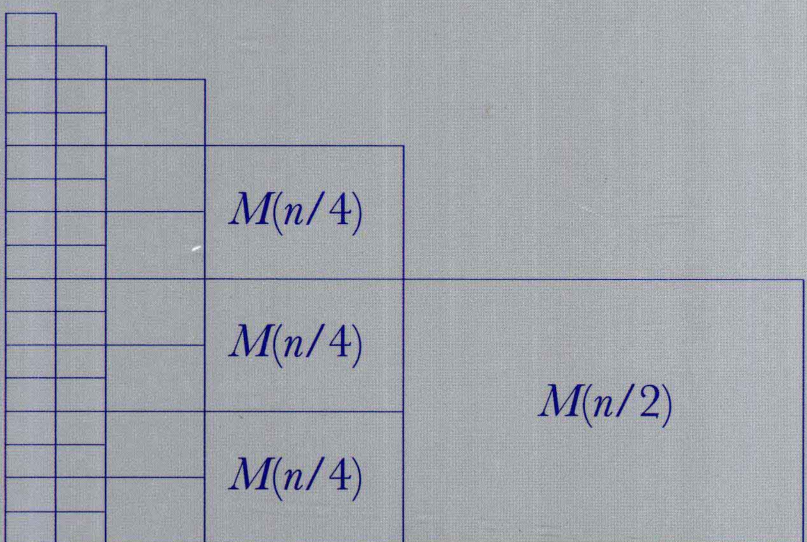
Cambridge Monographs on Applied and Computational Mathematics

Modern Computer Arithmetic

Richard Brent and Paul Zimmermann



Modern Computer Arithmetic

RICHARD P. BRENT
Australian National University, Canberra

PAUL ZIMMERMANN
INRIA, Nancy

CAMBRIDGE
UNIVERSITY PRESS

CAMBRIDGE UNIVERSITY PRESS
Cambridge, New York, Melbourne, Madrid, Cape Town, Singapore,
São Paulo, Delhi, Dubai, Tokyo, 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/9780521194693

First published 2011

Printed in the United Kingdom at the University Press, Cambridge

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

ISBN 978-0-521-19469-3 Hardback

CAMBRIDGE MONOGRAPHS ON APPLIED AND COMPUTATIONAL MATHEMATICS

18 Modern Computer Arithmetic

The *Cambridge Monographs on Applied and Computational Mathematics* series reflects the crucial role of mathematical and computational techniques in contemporary science. The series publishes expositions on all aspects of applicable and numerical mathematics, with an emphasis on new developments in this fast-moving area of research.

State-of-the-art methods and algorithms as well as modern mathematical descriptions of physical and mechanical ideas are presented in a manner suited to graduate research students and professionals alike. Sound pedagogical presentation is a prerequisite. It is intended that books in the series will serve to inform a new generation of researchers.

A complete list of books in the series can be found at
http://www.cambridge.org/uk/series/sSeries.asp?code=MACM
Recent titles include the following:

6. The theory of composites, *Graeme W. Milton*
7. Geometry and topology for mesh generation, *Herbert Edelsbrunner*
8. Schwarz–Christoffel mapping, *Tobin A. Driscoll & Lloyd N. Trefethen*
9. High-order methods for incompressible fluid flow, *M. O. Deville, P. F. Fischer & E. H. Mund*
10. Practical extrapolation methods, *Avram Sidi*
11. Generalized Riemann problems in computational fluid dynamics, *Matania Ben-Artzi & Joseph Falcovitz*
12. Radial basis functions, *Martin D. Buhmann*
13. Iterative Krylov methods for large linear systems, *Henk van der Vorst*
14. Simulating Hamiltonian dynamics, *Benedict Leimkuhler & Sebastian Reich*
15. Collocation methods for Volterra integral and related functional differential equations, *Hermann Brunner*
16. Topology for computing, *Afra J. Zomorodian*
17. Scattered data approximation, *Holger Wendland*
18. Modern computer arithmetic, *Richard P. Brent & Paul Zimmermann*
19. Matrix preconditioning techniques and applications, *Ke Chen*
21. Spectral methods for time-dependent problems, *Jan Hesthaven, Sigal Gottlieb & David Gottlieb*
22. The mathematical foundations of mixing, *Rob Sturman, Julio M. Ottino & Stephen Wiggins*
23. Curve and surface reconstruction, *Tamal K. Dey*
24. Learning theory, *Felipe Cucker & Ding Xuan Zhou*
25. Algebraic geometry and statistical learning theory, *Sumio Watanabe*
26. A practical guide to the invariant calculus, *Elizabeth Louise Mansfield*

Preface

This is a book about algorithms for performing arithmetic, and their implementation on modern computers. We are concerned with software more than hardware – we do not cover computer architecture or the design of computer hardware since good books are already available on these topics. Instead, we focus on algorithms for efficiently performing arithmetic operations such as addition, multiplication, and division, and their connections to topics such as modular arithmetic, greatest common divisors, the fast Fourier transform (FFT), and the computation of special functions.

The algorithms that we present are mainly intended for arbitrary-precision arithmetic. That is, they are not limited by the computer wordsize of 32 or 64 bits, only by the memory and time available for the computation. We consider both integer and real (floating-point) computations.

The book is divided into four main chapters, plus one short chapter (essentially an appendix). Chapter 1 covers integer arithmetic. This has, of course, been considered in many other books and papers. However, there has been much recent progress, inspired in part by the application to public key cryptography, so most of the published books are now partly out of date or incomplete. Our aim is to present the latest developments in a concise manner. At the same time, we provide a self-contained introduction for the reader who is not an expert in the field.

Chapter 2 is concerned with modular arithmetic and the FFT, and their applications to computer arithmetic. We consider different number representations, fast algorithms for multiplication, division and exponentiation, and the use of the Chinese remainder theorem (CRT).

Chapter 3 covers floating-point arithmetic. Our concern is with high-precision floating-point arithmetic, implemented in software if the precision provided by the hardware (typically IEEE standard 53-bit significand) is inadequate. The algorithms described in this chapter focus on *correct rounding*, extending the IEEE standard to arbitrary precision.

Chapter 4 deals with the computation, to arbitrary precision, of functions such as sqrt, exp, ln, sin, cos, and more generally functions defined by power series or continued fractions. Of course, the computation of special functions is a huge topic so we have had to be selective. In particular, we have concentrated on methods that are efficient and suitable for arbitrary-precision computations.

The last chapter contains pointers to implementations, useful web sites, mailing lists, and so on. Finally, at the end there is a one-page *Summary of complexities* which should be a useful *aide-mémoire*.

The chapters are fairly self-contained, so it is possible to read them out of order. For example, Chapter 4 could be read before Chapters 1–3, and Chapter 5 can be consulted at any time. Some topics, such as Newton's method, appear in different guises in several chapters. Cross-references are given where appropriate.

For details that are omitted, we give pointers in the *Notes and references* sections of each chapter, as well as in the bibliography. We have tried, as far as possible, to keep the main text uncluttered by footnotes and references, so most references are given in the Notes and references sections.

The book is intended for anyone interested in the design and implementation of efficient algorithms for computer arithmetic, and more generally efficient numerical algorithms. We did our best to present algorithms that are ready to implement in your favorite language, while keeping a high-level description and not getting too involved in low-level or machine-dependent details. An alphabetical list of algorithms can be found in the index.

Although the book is not specifically intended as a textbook, it could be used in a graduate course in mathematics or computer science, and for this reason, as well as to cover topics that could not be discussed at length in the text, we have included exercises at the end of each chapter. The exercises vary considerably in difficulty, from easy to small research projects, but we have not attempted to assign them a numerical rating. For solutions to the exercises, please contact the authors.

We welcome comments and corrections. Please send them to either of the authors.

Richard Brent and Paul Zimmermann
Canberra and Nancy
`MCA@rpbrent.com`
`Paul.Zimmermann@inria.fr`

Acknowledgements

We thank the French National Institute for Research in Computer Science and Control (INRIA), the Australian National University (ANU), and the Australian Research Council (ARC), for their support. The book could not have been written without the contributions of many friends and colleagues, too numerous to mention here, but acknowledged in the text and in the Notes and references sections at the end of each chapter.

We also thank those who have sent us comments on and corrections to earlier versions of this book: Jörg Arndt, Marco Bodrato, Wolfgang Ehrhardt (with special thanks), Steven Galbraith, Torbjörn Granlund, Guillaume Hanrot, Marc Mezzarobba, Jean-Michel Muller, Denis Roegel, Wolfgang Schmid, Arnold Schönhage, Sidi Mohamed Sedjelmaci, Emmanuel Thomé, and Mark Wezelenburg. Two anonymous reviewers provided very helpful suggestions. Jérémie Detrey and Anne Rix helped us in the copy-editing phase.

The *Mathematics Genealogy Project* (`http://www.genealogy.ams.org/`) and Don Knuth's *The Art of Computer Programming* [142] were useful resources for details of entries in the index.

We also thank the authors of the LaTeX program, which allowed us to produce this book, the authors of the `gnuplot` program, and the authors of the GNU MP library, which helped us to illustrate several algorithms with concrete figures.

Finally, we acknowledge the contribution of Erin Brent, who first suggested writing the book; and thank our wives, Judy-anne and Marie, for their patience and encouragement.

Notation

$\mathbb{C}$	set of complex numbers
$\widehat{\mathbb{C}}$	set of extended complex numbers $\mathbb{C} \cup \{\infty\}$
$\mathbb{N}$	set of natural numbers (nonnegative integers)
$\mathbb{N}^*$	set of positive integers $\mathbb{N}\backslash\{0\}$
$\mathbb{Q}$	set of rational numbers
$\mathbb{R}$	set of real numbers
$\mathbb{Z}$	set of integers
$\mathbb{Z}/n\mathbb{Z}$	ring of residues modulo n
C^n	set of (real or complex) functions with n continuous derivatives in the region of interest
$\Re(z)$	real part of a complex number z
$\Im(z)$	imaginary part of a complex number z
$\bar{z}$	conjugate of a complex number z
$\lvert z\rvert$	Euclidean norm of a complex number z, or absolute value of a scalar z
B_n	Bernoulli numbers, $\sum_{n\geq 0} B_n z^n/n! = z/(e^z - 1)$
C_n	scaled Bernoulli numbers, $C_n = B_{2n}/(2n)!$, $\sum C_n z^{2n} = (z/2)/\tanh(z/2)$
T_n	tangent numbers, $\sum T_n z^{2n-1}/(2n-1)! = \tan z$
H_n	harmonic number $\sum_{j=1}^{n} 1/j$ (0 if $n \leq 0$)
$\binom{n}{k}$	binomial coefficient "n choose k" = $n!/(k!\,(n-k)!)$ (0 if $k < 0$ or $k > n$)

β	"word" base (usually 2^{32} or 2^{64}) or "radix" (floating-point)
n	"precision": number of base β digits in an integer or in a floating-point significand, or a free variable
ε	"machine precision" $\beta^{1-n}/2$ or (in complexity bounds) an arbitrarily small positive constant
η	smallest positive subnormal number
$\circ(x), \circ_n(x)$	rounding of real number x in precision n (Definition 3.1)
$\mathrm{ulp}(x)$	for a floating-point number x, one unit in the last place
$M(n)$	time to multiply n-bit integers, or polynomials of degree $n-1$, depending on the context
$\sim M(n)$	a function $f(n)$ such that $f(n)/M(n) \to 1$ as $n \to \infty$ (we sometimes lazily omit the "$\sim$" if the meaning is clear)
$M(m,n)$	time to multiply an m-bit integer by an n-bit integer
$D(n)$	time to divide a $2n$-bit integer by an n-bit integer, giving quotient and remainder
$D(m,n)$	time to divide an m-bit integer by an n-bit integer, giving quotient and remainder
$a \mid b$	a is a divisor of b, that is $b = ka$ for some $k \in \mathbb{Z}$
$a = b \bmod m$	modular equality, $m \mid (a-b)$
$q \leftarrow a \operatorname{div} b$	assignment of integer quotient to q ($0 \le a - qb < b$)
$r \leftarrow a \bmod b$	assignment of integer remainder to r ($0 \le r = a - qb < b$)
(a,b)	greatest common divisor of a and b
$\left(\frac{a}{b}\right)$ or $(a \mid b)$	Jacobi symbol (b odd and positive)
iff	if and only if
$i \wedge j$	bitwise *and* of integers i and j, or logical *and* of two Boolean expressions
$i \vee j$	bitwise *or* of integers i and j, or logical *or* of two Boolean expressions
$i \oplus j$	bitwise *exclusive-or* of integers i and j
$i \ll k$	integer i multiplied by 2^k
$i \gg k$	quotient of division of integer i by 2^k
$a \cdot b, \quad a \times b$	product of scalars a, b
$a * b$	cyclic convolution of vectors a, b
$\nu(n)$	2-valuation: largest k such that 2^k divides n ($\nu(0) = \infty$)
$\sigma(e)$	length of the shortest addition chain to compute e
$\phi(n)$	Euler's totient function, $\#\{m : 0 < m \le n \wedge (m,n) = 1\}$

$\deg(A)$	for a polynomial A, the degree of A
$\mathrm{ord}(A)$	for a power series $A = \sum_j a_j z^j$, $\mathrm{ord}(A) = \min\{j : a_j \neq 0\}$ ($\mathrm{ord}(0) = +\infty$)
$\exp(x)$ or e^x	exponential function
$\ln(x)$	natural logarithm
$\log_b(x)$	base-b logarithm $\ln(x)/\ln(b)$
$\lg(x)$	base-2 logarithm $\ln(x)/\ln(2) = \log_2(x)$
$\log(x)$	logarithm to any fixed base
$\log^k(x)$	$(\log x)^k$
$\lceil x \rceil$	ceiling function, $\min\{n \in \mathbb{Z} : n \geq x\}$
$\lfloor x \rfloor$	floor function, $\max\{n \in \mathbb{Z} : n \leq x\}$
$\lfloor x \rceil$	nearest integer function, $\lfloor x + 1/2 \rfloor$
$\mathrm{sign}(n)$	$+1$ if $n > 0$, -1 if $n < 0$, and 0 if $n = 0$
$\mathrm{nbits}(n)$	$\lfloor \lg(n) \rfloor + 1$ if $n > 0$, 0 if $n = 0$
$[a, b]$	closed interval $\{x \in \mathbb{R} : a \leq x \leq b\}$ (empty if $a > b$)
(a, b)	open interval $\{x \in \mathbb{R} : a < x < b\}$ (empty if $a \geq b$)
$[a, b)$, $(a, b]$	half-open intervals, $a \leq x < b$, $a < x \leq b$ respectively
${}^t[a, b]$ or $[a, b]^t$	column vector $\begin{pmatrix} a \\ b \end{pmatrix}$
$[a, b; c, d]$	2×2 matrix $\begin{pmatrix} a & b \\ c & d \end{pmatrix}$
$\widehat{a}_j$	element of the (forward) Fourier transform of vector a
$\widetilde{a}_j$	element of the backward Fourier transform of vector a
$f(n) = O(g(n))$	$\exists c, n_0$ such that $\lvert f(n) \rvert \leq cg(n)$ for all $n \geq n_0$
$f(n) = \Omega(g(n))$	$\exists c > 0, n_0$ such that $\lvert f(n) \rvert \geq cg(n)$ for all $n \geq n_0$
$f(n) = \Theta(g(n))$	$f(n) = O(g(n))$ and $g(n) = O(f(n))$
$f(n) \sim g(n)$	$f(n)/g(n) \to 1$ as $n \to \infty$
$f(n) = o(g(n))$	$f(n)/g(n) \to 0$ as $n \to \infty$
$f(n) \ll g(n)$	$f(n) = O(g(n))$
$f(n) \gg g(n)$	$g(n) \ll f(n)$
$f(x) \sim \sum_0^n a_j/x^j$	$f(x) - \sum_0^n a_j/x^j = o(1/x^n)$ as $x \to +\infty$
123 456 789	123456789 (for large integers, we may use a space after every third digit)

$xxx.yyy_\rho$	a number $xxx.yyy$ written in base ρ; for example, the decimal number 3.25 is 11.01_2 in binary
$\frac{a}{b+}\,\frac{c}{d+}\,\frac{e}{f+}\cdots$	continued fraction $a/(b+c/(d+e/(f+\cdots)))$
$\|A\|$	determinant of a matrix A, e.g. $\begin{vmatrix} a & b \\ c & d \end{vmatrix} = ad - bc$
$\mathrm{PV}\int_a^b f(x)\,\mathrm{d}x$	Cauchy principal value integral, defined by a limit if f has a singularity in (a, b)
$s \,\|\|\, t$	concatenation of strings s and t
▷ `<text>`	comment in an algorithm
□	end of a proof

Contents

1
Integer arithmetic

In this chapter, our main topic is integer arithmetic. However, we shall see that many algorithms for polynomial arithmetic are similar to the corresponding algorithms for integer arithmetic, but simpler due to the lack of carries in polynomial arithmetic. Consider for example addition: the sum of two polynomials of degree n always has degree at most n, whereas the sum of two n-digit integers may have $n + 1$ digits. Thus, we often describe algorithms for polynomials as an aid to understanding the corresponding algorithms for integers.

1.1 Representation and notations

We consider in this chapter algorithms working on integers. We distinguish between the logical – or mathematical – representation of an integer, and its physical representation on a computer. Our algorithms are intended for "large" integers – they are not restricted to integers that can be represented in a single computer word.

Several physical representations are possible. We consider here only the most common one, namely a dense representation in a fixed base. Choose an integral *base* $\beta > 1$. (In case of ambiguity, β will be called the *internal* base.) A positive integer A is represented by the length n and the digits a_i of its base β expansion

$$A = a_{n-1}\beta^{n-1} + \cdots + a_1\beta + a_0,$$

where $0 \leq a_i \leq \beta - 1$, and a_{n-1} is sometimes assumed to be non-zero. Since the base β is usually fixed in a given program, only the length n and the integers $(a_i)_{0 \leq i < n}$ need to be stored. Some common choices for β are 2^{32} on a 32-bit computer, or 2^{64} on a 64-bit machine; other possible choices