Dilla University

Department of Mathematics

Topics In Algebra I Lecture Note

by

Dereje Kifle (PhD)

Contents

Chapter 1

Introduction

1.1 What is Computer Algebra?

Mathematicians in the old period, say before 1850 A.D., solved the majority of mathematical problems by extensive calculations. A typical example of this type of mathematical problem solver is Euler. So it is not astonishing that in the 18th and beginning 19th centuries many mathematicians were real wizards of computation. However, during the 19th century the style of mathematical research changed from quantitative to qualitative aspects. A number of reasons were responsible for this change, among them the importance of providing a sound basis for the vast theory of analysis. But the fact that computations gradually became more and more complicated certainly also played its role. This impediment has been removed by the advent of modern digital computers in general and by the development of program systems in computer algebra in particular. By the aid of computer algebra the capacity for mathematical problem solving has been decisively improved.

Even in our days many mathematicians think that there is a natural division of labor between man and computer: a person applies the appropriate algebraic transformations to the problem at hand and finally arrives at a program which then can be left to a "number crunching" computer. But already in 1844 Lady Augusta Ada Byron, countess Lovelace, recognized that this division of labor is not inherent in mathematical problem solving and may be even detrimental.

A modern digital computer is a "universal" machine capable of carrying out an arbitrary algorithm, i.e. an exactly specified procedure, algebraic algorithms being no exceptions.

Now what exactly is symbolic algebraic computation, or in other words computer algebra? In his introduction to (Buchberger et al. 1983) R. Loos gave the following attempt at a definition:

Computer algebra is that part of computer science which designs, analyzes, implements, and applies algebraic algorithms.

While it is arguable whether computer algebra is part of computer science or mathematics, we certainly agree with the rest of the statement. In fact, in our view computer algebra is a special form of scientific computation, and it comprises a wide range of basic goals, methods, and applications. More formally,

Definition 1.1.1. *Computer Algebra* is a discipline between mathematics and computer science which deals with designing, analyzing, implementing, and applying algebraic algorithms.

In contrast to numerical computation the emphasis is on computing with symbols representing mathematical concepts. Of course that does not mean that computer algebra is devoid of computations with numbers. Decimal or other positional representations of integers, rational numbers and the like appear in any symbolic computation. But integers or real numbers are not the sole objects. In addition to these basic numerical entities computer algebra deals with polynomials, rational functions, trigonometric functions, algebraic numbers, etc. That does not mean that we will not need numerical algorithms any more. Both forms of scientific computation have their merits and they should be combined in a computational environment. For instance, in order to compute an approximate solution to a differential equation it might be reasonable to determine the first n terms of a power series solution by exact methods from computer algebra before handing these terms over to a numerical package for evaluating the power series.

Summarizing, computer algebra has two fundamental goals: Provide algorithms for computations with algebraic structures, like fields, vector spaces, rings, ideals, and modules to the computer. And use the algorithms and their implementations to solve mathematical problems in theory and applications. Here, computations usually refer to exact, that is, symbolic ones. However, in some cases, numerical computations can be helpful in obtaining exact results.

1.2 Application Areas of Computer Algebra

Computer algebra is interdisciplinary in nature, with links to quite a number of areas in mathematics, with applications in mathematics, other branches of science, and engineering:

- Through computer algebra methods, a number of mathematical disciplines become accessible to experiments. This is in particular true for various parts of algebra, number theory, and geometry.

- Modern application areas of mathematics such as cryptography, coding theory, CAD, robotics, algebraic statistics, and algebraic biology heavily rely on computer algebra.

1.3 Why Should we use Computer Algebra?

Of course, there are practical problems, that can be solved by computer algebra, for example, in cryptography, robotics, algebraic statistics, computational biology, and

physics. On the other hand, experiments with the computer allow you to get an insight into theoretical problems and test conjectures. In many settings, you can even obtain theoretical results by handling just a single special case by computer.

As a consequence one can build decision algorithms on computer algebra, e.g. for the factorizability of polynomials, the equivalence of algebraic expressions, the solvability of integration problems in a specific class of expressions, the solvability of certain differential equation problems, the solvability of systems of algebraic equations, the validity of geometric statements, the parametrizability of algebraic curves.

What is an Algorithm?

An *Algorithm* is a set of instructions for solving a particular problem in *finitely many, well-defined steps*. Starting from a given *input*, the instructions describe a computation which eventually will produce an *output* and *terminate*. The transition from one step to the next one is not necessarily *deterministic*: *probabilistic algorithms* incorporate random input, which may lead to random performance and random output. For example,

Algorithm 1.1 `Sample Algorithm`

Input: some input.

Output: some output m.

instruction

What are Algebraic Algorithms?

Algebraic algorithms deal with algebraic objects, make us of algebraic methods, and are based on algebraic theorems. Objects are represented exactly and calculations are carried through exactly (no approximation is applied at any step).

Analyzing Algorithms

One way of measuring the efficiency of an algorithm is to give asymptotic bounds on its running time which depend on the size of the input.

Designing Algorithms

When designing algorithms, we will describe them in a somewhat informal way which makes use of the structural conventions of a programming language. We refer to such a description as *pseudocode*, see the above algorithm.

Implementations

There is a large variety of computer algebra systems suiting different needs. Some of the most widely used systems are Mathematica, Maple, Derive, Reduce, SINGULAR, MAXIMA, MAGMA, COCOA, GAP, JULIA, SAGE and MATLAB. Among these computer algebra systems, in this lecture, we work in this lecture with SINGULAR, MAXIMA, GAP, JULIA and SAGE which are open computer algebra systems, that is, they can be downloaded for free from internet.

1.4 Numbers

One of the most important algorithms in mathematics is Euclidean algorithm for finding the greatest common divisor. In a generalized form, it will be presented explicitly or implicitly in many algorithms we will discuss later on.

Definition 1.4.1. For integers a and b, $b \neq 0$, b is called a *divisor* of a, if there exists an integer c such that $a = bc$.

We denote by $b \mid a$ if b is a divisor of a and by $b \nmid a$ if it is not.

Lemma 1.4.2 (**Division with Remainder**). *For $a, b \in \mathbb{Z}$, $b \neq 0$, there are $q, r \in \mathbb{Z}$ with $a = b \cdot q + r$ and $0 \leq r < |b|$.*

Proof. Without loss of generality $b > 0$. The set

$$\{w \in \mathbb{Z} \mid b \cdot w > a\} \neq \emptyset$$

has a smallest element w. Then set $q := w - 1$ and $r := a - qb$. □

Definition 1.4.3. An integral domain R together with a function $d : R \to \mathbb{N} \cup \{\infty\}$ is a *Euclidean domain* if for all $a, b \in R$ with $b \neq 0$, we can divide a by b with remainder, so that there exist $q, r \in R$ such that $a = qb + r$ and $d(r) < d(b)$. We say that $q = a \operatorname{quo} b$ is the *quotient* and $r = a \operatorname{rem} b$ the *remainder*, although q and r need to be unique. Such a d is called a *Euclidean function* on R.

Example 1.4.4.

(i) The function $d : \mathbb{Z} \to \mathbb{N} \cup \{-\infty\}$ defined by $d(a) = |a|$ is an Euclidean function. Here the quotient and the remainder can be made unique with additional requirement that $r \geq 0$.

(ii) The function $d : F[x] \to \mathbb{N} \cup \{-\infty\}$ defined by $d(a) = \deg a$ is an Euclidean function. Here the quotient and the remainder are unique without any further requirement.

Definition 1.4.5. Let R be a ring and $a, b, c \in R$. Then

(1) c is a *greatest common divisor (or gcd)* of a and b if

i) $c|a$ and $c|b$,

ii) if $d|a$ and $d|b$, then $d|c$ for all $d \in R$.

(2) c is called a *least common multiple* of a and b if

i) $a|c$ and $b|c$,

ii) if $a|d$ and $b|d$, then $c|d$ for all $d \in R$.

(3) A *unit* $u \in R$ is any element with a multiplicative inverse $v \in R$, that is, $uv = 1$.

(4) The elements a and b are *associate*, denoted as $a \sim b$, if $a = ub$ for a unit $u \in R$

Remark 1.4.6.

- Neither the gcd nor the lcm are unique, but all gcds of a and b are precisely the associates of one of them and so is for the lcm. For example, 3 and -3 are all gcds of 12 and 15 in $\mathbb{Z}$ because 1 and -1 are the only units in $\mathbb{Z}$.

- For $R = \mathbb{Z}$, we may define $\gcd(a, b)$ as the unique nonnegative greatest common divisor and $\operatorname{lcm}(a, b)$ as the unique nonnegative least common multiple of a and b.

Remark 1.4.7. Let R be an integral domain and $a, b \in R$ such that $\gcd(a, b) = c$ exists. Clearly, all such divisors are obtained by multiplying c with a unit of R. In other words, the gcd's form an equivalence class under being associated. In this lecture, we always assume that in each such equivalence class a *normal form* is d selected. If the class is represented by $a \in R$, we write $N(a)$ for the normal form. Here N is defined as follows:

$$N(a) := \begin{cases} 0 & \text{if } a = 0 \\ 1 & \text{if } a = 1 \\ a/U(a) & \text{otherwise} \end{cases}$$

where $U(a)$ is called the *leading unit* of $a \in R$ such that $a \sim N(a)$, that is, $a = U(a)N(a)$. For $a = 0$, we set $U(a) = 1$.

Note that

- two elements of R have the same normal form if and only if they are associate, that is, $N(a) = N(b)$ iff $a \sim u \cdot b$ for some unit $u \in R$.

- the normal form of a product is equal to the products of the normal forms, that is, $N(a \cdot b) = N(a) \cdot N(b)$.

Example 1.4.8.

i) If $R = \mathbb{Z}$, $U(a) = \text{sign}(a)$ if $a \neq 0$ and $N(a) = |\, a \,|$ defines a normal form, so that an integer is normalized if and only if it is nonnegative.

ii) If $R = F[x]$ for a field F, then letting $U(a) = \text{lc}(a)$ (with the convention that $U(0) = 1$) and $N(a) = a/\text{U(a)}$ defines a normal form, and a nonzero polynomial is normalized if and only if it is monic.

Theorem 1.4.9 (**Euclidean Algorithm**). *Suppose* $a_1, a_2 \in \mathbb{Z}\backslash\{0\}$. *Successive division with remainder terminates*

$$\begin{aligned} a_1 &= q_1 a_2 + a_3 \\ &\vdots \\ a_j &= q_j a_{j+1} + a_{j+2} \\ &\vdots \\ a_{n-2} &= q_{n-2} a_{n-1} + a_n \\ a_{n-1} &= q_{n-1} a_n + 0 \end{aligned}$$

and

$$\gcd(a_1, a_2) = a_n \,.$$

Reading the equation backwards

$$\begin{aligned} a_n &= a_{n-2} - q_{n-2} a_{n-1} \\ &\vdots \\ a_3 &= a_1 - q_1 a_2 \end{aligned}$$

gives a representation

$$\gcd(a_1, a_2) = x \cdot a_1 + y \cdot a_2$$

with $x, y \in \mathbb{Z}$.

Proof. We have $|a_{i+1}| < |a_i|$ for $i \geq 2$ so after finitely many steps $a_i = 0$. Then a_n divides a_{n-1}, hence also $a_{n-2} = q_{n-2} a_{n-1} + a_n$ and inductively $a_{n-2}, \ldots, a_1$. If t is a divisor of a_1 and a_2, then also of $a_3, \ldots, a_n$. □

Example 1.4.10. We compute the gcd of 36 and 15:

$$\begin{aligned} 36 &= 2 \cdot 15 + 6 \\ 15 &= 2 \cdot 6 + 3 \\ 6 &= 2 \cdot 3 + 0 \end{aligned}$$

hence $\gcd(36, 15) = 3$. Furthermore, we can express $\gcd(36, 15)$ as a $\mathbb{Z}$ linear combination of 36 and 15:

$$3 = 15 - 2 \cdot 6 = 15 - 2 \cdot (36 - 2 \cdot 15) = 5 \cdot 15 + (-2) \cdot 36 \,.$$

Given a normal form, we define $\gcd(a, b)$ to be the unique normalized associate of all greatest common divisors of a and b, and similarly $\operatorname{lcm}(a, b)$ as the normalized associate of all least common multiples of a and b. Thus $\gcd(a, b) > 0$ for $R = \mathbb{Z}$ and $\gcd(a, b)$ is monic for $R = F[x]$ if at least one of a, b is nonzero, and $\gcd(0, 0) = 0$ in both cases.

The Euclidean algorithm in Theorem 1.4.9 can easily be summarized in pseudocode form as follows:

Algorithm 1.2 `Euclid's Algorithm for integers`

Input: $m, n \in \mathbb{Z}$.

Output: $\gcd(m, n)$.

1: $a := n, b := m$
2: **while** $b \neq 0$ **do**
3: $\quad r := a \operatorname{rem} b$// division with remainder
4: $\quad a := b, b := r$
5: $a := n(a)$// normal form
6: **return** a

Definition 1.4.11. An element $p \in \mathbb{Z}_{>1}$ is called *prime number*, if $p = a \cdot b, a, b \in \mathbb{Z}_{\geq 1}$ implies $a = 1$ or $b = 1$ and we call p a *composite number* if it is not prime.

Theorem 1.4.12 (**The Fundamental Theorem of Arithmetic**). *Every integer* $n \in \mathbb{Z} \setminus \{-1, 0, 1\}$ *has a unique representation may be expressed uniquely in the form*

$$n = \pm \prod_{i=1}^{k} p_i^{\alpha_i}$$

with ***prime factors*** $p_1 < \ldots < p_k$ *and* $\alpha_i \in \mathbb{N}$.

Algorithm 1.4.13. Let $n \in \mathbb{Z}$ be composite. The smallest prime factor p of n satisfies

$$p \leq m := \lfloor \sqrt{n} \rfloor \,.$$

If we know all primes $p \leq m$, then we can test $p|n$ by division with remainder and, hence, factor n.

Note that $\lfloor x \rfloor = \max\{a \in \mathbb{Z} \mid a \leq x, x \in \mathbb{R}\}$, the largest integer less than or equal to x. The function $\lfloor x \rfloor$ is called the *floor* function of x.

Algorithm 1.4.14 (Sieve of Eratosthenes). We can find all prime numbers smaller than n in the following way: Note all numbers from 2 to n. Starting with $p = 2$, delete all $a \cdot p$ for $a > 1$, and continue with the next largest number p which not has been deleted. Note that p is prime, since it is not a multiple of smaller prime. Stop if $p > \sqrt{n}$.

Example 1.4.15. We compute all primes ≤ 21:

2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
2	3		5		7		9		11		13		15		17		19		21
2	3		5		7				11		13				17		19		

In the first step we delete all multiple of 2, in the second step all multiple of 3. All remaining numbers are prime, since $5 > \sqrt{21}$.

One can even describe the distribution of the primes over all integers:

Theorem 1.4.16 (Prime Number Theorem). *For $x \in \mathbb{R}_{>0}$ let*

$$\pi(x) = |\{p \leq x \mid p \in \mathbb{N} \text{ prime }\}|\,.$$

Then

$$\lim_{x \to \infty} \frac{\pi(x)}{\frac{x}{\ln(x)}} = 1\,.$$

Example 1.4.17. The number of prime ≤ 21 is $\pi(21) = 8$, see Example 1.4.15.

Computer algebra in this spirit has many theoretical applications in number theory and algebraic geometry and practical applications, for example, in coding theory or RSA public key cryptography.

In general, number theory explores the properties of numbers, most importantly the interaction of addition and multiplication. This leads to many problems which are easy to formulate, but highly non-trivial to solve. The most famous one is Fermat's last theorem of 1637: There is no (non-trivial) integer solution of the equation

$$x^n + y^n = z^n$$

for $n \geq 3$. With the help of a computer one can test Fermat's last theorem for very large n (using the theoretical result that you only have to test it for so called irregular primes). Fermat's last theorem was finally proven in 1995 (by A. Wiles) after 350 years of work of many people, which led to many new concepts in mathematics.

1.5 Exercises

Exercise 1.5.1.

(a) Implement the Euclidean Algorithm for computing the greatest common divisor in $\mathbb{Z}$. Test your implementation at examples.

(b) Use your implementation to cancel

$$\frac{90189116021}{18189250063}.$$

Exercise 1.5.2 (Later). Let p be a prime and $\mathbb{F}_p = \mathbb{Z}/p$ the field with p elements.

(a) Use an analogue of the sieve of Eratosthenes to find all irreducible polynomials in $\mathbb{F}_2[x]$ of degree ≤ 3.

(b) Factor $x^5 + x^2 + x + 1 \in \mathbb{F}_2[x]$ into a product of irreducible polynomials.

(c) Determine all elements of $K = \mathbb{F}_2[x]/(x^2 + x + 1)$, the addition table of K, and the multiplication table of K. Prove that K is a field. What is the characteristic of K?

Exercise 1.5.3. Write a procedure to compute

$$\pi(x) = |\{p \leq x \mid p \in \mathbb{N} \text{ prime } \}|$$

for $x > 0$.

Exercise 1.5.4. Write a procedure to compute $n!$ for any $n \in \mathbb{Z}_{\geq 1}$.

Exercise 1.5.5. Write a procedure which returns an n-th Fibonacci number.

Definition 1.5.6. The Fibonacci numbers are the sequence of numbers $F_{n\,n=1}^{\infty}$ defined by the linear recurrence equation $F_n = F_{n-1} + F_{n-2}$ with $F_1 = F_2 = 1$.

How can I tell if a given number is a Fibonacci number? a is a Fibonacci number if and only if $5a^2 + 4$ or $5a^2 - 4$ is a square number.