Programming Techniques in Computer Algebra

Prof. Dr. Wolfram Koepf

Universität Kassel
http://www.mathematik.uni-kassel.de/~koepf

March 18, 2010
Yaounde, Cameroon

Abstract

Topics of This Talk

- In this talk important programming techniques and mathematical algorithms are discussed and presented.
- After starting with iteration and recursion, we show the efficiency of divide-and-conquer algorithms.
- The Extended Euclidean Algorithm and modular powers form the basis for the RSA cryptographic system.
- The talk finishes with an implementation of RSA and demonstrates an error-correcting code.

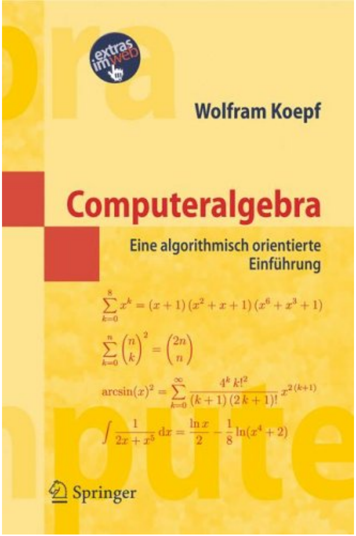
Summary

- Text Book on Computer Algebra
- Arithmetic of Large Integers
- Prime Numbers and Powers
- The RSA Cryptographic System

The programs that I will consider in this talk were developed for my (unfortunately German language!) text book

Computeralgebra, Springer, Berlin/Heidelberg, 2006

and can be downloaded from my web page:



http://www.mathematik.uni-kassel.de/ koepf/CA

Arithmetic of Large Integers

Greatest Common Divisor

- Already the internal simplification of rational numbers needs the computation of greatest common divisors.
- However: The factorization of large integers is very time consuming! *Mathematica*

Euclidean Algorithm

To compute the GCD (*recursively*), one uses the relations:

- $\mathrm{GCD}(a, b) = \mathrm{GCD}(|a|, |b|)$, if $a < 0$ or $b < 0$
- $\mathrm{GCD}(a, b) = \mathrm{GCD}(b, a)$, if $a < b$
- $\mathrm{GCD}(a, 0) = a$
- $\mathrm{GCD}(a, b) = \mathrm{GCD}(b, a \mod b)$ *Mathematica*

Extended Euclidean Algorithm

Extended Euclidean Algorithm

- An *iterative* application of the Euclidean Algorithm yields additional informations.
- For $a, b \in \mathbb{Z}$ this so-called *Extended Euclidean Algorithm* yields $g = \mathrm{GCD}(a, b)$ and coefficients $s, t \in \mathbb{Z}$ such that

$$g = s\,a + t\,b\,.$$

- Conversely: If for suitable $s, t \in \mathbb{Z}$ the relation $1 = s\,a + t\,b$ is valid, then a and b are relatively prime. *Mathematica*

Prime Number Test

Fermat's Little Theorem

For a prime number $p \in \mathbb{P}$ and $a \in \mathbb{Z}$ the relation

$$a^p = a \pmod{p}$$

is valid, or respectively

$$a^{p-1} = 1 \pmod{p}\ , \quad \text{if } \mathrm{GCD}(a,p) = 1\ .$$

Fermat Test

If this relation is *not* valid for a number $a \in \mathbb{Z}$, the p cannot be a prime number! *Mathematica*

Efficient Computation of Powers

Divide and Conquer Algorithm

- To utilize the Fermat test, modular powers should be computed very efficiently.
- The modular power $a^n \pmod{p}$ is computed efficiently by reducing powers of size n to powers of size $n/2$.
- Such a method is called a *Divide and Conquer Algorithm*.
- Recursive formulation of this algorithm:
  - $a^0 \mod p = 1$
  - $a^n \mod p = (a^{n/2} \mod p)^2 \mod p$ for even n
  - $a^n \mod p = (a^{n-1} \mod p) \cdot a \mod p$ for odd n
- *Mathematica*
- Question: How does an iterative version of this algorithm work?

Carmichael Numbers

Carmichael Numbers

- An integer p which is *not* prime, but nevertheless satisfies Fermat's criterion, is called a *Carmichael number*.
- Fact: There are infinitely many Carmichael numbers.
- Carmichael numbers are not recognized by Fermat's test.
- We compute the first Carmichael numbers. *Mathematica*

Criterion for Carmichael Numbers

An integer $p \in \mathbb{N} \subset \mathbb{P}$ is a Carmichael number if and only if

- $p = p_1 \cdots p_n$ with pairwise different primes $p_k \in \mathbb{P}$
- $p_k - 1 | p - 1$ for all $k = 1, \ldots, n$. *Mathematica*

Cryptographic Systems

Cryptographic Systems

- By a cryptographic system a message N is encoded by a function E and a *key* e

$$K = E_e(N) \; .$$

- The decoding is carried out by a function D with key d:

$$D_d(K) = D_d(E_e(N)) = N \; .$$

- The functions E and D should be efficiently computable.
- One problem is the *key exchange*.

Diffie-Hellman Key Exchange

Modular Logarithm

- The inverse of the real exponential function $x \mapsto 2^x$ is simple to compute.
- The inverse of the integer exponential function $x \mapsto 2^x$ is also simple to compute.
- However, the inverse of the modular exponential function $x \mapsto 2^x \pmod{p}$ is difficult to compute. *Mathematica*

Diffie-Hellman Key Exchange

Protocol of Diffie-Hellman Key Exchange (1976)

- Anna and Barbara want to exchange a common key. They choose numbers $g \in \mathbb{N}$ and $p \in \mathbb{P}$. These can be assumed to be public.

A chooses $a < p$	B chooses $b < p$
A computes $\alpha := g^a \mod p$	B computes $\beta := g^b \mod p$
A sends α to B	B sends β to A
A computes $s := \beta^a \mod p$	B computes $t := \alpha^b \mod p$

Correctness of algorithm

$$s = \beta^a = (g^b)^a = (g^a)^b = \alpha^b = t\ .$$

Asymmetric Cryptography

Asymmetric Cryptography

- The RSA algorithm is an example of an *asymmetric* cryptographic system.
- In such systems all participants have their own *personal* keys e and d.
- Their encoding keys e are made *public*, whereas their decoding keys d are kept secret (*private*).
- Key exchange of the private key d is therefore not necessary.

The RSA Cryptographic System

Cryptographic Protocol of the RSA System (1978)

The potential recipient and participant of the system

- computes a 200 digit decimal number $m = p \cdot q$ with $p, q \in \mathbb{P}$,
- computes $\varphi = (p-1)(q-1)$,
- computes and publishes a public key e which is relatively prime with φ
- and computes his private key d with the property $e \cdot d = 1 \mod \varphi$.
- The encoding and decoding functions are given by

$$K = E_e(N) = N^e \pmod{m} \quad \text{and} \quad D_d(K) = K^d \pmod{m} .$$

The RSA Cryptographic System

What do we Need fo RSA?

- Computation of large prime numbers: `NextPrime`
- Powers $N^e \mod m$ must be computed efficiently: `PowerMod`
- We must compute the modular inverse $d = e^{-1} \pmod{m}$: `PowerMod`
- The latter is actually an application of the Extended Euclidean Algorithm. *Mathematica*
- The correctness of the algorithm results from Fermat's Little Theorem.
- Above all: With suitable auxiliary functions messages are converted towards integers and eventually transformed back. *Mathematica*

Error Correcting Codes

Why is Error-Correction Necessary?

- We saw that cryptography assumes that messages are transferred error-free.
- If you use a music CD or a CD-ROM, it can contain up to hundreds of thousands of errors!
- All these errors must be corrected. Otherwise you cannot hear the music or your programs do not work.
- For this purpose highly specialized, so called Reed-Solomon codes, are used.
- Let's demonstrate such a code which can correct two errors. *Mathematica*

Many Thanks for Your Interest!