Computational Number Theory

Lecture Notes

N.R.Aravind

Acknowledgements

The latex source file was built using the LegrandOrangeBook template (copyright 2022, Goro Akechi) available at BOOK-WEBSITE.COM and licensed under the Creative Commons Attribution-NonCommercial 4.0 License (the "License"). A copy of the License is available at https://creativecommons.org/licenses/by-nc-sa/4.0. Minor modifications were made in the use of the template.

I thank the following students who pointed out typos and mistakes in the notes: Gunangad Pal Singh, Krishn Vikas Kher.

Contents

Preface 7

I Polynomials in One Variable

1 Two Equations from Ancient Times 11

1.1 The Cubic Equation 11

1.1.1 Geometric solution 11

1.1.2 Algebraic solution and depressed cubics 12

1.1.3 Solving a depressed cubic 13

1.1.4 Quartic and higher powers: More history 14

1.2 Fast Arithmetic Operations 14

1.3 The Equation $ax + by = c$ 14

1.3.1 Bezout's Lemma 15

1.3.2 Euclid's Algorithm 16

1.3.3 The Extended Euclidean algorithm 17

1.4 Notes and Further Reading 18

1.5 Exercises 19

2 The Fundamental Theorem of Arithmetic 21

2.1 Prime numbers 21

2.2 Euclid's lemma 21

2.3 The fundamental theorem 21

2.4 Notes and Further Reading 22

2.5 Exercises 22

3 Congruences 23

3.1 Definition and properties 23

3.2 Linear congruences and the Chinese Remainder Theorem 24

3.3 Fermat's little theorem 25

3.4 The ring $\mathbb{Z}_n$ 26

3.4.1 Chinese remainder theorem as an isomorphism 26

3.4.2 What's a ring? 26

3.5 Arithmetic in $\mathbb{Z}_n$ 27

3.6 The set $\mathbb{Z}_n^*$ and Euler's totient function 27

3.7 Order and primitive roots 28

3.8 Exercises 29

4 Polynomials over $\mathbb{Z}_n$ 31

4.1 The ring $\mathbb{Z}_n[x]$ 31

4.2 Lagrange's theorem 31

4.3 Euclid's algorithm and unique factorization for polynomials 31

4.4 The equations $x^d = 1$ and $x^d = a$ in $\mathbb{Z}_p$ 32

4.5 Application: The RSA Algorithm 33

4.6 Exercises 33

5 The Quadratic Equation in $\mathbb{Z}_p$ 35

5.1 Quadratic Residues 35

5.2 Application: Coin Tossing over a telephone 35

5.3 The Legendre Symbol 36

5.4 The equation $x^2 = a$: Two easy cases 38

5.4.1 $p \equiv 3 \pmod 4$ 38

5.4.2 $p \equiv 1 \pmod 4$, $a = -1$ 38

5.5 Wilson's theorem and the value of $\left(\frac{2}{p}\right)$ 38

5.6 Quadratic Reciprocity 39

5.7 The Tonelli-Shanks Algorithm: Exposition from 2022 40

5.7.1 $p \equiv 5 \pmod 8$ 40

5.7.2 The general case 41

5.8 The Tonell-Shanks algorithm: Exposition of 2025 43

5.9 Hensel Lifting: From $\mathbb{Z}_p$ to $\mathbb{Z}_{p^k}$ 44

5.10 A second algorithm for finding square-roots 46

5.11 Exercises 46

6 Finite Fields 47
6.1 Groups 47
6.1.1 Cayley Tables and Isomorphism 48
6.1.2 Direct products and subgroups 49
6.1.3 Cosets and Lagrange's theorem 49
6.2 Rings 50
6.3 Fields 50
6.4 Finite Fields 51
6.5 Irreducible polynomials in $\mathbb{Z}_p[x]$ 51
6.6 Application: Secret sharing 52

7 Polynomial Factorization over $\mathbb{Z}_p$ 53
7.1 Phase 1: Finding the square-free part 53
7.2 Phase 2: Distinct-degree factorization 54
7.3 Phase 3: Finding irreducible factors of degree i 55

8 Polynomial Factorization over $\mathbb{Z}$ 57

II Quadratic Equations in Two Variables

9 Primality Testing: Before 2002 61
9.1 Fermat and Mersenne primes 61
9.1.1 Primes of the form 2^n+1 61
9.1.2 Primes of the form 2^n-1 62
9.2 Testing Fermat's little theorem 62
9.3 Fibonacci and Lucas pseudoprimality tests 63
9.4 The Miller-Rabin Test 64
9.4.1 Analysis of time complexity 65
9.4.2 Analysis of correctness probability 65

10 The Integer Factoring Problem 67
10.1 Trial Division and Fermat's Method 67
10.2 Pollard rho Algorithm 67
10.3 Dixon's Algorithm 69

11 Primality Testing: The AKS algorithm 73
11.1 A Polynomial Identity 73
11.2 The Algorithm 73
11.2.1 Running Time: 74
11.3 Correctness 74
11.3.1 The Proof Strategy 75

11.3.2 The Proof 75

12 Quadratic Forms 77

Preface

The purpose of these notes is to present elementary algorithms in number theory from the point of view of solving polynomial equations - primarily over $\mathbb{Z}$ and over $\mathbb{Z}_p$ (the ring of integers modulo p with p prime). The simplest case, namely that of factorizing polynomials in one variable, already uses non-trivial ideas.

The two-variable case is non-trivial even when the degree is restricted to two. We will study three classical cases of quadratic forms: the Brahmagupta-Pell equation ($x^2 - ny^2 = 1$), representations as sum of two squares, and the equation $xy = n$, which lead to primality testing and integer factoring.

What about the multivariate cases? Linear equations are solvable efficiently, whether over $\mathbb{Z}$ or $\mathbb{Z}_n$. Factoring multivariate polynomials can also be done efficiently, i.e. in (randomized) polynomial time, over $\mathbb{Z}_p$. This requires some understanding of finite fields, so we shall study finite fields as well as one or two applications.

Over integers, the picture is very different. The problem of deciding if an arbitrary polynomial in any number of variables has an integer solution - mentioned by Hilbert among his 23 problems for the twentieth century, was famously shown to be undecidable by Matiyasevich in 1970 following a series of work by Julia Robinson, Martin Davis and Hilary Putnam.

2.2 Euclid's lemma 21
2.3 The fundamental theorem 21
2.4 Notes and Further Reading 22
2.5 Exercises 22

3 Congruences 23
3.1 Definition and properties 23
3.2 Linear congruences and the Chinese Remainder Theorem 24
3.3 Fermat's little theorem 25
3.4 The ring $\mathbb{Z}_n$ 26
3.5 Arithmetic in $\mathbb{Z}_n$ 27
3.6 The set $\mathbb{Z}_n^*$ and Euler's totient function 27
3.7 Order and primitive roots 28
3.8 Exercises 29

4 Polynomials over $\mathbb{Z}_n$ 31
4.1 The ring $\mathbb{Z}_n[x]$ 31
4.2 Lagrange's theorem 31
4.3 Euclid's algorithm and unique factorization for polynomials 31
4.4 The equations $x^d = 1$ and $x^d = a$ in $\mathbb{Z}_p$ 32
4.5 Application: The RSA Algorithm 33
4.6 Exercises 33

5 The Quadratic Equation in $\mathbb{Z}_p$ . . . 35
5.1 Quadratic Residues 35
5.2 Application: Coin Tossing over a telephone 35
5.3 The Legendre Symbol 36
5.4 The equation $x^2 = a$: Two easy cases 38
5.5 Wilson's theorem and the value of $\left(\frac{2}{p}\right)$ 38
5.6 Quadratic Reciprocity 39
5.7 The Tonelli-Shanks Algorithm: Exposition from 2022 40
5.8 The Tonell-Shanks algorithm: Exposition of 2025 43
5.9 Hensel Lifting: From $\mathbb{Z}_p$ to $\mathbb{Z}_{p^k}$ 44
5.10 A second algorithm for finding square-roots . . . 46
5.11 Exercises 46

6 Finite Fields 47
6.1 Groups 47
6.2 Rings 50
6.3 Fields 50
6.4 Finite Fields 51
6.5 Irreducible polynomials in $\mathbb{Z}_p[x]$ 51
6.6 Application: Secret sharing 52

7 Polynomial Factorization over $\mathbb{Z}_p$ 53
7.1 Phase 1: Finding the square-free part 53
7.2 Phase 2: Distinct-degree factorization 54
7.3 Phase 3: Finding irreducible factors of degree i 55

8 Polynomial Factorization over $\mathbb{Z}$. 57

I Polynomials in One Variable

1. Two Equations from Ancient Times

1.1 The Cubic Equation

Solutions to linear and even quadratic equations have been known from a very long time. In 1800 BC, Egyptians solved quadratic equations by the "method of false position", i.e. by finding successively smaller intervals containing a root. From a clay tablet dated between 1800 BC to 1600 BC, we know that Babylonians of the period knew how to solve quadratic equations exactly.

A natural follow-up to the quadratic equation is: what about cubic equations? Solving the simplest cubic equation $x^3 = a$, boils down to finding cube-roots, and finding cube-roots numerically was also known for a long time; for example, Aryabhatta (around AD 500),gave a method for finding both square-roots and cube-roots.

What about general cubic equations? This was first studied by Omar Khayyam (AD 1100), where he gave a geometric solution.

1.1.1 Geometric solution

Consider the intersection of the parabola $y = \dfrac{x^2}{a}$ and the circle $(x-r)^2 + y^2 = r^2$. An intersection point satisfies $\dfrac{x^4}{a^2} + x^2 - 2rx = 0$, i.e. $x^3 + a^2x = 2ra^2$. Thus we get a solution to the cubic equation $x^3 + px = q$ when $p \geq 0$. In Omar Khayyam's time, negative numbers were avoided and thus the same equation above with a negative value for p would instead be written as $x^3 = px + q$ with p positive. For example, to solve the equation $x^3 + x = 12$, we intersect $y = x^2$ with $(x-6)^2 + y^2 = 36$. This is illustrated in Figure 1.1.1 shown below.

Omar Khayyam divided the cubic equation into various categories so that the coefficients would be positive, and gave different geometric solutions for them.

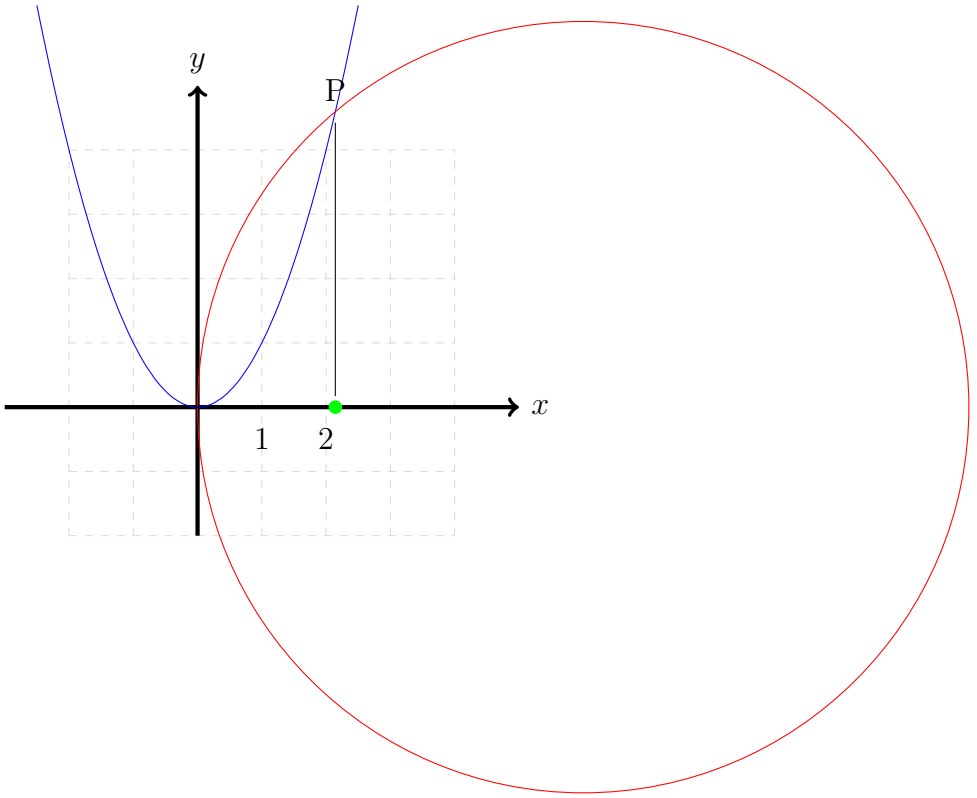

Figure 1.1: Illustration of the geometric solution of $x^3 + x = 12$

1.1.2 Algebraic solution and depressed cubics

While a geometric solution is useful, does the cubic equation have a "closed-form" expression for its solutions in terms of its coefficients, like the quadratic equation does? The answer is Yes and here is a solution. A solution to $x^3 + px + q = 0$ is given by:

$$x = \sqrt[3]{\frac{-q}{2} + \sqrt{D}} + \sqrt[3]{\frac{-q}{2} - \sqrt{D}}. \tag{1.1}$$

where $D = \frac{q^2}{4} + \frac{p^3}{27}$.

But how do we obtain this? And what about the more general cubic with a non-zero x^2 term? To answer the latter question first, it turns out that we can reduce the solution of any cubic polynomial to a solution of a cubic without the x^2 term: such a cubic polynomial is called a depressed cubic.

Consider the polynomial $f(x) = x^3 + ax^2 + bx + c$. Substitute $x = y + r$ to obtain

$$f(x) = y^3 + (3r + a)y^2 + (3r^2 + 2ar + b)y + (r^3 + ar^2 + br + c).$$

If we choose $r = -a/3$, then we obtain a depressed cubic in y, let's call it $g(y)$. Thus to solve $f(x) = 0$, we can solve $g(y) = 0$ and then find the corresponding roots of f.

History

Around AD 1500, Scipione del Ferro, professor at the University of Bologna, discovered a formula for depressed cubic equations (cubic equations with a missing x^2 term) and shortly before his death in 1526, communicated it to his disciple, Antonio del Fiore. After del Ferro's death, around 1535, Fiore issued a challenge to another mathematician Tartaglia, with a list of 30 problems all of whose solutions depended on knowing how to solve the cubic equations $x^3+px=q$ and $x^3=px+q$.
Interestingly, Tartaglia had five years earlier, independently figured out solutions to $x^3+ax^2=b$ and $x^3=ax^2+b$. After accepting the challenge he figured out shortly how to solve the other kind of cubic equation and thus solved all the 30 problems posed by Fiore; he himself posed both kinds of equations in the counter-challenge which Fiore could not solve, and hence won the duel. Here are two of the equations that Tartaglia solved in the duel: $x^3+x=12$ and $x^3+3x=15$.
After the duel, Tartaglia was approached by Gerolamo Cardano, to share his secret as Cardano was writing a book on arithmetic etc. [Incidentally, Cardano was also the first to write a book on probability although he made many mistakes in it.] Initially, Tartaglia refused, but later shared his formula by means of a poem on the promise that Cardano would keep it secret.
Cardano kept his secret for some time, but a few things changed his mind. Firstly, he himself figured out how to reduce the most general equation to a depressed cubic; secondly his student Ferrari figured out how to solve the biquadratic equation, i.e. an equation of degree four. Thirdly, he visited del Ferro's house and examined his manuscripts and was convinced that del Ferro was the original discoverer of the solution to the cubic. He published the solutions in his new book Ars Magna, which led to a fallout between Cardano and Tartaglia.

1.1.3 Solving a depressed cubic

We consider the polynomial $f(x)=x^3+px+q$. Suppose we write $x=\sqrt[3]{u}+\sqrt[3]{v}$. Then

$$x^3=u+v+3\sqrt[3]{uv}x. \tag{1.2}$$

Now we notice that if $u+v=-q$ and $3\sqrt[3]{uv}=-p$, then $x=\sqrt[3]{u}+\sqrt[3]{v}$ satisfies $f(x)=0$. Clearly we can find such a pair u,v as roots of the quadratic polynomial $z^2+qz-\frac{p^3}{27}$. Thus we find $u=\frac{-q}{2}+\sqrt{D}, v=\frac{-q}{2}-\sqrt{D}$, where $D=\frac{q^2}{4}+\frac{p^3}{27}$. The corresponding root is: $x=\sqrt[3]{\frac{-q}{2}+\sqrt{D}}+\sqrt[3]{\frac{-q}{2}-\sqrt{D}}$.

Exercise 1.1 Find a root of the polynomial $x^3 + x^2 - 10$. ∎

1.1.4 Quartic and higher powers: More history

Quartic (fourth-degree) polynomials also have a closed-form expression, found by Ferrari (as mentioned in the history), and naturally this led to the question of a formula for polynomials of degree five and higher. By formula, we mean an expression using the four standard arithmetic operations plus the operation of taking nth roots.

No such formula was however found despite many attempts and around 1800, Ruffini (and later Abel), proved that no general formula can exist. This does not however rule out individual polynomials having roots expressed in terms of radicals; it only rules out the absence of a common formula that works for all polynomials.

Nevertheless Galois in 1830 figured out the exact conditions under which a polynomial has a radical solution; in particular most polynomials of degree five and higher do not have closed form expressions for their roots. The simplest example of such a polynomial is $x^5 - x - 1$. Galois thus solved the problem completely and the theory he built (and further refined by subsequent mathematicians) is called Galois theory.

In a different direction, that every polynomial of degree n has exactly n complex roots (the fundamental theorem of algebra) was proved by d'Alembert, Gauss and Argand around 1800.

1.2 Fast Arithmetic Operations

One of our concerns in this course will be the design of efficient algorithms, often algorithms running in time polynomial in the input size. Thus, we recap the following facts about how efficiently we can do basic arithmetic operations.

1. Addition, subtraction of two n-bit integers can be done in $O(n)$ time.
2. Multiplication, division of two n-bit integers can be done in $O(n \log^n)$ time using the Fast Fourier Transform. The best known algorithm for this fundamental computation has time complexity $O(n \log n)$ and is due to Harvey and van der Hoeven, 2020.
3. Two polynomials of degree d can be added using $O(d)$ arithmetic operations and multiplied in $O(d \log d)$ arithmetic operations. Notice that for the total time complexity, the cost of these operations (in terms of the size of the coefficients) has to be taken into account.
4. Expressions of the form a^n can be computed using $\log n$ multiplications by binary exponentiation.

It is usually convenient to use the notation $O\tilde{(}n^c)$ to denote to functions of the form $O(n^c g(n))$, where $g(n) = o(n)$. Thus, in place of functions such as $n \log n$, $n \log^2 n, n^2 \log^{3/2} n \log \log n$, we may write $O\tilde{(}n)$, $O\tilde{(}n)$, $O\tilde{(}n^2)$ respectively.

1.3 The Equation $ax + by = c$

In contrast to solving equations over the reals ($\mathbb{R}$) or over the complex numbers ($\mathbb{C}$), the focus in number theory is to consider equations over integers ($\mathbb{Z}$) or over rational

numbers ($\mathbb{Q}$).

Such equations are called Diophantine equations in honor of Diophantus (ÃD 250). Diophantus wrote a book called Arithmetica in which he collected over 200 problems and explained their solutions. He was mainly interested in solutions in terms of positive rationals; for many problems he found general parametric solutions. His book famously inspired Fermat to write in its margins. He was also one of the first persons to use symbolic notation (combined with reasoning by words). Here are three problems from the Arithmetica.

1. Find two numbers such that each subtracted from the square of their sum gives a square.
2. Find three numbers such that the product of any two added to the third is a square.
3. Find two numbers such that the first cubed added to the second is a cube, and the second squared added to the first is a square.

Notice that for each of these problems, finding a solution is equivalent to finding a solution to a system of polynomial equations in one or more variables.

In this course, we shall deal with the problem of factorizing/finding solutions to polynomials in one or two variables, over the integers and also over the field of integers modulo a prime. For example, the factorization of the polynomial x^4+4 over integers is $x^4+4=(x^2-2x+2)(x^2+2x+2)$. We can also factorize this polynomial further modulo 5; $x^2-2x+2\equiv(x-3)(x-4)$ modulo 5 and $x^2+2x+2\equiv(x-1)(x-2)$ modulo 5.

1.3.1 Bezout's Lemma

The simplest equations, over $\mathbb{Z}$, are, as in the case of reals, linear equations. Aryabhatta was the first to explain how to solve an equation of the form $ax+by=c$; an example that he used was 137x+10=60y.

First, let's look at an example of an equation with no solutions. Consider the equation $4x+6y=5$. This has no solution in integers because 2 divides the LHS but not the RHS. In general, we see that if $d|a$ and $d|b$, then for $ax+by=c$ to have a solution, d must divide c. In particular, the greatest common divisor of (a,b) must divide c. This is a necessary condition but it also turns out to be sufficient.

Theorem 1.1 — Bezout's Lemma. Let a,b,c be natural numbers. The equation $ax+by=c$ has a solution in integers if and only if $gcd(a,b)$ divides c.

Proof. It is both necessary and sufficient to prove that we can express $gcd(a,b)$ as $ax+by$ for some integers x,y. Let $d=gcd(a,b)$. For a given value of d, we prove by induction on $a+b$ (over pairs (a,b) with $(gcd(a,b)=d)$ that d can be written as $ax+by$. The base case is when $a+b=d$, i.e. $a=d$ and $b=0$. In this clearly $x=1,y=0$ is a solution. Now consider an arbitrary pair (a,b) with $gcd(a,b)=d$ and without loss of generality, let $a\geq b$. Then $gcd(a-b,b)=d$ and by the induction hypothesis, we have: $d=(a-b)x+by$. This implies that $d=ax+b(y-x)$ and thus d is an integer-linear combination of a,b as desired. This completes the proof. ■

We make some remarks: firstly, the proof can be made algorithmic by finding successively smaller pairs (a,b) till we reach the pair $(d,0)$ and work backwards. A simple recursive algorithm is the following:

Algorithm 1 Recursive-Bezout:
Input: Integers a,b with $a \geq b \geq 0$.
Output: A triple (g,x,y) such that $ax+by=g$ and $g=gcd(a,b)$

```
1: procedure SIMPLE-EUCLID((a,b))
2:     if b = 0 then return (a,1,0)
3:     end if
4:     (g,x,y) ← SIMPLE-EUCLID(a−b,b)
5:     Return (g,x,y−x)
6: end procedure
```

Now the second remark: as in the case of Euclid's algorithm, we can make it more efficient by observing that if $a=bq+r$ where $0<r<b$, then the above algorithm will reduce - after q steps, the pair (a,b) to (b,r) which we may as well do in one step.

1.3.2 Euclid's Algorithm

Euclid ($\tilde{3}50$ BC) wrote his algorithm in his famous book The Elements, along with a few other statements in number theory.

Algorithm 2 Euclid's algorithm

```
1: procedure EUCLID(A,B)                          ▷ Returns gcd(A,B)
2:     a ← max(A,B), b ← min(A,B)
3:     while b ≠ 0 do
4:         r ← a mod b                            ▷ gcd(a,b) = gcd(b,r)
5:         a ← b
6:         b ← r
7:     end while
8:     return a
9: end procedure
```

Sample runs of Euclid's algorithm:

- **Example 1.1** gcd(186,75) = gcd(75,36) = gcd(36,3)=gcd(3,0)=3. ∎
- **Example 1.2** gcd(31,20)=gcd(20,11)=gcd(11,9) =gcd(9,2)=gcd(2,1)=gcd(1,0)=1 ∎

Correctness of Euclid's algorithm: We have $gcd(a,b)=gcd(b,a-qb)$ for every integer q; in each iteration of the while loop, the current pair (a,b) is updated to a pair of the form $(b,a-qb)$; thus $gcd(a,b)$ is an invariant across the iterations of the while loop.

Running Time of Euclid's algorithm:

Theorem 1.2 Let $n = max(\log_2 A, \log_2 B)$. Then the number of iterations in Euclid's algorithm is at most $2n$. The total time complexity is at most $\tilde{O(n^2)}$.

Proof. The second statement follows from the first one because the complexity of each iteration is essentially the cost of integer division, which is $\tilde{O(n)}$.

To prove the first statement, let (a_i, b_i) denote the value of the pair (a, b) after i iterations, with $(a_0, b_0) = (max(A, B), min(A, B))$. Further, let $a_i = b_i q_i + r_i$ with $0 < r_i < b_i$. Note that $a_{i+2} = b_{i+1} = r_i < b_i$.

We have $a_i = b_i q_i + a_{i+2} \geq b_i + a_{i+2} > 2a_{i+2}$. The a_is reduce by a factor of at least 2 after every two iterations, so that the number of iterations is at most $2\log_2(max(A, B)) = 2n$. ■

1.3.3 The Extended Euclidean algorithm

We now look at the extended Euclid's algorithm which, in addition to finding $gcd(A, B)$, finds two integers x, y such that $Ax + By = gcd(A, B)$.

The idea is to maintain each of the numbers a_i, b_i after every iteration, as an integer-linear combination of A, B. At the end of the algorithm, the value of a_i equals $gcd(A, B)$ so that we will have an integer-linear combination for it.

That is: we shall maintain a two by two integer-valued matrix $M_i = \begin{pmatrix} x_i & y_i \\ u_i & v_i \end{pmatrix}$ such that

$$\begin{pmatrix} a_i \\ b_i \end{pmatrix} = \begin{pmatrix} x_i & y_i \\ u_i & v_i \end{pmatrix} \cdot \begin{pmatrix} A \\ B \end{pmatrix} \tag{1.3}$$

Initially, we have M_0 as the 2 by 2 identity matrix. To compute the M_is successively, a useful observation is that the pair (a_{i+1}, b_{i+1}) is obtained from (a_i, b_i) by a linear transformation, namely:

$$\begin{pmatrix} 0 & 1 \\ 1 & -q_i \end{pmatrix} \cdot \begin{pmatrix} a_i \\ b_i \end{pmatrix} = \begin{pmatrix} a_{i+1} \\ b_{i+1} \end{pmatrix} \tag{1.4}$$

Suppose that we have maintained Equation 1.3 for iteration i and we wish to compute M_{i+1} to maintain it.

Left-multiplying Equation 1.3 by the elementary matrix $E_i = \begin{pmatrix} 0 & 1 \\ 1 & -q_i \end{pmatrix}$ and using Equation 1.4, we obtain: $\begin{pmatrix} a_{i+1} \\ b_{i+1} \end{pmatrix} = E_i M_i \begin{pmatrix} A \\ B \end{pmatrix}$. Thus, we see that $M_{i+1} = E_i M_i$ and we can use this relation to compute the M_is successively.

The algorithm is described below. It has the same asymptotic time complexity as Euclid's algorithm.

Algorithm 3 Extended Euclid's algorithm
Input: Integers $A \geq B \geq 0$
Output: Integers x, y, g such that $Ax + By = g$ and $g = gcd(A, B)$

1: **procedure** EXTENDED-EUCLID(A, B)
2: $a \leftarrow A$, $b \leftarrow B$
3: $x \leftarrow 1, y \leftarrow 0$ ▷ $Ax + By = a$ will be invariant.
4: $u \leftarrow 0, v \leftarrow 1$ ▷ $Au + Bv = b$ will be invariant.
5: **while** $b \neq 0$ **do**
6: $r \leftarrow a \bmod b$ ▷ $gcd(a, b) = gcd(b, r)$
7: $q \leftarrow \lfloor a/b \rfloor$
8: $a \leftarrow b$
9: $b \leftarrow r$
10: $\begin{pmatrix} x & y \\ u & v \end{pmatrix} \leftarrow \begin{pmatrix} u & v \\ x - qu & y - qv \end{pmatrix}$
11: **end while**
12: **return** (x, y, a)
13: **end procedure**

■ **Example 1.3** An illustration of the algorithm for $A = 12, B = 7$:

a	b	x	y	u	v	q
12	7	1	0	0	1	1
7	5	0	1	1	-1	1
5	2	1	-1	-1	2	2
2	1	-1	2	3	-5	2
1	0	3	-5	-7	12	

Thus, we find that the gcd is 1 and $(3, -5)$ is a solution to $12x + 7y = 1$. ■

1.4 Notes and Further Reading

Efficient numerical methods for finding real and complex roots of univariate polynomials do exist. One approach, for real roots, is to isolate the roots into disjoint intervals using results such as Sturm's theorem or Descartes' rule of signs, and then apply Netwon-Raphson or binary search within the interval. For complex (and real) roots, the Jenkins-Traub algorithm is used in many software for finding roots.

While the extended Euclid's algorithm has time complexity $O\tilde{(}n^2)$, there exist several $O\tilde{(}n)$-time algorithms for computing the gcd of two n-bit numbers. The first of these were by Knuth in 1970 and Schonhage in 1971. The article by Niels Moller in Mathematics of Computation (2008) and a simplified half-gcd algorithm by Daniel Lichtau, should be accessible.

Parallel algorithms for gcd computation have also been studied: Brent and Kung, in 1983, gave a $O(n)$ time algorithm using $O(n)$ processors; Chor and Goldreich gave a $O(n/\log n)$ time algorithm using $O(n^{1+\varepsilon})$ processors; Adleman and Kompella gave an algorithm with a different trade-off, running in poly-log(n) time using a subexponential number of processors.

1.5 Exercises

Exercise 1.2 Find an integer solution to 32x+57y=1. ∎

Exercise 1.3 Find all integer solutions to 4x+7y=1. ∎

Exercise 1.4 Find three integers x, y, z such that $6x + 10y + 15z = 1$. ∎

Exercise 1.5 Find all integers x, y, z such that $2x + 3y + 5z = 0$. ∎

Exercise 1.6 (*) Prove that if $d|a$ and $d|b$, then $d|gcd(a,b)$. ∎

Exercise 1.7 The *binary gcd* algorithm also has a similar complexity as Euclid's algorithm and works as follows:

In the first step, divide a, b by the largest power of 2 that divides both of them. When at least one of them is odd, define a recursive $binaryGCD$ function as follows: if one of them is zero, return the other number; if a is even and b is odd, call $binaryGCD(a/2, b)$; if a is odd and b is even, call $binaryGCD(a, b/2)$.

If both a, b are odd, with which parameters should you make the recursive call? Also find an upper bound on the number of iterations. ∎

Exercise 1.8 (+) The goal of this exercise is to find a method to show the existence of a radical formula for roots of quartic polynomials.

Let $f(x) = x^4 + ax^3 + bx^2 + cx + d$. Since complex roots appear in conjugate pairs, we can assume a factorization of $f(x)$ as $f(x) = (x^2 + \alpha x + \beta)(x^2 + \gamma x + \delta)$. Show that it is also sufficient to consider polynomials with the coefficient of x^3 being zero. Using the above factorization into two quadratics, show that finding one of the unknowns (say α) can be reduced to that of solving a cubic equation. ∎

Exercise 1.9 (+) Euclid's algorithm can also be applied for an arbitrary pair of positive real numbers (a, b) with $a > b$ by setting the quotient as $q = \lfloor \frac{a}{b} \rfloor$.

For example, the starting with the pair $(3.35, 2.35)$, we obtain the following pairs:

$(2.35, 1), (1, 0.35), (0.35, 0.3), (0.3, 0.05), (0.05, 0)$, with the sequence of quotients being $1, 2, 2, 1, 6$.

(a) Show that if α is a rational number larger than 1, then the algorithm terminates after a finite number of steps.

(b) By writing a program, compute the first 20 quotients for the pairs $(\sqrt{2}, 1)$ and $(e, 1)$, where $e = 2.71828...$ is Euler's number/the base of the natural logarithm. Do you observe a pattern? Can you prove it for the first pair? ∎

2. The Fundamental Theorem of Arithmetic

2.1 Prime numbers

A natural number n is defined to be a prime number if it has exactly 2 divisors (namely 1 and n). The sequence of prime numbers begins $2, 3, 5, 7, \ldots$. The first interesting fact about prime number is that there are infinitely many of them.

Theorem 2.1 There are infinitely many prime numbers.

Proof. Suppose for contradiction that there are only finitely many primes, say $p_1, p_2, \ldots, p_k$. Consider $N = p_1 p_2 \ldots p_k + 1$. The rest of the proof is left as an exercise. ■

2.2 Euclid's lemma

Lemma 2.1 If p is prime and $p|ab$, then $p|a$ or $p|b$.

Proof. Suppose for contradiction that p does not divide a and p does not divide b. Then $gcd(p,a) = 1$ and by Bezout's lemma, there exist integers x, y such that $px + ay = 1$. Similarly, $gcd(p,b) = 1$ and there exist integers u, v such that $pu + bv = 1$. Multiplying the two relations, we get: $(px + ay)(pu + bv) = 1$. Expanding the LHS, we get a contradiction because p divides each term on the LHS, but the RHS is equal to 1. This proves the lemma. ■

2.3 The fundamental theorem

Theorem 2.2 Every natural number $n > 1$ can be uniquely factored into a product of prime numbers, i.e. we can write $n = p_1 p_2 \ldots p_k$, where the p_is are prime (not necessarily distinct), and if $n = q_1 q_2 \ldots q_l$ with each q_i prime, then $k = l$ and $\{q_1, q_2, \ldots, q_l\}$ is equal (as a multiset) to $\{p_1, p_2, \ldots, p_k\}$.

Proof. There are two statements to prove, (a) that every natural number larger than 1 can be factored into primes and (b) that the factorization is unique (up to ordering).

We first prove (a) by induction. The first few base cases are $2,3,4$ which we see have a prime factorization. The induction step: We consider an arbitrary natural number $n>1$ and inductively assume that $1,2,\ldots,n-1$ have a prime factorization. If n is prime, then we are done, otherwise let $n=ab$ with $1<a,b<n$. By the induction hypothesis, a,b have a prime factorization. The two factorizations may be combined to give a factorization for n. This proves (a).

We now prove (b), also by induction. As before, we may verify that (b) holds for the base cases $2,3,4$. For the induction step, we consider an arbitrary $n>1$, assuming that (b) holds for all numbers lesser than n. Suppose that $n=p_1p_2\ldots p_k=q_1q_2\ldots q_k$. We apply Euclid's lemma to the product $q_1q_2\ldots q_k$ to deduce that p_1 divides some q_i. Since q_i is prime, we have $p_1=q_i$. Now we apply the induction hypothesis to $n/p_1=p_2\ldots p_k=\prod_{j\neq i}q_j$ and conclude that $\{p_2,\ldots,p_k\}=\{q_j|j\neq i\}$ (and $k-1=l-1$). Together with $p_q=q_i$, this gives $\{p_1,\ldots,p_k\}=\{q_1,\ldots,q_l\}$, completing the proof of (b). ■

As a consequence of the fundamental theorem, we can write every natural number $n>1$ as $p_1^{e_1}\ldots p_k^{e_k}$ with distinct primes p_i and e_i being a non-negative integer.

2.4 Notes and Further Reading

Historically, unique factorization turned out to be crucial in the development of algebraic number theory. For example, the set $\{a+bi:a,b\in\mathbb{Z}\}$ admits unique factorization, whereas the set $\{a+b\sqrt{-5}:a,b\in\mathbb{Z}\}$ does not: $6=(1+\sqrt{-5})(1-\sqrt{-5})$.

2.5 Exercises

Exercise 2.1 Prove that there are infinitely many primes of the form $4k+3$. ■

Exercise 2.2 Show that if $gcd(a,b)=1$, then $gcd(ab,c)=gcd(a,c)gcd(b,c)$. ■

Exercise 2.3 Show that if $gcd(a,b)=1$ and $a|bc$, then $a|c$. ■

Exercise 2.4 Suppose that $n=p_1^{e_1}\ldots p_k^{e_k}$ is the factorization of n, where the p_is are distinct primes. How many divisors does n have? ■

Exercise 2.5 (a) Show that if $n>1$ is not prime, then n has a prime factor p such that $p\leq\sqrt{n}$.

(b) Show that n (in $\mathbb{N}$) has at most $\log_2 n$ distinct prime factors.

(c) Describe an $O(\tilde{\sqrt{n}})$ time algorithm to find factorize a given natural number n, i.e. to find the prime factors p_i along with their exponents e_i such that $\prod_i p_i^{e_i}=n$. ■

3. Congruences

3.1 Definition and properties

Let $a, b \in \mathbb{Z}$ and $n > 1$ be a natural number. We say that $a \equiv b \pmod{n}$ (read as a is *congruent to* b modulo n) if $a - b$ is divisible by n.

Examples: $17 \equiv 3 \pmod{7}$, $-20 \equiv -8 \pmod{3}$, $360 \equiv 0 \pmod{60}$.

Properties satisfied by congruences:

1. If $a \equiv b \pmod{n}$ and $c \equiv d \pmod{n}$, then $a + c \equiv b + d \pmod{n}$.
2. If $a \equiv b \pmod{n}$ and $c \equiv d \pmod{n}$, then $ac = bd \pmod{n}$.
3. Note that division doesn't work the same way as reals: in general, $ac \equiv bc \pmod{n}$ does not imply that $a \equiv b \pmod{n}$ OR $c \equiv 0 \pmod{n}$. A counter-example is $n = 6, a = 4, b = 2, c = 3$.
   To understand what we can deduce from $ac \equiv bc \pmod{n}$, we rewrite it as $(a - b)c \equiv 0 \pmod{n}$. Now we see that in some cases, we can draw some conclusions. For example, if $gcd(c, n) = 1$, then we can conclude that $a \equiv b \pmod{n}$. Also, if n is prime, then we can conclude that n divides one of $(a - b), c$, i.e. $a \equiv b \pmod{n}$ or $c \equiv 0 \pmod{n}$.
4. The congruence relation is an *equivalence relation*, i.e. it satisfies the following properties:
   (i) Reflexivity: $a \equiv a \pmod{n}$;
   (ii) Symmetry: $a \equiv b \pmod{n}$ implies that $b \equiv a \pmod{n}$;
   (iii) Transitivity: $a \equiv b \pmod{n}$ and $b \equiv c \pmod{n}$ implies that $a \equiv c \pmod{n}$.
   An equivalence relation partitions the set into *equivalence classes*, in this case *congruence classes*. The congruence classes for a given n are $\{0, \pm n, \pm -2n, \dots, \}, \{1, 1 \pm n, 1 \pm 2n, \dots, \}, \dots, \{n - 1, n - 1 \pm n, n - 1 \pm 2n, \dots\}$.
   We consider the numbers $\{0, 1, \dots, n - 1\}$ to be the canonical representatives of these congruence classes, as these numbers are the remainders when divided by n.

3.2 Linear congruences and the Chinese Remainder Theorem

Consider the following linear congruence in one variable:

$$ax \equiv b(\text{mod n}). \tag{3.1}$$

We can rewrite the above equation as $ax = b + ny$; thus we see that this equation has a solution if and only if $gcd(a,n)$ divides b. It is usually convenient to divide this equation on both sides by $gcd(a,n)$; translating this back to congruences, this means considering Equation 3.1 when $(a,n) = 1$. In this case, it turns out that the solution is unique.

Theorem 3.1 Given $a \in \{1,\ldots,n-1\}$ such that $gcd(a,n) = 1$, the congruence $ax \equiv b$ (mod n) has a unique solution modulo n.

Proof. Suppose that x_1, x_2 are two solutions to the given congruence equation. Then $ax_1 \equiv b \pmod{n}$ and $ax_2 \equiv b \pmod{n}$; subtracting we get:

$$a(x_1 - x_2) \equiv 0(\text{ mod n}).$$

Since $gcd(a,n) = 1$, this implies that n divides $x_1 - x_2$, i.e. $x_1 \equiv x_2 \pmod{n}$. ■

Without the assumption that $gcd(a,n) = 1$, how many solutions does the linear congruence 3.1 have? You will figure this out in an exercise below!

Exercise 3.1 Find all the solutions in $\{0,1,\ldots,24\}$ to $10x \equiv 15 \pmod{25}$. ■

Exercise 3.2 Suppose that $gcd(a,n)$ divides b. How many distinct solutions to $ax \equiv b \pmod{n}$, modulo n, are there? Justify your answer. [Hint: You may be able to first guess the answer from what you find in the previous exercise.] ■

The second type of linear congruence that we look at is simultaneous congruences, the most general problem being the following: find all $x \in \mathbb{Z}$ such that $x \equiv a_1 \pmod{n_1}$, $x \equiv a_2 \pmod{n_2}$, $\ldots$, $x \equiv a_2 \pmod{n_k}$. That is, we want the integers x that simultaneously satisfy all the k congruences.

We first look at the simplest case, i.e. $k = 2$. Consider the following pair of congruences:

$$x \equiv a(\text{ mod n}), x \equiv b(\text{ mod m}). \tag{3.2}$$

We have $x = a + ny = b + mz$; thus we obtain the equation $ny - mz = b - a$. This equation has a solution for y, z in integers if and only if $gcd(n,m)$ divides $(b-a)$. As before, we simplify further and first consider the case that $gcd(n,m) = 1$. In this case, 3.2 has a solution, and further, this solution is unique modulo mn.

Theorem 3.2 Let $gcd(m,n)=1$. Then the map $f:\{0,1,\ldots,mn-1\}\to\{0,1,\ldots,m-1\}\times\{0,1,\ldots,n-1\}$, given by $f(x)=(x_1,x_2)$ where $x\equiv x_1 \pmod m$ and $x\equiv x_2 \pmod n$, is a bijection.

Proof. The fact that this map is surjective follows from the preceding paragraph: given (x_1,x_2), we may use the extended Euclidean algorithm to find y,z such that $ny-mz=x_2-x_1$; then $x_1+ny=x_2+mz$ is a pre-image of (x_1,x_2).

From the fact that this map is surjective and the fact that the domain and co-domain are finite sets of the same size, we may already conclude that f is a bijection.

Nevertheless, we may also prove that f is injective, as follows. Let x,y be two different numbers in $\{0,1,\ldots,mn-1\}$. Then $x-y\in\{1,\ldots,mn-1\}$, hence $x-y$ is not divisible by mn. Since $gcd(m,n)=1$, we can conclude: $x-y$ is not divisible by m OR $x-y$ is not divisible by n. In either case we have $f(x)\neq f(y)$, proving that f is an injective map. ■

Example: Consider $m=3,n=4$. The values of $f(0),f(1),\ldots,f(11)$ are, in order: $(0,0),(1,1),(2,2),(0,3),(1,0),(2,1),(0,2),(1,3),(2,0),(0,1),(1,2),(2,3)$.

The above result extends naturally to multiple moduli.

Theorem 3.3 — Chinese Remainder Theorem. Suppose that $n_1,\ldots,n_k$ are natural numbers such that $gcd(n_i,n_j)=1$ for every $i\neq j$. Then the congruences $x\equiv a_1 \pmod{n_1}$, $\ldots$, $x\equiv a_k \pmod{n_k}$ has a unique solution modulo $n_1\ldots n_k$. Equivalently, the map $f:\{0,1,\ldots,n_1\ldots n_k-1\}\to\{0,1,\ldots,n_1-1\}\times\ldots\times\{0,1,\ldots,n_k-1\}$ given by $f(x)=(x_1,\ldots,x_k)$ with $x\equiv x_i \pmod{n_i}$ for all i, is a bijection.

Further, we can find this unique value of x in polynomial-time.

Proof Sketch: The first part follows by a repeated application of Theorem 3.2. For the second part, we may use the Extended Euclidean algorithm to successively solve pairs of congruences. ■

Exercise 3.3 Find the least positive integer x such that $x\equiv 1 \pmod 3$, $x\equiv 4 \pmod{13}$ and $x\equiv 7 \pmod{23}$. ■

Exercise 3.4 This problem is attributed to Brahmagupta (7th century). A woman has a basket of eggs. When she takes them out two at a time, one is left over. The same thing happens when she takes them out in groups of three or four or five or six: one egg is left over in the basket. But if she takes them out seven at a time, none are left over. What is the smallest number of eggs she could have had in the basket? ■

3.3 Fermat's little theorem

Theorem 3.4 Let p be a prime. Then, for every natural number a, we have:

$$a^p \equiv a(\bmod p). \tag{3.3}$$

Equivalently, we may say: for every natural a such that p does not divide a, we have:

$$a^{p-1} \equiv 1 (\bmod p). \tag{3.4}$$

Proof. We give two proofs.

Proof by mathematical induction on a: We prove Eqn 3.3 for every natural number a. The base case is $a = 1$; for the induction step, write $(a+1)^p = a^p + \sum_{i=1}^{p-1} \binom{p}{i} a^{p-i} + 1$; observe that every term in the summation is divisible by p and complete using the induction hypothesis.

Proof by a bijection: We assume that p does not divide a and observe that $\{a, 2a, \ldots, (p-1)a\}$ must all have distinct values modulo p (to see this, consider their differences). This implies that the set of these values modulo p must be the same (possibly in different order) as $\{1, 2, \ldots, p-1\}$. Now we note that $a \times 2a \ldots \times (p-1)a \equiv (p-1)!$ (modulo p). We may divide by $(p-1)!$ on both sides; this is possible because p does not divide $(p-1)!$; this gives us the desired result. ■

Exercise 3.5 Find the remainder when 3^{1000} is divided by 23. ■

Exercise 3.6 (a) Show that the congruence $x^{41} \equiv 2 \pmod{83}$ has no solutions.

(b) Show that the congruence $x^2 \equiv 2 \pmod{83}$ has no solutions. ■

Exercise 3.7 Find a solution to the congruence $x^{17} \equiv 2 \pmod{43}$. ■

3.4 The ring $\mathbb{Z}_n$

We define $\mathbb{Z}_n$ to be the set $\{0, 1, \ldots, n-1\}$ together with operations $(, +, \times)$, where $+$ is defined as $a + b = c$ if $a + b \equiv c$ (modulo n) and $a \times b = c$ if $ab \equiv c$ (modulo n).

When we are working with the elements of $\mathbb{Z}_n$, the congruence relation becomes an equality; thus, in $\mathbb{Z}_7$, we have: $5 + 4 = 2$ and $5 \times 4 = 6$.

3.4.1 Chinese remainder theorem as an isomorphism

In this language, Theorem 3.2 actually encodes a stronger fact: if $gcd(m, n) = 1$, then $\mathbb{Z}_{mn} \cong \mathbb{Z}_m \times \mathbb{Z}_n$. This is read as the two sets being *isomorphic*; this means that there is a bijective map f between the two sets, and further it satisfies: $f(x+y) = f(x) + f(y)$ and $f(xy) = f(x)f(y)$.

3.4.2 What's a ring?

For a set R with two binary operations $(+, \times)$, we call $(R, +, \times)$ a ring if the following properties are satisfied:

(a)[**Closures**] For every $a, b \in R$, we have: $a + b \in R$ and $ab \in R$;

(b)[**Identities**] We have distinct elements $0,1 \in R$ such that $a+0=0+a=a$ and $a \times 1 = 1 \times a = a$ for every $a \in R$;

(c)[**Associativity**] For all $a,b,c \in R$, we have: $a+(b+c)=(a+b)+c$ and $a \times (b \times c) = (a \times b) \times c$.

(d)[**Distributivity**] For all $a,b,c \in R$, we have: $a(b+c)=ab+ac$ and $(a+b)c=ac+bc$.

(e)**Addition is commutative** For all $a,b \in R$, we have: $a+b=b+a$.

(f)**Additive inverses** For all $a \in R$, there is an element b such that $a+b=0$; we may just write $-a$ for this element.

If multiplication is also commutative, then we call R a commutative ring. Some other examples of rings are: the set $\mathbb{R}[x]$ of polynomials with real coefficients, the set of 2 by 2 matrices with integer entries, this being an example of a non-commutative ring.

Exercise 3.8 Let p be prime and $a \in \mathbb{Z}_p, a \neq 0$. Find the number of solutions of $ax = 1$. ∎

Exercise 3.9 Let p be prime. Find the number of solutions of $xy = 1$ in $\mathbb{Z}_p$. ∎

Exercise 3.10 Let p be prime and a,b be non-zero elements of $\mathbb{Z}_p$. Find the number of solutions of $ax+by=1$. ∎

3.5 Arithmetic in $\mathbb{Z}_n$

The basic arithmetic operations of addition and subtraction have time complexity $O(\log n)$; the complexity of multiplication is $O(\log n \log\log n)$; the complexity of finding $a^b \pmod n$ (by repeated squaring) is $O(\log b \log n \log\log n) = O(\log^2 n \log\log n)$.

An example of exponentiation by repeated squaring: we find 3^{100} modulo 35 as follows: we first find the values of 3^k modulo 35 for k being a power of two.

k	1	2	4	8	16	32	64
$3^k \pmod{35}$	3	9	11	16	11	16	11

We now find:

$$\begin{aligned} 3^{100} &\equiv 3^{64} \cdot 3^{32} \cdot 3^4 (\bmod 35) \\ &\equiv 11 \times 16 \times 11 (\bmod 35) \\ &\equiv 11 (\bmod 35) \end{aligned}$$

What about division? A reasonable definition of b/a modulo n is to define it as the integer $c \in \{0,1,\ldots,n-1\}$ such that $b \equiv ac \pmod n$. Ideally, we would like the solution to be unique: this is possible if $gcd(a,n)=1$ (see next section).

3.6 The set $\mathbb{Z}_n^*$ and Euler's totient function

We define $\mathbb{Z}_n^* = \{a \in \mathbb{Z}_n | gcd(a,n) = 1\}$. This set is important because it consists of exactly those elements of $\mathbb{Z}_n$ with a multiplicative inverse. This is in turn crucial

when we wish to solve an equation and we wish to divide on both sides by a common element - this is possible as long as that element belongs to $\mathbb{Z}_n^*$.

For $a \in \mathbb{Z}_n^*$, we define a^{-1} as the unique element $b \in \mathbb{Z}_n$ such that $ab = 1$. Fractions such aas $\frac{a}{b}$ are also well-defined if the denominator b is an element of $\mathbb{Z}_n^*$.

Another important fact is that the Chinese remaindering bijection preserves invertibility. That is: if $gcd(m,n) = 1$, then the natural isomorphism from $\mathbb{Z}_{mn}$ to $\mathbb{Z}_m \times \mathbb{Z}_n$ is also a bijection from $\mathbb{Z}_{mn}^*$ to $\mathbb{Z}_m^* \times \mathbb{Z}_n^*$.

A natural question is how many elements are in $\mathbb{Z}_n^*$. Euler's totient function defines this quantity: $\phi(n) = |\mathbb{Z}_n^*|$. As observed earlier, if $gcd(m,n) = 1$, then there's a bijection between $\mathbb{Z}_{mn}^*$ and $\mathbb{Z}_m^* \times \mathbb{Z}_n^*$, so that we have $\phi(mn) = \phi(m)\phi(n)$ in this case.

In particular, if $n = p_1^{e_1} \dots p_k^{e_k}$ where the p_is are distinct primes, then we have: $\phi(n) = \prod_i \phi(p_i^{e_i})$.

If p is prime, then we can directly compute the totient function for a power of p. We have: $\phi(p^k) = p^k - p^{k-1} = p^k(1 - \frac{1}{p})$. Thus, we obtain:

Theorem 3.5 If $n = p_1^{e_1} \dots p_k^{e_k}$, then

$$\phi(n) = n\left(1 - \frac{1}{p_1}\right) \dots \left(1 - \frac{1}{p_k}\right).$$

We now present a generalization of Fermat's little theorem.

Theorem 3.6 [Euler's Theorem] If $gcd(a,n) = 1$, then $a^{\phi(n)} \equiv 1 \pmod{n}$. Equivalently, if $a \in \mathbb{Z}_n^*$, then $a^{\phi(n)} = 1$ in $\mathbb{Z}_n$.

Proof. We generalize the second proof of Fermat's little theorem. Consider the set $\mathbb{Z}_n^*$. Firstly, we claim that if $a \in \mathbb{Z}_n^*$, then the sets $a\mathbb{Z}_n^*$ and $\mathbb{Z}_n^*$ are equal, as subsets of $\mathbb{Z}_n$. To see this observe that if $b \in \mathbb{Z}_n^*$, then $ab \in \mathbb{Z})_n^*$ as well. Thus, $a\mathbb{Z}_n^* \subseteq \mathbb{Z}_n^*$. Also, if b_1, b_2 are two distinct elements of $\mathbb{Z}_n^*$, then $ab_1 \neq ab_2$, because $b_1 - b_2 \not\equiv 0 \pmod{n}$ and $gcd(a,n) = 1$.

Let $P = \prod_{x \in \mathbb{Z}_n^*} x$, $Q = \prod_{x \in a\mathbb{Z}_n^*} x$. We have: $P = Q$ and $Q = a^{\phi(n)}P$ (in $\mathbb{Z}_n$). Since $P \in \mathbb{Z}_n^*$, we may cancel P on both sides from the equation $a^{\phi(n)}P = P$ to obtain the theorem statement. ■

Exercise 3.11 Find $\phi(360)$. ▪

Exercise 3.12 Find the remainder when 7^{1000} is divided by 360. ▪

3.7 Order and primitive roots

Definition 3.1 We say that an element $a \in \mathbb{Z}_n^*$ has order d if $a^d \equiv 1 \pmod{n}$ and $a^k \not\equiv 1 \pmod{n}$ for every positive integer $k < d$. We denote this by $ord_n(a)$.

This definition is well-defined because if $a \in \mathbb{Z}_n^*$, then $a^{\phi(n)} \equiv 1 \pmod{n}$, and thus there must exist a smallest positive ineger d such that $a^d \equiv 1 \pmod{n}$.

As an example, we compute the orders of every element in $\mathbb{Z}_7^*$.

a	1	2	3	4	5	6
$ord_7(a)$	1	3	6	3	6	2

Observation 1 For every $a \in \mathbb{Z}_n^*$, we have: $ord_n(a)$ divides $\phi(n)$.

Proof. We have $a^{\phi(n)} \equiv 1 \pmod{n}$. Let $d = ord_n(a)$ and suppose for contradiction that d does not divide $\phi(n)$; then we can write $\phi(n) = qd + r$ where $0 < r < d$. Since $a^d \equiv 1 \pmod{n}$, we obtain $a^r \equiv 1 \pmod{n}$, which is a contradiction to the definition of d as the order. This proves the observation. ■

In particular, when p is prime and $a \in \mathbb{Z}_{p,}^*$ we have $ord_p(a)$ divides $(p-1)$.

Definition 3.2 We call an element $a \in \mathbb{Z}_n^*$ *a primitive kth-root of unity* if $ord_n(a) = k$. We call an element $a \in \mathbb{Z}_n^*$ **a primitive root** if $ord_n(a) = \phi(n)$.

One reason that primitive roots are interesting is that their powers generate $\mathbb{Z}_n^*$, that is if g is a primitive root, then $\{g, g^2, \ldots, g^{\phi(n)}\} = \mathbb{Z}_n^*$. To see this, note that if $g^i \equiv g^j \pmod{n}$, then $g^{i-j} \equiv 1 \pmod{n}$; this is not possible if i, j are distinct and less than $\phi(n)$ (because $ord_n(g) = \phi(n)$).

Primitive roots exist only for some moduli; the relevant fact for us is that they exist when n is prime.

Theorem 3.7 Let p be a prime. Then there is an element $g \in \mathbb{Z}_p^*$ such that $ord_p(g) = p - 1$.

3.8 Exercises

4. Polynomials over $\mathbb{Z}_n$

4.1 The ring $\mathbb{Z}_n[x]$

4.2 Lagrange's theorem

Theorem 4.1 Let p be a prime. Then a polynomial $f(x)$ in $\mathbb{Z}_p[x]$ has at most $deg(f)$ roots.

Proof. We prove this by induction on the degree of f, with linear polynomials being the case case. We already noted that $ax-b$ has a unique root in $\mathbb{Z}_n$ when $gcd(a,n)=1$. Thus, if $a \neq 0$ and $a \in \mathbb{Z}_p$, then the polynomial $(ax-b)$ has exactly one root.

Now assume that $f(x)$ is a polynomial of degree $d>1$ and assume inductively that the statement of the theorem is true for all polynomials o degree less than d.

Let x_0 be one root of $f(x)$. By the remainder theorem for polynomials, we have: $f(x)=(x-x_0)g(x)$. Now if x_1 is a root of f, then it must be the case that $x_1=x_0$ or $g(x_0)=0$. This is because in $\mathbb{Z}_p$, if $ab=0$, we must have $a=0$ or $b=0$.

Thus the number of roots of f is at most one plus the number of roots of g, and using the induction hypothesis, this value is at most $1+(d-1)=d$.

Remark: Since x_0 can be a multiple root of f, it may be cleaner/more rigorous to first write $f(x)=(x-x_0)^k h(x)$ for the largest value of k possible and use the induction hypothesis on $h(x)$. ■

4.3 Euclid's algorithm and unique factorization for polynomials

Let p be prime. Given two polynomials $f(x), g(x) \in \mathbb{Z}_p[x]$, we may write

$$g(x)=f(x)q(x)+r(x) \tag{4.1}$$

with $deg(r(x)) < deg(f(x))$. We remark that 4.1 does not always hold over $\mathbb{Z}_n$ for n composite, for example, if $n = 4$ and $f(x) = 2x, g(x) = x$.

The Extended Euclidean algorirthm thus works for polynomials in the same way as for integers; the running time is linear in the degree of the polynomials and polynomial in the size of the coefficients. In particular, it finds the gcd of two polynomials, which we define below.

Definition 4.1 Given two polynomials $f(x), g(x) \in \mathbb{Z}_p[x]$, we define the greatest common divisor (gcd) of $f(x)$ and $g(x)$ as the unique monic polynomial of largest degree that divides $f(x)$ and $g(x)$.

Why is the monic polynomial of largest degree that divides both polynomials unique? To see this, we may argue that if $h(x)$ is one such polynomial and $t(x)$ some polynomial that also divides both $f(x)$ and $g(x)$, then $t(x)$ must divide $h(x)$. From this, the desired conclusion may be drawn.

As a consequence of the Euclidean algorithm, we also deduce Bezout's lemma for polynomials.

Theorem 4.2 — Bezout's Lemma. Let p be prime and $f(x), g(x)$ be polynomials in $\mathbb{Z}_p[x]$. Then there exist polynomials $u(x), v(x)$ such that $f(x)u(x) + g(x)v(x) = gcd(f(x), g(x))$. Further, the Euclidean algorithm finds $u(x), v(x)$ such that $deg(u(x)) < deg(g(x))$ and $deg(v(x)) < deg(f(x))$.

The final analog of natural numbers for polynomials is unique factorization. We say that a polynomial $f(x) \in \mathbb{Z}_p[x]$ is irreducible if $f(x) = g(x)h(x)$ implies $g(x) = 1$ or $h(x) = 1$.

Theorem 4.3 Every polynomial $f(x) \in \mathbb{Z}_p[x]$ can be uniquely factorized into irreducible polynomials.

An illustration of Euclid's algorithm for $f(x) = x^3 + x, g(x) = 5x^2 - 13x + 6$ over $\mathbb{Z}_{17}$:

A	B	r	q	u	v	s	t
$x^3 + x$	$5x^2 - 13x + 6$	$-5x - 14$	$7x + 8$	1	0	0	1
$5x^2 - 13x + 6$	$-5x - 14$	0	$-x + 2$	0	1	1	$-7x - 8$
$-5x - 14$	0			1	$-7x - 8$	$x - 2$	$-7x^2 + 5x$

Thus, the gcd of $x^3 + x$ and $5x^2 - 13x + 6$ in $\mathbb{Z}_{17}[x]$ is $(-5x - 14)/-5 = (x - 4)$. We further have: $(x^3 + x)u(x) + (5x^2 - 13x + 6)v(x) = -5x - 14$, where $u(x) = 1$ and $v(x) = (-7x - 8)$.

4.4 The equations $x^d = 1$ and $x^d = a$ in $\mathbb{Z}_p$

In this section, we assume that p is prime and state all results for $\mathbb{Z}_p$. We consider solutions of the equation $x^d = a$, which rest on the consideration of two "opposite" cases: $d|(p-1)$ and $gcd(d, p-1) = 1$.

Theorem 4.4 Let $d|(p-1)$. Then the polynomial $x^d - 1$ has d distinct roots.

Proof. We know that $x^{p-1}-1$ has $p-1$ distinct roots, namely $1,2,\ldots,p-1$. Further, $x^{p-1}-1$ is divisible by x^d-1 because $d|(p-1)$. Thus, we can write $x^{p-1}-1 = (x^d-1)g(x)$ for a polynomial $g(x)$ of degree $p-1-d$. If (x^d-1) has less than d roots, then the number of roots of the RHS would be at most $d-1+deg(g) < p-1$ (because of Lagrange's theorem applied to $g(x)$), which is a contradiction. This completes the proof. ■

We remark that if $f(x)$ is a polynomial with $deg(f)$ roots, then for every divisor $g(x)$ of $f(x)$, the polynomial $g(x)$ must have $deg(g))$ roots.

Theorem 4.5 Let $gcd(d,p-1)=1$ and $a \in \mathbb{Z}_p$. Then the polynomial x^d-a has a unique root, which we can find efficiently.

Proof. Since $gcd(d,p-1)=1$, we can find (efficiently) a positive integer k such that:

$$dk \equiv 1 \bmod (p-1).$$

Raising both sides of $x^d = a$ to the power k, we obtain $x^{dk} = a^k$ and applying Fermat's little theorem, we get $x = a^k$. ■

Combining the previous ideas, we can obtain the following corollary to Theorem 4.4.

Corollary 4.1 In $\mathbb{Z}_p$, the number of roots of x^d-1 is equal to $gcd(d,p-1)$.

The above result is obtained by finding k such that $dk \equiv gcd(d,p-1) \pmod{(p-1)}$ and noting that $x^d = 1$ raised to the power of k gives: $x^{gcd(d,p-1)} = 1$, which has $gcd(d,p-1)$ solutions.

Exercise 4.1 Show that the number of roots of x^d-a in $\mathbb{Z}_p$ is either zero or equal to $gcd(d,p-1)$. ■

Theorem 4.5 also generalizes as follows, with the theorem statement itself containing the explanation.

Theorem 4.6 Let $n \in \mathbb{N}$. If $gcd(d,\phi(n)) = 1$ and $gcd(a,n) = 1$, then the equation $x^d = a$ has a unique solution in $\mathbb{Z}_n$, given by $x = a^k$, where $dk \equiv 1 \pmod{\phi(n)}$.

4.5 Application: The RSA Algorithm

The idea behind Theorem 4.6 is used in the RSA algorithm. The original RSA paper is very readable; the link is here: Link to original RSA paper.

Sections 3,4 of the above paper can be skipped; after Section 2, just go to Section 5.

4.6 Exercises

Exercise 4.2 Find an upper bound for the worst-case time complexity of the Euclidean algorithm for the gcd of two polynomials $f(x), g(x) \in \mathbb{Z}_p[x]$ assuming that their degrees are at most d. Express your bound in terms of d, p. ■

Exercise 4.3 For each of the equations below, over $\mathbb{Z}_{67}$, decide whether it has a solution, and find a solution if it has.

(i) $x^5 = 3$

(ii) $x^3 = 2$.

(iii) $x^2 = 3$

(iv) $x^2 = 17$

■

Exercise 4.4 Suppose that you are given as input a prime p, and polynomials $f(x), g(x), q(x), r(x) \in \mathbb{Z}_p[x]$ such that $p(x), q(x)$ are irreducible.

Show that there is a polynomial $h(x) \in \mathbb{Z}_p[x]$ such that $h(x) \equiv f(x) \pmod{q(x)}$ and $h(x) \equiv g(x) \pmod{r(x)}$ and describe an efficient algorithm to find such a polynomial. ■

Exercise 4.5 (a) Show that the polynomials x^2+1 and x^2-2 are both irreducible in $\mathbb{Z}_{11}$.

(b) In $\mathbb{Z}_{11}[x]$: Find a polynomial $f(x)$ such that $f(x) \equiv x+2 \bmod (x^2+1)$ and $f(x) \equiv 2x-3 \bmod (x^2-2)$. ■

5. The Quadratic Equation in $\mathbb{Z}_p$

5.1 Quadratic Residues

The goal of this chapter is to solve the equation $x^2 = a$ in $\mathbb{Z}_p$, where p is prime. The first question is whether, for a given a, it is solvable at all. To this end, we show the following for the more general question of when a given element in $\mathbb{Z}_p$ is a dth power.

Theorem 5.1 Let $d|(p-1)$ and let $a \in \mathbb{Z}_p$. Then $x^d = a$ is solvable if and only if $a^{(p-1)/d} = 1$.

For arbitrary d, i.e. without the assumption that $d|(p-1)$, we can still answer the same question.

Corollary 5.1 Let d be a positive integer, and let $a \in \mathbb{Z}_p$. Then $x^d = a$ is solvable if and only if $a^{k(p-1)/gcd(d,p-1)} = 1$, where $kd \equiv gcd(d,p-1) \pmod{p-1}$.

5.2 Application: Coin Tossing over a telephone

Suppose that Alice and Bob would like to have a fair coin toss over a distance, such as via telephone. How can they do this without either of them being able to gain an unfair advantage? A solution to this based was suggested by Manuel Blum.

First, let's consider the following abstract protocol:

1. Alice and Bob agree on a function f, whose domain is two- valued, say $\{H,T\}$.
2. Alice tosses a coin, say the result is $x \in \{H,T\}$.
3. Alice sends $f(x)$ to Bob.
4. Now Bob calls Heads or Tails, let y be the value called by Bob.
5. Bob sends y to Alice.
6. Alice now sends x to Bob; both of them know x,y and know who has won the toss.

The key properties required of the function f for the above protocol to work are:

(a) Given the value of $f(x)$, Bob cannot find out the value of x. This means that $f(H)$ must equal $f(T)$, otherwise Bob can compute each of these two values and see which one was sent by Alice.

(b) Alice shouldn't be able to find $x_2 \neq x_1$ such that $f(x_1) = f(x_2)$; otherwise she can always switch to the other value if needed.

The two properties appear to be negations of each other; however the idea is that the two-valuedness of the domain is indirectly encoded.

The protocol suggested by Blum is the following.

1. Bob chooses two large primes p, q and sends their product $n = pq$ to Alice.
2. Alice picks a random number $x \in \mathbb{Z}_n$; with high probability, $x \in \mathbb{Z}_n^*$.
3. Alice sends $y = x^2 \in \mathbb{Z}_n$ to Bob.
4. Bob solves the equation $x^2 = y$ (with y being known and x being unknown) separately in $\mathbb{Z}_p$ and in $\mathbb{Z}_q$. Combining the solutions, Bob gets FOUR solutions in $\mathbb{Z}_n$.
5. Now Bob calls one of the four values, say Bob calls z and sends it to Alice.
6. Alice now sends x to Bob.
7. Bob wins the toss if $z = x$ or $z = -x$ (in $\mathbb{Z}_n$), else Alice wins the toss.

Let's see the two properties satisfied by the above protocol.

(a) Suppose the four square-roots of y in $\mathbb{Z}_n$ are $x, -x, x_2, -x_2$. Given these four values, Bob has exactly a 50% chance of guessing one of $x, -x$.

(b) Alice cannot switch to one of $x_2, -x_2$ because Alice cannot find the other pair of square-roots without essentially factoring n. This is because if Alice did find the value of x_2, then she can factor n by finding $gcd(x_1 - x_2, n)$ and $gcd(x_1 + x_2, n)$ which would be the values p, q.

The security of the protocol thus depends on the hardness of factoring an integer. Currently there is no known polynomial-time algorithm for factoring an integer and it is believed that such an algorithm is unlikely. If the numbers p, q are large enough, then Alice would need hundreds of years to factor n should she wish to do so.

5.3 The Legendre Symbol

The Legendre symbol is defined as follows, for a natural number a and prime number p. If p does not divide a, then it is defined as:

$$\left(\frac{a}{p}\right) = +1 \text{ if } x^2 = a \text{ has solutions in } \mathbb{Z}_p$$
$$= -1 \text{ otherwise}$$

If p divides a, then it is defined as $\left(\frac{a}{p}\right) = 0$.

By Theorem 5.1, we obtain the following:

Lemma 2 [Euler's Criterion] $\left(\frac{a}{p}\right) \equiv a^{\frac{(p-1)}{2}} \pmod{p}$.

The numbers $a \in \mathbb{Z}_p$ such that $\left(\frac{a}{p}\right) = 1$ are called *quadratic residues* and the numbers $a \in \mathbb{Z}_p$ such that $\left(\frac{a}{p}\right) = -1$ are called *quadratic non-residues*.

We now use Euler's criterion in the following exercises.

Class Exercises:

1. Find $\left(\frac{-1}{p}\right)$ for the following values of p: $3, 5, 7, 11, 13$.
   Solution: We compute $-1^{(p-1)/2}$ for each prime. We see that the above quantity is 1 if and only if $(p-1)/2$ is even, i.e. if $p \equiv 1 \pmod 4$. Thus, we get the following.

p	$\left(\frac{-1}{p}\right)$
3	-1
5	1
7	-1
11	1
13	1

2. Find $\left(\frac{2}{p}\right)$ for the following values of p: $5, 7, 11, 17$.
   Solution: We compute $2^{(p-1)/2}$ for each prime.
   (a) $p = 5$: $2^2 \equiv -1 \pmod 5$.
   (b) $p = 7$: $2^3 \equiv 1 \pmod 7$.
   (c) $p = 11$: $2^5 \equiv -1 \pmod{11}$.
   (d) $p = 17$: $2^8 = 16^2 \equiv 1 \pmod{17}$.
   Thus, we get the following.

p	$\left(\frac{2}{p}\right)$
5	-1
7	1
11	-1
17	1

3. (a) Does the equation $x^3 \equiv 2 \pmod{19}$ have solutions?
   (b) Does the equation $x^3 \equiv 2 \pmod{17}$ have solutions?
   Solution:
   (a) We have $p = 19$ and $3|(p-1)$. So we compute $2^(p-1)/3$, i.e. $2^6 \equiv 64 \equiv 7 \pmod{19}$. Thus, this equation does not have solutions.
   (b) We have $p = 17$ and $gcd(3, 16) = 1$. Thus, there is a unique solution, given by $x \equiv 2^k \pmod{17}$, where $3k \equiv 1 \pmod{16}$.
   More generally, given an arbitrary d, to check whether $x^d = a$ has solutions, we first find k such that $dk \equiv gcd(d, p-1) \pmod{p-1}$, and then check whether the equation $x^{gcd(d,p01)} = a^k$ has solutions.
4. Given the values of $\left(\frac{a}{p}\right)$ and $\left(\frac{b}{p}\right)$, find the corresponding values of $\left(\frac{ab}{p}\right)$.

Solution: Because of Euler's criterion, we find that $\left(\frac{ab}{p}\right) = \left(\frac{a}{p}\right)\left(\frac{b}{p}\right)$. Thus, we get the following.

$\left(\frac{a}{p}\right)$	$\left(\frac{b}{p}\right)$	$\left(\frac{b}{p}\right)$
1	1	1
1	-1	-1
-1	1	-1
-1	-1	1

5.4 The equation $x^2 = a$: Two easy cases

5.4.1 $p \equiv 3 \pmod 4$

When $p \equiv 3 \pmod 4$, we can solve $x^2 = a$ in $\mathbb{Z}_p$ directly, as the following exercise illustrates.

Exercise 5.1 Solve $x^2 = a$ in $\mathbb{Z}_{79}$, assuming that it has solutions. ∎

Solution: Since a is a quadratic residue, we must have: $a^{39} \equiv 1 \pmod{79}$. Multiplying by a on both sides, we get $a^{40} \equiv a \pmod{79}$, so that $x = a^{20}$ is a square-root of a.

The above idea generalizes for any prime $p \equiv 3 \pmod 4$ because the condition $a^{(p-1)/2} \equiv 1 \pmod p$ is equivalent to $a^{(p+1)/2} \equiv a \pmod p$ and $(p+1)/2$ is even when $p \equiv 3 \pmod 4$.

Thus, in this case, $\boxed{a^{(p+1)/4} \text{ is a square-root of } a.}$

5.4.2 $p \equiv 1 \pmod 4$, $a = -1$

A key idea that we will use is that we can find a quadratic non-residue; for this we can simply sample a random element a and check if $a^{(p-1)/2} \equiv -1 \pmod p$, as half of the elements in $\mathbb{Z}_p$ are quadratic non-residues. By sampling several elements, we can ensure that the probability of success is high.

Lemma 3 There's an efficient randomized algorithm that finds $b \in \mathbb{Z}_p$ such that $\left(\frac{b}{p}\right) = -1$ with high probability.

In particular, once we find such a b, notice that b satisfies $b^{(p-1)/2} = -1$ and if $p \equiv 1 \pmod 4$, then $\boxed{b^{(p-1)/4} \text{ is a square-root of -1.}}$

5.5 Wilson's theorem and the value of $\left(\frac{2}{p}\right)$

Theorem 5.2 Let p be a prime. Then $(p-1)! \equiv -1 \pmod p$.

Proof. It is possible to prove this in several ways. One proof idea is to note that the numbers in $\{1, 2, \ldots, p-1\}$ may be paired up as (a, a^{-1}), except for $1, (p-1)$

which are their own inverses. The product of each pair is 1, and the product of the remaining elements is -1.

Another proof idea is to observe that $x^{p-1}-1=(x-1)\ldots(x-p+1)$ because of Fermat's little theorem. Comparing the constant term gives the result. ■

Using Wilson's theorem, we derive the following which we shall use in computing $\left(\frac{2}{p}\right)$.

Claim 4 Let $r=\left(\frac{p-1}{2}\right)!$. Then $r^2\equiv -1 \pmod p$ if $p\equiv 1 \pmod 4$, and $r^2\equiv 1 \pmod p$ if $p\equiv 3 \pmod 4$.

Proof. Writing $p-1=-1, p-2=-2, \ldots, (p+1)/2=-(p-1)/2$, we obtain, $(p-1)!=r^2(-1)^{(p-1)/2}$. From this and Wilson's theorem, the claim follows. ■

We now give the value of $\left(\frac{2}{p}\right)$.

Theorem 5.3 Let p be an odd prime. If $p\equiv \pm 1 \pmod 8$, then 2 is a quadratic residue modulo p, otherwise, 2 is a quadratic non-residue modulo p.

Proof. All calculations are in $\mathbb{Z}_p$. We write $(p-1)!=ST$, where $S=1\times 3\times\ldots\times(p-2)$ and $T=2\times 4\times\ldots\times(p-1)$. We note that $T=2^{(p-1)/2}\left(\frac{p-1}{2}\right)!$. Also, we rewrite S by writing each value $p-x$ in the second half of the product as $-x$ to obtain: $S=\left(\frac{p-1}{2}\right)!(-1)^{\lfloor (p-1)/4\rfloor}$. Letting $r=\left(\frac{p-1}{2}\right)!$ and substituting for S,T, we get:

$$(p-1)!=2^{(p-1)/2}r^2(-1)^{\lfloor (p-1)/4\rfloor}.$$

The LHS is -1; on the RHS, we know the value of r^2 from the previous claim. Thus we get an expression for $2^{(p-1)/2}$, whose value we may obtain for each of the values of p modulo 8. ■

5.6 Quadratic Reciprocity

One of the most interesting results about quadratic residues is that of the law of quadratic reciprocity, proved by Gauss.

Theorem 5.4 — Law of quadratic reciprocity. If p,q are odd primes, then

$$\begin{aligned}\left(\frac{p}{q}\right)&=\left(\frac{q}{p}\right) \text{ if } p\equiv 1 (\text{mod } 4) \text{ or } q\equiv 1(\text{mod } 4)\\ &=-\left(\frac{q}{p}\right) \text{ if } p\equiv q\equiv 3(\text{mod } 4)\end{aligned}$$

For example, to calculate $\left(\frac{3}{97}\right)$, we can write:

$$\left(\frac{3}{97}\right) = \left(\frac{97}{3}\right) = \left(\frac{1}{3}\right) = 1.$$

5.7 The Tonelli-Shanks Algorithm: Exposition from 2022

Before we describe the algorithm, we recall that the main idea in solving $x^2 = a$ when p is a prime congruent to 3 mod 4 is that we had: $a^m = 1$, where $m = (p-1)/2$ is odd. By multiplying by a on both sides, we get $a^{m+1} = a$ and since $m+1$ is even, $a^{(m+1)/2}$ is a square-root of a.

This fails for primes of the form 1 mod 4; nevertheless a modification of the idea works. We first illustrate this in the case that p is congruent to 5 mod 8.

5.7.1 $p \equiv 5 \pmod 8$

Suppose that $p \equiv 5 \pmod 8$. Then, we can write $p-1 = 4m$, where m is odd. For example, if $p = 61$, we can write $p-1 = 4m$ with $m = 15$.

Consider the number a^m in $\mathbb{Z}_p$. If $a^m = 1$, then our earlier method works, i.e. $a^{(m+1)/2}$ is a square-root of a. Suppose that $a^m \neq 1$. We also know that $a^{2m} = a^{(p-1)/2} = 1$. This implies that $a^m = -1$. How can we use this information?

The key idea is to find a quadratic non-residue, i.e. a number r such that $r^{(p-1)/2} = -1$. We can find a such an element by sampling random elements from $\mathbb{Z}_p^*$ and testing if they satisfy the condition. Since at least half of the elements in $\mathbb{Z}_p^*$ are quadratic non-residues, the probability of success is $1/2$.

Lemma 5 Given a prime p, there is an efficient randomized algorithm to find an element $r \in \mathbb{Z}_p$ such that r is quadratic non-residue, i.e. $r^{(p-1)/2} = -1$.

In particular, for our current example, we have: $r^{2m} = -1$. This means that $(ar^2)^m = 1$. Multiplying by a on both sides we obtain: $a^{m+1}r^{2m} = a$. Now the LHS has both exponents even so that we find $a^{(m+1)/2}r^m$ as a square-root of a.

Example: Let $p = 61$ and $a = 3$. By quadratic reciprocity, we know that $\left(\frac{3}{61}\right) = \left(\frac{61}{3}\right) = 1$.

We have $p-1 = 4m$ with $m = 15$. We find $a^m = 3^{15} = -1$. We also know from our earlier calculations that 2 is a quadratic non-residue when $p \equiv 5 \pmod 8$. Thus we set $r = 2$. We have $r^{2m} = 2^{30} = -1$, thus $3^{15} \cdot 2^{30} = 1$. Multiplying by 3 on both sides, we obtain $3^8 \cdot 2^{15} = 8$ as a square-root of 3. The other square-root is $-8 = 53$.

To summarize, when $p \equiv 5 \pmod 8$, we have two cases.

If $a^m = 1$, then $a^{(m+1)2}$ is a square-root.

If $a^m = -1$, then $a^{(m+1)/2} \cdot 2^m$ is a square-root.

5.7.2 The general case

Given an odd prime p, we write $p-1=2^t m$, with m odd. Given a number $a \in \mathbb{Z}_p$ whose square-root we wish to find, the key idea is to find a number b such that

$$(ab^2)^m = 1.$$

By multiplying by a on both sides, we then find $a^{(m+1)/2}b^m$ as a square-root of a.

How can we find such a b? Let's first consider the powers $a^m, a^{2m}, \ldots, a^{2^{t-1}m} = 1$. Let k be the least non-negative integer such that $a^{2^k m} = 1$. If $k = 0$, then we are done (with $b = 1$), otherwise let $a_1 = ar^e$, where r is a quadratic non-residue modulo p and $e = 2^{t-k}$. Then we have $a_1^{2^{k-1}m} = 1$. We now find the smallest non-negative integer k_1 such that $a_1^{2^{k_1}m} = 1$; note that $k_1 < k$. By repeating this process, we obtain a sequence of values $a_0 = a, a_1, \ldots, a_l$ where a_{i+1} is of the form $a_{i+1} = a_i r^{e_i}$ with e_i being an even non-negative integer and $a_l^m = 1$.

Before describing the algorithm formally, we illustrate it with an example.

Example: Solve the equation $x^2 = 2$ in $\mathbb{Z}_{97}$.

Solution: We have $a = 2, m = 3$ and we find:

$a^3 = 8, a^6 = 64, a^{12} = 22, a^{24} = -1, a^{48} = 1$.

We now find r such that r is a quadratic non-residue modulo 97. $r = 5$ works (as can be checked using quadratic reciprocity, for example).

We now that $r^{48} = -1$, thus we set $a_1 = ar^2$ so that $a_1^{24} = 1$. Thus, $a_1 = 2 \times 25 = 50$.

Now we compute $a_1^3 = 64, a_1^6 = 22, a_1^{12} = -1, a_1^{24} = 1$. We set $a_2 = a_1 r^4$ so that $a_2^{12} = 1$. Thus, $a_2 = 50 \times 5^4 = 16$.

We find $a_2^3 = 22, a_2^6 = -1, a_2^{12} = 1$ and we set $a_3 = a_2 r^8$ so that $a_3^6 = 1$. Thus, $a_3 = 16 \times 5^8 = -1$.

Finally, we set $a_4 = a_3 r^{16} = 61$ and we have $a_4^3 = 1$. Thus, we get $\left(ar^{2+4+8+16}\right)^3 = 1$. From this, we find a square-root of a to be: $a^2 r^{3+6+12+24}$, which is equal to 83.

Thus, the two square-roots are 14,83.

We now describe the algorithm formally. Since every computation involves a mth power it is convenient to compute mth powers at the beginning of the algorithm itself.

In the algorithm, findk(z) is a function that returns the least non-negative integer k such that $z^{2^k} = 1$. This need not exist for every $z \in \mathbb{Z}_p$, but it exists for values z which are mth powers (on which the function is invoked).

The invariant $x^2 = ab$ is maintained at the beginning and end of each iteration. The values of b, k, S, x for our earlier example ($a = 2, p = 97$) would be as shown in the table below (with $r = 5$).

Algorithm 4 Tonelli-Shanks Algorithm

procedure TONELLI-SHANKS(a, p) ▷ Finds x such that $x^2 = a$ in $\mathbb{Z}_p$
  Write $p-1 = 2^t m$ with m odd.
  $b \leftarrow a^m$
  $k \leftarrow findk(b)$
  if k=t **then return** Not a square.
  end if
  $x \leftarrow a^{\frac{(m+1)}{2}}$.
  if k=0 **then return** x.
  end if
  Find r such that $r^{\frac{(p-1)}{2}} = -1$.
  $s \leftarrow r^m$
  $S \leftarrow s^{2^{t-k}}$
  while $k > 0$ **do**
    $b \leftarrow bS$
    $x \leftarrow xs^{2^{t-k-1}}$
    $k \leftarrow findk(b)$
    $S \leftarrow s^{2^{t-k}}$
  end while
  return x
end procedure

b	k	S	x
8	4	28	4
64	3	8	15
22	2	64	23
96	1	22	17
1	0	96	83

5.8 The Tonell-Shanks algorithm: Exposition of 2025

Given an odd prime p and $a \in \mathbb{Z}_p$, our goal is to solve the equation

$$x^2 = a \tag{5.1}$$

in $\mathbb{Z}_p$ in polynomial time, i.e. time polynomial in $\log p$.

Firstly, we check whether $a^{(p-1)/2} = 1$. If not, we report that the equation has no solution.

Now, we write $p-1 = 2^t m$, with m odd. Our first observation is that it is sufficient to solve the equation $y^2 = a^m$. Given a solution y to this equation, the solutions to Equation 5.2 are: $x = \pm \dfrac{a^{(m+1)/2}}{y}$.

Thus, we now focus on solving the equation

$$y^2 = b \tag{5.2}$$

where $b = a^m$; further since $a^{(p-1)/2} = 1$, we have:

$$b^{2^{t-1}} = 1 \tag{5.3}$$

Our second idea is to find an element $r \in \mathbb{Z}_p$ such that $r^{(p-1)/2} = -1$. To find such an element, observe that $(p-1)/2$ elements in $\mathbb{Z}_p$ satisfy the above relation; hence we may sample random elements until one satisfies the above relation.

Now consider the element $s = r^m$; this satisfies

$$s^{2^{t-1}} = -1 \tag{5.4}$$

We now show how to combine equations 5.3 and 5.4 to find a square-root of b. We shall maintain inductively, an exponent e such that $b^{2^i} = s^e$, where e is divisible by 2^{i+1}, with the initial values being $i = t-1$, $e = 1$. We aim to maintain such a relation for the values of i down to zero. At $i = 0$, we have an expression for b as a square.

For the inductive step, note that $b^{2^{i-1}} = s^{e/2}$ or $b^{2^{i-1}} = -s^{e/2}$. In the first case, we divide e by 2; in the latter case, we substitute for -1 using Equation 5.4 to obtain: $b^{2^{i-1}} = s^{(p-1)/2+e/2}$; thus, in this case, we replace e with $(e+p-1)/2$.

An example: We solve the equation $x^2 = 2$ in $\mathbb{Z}_{97}$. We first check that $2^{48} = 1$. We have $m = 3$, $t = 5$ and $b = 8$. We store the value $X = a^{(m+1)/2} = 4$, to divide this by the square-root of b later. We find that $r = 5$ satisfies $r^{48} = -1$ and set $s = r^3 = 28$.

Thus we start with $b^{16} = 1$ and $s^{16} = -1$ with $b = 8$, $s = 28$.

We first compute $b = 8$, $b^2 = 64$, $b^4 = 22$, $b^8 = -1$. We also compute $s = 28$, $s^2 = 8$, $s^4 = 64$, $s^8 = 22$. By chance, s itself happens to be a square-root of b; nevertheless let's run through the algorithm for now.

Starting with $b^8 = s^{16}$, we check if $b^4 = s^8$ (yes); then we check if $b^2 = s^4$ (yes) and if $b = s^2$ (yes). Thus we find $y = 28$ and $x = \pm\frac{4}{28} = \pm 14$.

Now let's run through it again with a different value of r, say $r = 7$ which also satisfies $r^{48} = -1$. Now we have $s = 7^3 = 52$ and $s^2 = 85$, $s^4 = 47$, $s^8 = 75$, $s^{16} = -1$.

We have $b^8 = s^{16}$ but $b^4 = -s^8$ so that we write $b^4 = s^{24}$; now we find that $b^2 = -s^{12}$, so we write it as $b^2 = s^{28}$ and finally we find that $b = -s^{14} = s^{30}$, and we find $y = \pm s^{15} = \pm 28$ as before.

We now describe the algorithm formally.

Algorithm 5 Tonelli-Shanks Algorithm

procedure TONELLI-SHANKS(a, p) ▷ Finds x such that $x^2 = a$ in $\mathbb{Z}_p$
  Write $p-1 = 2^t m$ with m odd.
  $b \leftarrow a^m$ ▷ We'll solve $y^2 = b$
  Find $B[j] = b^{2^j}$ for $j = 0, 1, \ldots, i$ by repeated squaring until $B[i] = 1$.
  if i=t **then return** Not a square. ▷ $b^{2^t} = 1$ and $b^{2^{t-1}} = -1$
  end if
  $x \leftarrow a^{\frac{(m+1)}{2}}$. ▷ We'll divide this by y.
  if i=0 **then return** x. ▷ $b = 1$
  end if
  Find r such that $r^{\frac{(p-1)}{2}} = -1$. ▷ Sample and check
  $s \leftarrow r^m$
  $e \leftarrow 2^t$ ▷ Initialize $b^{2^i} = s^e$
  while $i > 0$ **do**
    if $B[i-1] = s^{e/2}$ **then**
      $e \leftarrow e/2$
    else
      $e \leftarrow e/2 + (p-1)/2$
    end if
  end while
  $Y \leftarrow s^{p-1-e/2}$ ▷ $y = s^{e/2}$; $Y = y^{-1}$
  return xY
end procedure

5.9 Hensel Lifting: From $\mathbb{Z}_p$ to $\mathbb{Z}_{p^k}$

We have seen one method to solve the quadratic equation in $\mathbb{Z}_p$; we will now see how to extend this to solving quadratic equations in $\mathbb{Z}_p^k$ for any given positive integer k.

By using the algorithm in the previous section we first find a solution b to $x^2 \equiv a \pmod{p}$. To solve the equation $x^2 \equiv a \pmod{p^2}$, we write $x = py + b$; this gives us:

$$(py + b)^2 \equiv a (\bmod\ p^2),$$

which is equivalent to:

$$2pby \equiv -(b^2 - a) (\bmod\ p^2).$$

Writing $b^2 - a = pc$ for an integer c (since p divides $b^2 - a$), and dividing by p, we get:

$$2by \equiv -c(\text{mod } p).$$

For an odd prime p, we can now solve this equation as $gcd(2b, p) = 1$ (assuming $gcd(a, p) = 1$).

The same method works for finding a solution modulo p^{2k} given a solution modulo p^k; this is called Hensel Lifting. Thus, from a solution modulo p, we can find a solution modulo p^k using $O(\log k)$ calls to a linear-equation solver.

This method generalizes further to finding the roots of polynomials of arbitrary degree as well as for factoring polynomials.

We illustrate this with a couple of examples:

Example 1: Solve the equation $x^2 \equiv 5$ (modulo 361).

Solution: Note that $361 = 19^2$. We first solve $x^2 \equiv 5$ (modulo 19). We find $5^9 \equiv 1$ (modulo 19) so that 5 is a quadratic residue. Multiplying by 5 on both sides, we obtain $5^5 \equiv 9$ modulo 19.

Now we write $x = 19y + 9$ and substitute to obtain:

$$(19y + 9)^2 \equiv 5(\text{ mod } 361).$$

That is:

$$18y \equiv -4 (\text{ mod } 19).$$

Solving this gives: $y \equiv 4$ (mod 19), so that $x = 85$ is a solution.

Example 2: Solve the equation $x^3 \equiv 3$ (mod 289).

Solution: We have $289 = 17^2$. We first solve the equation $x^3 \equiv 3$ (mod 17). The exponent 3 is co-prime to 16, so we first solve the auxiliary equation $3k + 16l = 1$, which gives us $k = 5, l = -1$ as a solution. We now raise our congruence equation to the 5th power on both sides to obtain:

$$x^{15} \equiv 3^5(\text{ mod } 17).$$

Since $x^{16} \equiv 1$ (mod 17), this is equivalent to $x \equiv \frac{1}{3^5}$ (mod 17), which gives us $x \equiv 7$ (mod 17).

Now we write $x = 17y + 7$ to obtain:

$$(17y + 7)^3 \equiv 3(\text{ mod } 289),$$

which is equivalent (after simplification) to

$$3 \times 49 \times y \equiv -20 \text{ mod } 17.$$

From this we solve for y to find $y \equiv 9$ (mod 17), so that $x \equiv 160$ (mod 289) is a solution.

5.10 A second algorithm for finding square-roots

We know describe another method to solving the equation

$$x^2 = a$$

in $\mathbb{Z}_p$.

The algorithm itself is easy to describe and is as follows.

Algorithm 6 Square-root Algorithm

1: **procedure** FIND SQUARE-ROOT(a,p) ▷ Finds x such that $x^2 = a$ in $\mathbb{Z}_p$
2: Pick random $r \in \mathbb{Z}_p$.
3: **if** $gcd(x^2 - a, (x-r)^{(p-1)/2} - 1) = (x-b)$ **then return** $b, -b$.
4: **end if**
5: **end procedure**

The gcd is found by repeatedly squaring $(x-r)^{(p-1)/1}$ modulo (x^2-a) (in $\mathbb{Z}_p[x]$). Note that $(sx+t)^2 \equiv 2stx + s^2a + t^2$ modulo (x^2-a); this expression may also be used to find the successive remainders.

If the gcd in step 2 is 1 instead, then we repeat the algorithm with another random value of r, until success. The probability that the above algorithm succeeds is at least $1/2$.

Example: $x^2 = 10$ **in** $\mathbb{Z}_{41}$

Solution: For $r = 1$, we find the gcd to be 1; for $r = 2$, the calculation is as follows, the congruences being modulo $(x^2 - 10)$.

$$\begin{aligned}(x-2)^2 &\equiv 14 - 4x \\ (x-2)^4 \equiv (14-4x)^2 &\equiv 28 + 11x \\ (x-2)^8 \equiv (28+11x)^2 &\equiv x + 26 \\ (x-2)^{16} \equiv (x+26)^2 &\equiv 11x + 30\end{aligned}$$

Thus, $(x-2)^{20} \equiv (11x+28)(11x+30) \equiv 23x$. The gcd of (x^2-10) and $(x-2)^{20} - 1$ is $gcd(x^2 - 10, 23x - 1) = (x + 16)$.

Thus the square-roots of 10 in $\mathbb{Z}_{41}$ are ± 16.

5.11 Exercises

1. Solve $x^2 = 2$ in $\mathbb{Z}_{289}$.
2. Describe an efficient method to decide whether a polynomial $ax^2 + bx + c \in \mathbb{Z}_p[x]$ has solutions and how to find them.
3. Find the value of $\left(\frac{510}{1009}\right)$, given that 1009 is a prime.

6. Finite Fields

In this chapter, we study finite fields which are of importance in several areas of computer science, such as coding theory, cryptography and complexity theory. This will also lead us to a factorization algorithm for polynomials in $\mathbb{Z}_p[x]$.

6.1 Groups

In abstract algebra, the most basic objects are groups.

Definition 6.1 A group is a pair $(G,*)$, where G is a set and $*$ is a binary operation on G which satisfies all of the following properties.

(a) **Closure:** For all $g,h \in G$, we have $g*h \in G$.

(b) **Associativity:** For all $g,h,k \in G$, we have: $g*(h*k) = (g*h)*k$.

(c) **Identity:** There is a unique element $e \in G$ such that for all $g \in G$, we have: $g*e = e*g = g$.

(d) **Inverse:** For every element $g \in G$, there is an element h such that $g*h = h*g = e$. This element is usually denoted by g^{-1}.

Examples of groups:

1. $(\mathbb{R},+)$, $(\mathbb{C},+)$, $(\mathbb{Q},+)$, $(\mathbb{Z},+)$, $(\mathbb{R}^n,+)$, $(\mathbb{R}[x],+)$
2. $(\mathbb{R}\setminus\{0\},\times)$, $(\mathbb{C}\setminus\{0\},\times)$, $(\mathbb{Q}\setminus\{0\}),\times)$
3. $(\mathbb{Z}_n,+)$ (the cyclic group), $(\mathbb{Z}_n^*,\times)$
4. For every fixed n, the set of all $n\times n$ matrices over $\mathbb{R}$ under addition forms a group.
5. For every fixed n, the set of all non-singular $n\times n$ matrices over $\mathbb{R}$ under multiplication forms a group.
6. The set of all permutations of $\{1,2,\ldots,n\}$ under composition forms a group; this group is called the *symmetric group* and is denoted by S_n.
7. $(\mathbb{R},*)$ where $a*b = a+b+1$.

Non-examples:

1. The set $\{0,1,2\}$ under addition is not a group because it fails the Closure property (a).
2. $(\mathbb{R},-),(\mathbb{C},-)$ are not groups because they fail the Associativity property (b).
3. The pair $(R,*)$ with $*$ being defined as $a*b=1$ is not a group because it fails to have an Identity element.
4. $(2^S,\cup)$, $(2^S,\cap)$ are not groups because they fail the existence of an inverse for every element.
5. The sets $\mathbb{R},\mathbb{C},\mathbb{Z}_n$ are not groups under multiplication but note that we can obtain groups from them by removing the elements without an inverse (Zero in the first two cases, elements having a common factor with n in the third).

An important property of groups is cancellation and it follows from the existence of inverses.

Observation 6 — Cancellation property. If $(G,*)$ is a group and $a*b=a*c$, for elements $a,b,c\in G$, then $b=c$. Similarly, if $g*h=j*h$ for elements $g,h,j\in G$, then $g=j$.

6.1.1 Cayley Tables and Isomorphism

The Cayley table for a group $(G,*)$ is a matrix with rows and columns indexed by the elements of G and the entry in row g and column h being the element $g*h$. The Cayley table defines the operation $*$. From the cancellation property, it follows that every row (similarly, every column) must contain all the elements of G exactly once (in some order).

Here's an example: we define a two-element group $G=(\{a,e\},*)$ (with e being the identity element) using the following table:

*	e	a
e	e	a
a	a	e

Now consider the Cayley table of the group $(\mathbb{Z}_2,+)$.

+	0	1
0	0	1
1	1	0

We may notice that the tables of the two groups are identical except for a changing of the element names (with 0 in place of e and 1 in place of a, + in place of *). When this happens for two groups, we call them *isomorphic*. Here's a more formal definition.

Definition 6.2 Two groups $(G,*)$ and $(H,\cdot)$ are said to be isomorphic if there exists a bijection $\phi:G\to H$ such that the following holds:

For every $g_1,g_2,g_3\in G$, $g_1*g_2=g_3$ if and only if $\phi(g_1)\cdot\phi(g_2)=\phi(g_3)$.

Examples of isomorphism:

1. Every group of size 3 is isomorphic to $(\mathbb{Z}_3,+)$.
2. The group of all nth roots of unity (under multiplication) is isomorphic to the group $(\mathbb{Z}_n,+)$. This group is also known as *the cyclic group* on n elements, and is denoted by C_n.

6.1.2 Direct products and subgroups

We now look at two ways to construct new groups from existing groups.

Definition 6.3 Given two groups $(G,*)$ and $(H,\cdot)$, the *direct product* of G and H is the group consisting of $G\times H$ with the group operation being co-ordinate wise, i.e. $(g_1,h_1)\times(g_2,h_2)=(g_1*g_2,h_1\cdot h_2)$.

It is easy to verify that $G\times H$ satisfies the definition of a group.

The direct product is useful to construct groups larger than (and containing)a given group. In contrast, a group that is smaller than (and contained) in a given group is called a subgroup.

Definition 6.4 We say that H is a subgroup of $(G,*)$ (written $H\leq G$) if $H\subseteq G$ and $(H,*)$ is a group.

From this definition, it is not clear how to construct subgroups of a given group, but one method is the following. Let $S\subseteq G$. We define the *group generated by* S as:

$$< S >= \{g_1*g_2*\ldots*g_k | k\in\mathbb{N} \text{ and } g_1,\ldots,g_k\in S\cup S^{-1}\}.$$

As in the case of direct product, we can easily verify that $(< S >,*)$ satisfies the group axioms, and is hence a subgroup of G. Conversely, if H is a subgroup of G and $< S >= H$, then we call $< S >$ a *generating set* for H.

Examples of subgroups:

Group	Subgroup(s)
$(\mathbb{R},+)$	$\mathbb{Q}$, $\mathbb{Z}$
$(\mathbb{Z},+)$	$2\mathbb{Z},3\mathbb{Z},4\mathbb{Z},\ldots$
$(\mathbb{Z}_6,+)$	$< 3 >= \{0,3\}$,$< 2 >= \{0,2,4\}$
$(\mathbb{Z}_p^*,\times)$	Quadratic residues
$(\mathbb{Z}_2\times\mathbb{Z}_2,+)$	$< (1,1) >= \{(1,1),(0,0)\}$
$\mathrm{GL}_n(\mathbb{R})$	$\mathrm{SL}_n(\mathbb{R})$

$\mathrm{GL}_n(\mathbb{R})$: Real, invertible $n\times n$ matrices; $\mathrm{SL}_n(\mathbb{R})$: Real $n\times n$ matrices with determinant 1

6.1.3 Cosets and Lagrange's theorem

In this section, we prove the following result, which is the main tool from group theory that we will use.

Theorem 7 — Lagrange's Theorem. Let G be a finite group and H be a subgroup of G. Then $|H|$ divides $|G|$. In particular, for all $g\in G$, we have: $g^{|G|}=e$.

Before we begin the proof, we first define cosets which we will need.

Definition 6.5 Given a group $(G,*)$ and a subgroup H of G, a *left coset* of H is a set of the form $g*H=\{g*h|h\in H\}$ for $g\in G$.

For example, for $G=(\mathbb{Z},+)$ and $H=4\mathbb{Z}$, a coset is $3+H=\{3+4k:k\in\mathbb{Z}\}$. In fact, there are exactly four left cosets, namely $4\mathbb{Z}$, $4\mathbb{Z}+1$, $4\mathbb{Z}+2$, $4\mathbb{Z}+3$. For example, the coset $7+H$ is the same as $3+H$.

Lemma 8 If $(G,*)$ is a group and H is a subgroup of G, then for all $g,h \in G$, we have: $g*H = h*H$ or $g*H \cap h*H = \emptyset$.

We first prove Lagrange's theorem using the lemma (which we will prove subsequently).

Proof of Theorem 7: Lemma 8 implies that the left cosets of G form a partition of G. Thus, we can write $G = H \cup g_1 * H \cup \ldots \cup g_{r-1} * H$ for some $g_1, \ldots, g_{r-1}$.

Each coset has the same size, i.e. $|H|$. Thus, we obtain: $r|H| = |G|$, i.e. $|H|$ divides $|G|$, as desired.

The second part follows by considering the subgroup $H = \{a, a^2, \ldots, a^k = e\}$ (for some $k \in \mathbb{N}$). We then have: k divides $|G|$, and hence $a^{|G|} = e$. ■

6.2 Rings

Definition 6.6 A ring is a triple $(R,+,\cdot)$ with $+,\cdot$ being binary operations on R, satisfying the following properties.

1. $(R,+)$ is an abelian group; its identity element is denoted by 0.
2. $(R,\cdot)$ is associative and has an identity element which is denoted by 1.
3. For all $a,b,c \in R$, we have: $a\cdot(b+c) = a\cdot b + a\cdot c$ and $(a+b)\cdot c = a\cdot c + b\cdot c$.

Examples of rings: In each example, the addition and multiplication are the natural operators.

1. $\mathbb{Z}, \mathbb{R}, \mathbb{C}$
2. $\mathbb{Z}[x]$, $\mathbb{R}[x]$
3. $\mathbb{Z}_n$, $\mathbb{Z}_n[x]$ (for every fixed n)
4. $\mathbb{Z}_n[x]/(f(x))$, i.e. the ring of polynomials in $Z_n[x]$ modulo $f(x)$
5. The ring of $n \times n$ real matrices (for each fixed n)

6.3 Fields

A field is a ring with the additional property that every non-zero element has a multiplicative inverse.

Definition 6.7 A field is a triple $(F,+,\cdot)$ with $+,\cdot$ being binary operations on R, satisfying the following properties.

1. $(F,+)$ is an abelian group; its identity element is denoted by 0.
2. $(F\setminus\{0\},\cdot)$ is a group, whose identity element is denoted by 1.
3. For all $a,b,c \in F$, we have: $a\cdot(b+c) = a\cdot b + a\cdot c$ and $(a+b)\cdot c = a\cdot c + b\cdot c$.

Examples of fields: $\mathbb{R}$, $\mathbb{Q}$, $\mathbb{C}$, $\mathbb{Z}_p$ (p prime). Another important example for us will be: $\mathbb{Z}_p[x]/(f(x))$, where $f(x)$ is an irreducible polynomial in $\mathbb{Z}_p[x]$.

Fields are highly structured objects with many nice properties. Here are two useful ones:

- If F is a field, then the Euclidean algorithm and Bezout's Lemma work for polynomials in $F[x]$.
- If F is a field, then Gaussian elimination (and all of linear algebra) work over F in the same way as for reals.

Theorem 6.1 If F is a field, then $F[x]/(f(x))$ is a field if and only if $f(x)$ is an irreducible polynomial in $F[x]$.

Proof. Suppose that $f(x)$ is an irreducible polynomial. Let $g(x) \in F[x]/(f(x))$ be a polynomial of degree less than $deg(f)$. Then $gcd(g(x), f(x)) = 1$ and in $F[x]$, we can apply Bezout's Lemma to obtain polynomials $u(x)$ and $v(x)$ such that $f(x)u(x) + g(x)v(x) = 1$. That is, $g(x)v(x) \equiv 1 \pmod{(f(x))}$, so that $g(x)$ has a multiplicative inverse in $F[x]/(f(x))$. This shows that $F[x]/(f(x))$ is a field.

Conversely, suppose that $f(x)$ is not irreducible, let $f(x) = g(x)h(x)$ for a polynomial $g(x)$ of degree less than $deg(f)$. Then $g(x)$ does not have a multiplicative inverse in $F[x]/(f(x))$ and hence $F[x]/(f(x))$ is not a field. This completes the proof. ■

6.4 Finite Fields

Facts about finite fields: Let $q = p^k$ where p is prime and $k \in \mathbb{N}$.

1. There exists a finite field of size q; such a field may be constructed as $\mathbb{Z}_p[x]\ (f(x))$ for some irreducible polynomial of degree k in $\mathbb{Z}_p[x]$.
2. Any 2 fields of size q are isomorphic. Thus, we often write $\mathbb{F}_q$ to denote the field of size q.
3. $\mathbb{F}_q^*$ is cyclic.
4. For every $a \in \mathbb{F}_q$, we have: $a^q = a$.

6.5 Irreducible polynomials in $\mathbb{Z}_p[x]$

Theorem 9 In $\mathbb{Z}_p[x]$, we have the following:

- **(a):** If $f(x)$ is an irreducible polynomial of degree k, then $f(x)$ divides $x^{p^k} - x$.
- **(b):** $x^{p^k} - x = \prod_{d|k} \prod\{f(x) : f(x) \text{ is monic and irreducible, deg(f)} = d\}$.

Proof. (a) Let $q = p^k$. Then $F[x]/(f(x)) \cong \mathbb{F}_q$. Considering the polynomaial x as an element of $F[x]/(f(x))$, we therefore have: $x^q - x = 0 \pmod{f(x)}$ which is the desired statement.

(b) We first show that the RHS divides the LHS. Let $f(x)$ be an irreducible polynomial of degree d, with $d|k$, let $k = rd$. Then by part(a), we have: $x^{p^d} \equiv x \pmod{f(x)}$. Raising both sides to the p^dth power we obtain $x^{p^{2d}} \equiv x^{p^d} \equiv x \pmod{f(x)}$. Thus inductively we obtain $x^{p^{id}} \equiv x \pmod{f(x)}$ for every i; setting $i = r$, we get the desired result. ■

For example, in $\mathbb{Z}_3[x]$, we have: $x^9 - x = x(x-1)(x-2)(x^2-2)(x^2+x+2)(x^2+2x+2)$.

6.6 Application: Secret sharing

There's a secret S that has to be distributed to n persons; each person gets a *share* and what we want is the following: if any k persons combine their share, then they can recover the secret, but from any $k-1$ shares, zero information is obtained about the secret. Such a scheme is called a k-out-of-n threshold scheme (k being the threshold). We can imagine that a majority threshold (say 3-out-of-5) is relevant for a consensus decision to recover the secret.

The following scheme is due to Shamir ("How to Share a Secret", 1979).

1. Encode the secret as an element of $\mathbb{F}_q$ for suitable q; if the secret is a sequence of bits, then q may be a power of 2.
2. Generate $k-1$ random values of $\mathbb{F}_q$: $a_1, \ldots, a_{k-1}$.
3. Construct the polynomial $f(x) = a_0 + a_1 x + \ldots + a_{k-1}x^{k-1}$, where a_0 is the secret.
4. Generate n distinct values $x_1, \ldots, x_n \in \mathbb{F}_q$ and compute $y_i = f(x_i)$ for each i.
5. The pair (x_i, y_i) is the share of the ith person.

Explanation: Given k pairs $(x_i, f(x_i))$, it is possible to recover all the coefficients of $f(x)$, as there are k unknowns (the coefficients) and k linear equations. In particular, the coefficient a_0, which is the secret, can be obtained.

With $k-1$ shares/equations, the system of equations results in a solution space which is a 1-dimensional affine subspace of $\mathbb{F}_q^k$ and each of the q values is equally likely for a_0.

7. Polynomial Factorization over $\mathbb{Z}_p$

In this chapter, we'll see the Cantor-Zassenhaus algorithm for factoring univariate polynomials in $\mathbb{Z}_p[x]$.

We divide the algorithm into three components/phases and finally combine them together into a single algorithm.

In the first phase, given a polynomial $f(x) \in \mathbb{Z}[x]$, we'll obtain a polynomial $g(x)$ which is the square-free part of $f(x)$. The square-free part of a polynomial is defined as follows. Let $f(x) = h_1(x)^{e_1} \dots h_k(x)^{e_k}$ with $h_i(x)$ being distinct irreducibles and e_is being natural numbers. Then the square-free part of $f(x)$ is the polynomial $g(x) = h_1(x) \dots h_k(x)$.

In the second phase, we'll partition the square-free polynomial $g(x)$ into $f_1(x)f_2(x) \dots f_r(x)$, where $f_i(x)$ is the product of all monic irreducible factors of $g(x)$ that have degree equal to i.

In the third phase, we'll factor each $f_i(x)$ into irreducible factors of degree i.

Finally, for each irreducible factor $h(x)$ that divides $f(x)$, we find the largest natural number r such that $h(x)^r$ divides $f(x)$.

We now describe algorithms for each of the three phases.

7.1 Phase 1: Finding the square-free part

The main idea behind finding the square-free part is the following claim.

Claim 10 Let $f(x) \in \mathbb{Z}_p[x]$ be such that $f(x)$ is not divisible by $h(x)^p$ for any polynomial $h(x)$ of degree at least one. Then the square-free part of $f(x)$ is given by $\frac{f(x)}{gcd(f(x), f'(x))}$.

Notice that if $f(x) = h(x)^p g(x)$, then $f'(x) = h(x)^p g'(x)$; thus we are unable to use the above idea to find the square-free part of $h(x)$ if $h(x)^p$ divides $f(x)$.

To deal with this issue, we do the following: we first obtain the largest degree polynomial $h(x)$ such that $h(x)^p$ divides $f(x)$, then find the square-free part of $h(x)$ recursively. We then multiply this by the square-free part of $g(x)$ to obtain the square-free part of $f(x)$.

For the first step in the above idea, we use the following claim.

Claim 11 Let $f(x) \in \mathbb{Z}_p[x]$ and let $h(x)$ be the largest degree polynomial such that $f(x)$ is divisible by $h(x)^p$. Then there exists k such that $f^{(k)}(x) = h(x)^p$ and $f^{(k+1)}(x) = 0$. Here $f^{(j)}(x)$ denotes the jth derivative of $f(x)$.

To check whether a given polynomial in $\mathbb{Z}_p[x]$ is a pth power and to finds its pth root, we will use the following observation.

Claim 12 Let $f(x) = \sum_{i=0}^{d} a_i x^i \in \mathbb{Z}_p[x]$. Then $f(x)$ is equal to $g(x)^p$ for some polynomial $g(x)$ if aand only if: for every i, $a_i \neq 0$ implies $p|i$. Further, in this case, the polynomial $g(x)$ equals $\sum_{p|i} a_i x^{i/p}$.

We now have all the ingredients to find the square-free part of a given polynomial $f(x) \in \mathbb{Z}_p[x]$.

Algorithm 7 Algorithm to find square-free part of $f(x) \in \mathbb{Z}_p[x]$

```
1: procedure SQUARE-FREE PART(f(x), p)
2:     g(x) ← f(x), F(x) ← f(x)
3:     while g(x) ≠ 0 do
4:         h(x) ← g(x), g(x) ← g'(x)
5:     end while
6:     if h(x) ∈ Z_p then
7:         Return f(x)/gcd(f(x), f'(x))
8:     end if
9:     F(x) ← F(x)/h(x)
10:    if h(x) = H(x^p) then
11:        h(x) ← H(x)
12:        h(x) ← SQUARE-FREE PART(h(x), p)
13:    end if
14:    F(x) ← h(x)F(x)/gcd(F(x), F'(x))
15:    Return F(x)
16: end procedure
```

7.2 Phase 2: Distinct-degree factorization

In this phase, we now have a polynomial which is square-free. Our goal is to factorize it as $f_1(x) \dots f_r(x)$, where $f_i(x)$ is the product of all irreducible factors of the given polynomial of degree i.

The idea behind this algorithm is to use Theorem 9. Since the only irreducible polynomials that divides $x^p - x$ are those of degree one, $gcd(f(x), x^p - x)$ gives us $f_1(x)$. Note: Here, $f(x)$ is an input polynomial which is square-free, and corresponds to the output $g(x)$ of the first phase rather than to the original input polynomial.

Repeating this idea, suppose that we have found $f_1(x), \dots, f_i(x)$. Then $f_{i+1}(x) =$

$gcd\left(\frac{f(x)}{f_1(x)\dots f_i(x)}, x^{p^{i+1}} - x\right)$. We note that in finding the gcds, we shall use repeated squaring for the power of x and subsequently apply the Euclidean algorithm.

Algorithm 8 Distinct-degree factorization

```
1: procedure DISTINCT-DEGREE FACTORS(f(x), p)                    ▷
2: #: Finds f_1(x), f_2(x), ... where f_i(x) is the product of monic irreducible factors
   of f(x) of degree i in Z_p[x].                                ▷
3: Assumes that f(x) is square-free.
4:     i ← 1, g(x) ← f(x).
5:     while g(x) ≠ 1 do
6:         f_i(x) ← gcd(g(x), x^{p^i} − x).
7:         g(x) ← g(x)/f_i(x).
8:         i ← i + 1.
9:     end while
10:    Return {f_1(x), f_2(x), ...}.
11: end procedure
```

7.3 Phase 3: Finding irreducible factors of degree i

In this stage, our goal is to solve the following problem. Given a polynomial $f_i(x)$, all of whose irreducible factors have the same degree, i, find those factors. There are two ideas that we will use for this phase, and they are the content of the following two results.

Theorem 13 — Chinese Remainder Theorem for polynomials. If $h(x) = h_1(x)\dots h_k(x)$, where the $h_i(x)$s are pairwise co-prime polynomials in $\mathbb{Z}_p[x]$, then:

$$\mathbb{Z}_p[x]/(h(x)) \cong \mathbb{Z}_p[x]/(h_1(x)) \times \dots \times \mathbb{Z}_p[x]/(h_k(x)).$$

Claim 14 If $h(x)$ is an irreducible polynomial of degree i and $g(x)$ is a randomly chosen non-zero polynomial of degree less than i in $\mathbb{Z}_p[x]$, then:

$$Prob[g(x)^{(p^i-1)/2} \equiv 1(\bmod h(x))] = Prob[g(x)^{(p^i-1)/2} \equiv -1(\bmod h(x))] = \frac{1}{2}.$$

Let $f_i(x)$ be a polynomial with an unknown factorization $f_i(x) = h_1(x)\dots h_t(x)$, where each $h_j(x)$ is irreducible of degree i. Consider a polynomial $g(x)$ of degree less than $f_i(x)$, chosen uniformly at random. By Theorem 13, $g(x)$ modulo $h_j(x)$ will be a uniformly-at-random element of $\mathbb{Z}[x]/h_j(x)$, for every j. Applying Claim 14, for each j, the probability that $h_j(x)$ divides $gcd(g(x)^{(p^i-1)/2} - 1)$ is $1/2$. Further these events (various $h_j(x)$s dividing this gcd) are mutually independent. Thus, if $t \geq 2$, then with probability at least $1/2$, the above gcd will contain as factors some and not all of the irreducible factors of $f_i(x)$, which gives us a non-trivial factorization of $f_i(x)$. We can then recursively factorize each of the two factors thus obtained. The algorithm just described is presented below formally.

Finally, for each $h(x)$ in the list of irreducible factors obtained from Algorithm 9, we find the largest exponent e such that $h(x)^e$ divides $f(x)$. This completes the description of the factorization algorithm for polynomials in $\mathbb{Z}_p[x]$.

Algorithm 9 Uniform-degree irreducible factorization

procedure IRREDUCIBLEFACTORS($f(x), i, p$) ▷
#: Returns list of irreducible factors of $f(x)$ of degree i in $\mathbb{Z}_p[x]$ ▷
#: Assumes that $f(x)$ is square-free and has only irreducible factors of degree i
    if $deg(f) = i$ **then**
        Return $\{f\}$.
    end if
    $h_1(x) \leftarrow 1$, $h_2(x) \leftarrow 1$.
    while $(h_1(x) == 1)$ OR $(h_2(x) == 1)$ **do**
        Pick random $g(x) \in \mathbb{Z}_p[x]$ of degree less than $f(x)$.
        $h_1(x) = gcd(f(x), g(x)^{(p^i-1)/2} - 1)$.
        $h_2(x) = f(x)/h_1(x)$.
    end while
    $List1 \leftarrow IRREDUCIBLEFACTORS(h_1(x), i, p)$
    $List2 \leftarrow IRREDUCIBLEFACTORS(h_2(x), i, p)$
    Return $List1 \cup List2$.
end procedure

Remarks: In the above descriptions, we assumed that p is an odd prime. If $p = 2$, then a small change is needed in Phase 3; however we skip the details. This method also:

1. works over any finite field (apart from $\mathbb{Z}_p$);
2. can be combined with Hensel lifting to factorize polynomials in $\mathbb{Z}_{p^k}$;
3. can be extended to factorize polynomials in several variables over a finite field.

8. Polynomial Factorization over $\mathbb{Z}$

II Quadratic Equations in Two Variables

9 Primality Testing: Before 2002 .. 61

9.1 Fermat and Mersenne primes 61
9.2 Testing Fermat's little theorem 62
9.3 Fibonacci and Lucas pseudoprimality tests 63
9.4 The Miller-Rabin Test 64

10 The Integer Factoring Problem .. 67

10.1 Trial Division and Fermat's Method 67
10.2 Pollard rho Algorithm 67
10.3 Dixon's Algorithm 69

11 Primality Testing: The AKS algorithm 73

11.1 A Polynomial Identity 73
11.2 The Algorithm 73
11.3 Correctness 74

12 Quadratic Forms 77

9. Primality Testing: Before 2002

9.1 Fermat and Mersenne primes

Some of the largest primes known have been primes of the form $a^b \pm 1$. We consider two such forms.

9.1.1 Primes of the form $2^n + 1$

The 17th century mathematician Fermat noticed that all of the numbers $2+1, 2^2+1, 2^4+1, 2^8+1, 2^{16}+$ are prime and he conjectured that all the numbers $2^{2^n}+1$ are prime. But this turned out to be false. Euler showed in the 1700s that $2^{32}+1$ is divisible by 641, and modern computation has revealed only composite numbers so far among numbers of the form $2^{2^n}+1$. Indeed, it is possible that the sequence contains no further primes at all, although this hasn't been proved either. Numbers of the form $2^{2^n}+1$ are called Fermat numbers and if they are prime, they are called Fermat primes.

Let us see why it is necessary that n itself should be a power of 2 for 2^n+1 to be a prime.

Claim 15 If 2^n+1 is a prime, then n is a power of 2.

Proof. Suppose for contradiction that $n = ab$, where $a > 1$ is odd. Then we have: $2^n+1 = 2^{ab}+1 = (2^b)^a+1$, which is divisible by 2^b+1, because more generally if a is odd, then x^a+1 is divisible by $(x+1)$ (using the remainder theorem for polynomials). This contradicts the assumption that 2^n+1 is prime. Therefore n cannot have any odd divisors and must be a power of 2. ■

A factor of $2^{32}+1$: How did Euler find that 641 was a factor of $2^{32}+1$? Here's an observation: suppose that $p|2^{2^n}+1$. Then by considering the order of 2 with respect to p, we note that 2^{n+1} must divide $p-1$, i.e. $p \equiv 1 \pmod{2^{n+1}}$. Thus it is sufficient to look for such factors. For $n = 5$, it is sufficient to look for numbers of the form

$64k+1$. We can speculate that Euler tested small divisibility for small values of k and found a hit for $k=10$.

Let's prove that 641 does divide $2^{32}+1$. We have: $10\times 2^6 \equiv -1 \pmod{641}$, that is: $5\times 2^7 \equiv -1 \pmod{641}$. Raising both sides to the fourth power, we obtain: $625\times 2^{28} \equiv 1 \pmod{641}$. Since $625 \equiv -16 \pmod{641}$, we obtain: $-2^{32} \equiv 1 \pmod{641}$ which is the desired claim.

9.1.2 Primes of the form 2^n-1

Mersenne primes are primes of the form 2^p-1 where p is a prime. As in the case of Fermat primes, let us see why the restriction on the exponent is necessary.

Claim 16 If 2^n-1 is prime, then n is prime.

Proof. Suppose for contradiction that $n=ab$ where $a,b>1$. Then $2^n-1=2^{ab}-1$ is divisible by 2^a-1 which is a proper divisor of 2^n-1. This follows as in the case of Fermat primes, from the more general divisibility of $x^{ab}-1$ by x^a-1 using the remainder theorem. Thus, we obtain a contradiction if n is composite and hence n must be prime. ■

The largest explicit primes that we know are and have been Mersenne primes, and the reason for this is that there is an algorithm running in time $\tilde{O}(p^2)$ time to check whether 2^p-1 is prime.

Some large values of p such that 2^p-1 is prime: 13466917 (record-holder in 2001),32582657 (record-holder in 2006), 57885161 (record-holder in 2013), 82589933 (current record-holder). These primes are also the record-holders for the largest known prime for those years.

9.2 Testing Fermat's little theorem

A natural candidate for primality test is Fermat's little theorem, i.e. we can check whether the relation holds for some values of a: $a^n \equiv a \pmod{n}$. If n is prime, then the above congruence should hold for every a. What if n is composite? For "most" composites n, there will be "many" values of a for which the above congruence will fail to hold. Unfortunately though, there do exist composite numbers n for which the above relation holds for every a. Such numbers are called Carmichael numbers and the smallest of them is 561.

Thus, a direct testing of Fermat's little theorem is insufficient as a primality test, nevertheless it is still at the basis of the two well-known primality testing algorithms: the Miller-Rabin algorithm (Section 7.4) and the AKS algorithm (Chapter 9).

Further, Fermat's little theorem can still show up the compositeness of many composite numbers, which we make precise in the following claim.

Claim 17 Suppose that n is a composite number and let $A=\{\mathbb{Z}_n^* | a^{n-1} \equiv 1 \bmod n\}$. Then $A=\mathbb{Z}_n^*$ or $|A| \le |\mathbb{Z}_n^*|/2$.

If n is such that $|A| \le \mathbb{Z}_n^*/2$, then picking random $a \in \mathbb{Z}_n \setminus \{0\}$ and applying the Fermat's little theorem test will, with probability at least $1/2$, show that n is composite.

To prove Claim 17, we observe that the set A is a subgroup of $\mathbb{Z}^n*$ and combining this with Lagrange's theorem, the claim follows.

9.3 Fibonacci and Lucas pseudoprimality tests

The second primality testing idea we look at is also a pseudoprimality test, i.e. every prime will pass the test, but some composites may also pass. In practice though, such tests can be fast and can be combined with other tests (like Fermat's little theorem).

The Lucas sequence $U_n(a,b)$ is defined as:

$$U_0 = 0, U_1 = 1, U_n = aU_{n-1} + bU_{n-2};$$

For each fixed value of $a,b \in \mathbb{N}$, we get a sequence of natural numbers. When $a=1,b=1$, we get the Fibonacci sequence F_n.

The polynomial $f_{a,b}(x) = x^2 - ax - b$ is called the characteristic polynomial of $U_n(a,b)$. We shall henceforth assume that a,b are fixed and write $f(x), U_n$ in place of $f_{a,b}(x), U_n(a,b)$.

Claim 18 Let α, β be the roots of $f(x)$ and $d = a^2 - 4b$. If $d \neq 0$, then

$$U_n = \frac{\alpha^n - \beta^n}{\sqrt{d}}.$$

Claim 19 Let $M = \begin{bmatrix} a & b \\ 1 & 0 \end{bmatrix}$.

Then

$$M^n = \begin{bmatrix} U_n & bU_{n-1} \\ U_{n-1} & bU_{n-2} \end{bmatrix}.$$

In particular, we can compute U_n modulo m in time $poly(\log n, \log m)$ by using repeated squaring on the matrix M.

We also note that the characteristic polynomial of M is also $f(x)$, so that $M^2 - aM + bI_2 = 0$.

Let $r = \left(\frac{d}{p}\right)$, i.e. $r=1$ if d is a square modulo p and $r=-1$ otherwise.

Theorem 9.1 $U_{p-r} \equiv 0 \pmod{p}$.

In particular, for the Fibonacci sequence $F_n = U_n(1,1)$, we have:

Corollary 9.1 If n is a prime, then $n | F_{n-1}$ if $n \equiv \pm 1 \pmod 5$ and $n | F_{n+1}$ if $n \equiv \pm 2 \pmod 5$.

The Lucas primality test: Fix a Lucas sequence $U(a,b)$. Given an integer n, find $r = \left(\frac{d}{n}\right)$; this is easier when d is a prime $\equiv 1 \bmod 4$. Then find $U_{n-r} \bmod n$; if it is

zero, then n passes the test. When the sequence is F_n, the test is called Fibonacci pseudo-primality test.

Proof. **of Theorem 9.1:** We consider the cases $r = 1$ and $r = -1$ separately.

Case 1: $r = 1$

We work in $\mathbb{Z}_p$. Let $c \in \mathbb{Z}_p$ be such that $c^2 = d$. Thus, both α, β are elements of $\mathbb{Z}_p^*$ and we have $\alpha^{p-1} = \beta^{p-1} = 1$; hence $U_{p-1} = \dfrac{\alpha^{p-1} - \beta^{p-1}}{c} = 0$.

Case 2: $r = -1$

Now we work in $\mathbb{Z}_p[x]$ modulo $(x^2 - d)$.

We have $x^{p-1} = (x^2)^{(p-1)/2} = d^{(p-1)/2} = -1$. Now $(a+x)^p = a^p + x^p = a + x^p = a - x$; hence $(a+x)^{p+1} = a^2 - x^2 = a^2 - d = 4b$.

Similarly, $(a-x)^p = a + x$ and $(a-x)^{p+1} = a^2 - x^2 = 4b$.

Thus, $(a+x)^{p+1} = (a-x)^{p+1} \pmod{(x^2-d)}$, i.e. $U_{p+1} = 0$.

This completes the proof of Theorem 9.1. ■

Remarks:

1. The calculations for U_{p-r} (or for V_{p-r}) essentially come from the identity $(x+a)^p = x^p + a$ in $\mathbb{Z}_p[x]$, where we further reduced the term x^p modulo some quadratic polynomial. Writing $F(x) = (x+a)^p$ and $G(x) = x^p + a$, the Lucas test verifies that $F(x) = G(x)$ modulo some quadratic polynomial. This identity, which is true only for primes, is the starting point of the AKS algorithm, which checks that $F(x) = G(x)$ modulo $h(x)$ for a bunch of polynomials $h(x)$.
2. It appears that we do not know any composite $n \equiv \pm 2 \pmod 5$, which passes both the Fibonacci test as well as the test $2^{n-1} \equiv 1 \pmod n$; thus, performing just these two tests should detect composites effectively for such values.
   If you find such a composite n which does pass both tests, you win \$620 with \$500 from Selfridge, and \$20 from Pomerance. If you prove that no such composite exists, you stilll win the same amount, with \$500 from Pomerance, and \$20 from Selfridge. The remaining \$100 comes from Wagstaff in both cases.

9.4 The Miller-Rabin Test

The Miller-Rabin test is a randomized primality test that runs in polynomial time and it was the first of this kind. It is based on the idea that if n has at least two distinct prime factors p, q, then there are at least four distinct square-roots for 1 in $\mathbb{Z}_n$, whereas if n is prime, then there are exactly two square-roots, namely $-1, 1$, in $\mathbb{Z}_n$.

We also know that if n is prime, then for $a < n$, we must have: $a^{n-1} \equiv 1 \pmod n$. Thus, the idea is to consider the numbers $a^{(n-1)/2}, a^{(n-1)/4}, \ldots$, until we find a number c which is not equal to 1. If $c \neq -1$, then we know that n is not prime.

However, it may also happen that $c = -1$ for composites n. The claim however is that $c \neq -1$ with sufficiently large probability if n is composite and if a is a random element of $\mathbb{Z}_n$.

We first describe the algorithm in detail.

Algorithm 10 Miller-Rabin Algorithm for primality testing

procedure MILLER-RABIN TEST(n)
  If $n > 2$ is even, return COMPOSITE.
  If n is a perfect power, return COMPOSITE.
  Find t, m such that $n - 1 = 2^t m$ with $t \geq 1$, m odd.
  Pick random $a \in \{2, \ldots, n-1\}$.
  If $gcd(a, n) \neq 1$, return COMPOSITE.
  $b \leftarrow a^m$, $i \leftarrow 0$, $c \leftarrow -1$.
  while $i \leq t$ AND $b \neq 1$ **do**
    $c \leftarrow b$, $b \leftarrow b^2$. ▷ Calculations in $\mathbb{Z}_n$
    $i \leftarrow i + 1$.
  end while
  if $b \neq 1$ **then**
    return COMPOSITE.
  end if
  if $c \neq -1$ **then**
    return COMPOSITE.
  end if
  Return PRIME.
end procedure

9.4.1 Analysis of time complexity

Step 3: To check if n is a perfect power, it is sufficient if n is a kth power for $2 \leq k \leq \log_2 n$. For every fixed k, this can be done by binary search. Thus the time complexity of this step is at most $O(\log^2 n)$ (excluding the complexity of the arithmetic operations involved).

Step 4: The values of t, m can be found in $O(\log_2 n)$ time.

Step 8: The number of iterations is at most $t \leq \log_2 n$.

The other steps have time complexity $O(1)$, thus the overall time complexity is $O(\log^2 n)$ times the complexity of arithmetic operations in $\mathbb{Z}_n$, i.e. $\tilde{O}(\log^3 n)$.

9.4.2 Analysis of correctness probability

The main claim is that the Miller-Rabin algorithm is correct with probability at least $1/2$ on every input.

Claim 20
- If n is prime, then the algorithm will return PRIME.
- If n is composite, then the algorithm will return COMPOSITE with probability at least $1/2$.

Proof. If n is prime, then consider the values of c, b at the end of the while loop. If

the while loop terminates because $b = 1$, then the value of b must be equal to 1 and c is a square-root of 1 in $\mathbb{Z}_n$. Further $c \neq 1$ since the algorithm terminates the first time that b becomes 1. Thus $c = -1$ and the algorithm goes to step 18.

If the while loop terminates because $i = t$, then by Fermat's little theorem, the value of b must still be 1 at the end of the while loop, because $a^{m2^t} = a^{p-1} \equiv 1 \pmod{p}$. As before, the algorithm goes to step 18.

Now, suppose that n is composite. If n is a perfect power of a prime, then the algorithm outputs COMPOSITE in step 3. Thus, suppose that n is composite and has at least two prime factors. Let $A = \{s \in \mathbb{Z}_n^* | s^{n-1} = 1\}$. If $|A| \leq |\mathbb{Z}_n^*|/2$, then with probability at least 1/2, the algorithm will return COMPOSITE in line 13. Otherwise, using Claim 17, we deduce that $A = \mathbb{Z}_n^*$.

Let $r \geq 4$ be the number of square-roots of unity in $\mathbb{Z}_n$. For $d \in \mathbb{N}$, we define the set $S_d = \{r \in \mathbb{Z}_n^* : r^d = 1\}$. Note that S_d is a subgroup of $\mathbb{Z}_n^*$ and also that $S_{n-1} = A = \mathbb{Z}_n^*$.

We also have: if $d = d_1 d_2$ such that $gcd(d_1, d_2) = 1$, then S_d is isomorphic to $S_{d_1} \times S_{d_2}$. Thus, we get: $\mathbb{Z}_n^* = S_{n-1}$ is isomorphic to $S_{2^t} \times S_m$.

Consider the directed graph $G = (V, E)$ where $V = S_{2^t}$ and $E = \{(a, a^2) | a \in S_{2^t}\}$, with calculations modulo n. Then the underlying graph of G is a tree (with a self-loop at 1); if we fix 1 as the root, then it is a rooted tree T with the parent of vertex a being a^2. Also by our deduction that $\mathbb{Z}_n^*$ is isomorphic to $S_{2^t} \times S_m$, we find that the set $\{a^m : a in \mathbb{Z}_n^*\}$ is equal to V.

Thus, in terms of the tree T, the algorithm picks a random vertex in V and traverses up the tree until it reaches the root. For an element $a \in V$, let $T(a)$ denote the subtree rooted at a. To show that the algorithm returns COMPOSITE with probability at least 1/2, we must show that $|T(-1)| \leq |V|/2$.

The root vertex has $d-1$ children, let them be $a_1 = -1, a_2, \ldots, a_d$. Ever other node in the tree has either zero children (if it is not a square in $\mathbb{Z}_n$) or d children (if it is a square in $\mathbb{Z}_n$). Since $d \geq 4$, if we show that the height of $T(a_i)$ is greater than or equal to the height of $T(-1)$ for each $i \geq 2$, this would imply that $|T(-1)| \leq |V|/3$.

Let $T(-1)$ have height k. That is, there exists an element $\alpha \in \mathbb{Z}_n$ such that $\alpha^{2^k} = -1$. Let $n = p_1^{e_1} \ldots p_r^{e_r}$. Let ϕ be the natural isomorphism from $\mathbb{Z}_n$ to $\mathbb{Z}_{p_1^{e_1}} \times \ldots \times \mathbb{Z}_{p_r^{e_r}}$, i.e. $\phi(x) = (x_1, \ldots, x_r)$ where $x \equiv x_i \pmod{p_i^{e_i}}$. Notice that $\phi(-1) = (-1, \ldots, -1)$, whereas for $i \geq 2$, $\phi(a_i)$ is a tuple consisting of 1s and -1s, with at least one -1 and at least one 1. Suppose that $\phi(\alpha) = (\alpha_1, \ldots, \alpha_r)$. Let β be such that $\phi(\beta)_j = \alpha_j$ whenever $\phi(a_i)_j = 1$ and let $\phi(\beta)_j = 1$ whenever $\phi(a_i)_j = -1$. Then we can observe that $\beta^{2^k} = a_i$. Thus, the height of $T(a_i)$ is greater than or equal to the height of $T(-1)$. This shows that $|T(-1)| \leq |V|/3$ and completes the proof of Claim 20.

■

10. The Integer Factoring Problem

10.1 Trial Division and Fermat's Method

The simplest method of factoring a given integer n is trial division, which is to divide by each number $a \in \{2,\ldots,n-1\}$. The time complexity of this method is $\tilde{O}(\sqrt{n})$. In later sections, we will see factoring algorithms which are asymptotically faster, but we first see an alternative method to trial division, whose worst-case complexity is however $\Theta(n)$.

A simple observation that can help factor the number 323 is that $323+1=324=18^2$. Hence, $323=18^2-1=17\times 19$. More generally, suppose that we consider the numbers $n+1^2, n+2^2, \ldots$ until we find a perfect square. Then we have: $n+a^2=b^2$ and we get the factorization $n=(b-a)(b+a)$. This idea is called Fermat's method, and it can be effictive if $n=AB$ with A,B very close to each other. For such a factorization, the corresponding a,b are $a=(B-A)/2, b=(B+A)/2$, thus the number of steps is $(B-A)/2$. The worst-case complexity of this method is $\Theta(n)$, but it may be a useful test in combination with trial division and can be stopped in case we don't find a factor after some threshold number of steps.

10.2 Pollard rho Algorithm

In 1975, J. Pollard came up with an interesting algorithm that takes $O(n^{1/4})$ time to find a factor of n (if it exists) with high probability.

There are two ideas behind the Pollard rho algorithm, which we first explain.
Lemma 21 If we pick a random sequence $a_1, a_2, \ldots, a_k$ with each a_i being independently chosen from $\{1,2,\ldots,N\}$, and if $k=4\sqrt{N}$, then the probability that there exist $i<j$ with $a_i=a_j$, is at least 0.6.

The above lemma is a generalization of the so-called birthday paradox, which is that if we pick 23 people randomly, then with probability more than $1/2$, there will be two people with the same birthday (assuming independence of birthdays). We skip the

proof of the lemma but we note that an exact expression for the desired probability is $1 - \frac{(N-1)(N-2)\ldots(N-k+1)}{N^k}$.

Lemma 22 Given a function $f : \{0,1,\ldots,N-1\} \to \{0,1,\ldots,N-1\}$ and a sequence $a_1,\ldots,a_m$ where $a_i = f(a_{i-1})$ for every $i \geq 2$, there's an algorithm which can test in $O(m)$ time whether there exist distinct i,j such that $a_i = a_j$ and further find such i,j if they exist.

Proof. The problem described in the statement is known as the *cycle detection problem* and the solution that we will describe is known as Floyd's cycle detection method.

Let j be the least index such that a_j is equal to some previous element, say a_i, and let $L = i - j + 1$. Then for every $k \geq i$, we have $a_k = a_{k+L}$, while the elements $a_1,\ldots,a_{i-1}$ appear only once in the sequence. Note that even though the sequence given to us is only the first m elements, the elements $a_{m+1},\ldots$ are also well-defined by the relation $a_i = f(a_{i-1})$.

We claim that there exists $t \leq m$ such that $a_t = a_{2t}$. Let $t = i + T$. Then $a_t = a_r$ where $r = T\%L$ and $a_{2t} = a_s$, where $s = (i+2T)\%L$. Thus, if $T \equiv i + 2T$ (modulo L), then we have $a_t = a_{2t}$, and this certainly happens when $T \equiv -i \pmod{L}$, i.e. when t is of the form $t = qL$.

Now, we describe the algorithm. For $t = 1,2,\ldots$, we consider the pair of elements a_t, a_{2t} in the tth iteration. If $a_t \neq a_{2t}$ for $t \leq m$, then we conclude that all the elements are distinct.

Otherwise, the least t for which $a_t = a_{2t}$ must be the value of L We may know find the value of i in any number of ways, for example, by considering the pairs $(a_{t-1}, a_{2t-1}), (a_{t-2}, a_{2t-2})$ etc until the values in the pair are different.

The number of iterations in this algorithm is at most $2L \leq 2m$; this completes the proof of the lemma. ■

Algorithm 11 Pollard rho algorithm

```
1: procedure POLLARD RHO(n)
2:     m ← ⌈4n^{1/4}⌉
3:     Pick a_1, r ∈ {1,2,...,n−1} uniformly at random.
4:     for i=2 to m do
5:         a_i ← a_{i−1}^2 + r.                    ▷ f(x) = x^2 + r in Z_n.
6:     end for
7:     if gcd(a_i − a_j, n) ∉ {1,n} for some i,j then
8:         Return gcd(a_i − a_j, n) as a factor.
9:     end if
10: end procedure
```

The time complexity of the algorithm is $O(m) = O(n^{1/4})$, where we use Floyd's algorithm for Step 7. Suppose that n is composite and p is the least prime factor of n. Then $p \leq \sqrt{n}$ and hence by Lemma 21, with probability at least 0.6, there exist i,j such that $a_i \equiv a_j \pmod{p}$ so that $gcd(a_i - a_j, n) \neq 1$ with probability at least 0.6.

10.3 Dixon's Algorithm

The previous algorithms for factoring were exponential-time algorithms, as they had time complexity of the form $O(n^c) = O(e^{c \log n})$. We now look at a subexponential time algorithm, of complexity $e^{O(\sqrt{\log n \log\log n})}$.

The main idea behind Dixon's algorithm is to find two numbers α, β such that $\alpha^2 \equiv \beta^2 \pmod{n}$. If α, β are also random, then we can expect that $gcd(\alpha - \beta, n)$ is a non-trivial factor of n with good probability.

How can we find such α, β? Let us see an example. Let $n = 8857$. In the calculations below are the values of some squares modulo n (the congruences written are modulo n) and their factorizations.

$$95^2 \equiv 168 = 2^5 \times 3 \times 7 \tag{10.1}$$
$$97^2 \equiv 552 = 2^3 \times 3 \times 23 \tag{10.2}$$
$$107^2 \equiv 2512 = 2^5 \times 3^4 \tag{10.3}$$
$$173^2 \equiv 3358 = 2 \times 23 \times 73 \tag{10.4}$$
$$206^2 \equiv 7008 = 2^5 \times 3 \times 73 \tag{10.5}$$
$$742^2 \equiv 1430 = 2 \times 5 \times 11 \times 13. \tag{10.6}$$

Consider the equations (8.2),(8.3),(8.4),(8.5). Multiplying all of them, we obtain:

$$97^2 \times 107^2 \times 173^2 \times 206^2 \equiv 2^{14} \times 3^6 \times 23^2 \times 73^2 \pmod{n}. \tag{10.7}$$

Thus, we have $\alpha^2 \equiv \beta^2 \pmod{n}$, where $\alpha = 97 \times 107 \times 173 \times 206 \equiv 768 \pmod{n}$ and $\beta = 2^7 \times 3^3 \times 23 \times 73 \equiv 1289 \pmod{n}$. We find $gcd(\alpha - \beta, n) = 521$ which is a divisor of n.

Now we explain the idea. Firstly, we compute several random squares modulo n. We then maintain a list of those which have prime factors only in $\{p_1, \ldots, p_k\}$, where p_i is the ith prime number and where the threshold k is fixed in advance. We also maintain the list of their corresponding factorizations.

Let a number in the list have the factorization $p_1^{e_1} \ldots p_k^{e_k}$. Consider the exponent vector $(e_1, \ldots, e_k)$. A key idea is that if there are at least $k+1$ such exponent vectors, then there must be a subset of them whose sum is an even number in every coordinate. This is because the vector space $\mathbb{F}_2^k$ can have at most k linearly independent vectors.

We then multiply the corresponding relations; this gives us on the RHS a number of the form $p_1^{2f_1} p_2^{2f_2} \ldots p_k^{2f_k}$, which is a square. Thus, we can set $\beta = p_1^{f_1} \ldots p_k^{f_k}$. The linear relation among the exponent vectors (modulo 2) can be found by Gaussian elimination.

We know describe the algorithm in detail.

Analysis of running time and success probability:

Algorithm 12 Dixon's algorithm

1: **procedure** DIXON(n)
2: Fix B, m ▷ $B = m = \lceil e^{4\sqrt{\log n \log\log n}} \rceil$.
3: Find $P_B = \{p_1, \ldots, p_k\}$, the set of primes in $\{1, 2, \ldots, B\}$.
4: Pick $a_1, \ldots, a_m \in \{1, 2, \ldots, n-1\}$ uniformly at random.
5: $L \leftarrow \emptyset$, $I \leftarrow \emptyset$.
6: **for** i=2 to m **do**
7: $b_i \leftarrow a_i^2$. ▷ In $\mathbb{Z}_n$.
8: Divide b_i by each prime in P_B.
9: **if** b_i is B-smooth **then**
10: $v_i \leftarrow (e_{i,1}, \ldots, e_{i,k})$, where $b_i = \prod_{j=1}^{k} p_j^{e_{i,j}}$.
11: Add (a_i, b_i, v_i) to L; add i to I.
12: **end if**
13: **end for**
14: Find $T \subseteq I$ such that $\sum_{i \in T} v_i \equiv (0, 0, \ldots, 0) \pmod 2$. ▷ $|T| \leq k+1$.
15: $\alpha \leftarrow \prod_{i \in T} a_i$; $\beta \leftarrow \prod_{j=1}^{k} p_j^{\frac{\sum_{i \in T} e_{i,j}}{2}}$. ▷ $\alpha^2 \equiv \beta^2 \pmod n$
16: Find $n_1 = gcd(\alpha - \beta, n)$, $n_2 = n/n_1$ as possible non-trivial factors of n.
17: **end procedure**

We first analyze the running time of in terms of n, B, m without fixing the values of B, m. We do not explicitly include the cost of arithmetic operations in $\mathbb{Z}_n$ which we know to be $\tilde{O}(\log^2 n)$.

- **Lines 1-5:** $O(m + B)$
- **Lines 6-13:** $O(mB)$
- **Line 14:** $O(k^3) = O(B^3)$ via Gaussian elimination; it turns out that this can be improved to $O(B^2)$ by using the sparseness of the matrix.
- **Lines 15,16:** $O(k + \log n) = O(B + \log n)$

We thus find that the total time complexity is $O(mB + B^2)$ with some additional $poly(\log n)$ factors. Next, we need to fix m, B as functions of n.

The probability that the algorithm succeeds is related to the probability of finding at least $k+1$ numbers among $a_1, \ldots, a_m$ which are B-smooth, and further the probability that the final congruence $\alpha^2 \equiv \beta^2 \pmod n$ gives us a non-trivial factor of n. We'll only focus on the first part.

Let $S(n, B)$ denote the number of B-smooth numbers in $\{1, 2, \ldots, n\}$. Then the expected number of random elements we must pick in order to find $k+1$ numbers that are B-smooth is $\dfrac{(k+1)n}{S(n, B)}$. Thus, the choice of m will be a constant multiple of the above expression. We also need an estimate on $S(n, B)$ which is given below.

Lemma 23 If $B = n^{1/u}$, then $S(n, B) \sim \dfrac{n}{u^{u+o(1)}}$.

Writing $B = n^{1/u}$ and applying the above expression, we get the running time in terms of n, u to be:

$$T(n, u) = \tilde{O}\left(u^{u+1} n^{2/u}\right).$$

Now we can find u that minimizes $T(n,u)$ by taking logarithms and then differentiating with respect to u. This gives us: $\log u - \frac{\log n}{u^2} \sim 0$, so that $u \sim \sqrt{\frac{\log n}{\log\log n}}$. Thus, we get the running time to be $T(n) = \tilde{O}\left(e^{2\sqrt{\log n \log\log n}}\right)$. This completes our analysis of Dixon's algorithm.

11. Primality Testing: The AKS algorithm

In 2002, Manindra Agrawal, Neeraj Kayal and Nitin Saxenah came up with the first (and only) known deterministic polynomial time algorithm. It is now commonly referred to as the AKS algorithm.

11.1 A Polynomial Identity

The first key ingredient in the AKS algorithm is the following observation.

Lemma 24 Let $a \in \mathbb{Z}_n^*$. Then n is prime if and only if

$$(x+a)^n = x^n + a \tag{11.1}$$

holds in $\mathbb{Z}_n[x]$.

Proof. $\Rightarrow$: Let $n = p$ be prime. Then in $\mathbb{Z}_p[x]$, we have: $(x+a)^p = x^p + \sum_{i=1}^{p-1} \binom{p}{i} + a^p = x^p + a$. The last equality follows from the fact that p divides $\binom{p}{i}$ for $1 \leq i \leq p-1$ and from the fact that $a^p \equiv a \pmod{p}$.

$\Leftarrow$: Let n be composite with a prime factor p and let p^k be the largest power of p that divides n. Then the coefficient of x^p in $\binom{n}{p}$ is equal to $\frac{n(n-1)\ldots(n-p+1)}{p!}$ which is divisible by p^{k-1} but not by p^k. Thus this coefficient is non-zero in $\mathbb{Z}_n$, and $(x+a)^n \neq x+a$. ■

11.2 The Algorithm

We now describe the AKS algorithm. The polynomial computation is in $\mathbb{Z}_n[x]$.

Algorithm 13 AKS algorithm

1: **procedure** AKS(n)
2: Test whether n is a perfect power. If yes, return COMPOSITE.
3: Find $r \leq \lceil 16\log^5 n \rceil$ such that $ord_r(n) > 4\log^2 n$.
4: **for** $i = 1$ to r **do**
5: If $gcd(i,n) > 1$, return COMPOSITE.
6: **end for**
7: $k \leftarrow \lfloor 2\sqrt{r}\log n \rfloor$.
8: **for** $a = 1$ to k **do**
9: If $(x+a)^n \not\equiv x^n + a \bmod (x^r - 1)$ return COMPOSITE.
10: **end for**
11: Return PRIME.
12: **end procedure**

11.2.1 Running Time:

Step 3 can be done as follows: For each $r \leq 16\log^5 n$, do the following: for each $d < 4\log^2 n$, check whether $n^d - 1$ is not divisible by r. If it is not divisible for every d, the corresponding r is chosen. The time complexity of this step is $\tilde{O}(\log^8 n)$.

The other main contribution to the running time is lines 8-10; the number of iterations is $O(\log^{3.5} n)$ and each computation can be done in $\tilde{O}(\log^6 n)$ time (since $deg(x^r - 1) = O(\log^5 n)$). Thus the total time complexity is $\tilde{O}(\log^9 n)$.

11.3 Correctness

We will show that the algorithm returns PRIME if and only if n is prime. We will also prove why r exists as in step 3.

If n is prime, then the algorithm clearly does not return COMPOSITE in lines 2,5. It also does not return COMPOSITE in line 9 because of Lemma 24.

Suppose now that n is composite. If the algorithm does not return COMPOSITE in line 2, then n must have at least two distinct prime factors; let p denote the least prime factor of n.

Definition 11.1 Let r be a fixed positive integer. Let $m \in \mathbb{N}$, $f(x) \in \mathbb{Z}_p[x]$. We say that the pair $(m, f(x))$ is **introspective** (with respect to p) if

$$f(x)^m \equiv f(x^m) \bmod (p, (x^r - 1)). \tag{11.2}$$

We denote by IP the set of all introspective pairs (with respect to p).

Examples:

- $(1, f(x))$ for every $f(x), p, r$;
- $(p, f(x))$ for every $f(x), p, r$;
- (m, x) for every m, p, r;
- $(561, x+1)$ for $p = 3, 7, 11$ and $r = 4$.

The connection of this definition to the algorithm is that in line 9, we are checking

whether the pair $(n, x+a)$ is introspective.

Introspective pairs satisfy the following two multiplicative properties.

Claim 25 (a) Let $(m_1, f(x))$ and $(m_2, f(x))$ be introspective. Then $(m_1 m_2, f(x))$ is also introspective.

(b) Let $(m, f(x))$ and $(m, g(x))$ be introspective. Then $(m, f(x)g(x))$ is also introspective.

We can now explain the broad proof strategy.

11.3.1 The Proof Strategy

The strategy behind the proof of correctness is the following: We will show that if the algorithm does not return COMPOSITE in lines 8-10, then there are "many" introspective pairs by using the multiplicative property.

On the other hand, we will argue that if there are too many of them, then for a suitable polynomial of some degree m, there are more than $m+1$ roots. These roots will correspond to the polynomials in the introspective pairs, and the polynomial will be in a finite field, which gives the desired contradiction.

11.3.2 The Proof

Let $I = \{n^i p^j | i, j \geq 0\}$ and let $P = \{\prod_{a=1}^{k} (x+a)^{e_a}\}$, where the exponents e_a range over all values in $\mathbb{N} \cup \{0\}$.

The following observation comes from the multiplicative properties of introspective pairs and also uses the fact that the algorithm has not returned COMPOSITE in lines 8-10.

Observation 26 For every $m \in I$ and every $f(x) \in P$, the pair $(m, f(x))$ is introspective.

The sets I, P are infinite; we first define some finite subsets of them. We define G to be the set obtained by considering the values of I modulo r. Formally, $G = \{a \in \mathbb{Z}_r : a \equiv i \text{ mod r for some } i \in I\}$. Note that G is a subgroup of $\mathbb{Z}_r^*$ under multiplication.

Let $h(x)$ be an irreducible factor of $x^r - 1$ of largest degree. We define R to be the set obtained by considering the values of P modulo $h(x)$. Formally, $R = \{f(x) \in \mathbb{Z}_n[x]/(h(x)) : f(x) \equiv g(x) \text{mod } h(x) \text{ for some } g(x) \in P\}$. We note that R is a subgroup of $(\mathbb{Z}_n[x]/(h(x)))^*$.

We now make three claims about the sizes of $|G|, |R|$.

Claim 27 Let $t = |G|$. Then $t > 4\log^2 n$.

Claim 28 $|R| \geq \binom{t+k-2}{k-1}$.

Claim 29 $|R| \leq \left(\frac{n^2}{2}\right)^{\sqrt{t}}$.

Assuming the claims, the contradiction follows from Claims 28 and 29 after substituting the known bounds on t, r, k. We skip this calculation and instead give sketches of the proofs of the claims.

Proof. of Claim 27. Let $d = ord_r(n)$. Then the elements $1, n, \ldots, n^{d-1}$ are all distinct modulo r and from line 3, we have: $ord_r(n) > 4\log^2 n$. This proves the claim. ■

Proof. of Claim 28. Consider the polynomials in P of degree less than t. The number of such polynomials is equal to $\binom{t+k-2}{k-1}$. All these polynomials are distinct modulo $h(x)$ if $deg(h(x)) \geq t$. Thus it suffices to prove that $deg(h(x)) \geq t$. This can be done by considering the minimal polynomial of an element in $\mathbb{Z}_r$ that generates G, but we skip the details. ■

Proof. of Claim 29. Let $\hat{I} = \{n^i p^j | 0 \leq i \leq j \leq \lfloor\sqrt{t}\rfloor\}$. We have $|\hat{I}| > t$ and therefore there exist distinct $m_1, m_2 \in \hat{I}$ such that $m_1 \equiv m_2 \pmod{r}$.

Let $f(x)$ be an arbirary polynomial in R. Since $f(x)$ is introspective, we have:

$$f(x)^{m_1} \equiv f(x^{m_1})(\bmod\ h(x)) \tag{11.3}$$
$$and f(x)^{m_2} \equiv f(x^{m_2})(\bmod\ h(x)) \tag{11.4}$$
$$\tag{11.5}$$

Also, since $m_1 \equiv m_2 \pmod{r}$, we have $x^{m_1} \equiv x^{m_2} \pmod{x^r - 1}$, and hence: $x^{m_1} \equiv x^{m_2} \pmod{h(x)}$.

Thus, we find that $f(x)^{m_1} - f(x)^{m_2} \equiv 0 \pmod{h(x)}$. Now consider $F = \mathbb{Z}_p[x]/(h(x))$ which is a field. Seeing $f(x)$ as an element of F, we have that $f(x)$ is a root of the polynomial $Y^{m_1} - Ym_2$.

Since F is a field, the polynomial $Y^{m_1} - Y^{m_2}$ has at most $max(m_1, m_2)$ roots. Thus $|R| \leq max(m_1, m_2) \leq (np)^{\sqrt{t}} \leq \left(\frac{n^2}{2}\right)^{\sqrt{t}}$. ■

With the proof of the claims, we have shown that if n is composite, then the algorithm will return COMPOSITE.

We now consider the problem of showing that there exists $r < 16\log^5 n$ such that $ord_r(n) > 4\log^2 n$. For this, we the following lemma is useful.

Lemma 30 Let $n \geq 1$ be a natural number. Then $lcm(1, 2, \ldots, n) \geq 2^{(n-1)/2}$.

Proof. Let $L = lcm(1, 2, \ldots, 2n+1)$, and let $I = \int_0^1 x^n(1-x)^n dx$. Then $0 < I < \frac{1}{4^n}$ and LI is a positive integer. Thus, $L > 4^n$, from which the statement in the lemma may be obtained. ■

Let $D = \lfloor 4\log^2 n \rfloor$ and $R = \log^5 n$. Suppose that every $r \in \{1, 2, \ldots, R\}$ divides some number $n^d - 1$ for $d \leq D$. Then $LCM(1, 2, \ldots, R)$ must divide $(n-1)(n^2-1)\ldots(n^D - 1)$. Thus, $LCM(1, 2, \ldots, R) \leq n^{\frac{D^2}{2}}$. Applying Lemma 30 for the LHS and substituting for D, we may obtain the desired result.

This completes the proof sketch of the correctness of the AKS algorithm. ■

12. Quadratic Forms