A \to B$ to indicate that f is a function (also called a **map**) from a set A to a set B. If $A' \subseteq A$, then $f(A') := \{f(a) : a \in A'\}$ is the **image** of A' under f, and $f(A)$ is simply referred to as the **image** of f; if $B' \subseteq B$, then $f^{-1}(B') := \{a \in A : f(a) \in B'\}$ is the **pre-image** of B' under f.

A function $f : A \to B$ is called **one-to-one** or **injective** if $f(a) = f(b)$ implies $a = b$. The function f is called **onto** or **surjective** if $f(A) = B$. The function f is called **bijective** if it is both injective and surjective; in this case, f is called a **bijection**, or a **one-to-one correspondence**. If f is bijective, then we may define the **inverse function** $f^{-1} : B \to A$, where for $b \in B$, $f^{-1}(b)$ is defined to be the unique $a \in A$ such that $f(a) = b$; in this case, f^{-1} is also a bijection, and $(f^{-1})^{-1} = f$.

If $A' \subseteq A$, then the **inclusion map** from A' to A is the function $i : A' \to A$ given by $i(a) := a$ for $a \in A'$; when $A' = A$, this is called the **identity map** on A. If $A' \subseteq A$, $f' : A' \to B$, $f : A \to B$, and $f'(a) = f(a)$ for all $a \in A'$, then we say that f' is the **restriction** of f to A', and that f is an **extension** of f' to A.

If $f : A \to B$ and $g : B \to C$ are functions, their **composition** is the function $g \circ f : A \to C$ given by $(g \circ f)(a) := g(f(a))$ for $a \in A$. If $f : A \to B$ is a bijection, then $f^{-1} \circ f$ is the identity map on A, and $f \circ f^{-1}$ is the identity map on B. Conversely, if $f : A \to B$ and $g : B \to A$ are functions such that $g \circ f$ is the identity map on A and $f \circ g$ is the identity map on B, then f and g are bijections, each being the inverse of the other. If $f : A \to B$ and $g : B \to C$ are bijections, then so is $g \circ f$, and $(g \circ f)^{-1} = f^{-1} \circ g^{-1}$.

Function composition is associative; that is, for all functions $f : A \to B$, $g : B \to C$, and $h : C \to D$, we have $(h \circ g) \circ f = h \circ (g \circ f)$. Thus, we can simply write $h \circ g \circ f$ without any ambiguity. More generally, if we have functions $f_i : A_i \to A_{i+1}$ for $i = 1, \ldots, n$, where $n \geq 2$, then we may write their composition as $f_n \circ \cdots \circ f_1$ without any ambiguity. If each f_i is a bijection, then so is $f_n \circ \cdots \circ f_1$, its inverse being $f_1^{-1} \circ \cdots \circ f_n^{-1}$. As a special case of this, if $A_i = A$ and $f_i = f$ for $i = 1, \ldots, n$, then we may write $f_n \circ \cdots \circ f_1$ as f^n. It is understood that $f^1 = f$, and that f^0 is the identity map on A. If f is a bijection, then so is f^n for every non-negative integer n, the inverse function of f^n being $(f^{-1})^n$, which one may simply write as f^{-n}.

If $f : I \to S$ is a function, then we may view f as the family $\{x_i\}_{i \in I}$, where $x_i := f(i)$. Conversely, a family $\{x_i\}_{i \in I}$, where all of the x_i's belong to some set S, may be viewed as the function $f : I \to S$ given by $f(i) := x_i$ for $i \in I$. Really, functions and families are the same thing, the difference being just one of notation and emphasis.

Binary operations

A **binary operation** $\star$ on a set S is a function from $S \times S$ to S, where the value of the function at $(a, b) \in S \times S$ is denoted $a \star b$.

A binary operation $\star$ on S is called **associative** if for all $a, b, c \in S$, we have $(a \star b) \star c = a \star (b \star c)$. In this case, we can simply write $a \star b \star c$ without any ambiguity. More generally, for $a_1, \ldots, a_n \in S$, where $n \geq 2$, we can write $a_1 \star \cdots \star a_n$ without any ambiguity.

A binary operation $\star$ on S is called **commutative** if for all $a, b \in S$, we have $a \star b = b \star a$. If the binary operation $\star$ is both associative and commutative, then not only is the expression $a_1 \star \cdots \star a_n$ unambiguous, but its value remains unchanged even if we re-order the a_i's.

If $\star$ is a binary operation on S, and $S' \subseteq S$, then S' is called **closed under** $\star$ if $a \star b \in S'$ for all $a, b \in S'$.