Point Counting On Genus 2 Curves

(Thesis format: Monograph)

Javad Doliskani

Graduate Program in Computer Science

A thesis submitted in partial fulfillment
of the requirements for the degree of
Master of Science

The School of Graduate and Postdoctoral Studies
The University of Western Ontario
London, Ontario, Canada

Abstract

For cryptographic purposes, counting points on the jacobian variety of a given hyperelliptic curve is of great importance. There has been several approaches to obtain the cardinality of such a group, specially for hyperelliptic curves of genus 2. The best known algorithm for counting points on genus 2 curves over prime fields of large characteristic is a variant of Schoof's genus 1 algorithm. Following a recent work of Gaudry and Schost, we show how to speed up the current state of the art genus 2 point counting algorithm by proposing various computational improvements to its basic arithmetical ingredients.

Keywords. Point counting, Hyperelliptic curves, Schoof's algorithm, Elliptic curves

Contents

Chapter 1

Introduction

A one way function is, roughly speaking, a function that is easy to compute and hard to invert. The theory of one way functions forms the foundations of the modern cryptography [55, 38]. Given a certain finite cyclic group G, and a generator g of G, the function

$$\begin{array}{rccc} \varphi : & \mathbb{N}_{\leq |G|} & \longrightarrow & G \\ & x & \longmapsto & g^x \end{array}$$

is expected to be a one way function. The problem of computing φ^{-1} is called the *discrete logarithm problem* (DLP) to the base g in G. The security of many cryptographic schemes is based on DLP [43, 68]. The group G is suitable for this purpose if the DLP is hard in G, the elements of G are easily and efficiently representable, computation in G is fast, and the order of G can be computed efficiently. As a general assumption, it is always assumed that G has a non-smooth order, i.e. its order is a prime or a product of a large prime and some other very small factors; otherwise it is vulnerable to the Pohlig-Hellman attack [71].
The traditional candidates for the group G are multiplicative subgroups of finite fields. The arithmetic on finite fields is fast, but on the other hand, there are subexponential algorithms for computing discrete logarithm in these fields [61]. Another candidate for the group G, proposed by Koblitz [41] and Miller [62] independently in 1985, is a cyclic subgroup of the group of points of an elliptic curve over a finite field. The advantage of systems based on such groups is that there is no known subexponential algorithm for DLP in these groups, for example the index-calculus algorithm does not apply to elliptic curves except to those of very special structures [87, 62]. This leads to equally secure systems with smaller parameter sizes. The primary disadvantage of elliptic curve systems is that the addition of points is a relatively costly operation which is now reasonably fast due to the advances made in both theory and implementations, e.g. see [32] and the references therein. Furthermore, Koblitz [45, 47] proposed a special family of curves allowing fast arithmetic. Computing the order of the group of a given elliptic curve is called point counting. There are polynomial time algorithms for counting points on elliptic curves, e.g. [80].
As another candidate for group G suitable for cryptosystems, Koblitz [44] proposed the jacobian variety of a hyperelliptic curve over a finite field, which is indeed a generalization of the elliptic case. This leads to even smaller field sizes, and larger number of choices. Also, the elements of the jacobian can be represented using pairs of finite degree polynomials called the Mumford representation. According to Riemman-Roch theorem, there is a unique

integer attached to every curve, called the genus of the curve. For example, a curve is elliptic if and only if it is genus one with at least one rational point. The main drawback of the hyperelliptic curve systems is that the addition on the jacobian is generally slower than the group operation on elliptic curves, and the higher the genus goes the slower would be the arithmetic. Moreover, for large enough genera, there is a version of index-calculus method for computing the discrete logarithm in the jacobian [3, 23, 65, 29].
In terms of efficiency, genus 2 curves provide the closest hyperelliptic alternative for elliptic curve cryptosystems [69, 28, 50]. Furthermore, for an equally secure system, and the same parameter lengths, the base field for the genus 2 curve is almost twice smaller. To find secure hyperelliptic curves, there are two main approaches:

Point counting In which one tries several random curves over a given finite field until a good one is found, and then computes the cardinality of the jacobian.

CM method In which, instead of trying random curves, one starts with the endomorphism ring and vary the base field until a curve with a good jacobian order is found, e.g. [17, ch. 18], and [105].

There are other methods, e.g. [90], that work under special conditions. There are also known special families of curves with good jacobian order, e.g. [42]. The complex multiplication method is efficient, but curves found by this method have more special structures, and it is not generally known if they are stronger or weaker than general curves. In this thesis, we investigate point counting on genus 2 curves.
Various approaches have been proposed for point counting on hyperelliptic curves, from which we can name (**i**) Schoof's algorithm which is the generalization of the Schoof's algorithm for elliptic curves, and it is polynomial time. (**ii**) Kedlaya's algorithm [39] which is exponential time in general, but efficient in small characteristics. (**iii**) Satoh's algorithm [75] which is polynomial time in small characteristic, but exponential in general. The best of the above approaches, for large characteristics, is the Schoof's algorithm. The generalization is essentially due to Pila [70]. The Schoof's algorithm for hyperelliptic curves of genus 2 was then presented in Gaudry and Harley [26], and Gaudry and Schost [27, 30]. In particular, the following is the timeline for point counting on genus 2 curves.

- Gaudry and Harley (2000): 126 bit Jacobian
- Gaudry and Schost (2004): 164 bit, secure Jacobian
- Sutherland (2007): 188 bit, secure Jacobian (works for special curves)
- Gaudry and Schost (2008): 256 bit, secure Jacobian
- Gaudry and Schost (present): 256 bit, doubly-secure Jacobian

In this thesis, we show how to improve the work of Gaudry and Schost by improving the arithmetical ingredients of their algorithm both theoretically and in terms of implementation. According to their most recent work, their algorithm makes extensive use of square (or higher) root computing, modular polynomial composition, and power projection. Both modular polynomial composition, and power projection are based on matrix multiplication.

For example, to have a fast modular polynomial composition, the natural choice is the algorithm of Brent and Kung [9], which is still practically the best. So, we have proposed and implemented an algorithm for fast multiplication of matrices with integer entries of arbitrary length. This small parallel low level library, with high level interfaces, is embedded into the NTL library [85]. Consequently, the new modular composition, and power projection implementations have been embedded into NTL. For the square root computation, we have proposed a new algorithm, implemented it, and integrated it into NTL. This algorithm uses polynomial composition for computing a trace-like map that reduces the problem to the same problem over the prime field.

In this thesis, rather than an abstract general description of materials, specially the theory of elliptic and hyperelliptic curves, we have tried to conduct a concrete and intuitive approach. We have made an effort to make this document as self contained as possible with respect to the needed materials by giving clean proofs, unless the proofs are too long or there is no point in quoting other's proofs. The reader is assumed to have a basic knowledge of algebra, and polynomial arithmetic. We shall denote the asymptotic complexity bounds of polynomial multiplication, and modular polynomial composition by $O(M(n))$, and $O(C(n))$ respectively. The best known $M(n)$ is $n \log n \log\log n$ achieved by using Fast Fourier Transform (FFT) [79], and the best know $C(n)$ is $n^{1.69}$. See Section 5.3 for more details on $C(n)$.

A brief summary of the thesis follows.

To make the introduction of hyperelliptic curves, specially the Schoof's algorithm, smooth, and for the sake of completeness, Chapters 2 and 3 are dedicated to elliptic curves. A short introduction to the theory of elliptic curves over general fields, including the group law, endomorphisms, division polynomials, and torsion subgroups is given in Chapter 2. Chapter 3 includes more specific properties of elliptic curves over finite fields, and it is concluded with some point counting methods on elliptic curves. In Chapter 4, we give an introduction to the theory hyperelliptic curves of general genera. Chapter 5 consists of the description and implementation of a fast integer matrix multiplication algorithm, and it is concluded with applications of this algorithm to modular polynomial composition, and power projection. In Chapter 6, we first present various algorithms, including a new one, for computing square roots in finite field, and then extend each of those algorithms to compute k-th roots for arbitrary $k \in \mathbb{N}$. The last chapter consists of a quick review of a genus 2 Schoof algorithm, and some experimental results.

Chapter 2

Elliptic Curves

The origin of the elliptic curves goes back to the 18th century when mathematicians tried to calculate the arc length of an ellipse. This led to the study of elliptic integrals, and then elliptic functions, and the brilliant works of Weierstrass. Of course, all these were happening in the complex field $\mathbb{C}$. But the theory was extended over arbitrary fields, specially finite fields, afterwards, and today, there is a huge literature on elliptic curves and their applications. In this chapter, we give a brief introduction to the basic concepts of the theory of elliptic curves over general fields.

2.1 The Weierstrass equation and the group law

Let k be a field. The set $\mathbb{A}_k^2 = \{(x, y) \in k \times k\}$ is called the affine plane over k. Any nonconstant squarefree polynomial $f \in k[x, y]$ defines an affine plane curve C_f whose points are the zero set of f in $\mathbb{A}_k^2$. We say that C_f is defined over k. For an extension $L \supseteq k$, the zero set of f in $\mathbb{A}_L^2$ is denoted by $C_f(L)$. The projective plane over k, denoted by $\mathbb{P}_k^2$, is the set of all triples $(x, y, z) \in k^3$ with $(x, y, z) \neq (0, 0, 0)$ modulo an equivalence relation $\sim$ where $(x, y, z) \sim (x_1, y_1, z_1)$ if and only if there exists a nonzero $\mu \in k$ such that $(z_1, y_1, z_1) = (\mu x, \mu y, \mu z)$. The class of the point (x, y, z) is denoted by $(x : y : z)$. A projective plane curve is defined similar to the affine curve except that the defining polynomial $f \in k[x, y, z]$ should be homogeneous, otherwise the zero set of f in $\mathbb{P}_k^2$ is not well defined. A point $(x : y : z) \in \mathbb{P}_k^2$ is a finite point if $z \neq 0$, and a point at infinity if $z = 0$. There is a natural embedding

$$\begin{aligned} \varphi : \mathbb{A}_k^2 &\hookrightarrow \mathbb{P}_k^2 \\ (x, y) &\mapsto (x : y : 1) \end{aligned}$$

A plane curve C_f is said to be singular at a point P if $\frac{\partial f}{\partial x}(P) = \frac{\partial f}{\partial y}(P) = 0$, where the partial derivative of a polynomial is defined in the usual way, and nonsingular at P otherwise. The curve C_f is nonsingular if it has no singular points. An *elliptic curve* over k, denoted by E_k, is a nonsingular projective plane curve defined over k by a polynomial of the form $f(x, y) = y^2z + a_1xyz + a_3yz^2 - (x^3 + a_2x^2z + a_4xz^2 + a_6z^3) \in k[x, y, z]$. Thus, the points of E_k are the solution set in $\mathbb{P}_k^2$ of an equation of the form

$$E_k : \quad y^2z + a_1xyz + a_3yz^2 = x^3 + a_2x^2z + a_4xz^2 + a_6z^3 \tag{2.1}$$

with $a_1, a_3, a_2, a_4, a_6 \in k$. If we let $z = 0$ in Equation (2.1) then $x^3 = 0$, and hence $(0 : 1 : 0)$ is the only point at infinity of E_k. Therefore, all point of E_k are of the form $(x : y : 1)$, i.e. are in the finite plane, except the above point which we denote by ∞. So, the set of points on E_k is the solution set in $\mathbb{A}_k^2$ of

$$y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6 \tag{2.2}$$

together with ∞. For an extension $L \supseteq k$, the points on E_k with coordinates in L will be denoted by $E_k(L)$, i.e.

$$E_k(L) = \{\infty\} \cup \{(x, y) \in \mathbb{A}_L^2 \mid y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6\}$$

Equation (2.2) is called the generalized Weierstrass equation of E_k. If the characteristic of k is not 2 then, applying the change variables $y \mapsto y - a_1x/2 - a_3/2$, Equation (2.2) can be rewritten as

$$y^2 = x^3 + \frac{b_2}{4}x^2 + \frac{b_4}{2}x + \frac{b_6}{4} \tag{2.3}$$

where $b_2 = a_1^2 + 4a_2$, $b_4 = a_1a_3 + 2a_4$, $b_6 = a_3^2 + 4a_6$. Let us also define $b_8 = a_1^2a_6 + 4a_2a_6 - a_1a_3a_4 + a_2a_3^2 - a_4^2$ for future references. If the characteristic is also not 3 then the change of variables $x \mapsto x - b_2/7$ results in

$$y^2 = x^3 + Ax + B \tag{2.4}$$

for some constants $A, B \in k$. Equation (2.4) is called the Weierstrass equation of E_k. We shall simply use E instead of E_k when k is uniquely known from the context. Unless otherwise specified, by an elliptic curve E we shall mean an elliptic curve defined by Equation (2.4). Since E is nonsingular, the cubic $x^3 + Ax + B$ cannot have repeated roots and hence $\Delta = -4A^3 - 27B^2 \neq 0$ where Δ is the discriminant of the cubic. So, we always have $4A^3 + 27B^2 \neq 0$. Let $P = (x_P, y_P)$ and $Q = (x_Q, y_Q)$ be two points on E. Define the addition $R = (x_R, y_R) = P + Q$ as follows. For any point P, $P + \infty = P$, so for example $\infty + \infty = \infty$. For $P, Q \neq \infty$:

1. If $x_P \neq x_Q$ then $x_R = \left(\frac{y_Q - y_P}{x_Q - x_P}\right)^2 - x_P - x_Q$ and $y_R = \frac{y_Q - y_P}{x_Q - x_P}(x_P - x_R) - y_P$.
2. If $x_P = x_Q$ but $y_P \neq y_Q$ then $P + Q = \infty$.
3. If $P = Q$ and $y_P \neq 0$ then $x_R = \left(\frac{3x_P^2 + A}{2y_P}\right)^2 - 2x_P$ and $y_R = \frac{3x_P^2 + A}{2y_P}(x_P - x_R) - y_P$.
4. If $P = Q$ and $y_P = 0$ then $P + Q = \infty$.

It is not hard to prove that under the above addition, the points on E form an abelian group with ∞ as identity (e.g. [102, Sec. 2.4] or [22, Sec. 2.11]). From rule 2 and 4, and Equation (2.4) we have $-P = (x_P, -y_P)$. Figure 2.1 illustrates the geometrical view of the group law for the curve $y^2 = x^3 + x^2 - 2x$ over the real field $\mathbb{R}$ when $x_P \neq x_Q$, i.e. Case 1 of the above. The two ends of the y-axis are labelled with ∞ to show that they meet at infinity.

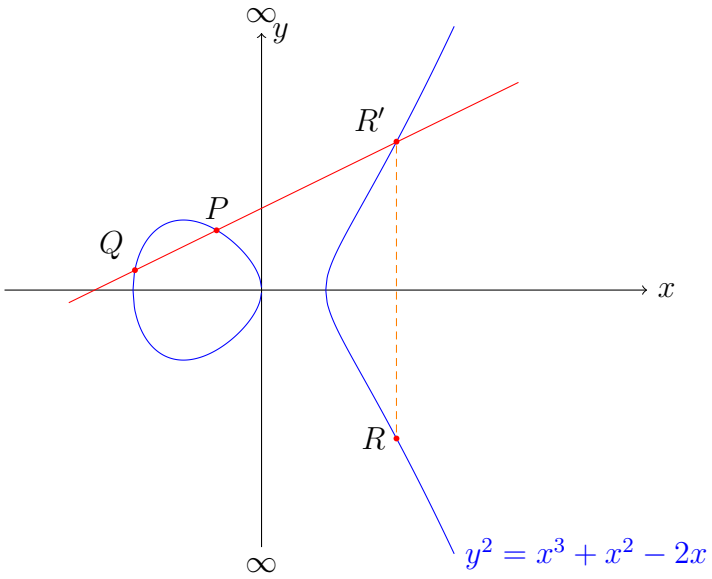

Figure 2.1: $P + Q = R$

2.2 Endomorphisms

Let E^1 and E^2 be elliptic curves defined over the field k and let $\overline{k}$ be the algebraic closure of k. An **isogeny** from E^1 to E^2 is a group homomorphism

$$\begin{aligned} \sigma : E^1(\overline{k}) &\longrightarrow E^2(\overline{k}) \\ (x, y) &\longmapsto (f(x, y), g(x, y)) \end{aligned}$$

where $f, g \in \overline{k}(x, y)$ are rational functions. Replacing y^2 by x^3+Ax+B in $f(x, y)$ we can write $f(x, y) = (a(x)+yb(x))/(c(x)+yd(x))$ where a, b, c, d are polynomials. Removing y from the denominator by multiplying $c(x) - yd(x)$ on both denominator and numerator, and using the fact that $(f(x, -y), g(x, -y)) = \sigma(x, -y) = \sigma(-(x, y)) = (f(x, y), -g(x, y))$ we have $f(x, y) = r(x)$ where $r(x)$ is a rational function in x. By a similar process $g(x, y) = ys(x)$ for some rational function $s(x)$. So, we can always write $\sigma(x, y) = (r(x), ys(x))$ for some rational functions r and s.
Let $r(x) = r_1(x)/r_2(x)$ where $\gcd(r_1(x), r_2(x)) = 1$. The degree of σ is defined to be $\max\{\deg r_1(x), \deg r_2(x)\}$ when σ is nontrivial. If $\sigma = 0$ then $\deg \sigma = 0$. A nontrivial isogeny σ is **separable** if the derivative $r'(x)$ is not the zero function.

Lemma 2.1. *Every nontrivial isogeny $\sigma : E^1(\overline{k}) \to E^2(\overline{k})$ of elliptic curves is surjective.*

Proof. See [102] for an elementary proof, and [25] for a proof for general projective curves. □

By the kernel of an isogeny σ we mean its kernel as a group homomorphism.

Proposition 2.2. *For any nontrivial isogeny σ, $\#\ker(\sigma) \leq \deg(\sigma)$ with equality if σ is separable.*

Proof. See [88, Sec. 3.4] or [102]. □

An **endomorphism** of elliptic curve E is an isogeny $\varphi : E(\overline{k}) \to E(\overline{k})$. The set of all endomorphisms of E is denoted by $\mathrm{End}(E)$. For every two endomorphisms $\alpha, \beta \in \mathrm{End}(E)$ define $(\alpha+\beta)(P) = \alpha(P)+\beta(P)$, and $(\alpha\beta)(P) = \alpha(\beta(P))$ for $P \in E$. Then it can easily be verified that $\mathrm{End}(E)$ is a ring. Two basic maps on E are

$$\begin{array}{rcl} \mathrm{id} : E(\overline{k}) & \to & E(\overline{k}) \\ P & \mapsto & P \end{array} \qquad \text{and} \qquad \begin{array}{rcl} [n] : E(\overline{k}) & \to & E(\overline{k}) \\ P & \mapsto & [n]P = P + P + \cdots + P \quad (n \text{ times}) \end{array}$$

The first one is clearly an endomorphism. Using the addition laws and a simple induction shows that the map $[n]$ is also an endomorphism. This means that $\mathrm{End}(E)$ always contains a copy of the integer ring $\mathbb{Z}$. We will show that $[n]$ is separable if and only if n is relatively prime to $\mathrm{char}(k)$. To this end, let first prove the following result.

Lemma 2.3. *Let $\varphi_1, \varphi_2 \in \mathrm{End}(E)$ such that $\varphi_i(x,y) = (r_i(x), ys_i(x))$, $i = 1,2$ where r_i, s_i are rational functions. Let $\varphi(x,y) = (r(x), ys(x))$ such that $\varphi = \varphi_1 + \varphi_2$. If $r_i'(x)/s_i(x) = c_i$ for some constants c_i, $i = 1,2$ then $r'(x)/s(x) = c_1 + c_2$.*

Proof. Let $\tau_Q(x,y) = (f,g)$ be the translation-by-Q map on E. Then it is straightforward, but lengthy, to show that $\frac{\partial f}{\partial x} + \frac{dy}{dx}\frac{\partial f}{\partial y} = \frac{g}{y}$. Let $\varphi(x,y) = (x_3, y_3)$, and let $\varphi_i(x,y) = (x_i, y_i)$, $i = 1,2$. If we let $Q = (x_1, y_1)$ then $\frac{\partial x_3}{\partial x_2} + \frac{dy_2}{dx_2}\frac{\partial x_3}{\partial y_2} = \frac{y_3}{y_2}$, and if $Q = (x_2, y_2)$ then $\frac{\partial x_3}{\partial x_1} + \frac{dy_1}{dx_1}\frac{\partial x_3}{\partial y_1} = \frac{y_3}{y_1}$. Therefore

$$\begin{aligned} r'(x) = \frac{dx_3}{dx} &= \frac{dx_1}{dx}\frac{\partial x_3}{\partial x_1} + \frac{dx_2}{dx}\frac{\partial x_3}{\partial x_2} + \frac{dx_1}{dx}\frac{dy_1}{dx_1}\frac{\partial x_3}{\partial y_1} + \frac{dx_2}{dx}\frac{dy_2}{dx_2}\frac{\partial x_3}{\partial y_2} \\ &= \frac{dx_1}{dx}\frac{y_3}{y_1} + \frac{dx_2}{dx}\frac{y_3}{y_2} = c_1\frac{y_3}{y} + c_2\frac{y_3}{y} = (c_1+c_2)s(x) \end{aligned}$$

as desired. □

Corollary 2.4. *Let $P = (x,y)$ be a point on E, and let $[n]P = (r(x), ys(x))$ for some rational functions r, s. Then $r'(x)/s(x) = n$. Therefore, the mapping $[n]$ is separable if and only if $\gcd(n, \mathrm{char}(k)) = 1$.*

Proof. The statement is clear for $n = 1$. Assume it is true for all $k < n$. Let $[n-1]P = (r_1(x), ys_1(x))$. Then $r_1'(x)/s_1(x) = n-1$. We have $(r(x), ys(x)) = [n]P = [n-1]P + P = (r_1(x), ys_1(x)) + (x,y)$, so by Lemma 2.3, $r'(x)/s(x) = n - 1 + 1 = n$. □

We will give explicit formulae for the endomorphism $[n]$ in Section 2.3. In the theory of elliptic curves over finite fields, i.e. $k = \mathbb{F}_q$, the following map plays an important role.

$$\begin{array}{rcl} \phi_q : E(\overline{\mathbb{F}_q}) & \to & E(\overline{\mathbb{F}_q}) \\ (x,y) & \mapsto & (x^q, y^q), \ \infty \mapsto \infty \end{array} \tag{2.5}$$

It is called the Frobenius map. It is, indeed, the curve version of the Frobenius automorphism $\phi_q : \overline{\mathbb{F}_q} \to \overline{\mathbb{F}_q}$. Since $a^q = a$ for every $a \in \mathbb{F}_q$, the field $\mathbb{F}_q$ is characterized by ϕ_q. Thus, applying ϕ_q to the equation of E, it is clear that $\phi_q(x,y) \in E(\overline{\mathbb{F}_q})$. We also have $P \in E(\mathbb{F}_q)$ if and only if $\phi_q(P) = P$. By the group laws defined in Section 2.1, it can be easily seen that ϕ is an endomorphism of E. Also it is clear that ϕ_q is not separable. In fact we have

Corollary 2.5. *Let E be an elliptic curve defined over $\mathbb{F}_q$. For integers a and b, not both zero, and the Frobenius endomorphism ϕ_q, the endomorphism $a\phi_q + b$ is separable if and only if* $\gcd(b, q) = 1$.

Proof. Lemma 2.3 and Corollary 2.4. □

Proposition 2.6. *Let E be an elliptic curve defined over $\mathbb{F}_q$, and ϕ_q the Frobenius endomorphism. Then 1. $\#E(\mathbb{F}_{q^n}) = \deg(\phi_q^n - 1)$, and 2. $\ker(\phi_q^n - 1) = E(\mathbb{F}_{q^n})$*

Proof. The map ϕ_q^n is equivalent to the Frobenius map on $E(\overline{\mathbb{F}_{q^n}})$. So, $(\phi_q^n - 1)(P) = 0$ if and only if $P \in E(\mathbb{F}_{q^n})$ which proves 2. By Corollary 2.5, $(\phi_q^n - 1)$ is separable hence $\#E(\mathbb{F}_{q^n}) = \#ker(\phi_q^n - 1) = \deg(\phi_q^n - 1)$ by Proposition 2.2. This proves 1. □

2.3 Division polynomials

For a positive integer n and a point P on an elliptic curve E, $[n]P$ can be computed using the repeated squaring algorithm. Another way of computing $[n]P$ is using explicit formulae expressed by division polynomials. The division polynomials for an elliptic curve defined by Equation (2.2) are defined recursively as follows.

$$
\begin{aligned}
&\psi_0 = 0 \\
&\psi_1 = 1 \\
&\psi_2 = 2y + a_1x + a_3 \\
&\psi_3 = 3x^4 + b_2x^3 + 3b_4x^2 + 3b_6x + b_8 \\
&\psi_4 = \psi_2 \cdot (2x^6 + b_2x^5 + 5b_4x^4 + 10b_6x^3 + 10b_8x^2 + (b_2b_8 - b_4b_6)x + (b_4b_8 - b_6^2)) \\
&\psi_{2m+1} = \psi_{m+2}\psi_m^3 - \psi_{m-1}\psi_{m+1}^3 && \text{for } m \geq 2 \\
&\psi_{2m} = \psi_2^{-1}\psi_m(\psi_{m-1}^2\psi_{m+2} - \psi_{m-2}\psi_{m+1}^2) && \text{for } m \geq 2
\end{aligned}
$$

where b_2, b_4, b_6, and b_8 are the values defined in Equation (2.3). Also define

$$
\begin{aligned}
\phi_m &= x\psi_m^2 - \psi_{m-1}\psi_{m+1} \\
\omega_m &= (2\psi_m)^{-1}(\psi_{2m} - (a_1\phi_m + a_3\psi_m^2)\psi_m^2)
\end{aligned}
$$

where ω_m is defined when the field k is not of characteristic 2. In the following let $R = \mathbb{Z}[a_1, \ldots, a_6, x, \psi_2^2]$.

Lemma 2.7. *If m is odd then $\psi_m \in R$. If m is even then $(\psi_2)^{-1}\psi_m \in R$.*

Proof. The lemma is clear for $m \leq 4$. Assume it is true for $\mathbb{N}_{<k}$. If k is odd, say $k = 2\ell + 1$, then $\psi_k = \psi_{\ell+2}\psi_\ell^3 - \psi_{\ell-1}\psi_{\ell+1}^3$. If ℓ is odd then $\psi_{\ell-1}\psi_{\ell+1}^3 \in \psi_2^4 R \subseteq R$ hence $\psi_k \in R$. If ℓ is even then $\psi_{\ell+2}\psi_\ell^3 \in \psi_2^4 R \subseteq R$ hence $\psi_k \in R$. If k is even, say $k = 2\ell$, then $\psi_k = \psi_2^{-1}\psi_\ell(\psi_{\ell-1}^2\psi_{\ell+2} - \psi_{\ell-2}\psi_{\ell+1}^2)$. If ℓ is odd then $\psi_{\ell-1}^2, \psi_{\ell+1}^2 \in \psi_2^2 R$ hence $\psi_k \in \psi_2^{-1}(\psi_2^2 R) = \psi_2 R$. If ℓ is even then $\psi_\ell, \psi_{\ell+2}, \psi_{\ell-2} \in \psi_2 R$ hence $\psi_k \in \psi_2^{-1}(\psi_2^2 R) = \psi_2 R$. □

A direct consequence of Lemma 2.7 is that $\psi_m^2, \phi_m \in R$ for all m. From Equation (2.3) we have $\psi_2^2 = (2y+a_1x+a_3)^2 = 4(y^2+a_1xy+a_3y)+(a_1x+a_3)^2 = 4(x^3+a_2x^2+a_4x+a_6)+(a_1x+a_3)^2 \in \mathbb{Z}[a_1,\ldots,a_6,x]$ hence $R = \mathbb{Z}[a_1,\ldots,a_6,x]$. Therefore, ψ_m^2 and ϕ_m are polynomials in x over the ring $\mathbb{Z}[a_1,\ldots,a_6]$ for all m. For a polynomial f over an arbitrary polynomial ring $A[x,y]$, let $\Lambda(f)$ denote the leading term of f as a polynomial of x.

Lemma 2.8.

$$\Lambda(\psi_m) = \begin{cases} mx^{(m^2-1)/2} & \text{if } m \text{ is odd} \\ \frac{m}{2}\psi_2 x^{(m^2-4)/2} & \text{if } m \text{ is even} \end{cases}$$

Proof. We will proceed by induction on m. The lemma is true for $m \leq 4$. Let $m = 2\ell+1$ with ℓ even. Since $\Lambda(\psi_2^2) = 4x^3$ and $(\ell+2)\ell^3 \neq (\ell-1)(\ell+1)^3$, we have $\Lambda(\psi_{\ell+2}\psi_\ell^3) \neq \Lambda(\psi_{\ell-1}\psi_{\ell+1}^3)$. Thus

$$\begin{aligned}
\Lambda(\psi_m) &= \Lambda(\psi_{\ell+2}\psi_\ell^3 - \psi_{\ell-1}\psi_{\ell+1}^3) \\
&= \Lambda(\Lambda(\psi_{\ell+2}\psi_\ell^3) - \Lambda(\psi_{\ell-1}\psi_{\ell+1}^3)) \\
&= \Lambda\left(\frac{\ell+2}{2}x^{(\ell^2+4\ell)/2}\frac{\ell^3}{8}x^{3(\ell^2-4)/2}(16x^6) - (\ell-1)x^{(\ell^2-2\ell)/2}(\ell+1)^3x^{3(\ell^2+2\ell)/2}\right) \\
&= ((\ell+2)\ell^3x^{(4\ell^2+4\ell)/2} - (\ell-1)(\ell+1)^3x^{(4\ell^2+4\ell)/2}) \\
&= (2\ell+1)x^{(4\ell^2+4\ell)/2} = mx^{(m^2-1)/2}
\end{aligned}$$

as desired. The other cases of m can be verified similarly. □

Corollary 2.9. $\Lambda(\psi_m^2) = m^2x^{m^2-1}$, *and* $\Lambda(\phi_m) = x^{m^2}$.

Proof. The first identity is trivial from Lemma 2.8. For the second identity assume m is odd. Then $\Lambda(x\psi_m^2) \neq \Lambda(\psi_{m-1}\psi_{m+1})$. Thus

$$\begin{aligned}
\Lambda(\phi_m) &= \Lambda(\Lambda(x\psi_m^2) - \Lambda(\psi_{m-1}\psi_{m+1})) \\
&= \Lambda(m^2x^{m^2} - \frac{m-1}{2}x^{(m^2-2m-3)/2}\frac{m+1}{2}x^{(m^2+2m-3)/2}(4x^3)) \\
&= m^2x^{m^2} - (m^2-1)x^{m^2} = x^{m^2}
\end{aligned}$$

The case of even m is treated similarly. □

It can be shown that polynomials ψ_m^2 and ϕ_m are coprime [78, Sec. 1.3]. Let $P = (x,y)$ be a point on the elliptic curve E defined by the generalized Weierstrass equation, (2.2), over a field k with $\mathrm{char}(k) \neq 2$. Then for any integer $n \in \mathbb{N}$

$$[n]P = \left(\frac{\phi_n(P)}{\psi_n^2(P)}, \frac{\omega_n(P)}{\psi_n^3(P)}\right) \tag{2.6}$$

This is usually proved by a complex analytic approach using the Weierstrass $\wp$ function, see for example [48]. One of the important properties of the division polynomials implied from Equation (2.6) is that if n is relatively prime to $\mathrm{char}(k)$ then $[n]P = \infty$ if and only if $\psi_n(x,y) = 0$; For if $\psi_n(x,y) \neq 0$ then $\phi_n(P)/\psi_n^2(P), \omega_n(P)/\psi_n^3(P) \in k$ hence $[n]P \neq$

∞. Conversely, if $[n]P \neq \infty$ then $\psi_n(x,y) \neq 0$, since $\phi_n(P)/\psi_n^2(P) \in k$ is defined and $\gcd(\psi_n^2, \phi_n) = 1$.
For an elliptic curve E defined over a field k, and a positive integer n, define the n-torsion subgroup of E to be

$$E[n] = \{P \in E(\overline{k}) \mid [n]P = \infty\}$$

which is the kernel of the endomorphism $[n] : E(\overline{k}) \to E(\overline{k})$. From Equation (2.6), the degree of $[n]$ is n^2, and by Corollary 2.4, $[n]$ is separable if and only if $\mathrm{char}(k) \nmid n$. So, $\#E[n] < n^2$ if $\mathrm{char}(k) \mid n$, and $\#E[n] = n^2$ if $\mathrm{char}(k) \nmid n$ by Proposition 2.2. For the field k of characteristic $p > 0$, E is called **ordinary** if $E[p] \cong \mathbb{Z}_p$, and **supersingular** if $E[p] \cong 0$. The structure of $E[n]$ for an arbitrary n is determined by the following.

Theorem 2.10. *Let E be an elliptic curve over a field K with $\mathrm{char}(K) = p$, and let n be a positive integer. Let $m = 1$ if $p = 0$. Otherwise, let $m = p^r$ where r is the largest integer such that $p^r \mid n$. Then*

$$E[n] = \mathbb{Z}_m^\delta \oplus \mathbb{Z}_{n/m} \oplus \mathbb{Z}_{n/m}$$

where $\delta = 0$ if the curve is supersingular, and $\delta = 1$ if it is ordinary.

Proof. Let t be a prime divisor of n. Then we consider two cases for t:
Case 1: $t = p$. We have $\#E[p] < p^2$ hence $E[p] \cong 0$ or $\mathbb{Z}_p$. If $E[p] \cong 0$ then $E[p^k] \cong 0$ for all k. So let $E[p] \cong \mathbb{Z}_p$. Since the endomorphism $[p]$ is surjective, there are points of order p^j for all j. Therefore $E[p^k]$ is cyclic of order p^k hence $E[p^k] \cong \mathbb{Z}_{p^k}$.
Case 2: $t \neq p$. We have $\#E[t^k] = t^{2k}$ so that $E[t^k]$ is a finite abelian t-group. Every finite abelian t-group can be expressed as a direct product of cyclic groups, hence $E[t^k] \cong \bigoplus_{i=1}^{\ell} \mathbb{Z}_{t^{\beta_i}}$ where $\beta_i \geq 1$. Thus, $E[t^k]$ contains $t^\ell - 1$ elements of order t which implies that $E[t] \subseteq E[t^k]$ is of order t^ℓ hence $\ell = 2$. Therefore, $E[t^k] \cong \mathbb{Z}_{t^{\beta_1}} \oplus \mathbb{Z}_{t^{\beta_2}}$ with $\beta_1 + \beta_2 = 2k$. Since $E[t^k]$ is not cyclic, we have $\beta_i \leq k$ and hence $\beta_i = k$, $i = 1, 2$.
Now, assume first that $p \neq 0$. Let $n = p^\alpha p_1^{\alpha_1} \cdots p_k^{\alpha_k}$ be the prime factorization of n in which $\alpha \geq 0$ and $\alpha_i > 0$ for $i = 1, \ldots, k$, and let $m = p^\alpha$. Then by case 1 and 2

$$\begin{aligned} E[n] &= E[p^\alpha] \oplus \left(\bigoplus_{i=1}^{k} E[p_i^{\alpha_i}] \right) = E[p^\alpha] \oplus \left(\bigoplus_{i=1}^{k} \mathbb{Z}_{p_i^{\alpha^i}} \oplus \mathbb{Z}_{p_i^{\alpha^i}} \right) \\ &= E[p^\alpha] \oplus \left(\bigoplus_{i=1}^{k} \mathbb{Z}_{p_i^{\alpha^i}} \right) \oplus \left(\bigoplus_{i=1}^{k} \mathbb{Z}_{p_i^{\alpha^i}} \right) = E[p^\alpha] \oplus \mathbb{Z}_{n/m} \oplus \mathbb{Z}_{n/m} \\ &= \mathbb{Z}_m^\delta \oplus \mathbb{Z}_{n/m} \oplus \mathbb{Z}_{n/m} \end{aligned}$$

If $p = 0$ then assume $n = p_1^{\alpha_1} \cdots p_k^{\alpha_k}$ is the prime factorization of n, and let $m = 1$. Then the same process results in $E[n] = \mathbb{Z}_n \oplus \mathbb{Z}_n = \mathbb{Z}_{n/m} \oplus \mathbb{Z}_{n/m}$ which completes the proof. $\square$

Chapter 3

Elliptic Curves Over Finite Fields

Let $\mathbb{F}_q$ be a finite field where $q = p^n$ is a power of a prime. Then, the group $E(\mathbb{F}_q)$ is finite. As a finite abelian group, its order is one of the most important quantities attached to it. Computing the quantity $\#E(\mathbb{F}_q)$ is referred to as *point counting*. In this chapter, we present some point counting algorithms. We start by deducing some general properties of the endomorphisms of $E(\mathbb{F}_q)$ by means of the Weil pairing. Then we prove the Hasse's theorem which puts a bound on the number of points. Before the final section, we will give some comments on the structure of $E(\mathbb{F}_q)$.

3.1 The Weil pairing

The Weil pairing, introduced by Weil [104], is a major computational and theoretical tool in the theory of elliptic curves, connecting the torsion subgroups to the roots of unity[1].

Theorem 3.1. *Let E be an elliptic curve over a field k and let m be a positive integer such that $\mathrm{char}(k) \nmid m$. Let also $\mu_m \subset \overline{k}^{\times}$ be the subgroup of m-th roots of unity. Then there exist a mapping*

$$e_m : E[m] \times E[m] \longrightarrow \mu_m$$

called the Weil pairing with the following properties:

1. *Bilinearity*

$$\begin{aligned} e_m(S_1 + S_2, T) &= e_m(S_1, T)e_m(S_2, T) \\ e_m(S, T_1 + T_2) &= e_m(S, T_1)e_m(S, T_2), \qquad \text{for all } T, T_1, T_2, S, S_1, S_2 \in E[m]. \end{aligned}$$

2. *Nondegeneracy.*

$$\begin{aligned} e_m(S, T) = 1 \text{ for all } T \in E[m] &\Leftrightarrow S = \infty \\ e_m(S, T) = 1 \text{ for all } S \in E[m] &\Leftrightarrow T = \infty \end{aligned}$$

[1] The Weil pairing is of great importance in pairing based cryptography. See chapters IX and X of [8] for details. Also see [63] for an efficient computation of the pairing.

3. *Identity.* $e_m(T,T) = 1$ *for all* $T \in E[m]$.

4. *Alternation.* $e_m(S,T) = e_m(T,S)^{-1}$ *for all* $S,T \in E[m]$.

5. *Endomorphism compatibility. For any* $\alpha \in \mathrm{End}(E)$, $e_m(\alpha(S),\alpha(T)) = e_m(S,T)^{\deg\alpha}$.

6. *Galois invariancy.* $e_m(\sigma S,\sigma T) = \sigma(e_m(S,T))$ *for any* $\sigma \in \mathrm{Gal}(\overline{k}/k)$, *where* $\mathrm{Gal}(\overline{k}/k)$ *is the Galois group of the extension* $\overline{k}/k$.

Proof. The proof needs some knowledge of divisors on elliptic curves, which is beyond the scope of this chapter. See [88, Sec. 3.8]. □

Remark. In some cases, one may work in a subfield of $\overline{k}$ over which the full n-torsion may not be available. There is a pairing, called **Tate-Lichtenbaum Pairing**, that can be used in those cases [24, 56].
Since $\mathrm{char}(k) \nmid m$, we have $E[n] \cong \mathbb{Z}_n^2$ by Theorem 2.10 i.e. $E[m]$ is a $\mathbb{Z}$-module of rank 2. If α is an endomorphism of E then $\alpha|_{E[m]}$ is a homomorphism of $\mathbb{Z}$-modules. Therefore, the action of α on a basis $\{B_1, B_2\}$ of $E[m]$ is a matrix $\alpha_m = [\alpha_{ij}]_{2\times 2}$ over $\mathbb{Z}_n$.

Theorem 3.2. *Let* E *be an elliptic curve defined over a field* k, *and* α *be a nontrivial endomorphism of* E. *Let* n *be a positive integer such that* $\mathrm{char}(k) \nmid n$. *Then* $\det(\alpha_n) \equiv \deg\alpha \pmod{n}$.

Proof. Let $e_n(B_1,B_2)^k = 1$. Then for every $T \in E[n]$,

$$\begin{aligned} e_n(kB_1,T) &= e_n(kB_1, aB_1 + bB_2) \quad \text{for some } a,b \in \mathbb{Z}_n \\ &= e_n(B_1,B_1)^{ak} e_n(kB_1,B_2)^b \quad \text{by bilinearity} \\ &= e_n(B_1,B_2)^{kb} = 1 \quad \text{by bilinearity and identity} \end{aligned}$$

which implies that $kB_1 = \infty$, and hence $n \mid k$. Thus, $e_n(B_1,B_2)$ is a primitive n-th root of unity. Therefore, by the properties of the Weil pairing,

$$\begin{aligned} e_n(B_1,B_2)^{\deg\alpha} &= e_n(\alpha(B_1),\alpha(B_2)) = e_n(\alpha_{11}B_1 + \alpha_{21}B_2, \alpha_{12}B_1 + \alpha_{22}B_2) \\ &= e_n(B_1,B_1)^{\alpha_{11}\alpha_{12}} e_n(B_1,B_2)^{\alpha_{11}\alpha_{22}} e_n(B_2,B_1)^{\alpha_{21}\alpha_{12}} e_n(B_2,B_2)^{\alpha_{21}\alpha_{22}} \\ &= e_n(B_1,B_2)^{\alpha_{11}\alpha_{22}-\alpha_{12}\alpha_{21}} = e_n(B_1,B_2)^{\det(\alpha_n)} \end{aligned}$$

which implies $\det(\alpha_n) \equiv \deg\alpha \pmod{n}$, since $e_n(B_1,B_2)$ is a primitive n-th root of unity. □

Corollary 3.3. *Let* $\varphi,\psi \in \mathrm{End}(E)$, *and* a,b *be integers. Then for the endomorphism* $a\varphi + b\psi \in \mathrm{End}(E)$, $\deg(a\varphi + b\psi) = a^2\deg\varphi + b^2\deg\psi + ab(\deg(\varphi+\psi) - \deg\varphi - \deg\psi)$.

Proof. Let φ_n and ψ_n be the matrices representing $\varphi|_{E[n]}$ and $\psi|_{E[n]}$ respectively. Then $(a\varphi + b\psi)|_{E[n]}$ is represented by $a\varphi_n + b\psi_n$. It can easily be seen that $\det(a\varphi_n + b\psi_n) = a^2\det(\varphi_n) + b^2\det(\psi_n) + ab(\det(\varphi_n+\psi_n) - \det(\varphi_n) - \det(\psi_n))$ which implies $\deg(a\varphi+b\psi) \equiv a^2\deg\varphi + b^2\deg\psi + ab(\deg(\varphi+\psi) - \deg\varphi + \deg\psi) \pmod{n}$. Since the degrees are finite and this holds for infinitely many n, it is an equality. □

3.2 The Hasse's Theorem

In this section, we prove the following result, first proved by Hasse [33] in 1934.

Theorem 3.4 (Hasse). *Let E be an elliptic curve over the finite field $\mathbb{F}_q$, and let ϕ_q be the Frobenius endomorphism of E. Assume $\#E(\mathbb{F}_q) = q + 1 - t$ for some integer t. Then*

1. *$|t| \leq 2\sqrt{q}$.*
2. *the endomorphism $\phi_q^2 - t\phi_q + q \in \mathrm{End}(E)$ is trivial.*

Moreover, t is the unique integer ℓ such that $\phi_q^2 - \ell\phi_q + q = 0$.

Proof of 1. Let a and b be nonzero integers. Then by Corollary 3.3,

$$\begin{aligned}\deg(a\phi_q - b) &= a^2 \deg \phi_q + b^2 \deg(-id) + ab(\deg(\phi_q - 1) - a\deg\phi_q - b\deg(-id)) \\ &= a^2 q + b^2 + ab(\#E(\mathbb{F}_q) - q - 1) \quad \text{by Proposition 2.6} \\ &= a^2 q + b^2 - abt\end{aligned}$$

Since $\deg(a\phi_q - b) \geq 0$, we have $a^2 q + b^2 - abt \geq 0$ hence $q + (b/a)^2 - (b/a)t \geq 0$. This is true for all rational numbers b/a, and since the set of such numbers is dense in $\mathbb{R}$, we have $x^2 - tx + q \geq 0$ for all $x \in \mathbb{R}$. This implies that $t^2 - 4q \geq 0$ which yields the result. □

To prove part 2, we need the following result from commutative algebra.

Lemma 3.5. *Let R be a commutative ring, and let M be a finitely generated R-module of rank n. Let I be an Ideal of R, and let φ be an R-module endomorphism of M such that $\varphi(M) \subseteq IM$. Then φ satisfies an equation of the form*

$$\varphi^n + a_1\varphi^{n-1} + \cdots + a_{n-1}\varphi + a_n = 0$$

where $a_i \in I$, $i = 1, \ldots, n$.

Proof. See [5, page 21]. □

Proof of 2. Let n be a positive integer relatively prime to q. As noted earlier, $E[n] \cong \mathbb{Z}_n^2$ is a $\mathbb{Z}_n$-module of rank 2. Let $I = \mathbb{Z}_n$ be the unit ideal of $\mathbb{Z}_n$. For the endomorphism $(\phi_q)_n = \phi_q|_{E[n]}$ of $E[n]$, where ϕ_q is the Frobenius endomorphism, we have $(\phi_q)_n(E[n]) \subseteq E[n] = IE[n]$. So, by Lemma 3.5, $(\phi_q)_n^2 + a(\phi_q)_n + b = 0$ for some $a, b \in \mathbb{Z}_n$. Let $(\phi_q)_n$ be represented by $[a_{ij}]_{2\times 2}$. Then, by elementary linear algebra, $a_{11} + a_{22} = \mathrm{tr}((\phi_q)_n) = a$, and $a_{11}a_{22} - a_{12}a_{21} = \det((\phi_q)_n) = b$. On the other hand, since $\phi_q - 1$ is separable,

$$\begin{aligned}\#E(\mathbb{F}_q) &= \ker(\phi_q - 1) \quad \text{by Proposition 2.6} \\ &= \deg(\phi_q - 1) \equiv \det((\phi_q)_n - I_2) \pmod{n} \quad \text{by Theorem 3.2} \\ &= a_{11}a_{22} - a_{12}a_{21} - (a_{11} + a_{22}) + 1 = \det((\phi_q)_n) - \mathrm{tr}((\phi_q)_n) + 1 \\ &= q - a + 1 \quad \text{by Theorem 3.2}\end{aligned}$$

Therefore, $a = t, b = q$ hence $(\phi_q)_n^2 + t(\phi_q)_n + q = 0$. This means that $(\phi_q^2 + t\phi_q + q)|_{E[n]} = 0$. Since this holds for infinitely many n, the endomorphism $\phi_q^2 + t\phi_q + q$ has an infinite kernel which is not possible by Proposition 2.2. So, it is the zero endomorphism. For the uniqueness assume that $\phi_q^2 + s\phi_q + q = 0$ for some integer s. Then $(t-s)\phi_q = (\phi_q^2 + t\phi_q + q) - (\phi_q^2 + s\phi_q + q) = 0$. Since ϕ_q is surjective, $[t - s]E(\overline{\mathbb{F}_q}) = \infty$ which implies that the endomorphism $[t - s]$ is trivial. This is not true unless $t - s = 0$. This completes the proof. □

3.3 The structure of $E(\mathbb{F}_q)$

The group $E(\mathbb{F}_q)$ is a finite abelian group. So, by the fundamental theorem of finite abelian groups, $E(\mathbb{F}_q)$ is isomorphic to a direct sum of cyclic groups $E(\mathbb{F}_q) \cong \mathbb{Z}_{n_1} \oplus \mathbb{Z}_{n_2} \oplus \cdots \oplus \mathbb{Z}_{n_k}$ such that $n_i \mid n_{i+1}$ for $i = 1, \ldots, k-1$. This means that there are n_1^k elements of $E(\mathbb{F}_q)$ of order n_1. But, by Theorem 2.10, there are $\leq n_1^2$ such elements. Thus, $k \leq 2$ hence

$$E(\mathbb{F}_q) \cong \mathbb{Z}_n \quad \text{or} \quad E(\mathbb{F}_q) \cong \mathbb{Z}_{n_1} \oplus \mathbb{Z}_{n_2}$$

for some positive integer n, or some positive integers n_1, n_2 with $n_1 \mid n_2$. We show that $n_1 \mid q-1$ in the second case. Let first prove the following.

Lemma 3.6. *Let E be an elliptic curve defined over a field k, and let n be a positive integer not divisible by* $\mathrm{char}(k)$. *If $E[n] \subseteq E(k)$ then $\mu_n \subset k$ where μ_n is the group of n-th roots of unity.*

Proof. Let e_n be the Weil pairing on the n-torsion $E[n]$, and let $\{B_1, B_2\}$ be a basis for $E[n]$. Then $e_n(B_1, B_2)$ is a primitive root of unity by the proof of Theorem 3.2. Since $E[n] \subset E(k)$, $B_1, B_2 \in E(k)$. For any $\sigma \in \mathrm{Gal}(\overline{k}/k)$ we have $\sigma(e_n(B_1, B_2)) = e_n(\sigma B_1, \sigma B_2) = e_n(B_1, B_2)$ by part 6 of Theorem 3.1. By the fundamental theorem of Galois theory, the primitive n-th root of unity is contained in k hence $\mu_n \subset k$. □

Since $\mathbb{Z}_{n_1} \oplus \mathbb{Z}_{n_1} \subset E(\mathbb{F}_q)$, $E(\mathbb{F}_q)$ contains all n_1^2 elements of the n_1-torsion subgroup hence $p \nmid n_1$. By Lemma 3.6, $\mu_{n_1} \subset \mathbb{F}_q$ hence $n_1 \mid q-1$.
So, we have determined the structure of $E(\mathbb{F}_q)$. The converse to this problem is that given a finite field $\mathbb{F}_q$ and a positive integer ℓ, is there an elliptic curve E over $\mathbb{F}_q$ such that $\#E(\mathbb{F}_q) = \ell$? The answer to this is given in [74] and [103].

3.4 Point counting on $E(\mathbb{F}_q)$

There have been various approaches for determining the number of rational points on elliptic curves over finite fields, see [7] for a survey. In this section, we present three point counting algorithms on elliptic curves over finite fields: the naive counting, the baby-step giant-step, and the Schoof's algorithm. Over this section, by a polynomial time algorithm we shall mean an algorithm with running time polynomial in $\log q$. Therefore, for example, an algorithm with the running time $O(\sqrt[4]{q})$, or more generally $O(q^{O(1)})$, is not a polynomial time algorithm. Let first make the following observation.

Theorem 3.7 (Weil). *Let E be an elliptic curve over $\mathbb{F}_q$, and let $\#E(\mathbb{F}_q) = q+1-t$. Write $x^2 - tx + q = (x-\lambda_1)(x-\lambda_2)$ with $\lambda_1, \lambda_2 \in \mathbb{C}$. Then $\#E(\mathbb{F}_{q^n}) = q^n + 1 - (\lambda_1^n + \lambda_2^n)$ for all $n \geq 0$.*

Proof. Let ℓ be a positive integer relatively prime to q, and let ϕ_q be the Frobenius endomorphism of E. Then $(\phi_q)_\ell^2 - t(\phi_q)_\ell + q = 0$ by Theorem 3.4. So, λ_1 and λ_2 are the eigenvalues of $(\phi_q)_\ell$ hence $\mathrm{tr}((\phi_q)_\ell) = \lambda_1 + \lambda_2$. From linear algebra we have $\mathrm{tr}((\phi_q)_\ell^n) = \lambda_1^n + \lambda_2^n$ for any

positive integer n. Also, it is trivial that $\det(A - I_2) = 1 + \det(A) - \operatorname{tr}(A)$ for all 2×2 matrices A. Thus

$$\begin{aligned}\#E(\mathbb{F}_{q^n}) &= \#\ker(\phi_q^n - 1) = \deg(\phi_q^n - 1) \equiv \det((\phi_q)_\ell^n - I_2) \pmod{\ell} \\ &= 1 + \det((\phi_q)_\ell^n) - \operatorname{tr}((\phi_q)_\ell^n) = 1 + q^n - (\lambda_1^n + \lambda_2^n)\end{aligned}$$

Since this holds for infinitely many ℓ, it must be an equality. □

Theorem 3.7 says that if we know $\#E(\mathbb{F}_q)$ then we can easily compute $\#E(\mathbb{F}_{q^n})$. So if, for example, the elements of $\mathbb{F}_q$ are represented by polynomials, which is usually the case, then computing $\#E(\mathbb{F}_{q^n})$ amounts to computing $\#E(\mathbb{F}_p)$.

3.4.1 The naive method

When the size of the field $\mathbb{F}_q$ is small we can simply run through all its elements to find pairs satisfying the equation of E. This amounts to check, for every $x \in \mathbb{F}_q$, if $x^3 + Ax + B$ is a square in $\mathbb{F}_q$. Let

$$\left(\frac{a}{\mathbb{F}_q}\right) = \begin{cases} 1 & \text{if } a \text{ is a square in } \mathbb{F}_q^\times \\ -1 & \text{if } a \text{ is not a square in } \mathbb{F}_q^\times \\ 0 & \text{otherwise} \end{cases}$$

be the Legendre symbol over $\mathbb{F}_q$. For every $\alpha \in \mathbb{F}_q$ if $(\frac{\alpha^3+A\alpha+B}{\mathbb{F}_q}) = 1$ or -1 or 0 then there are two points $(\alpha, \pm y)$ or no points or one point $(\alpha, 0)$ on $E(\mathbb{F}_q)$ respectively. Therefore, the number of points with first coordinate α is $1 + (\frac{\alpha^3+A\alpha+B}{\mathbb{F}_q})$. Summing over all $\alpha \in \mathbb{F}_q$, and taking into account the point ∞, gives

$$\#E(\mathbb{F}_q) = 1 + \sum_{\alpha \in \mathbb{F}_q} \left(1 + \left(\frac{\alpha^3 + A\alpha + B}{\mathbb{F}_q}\right)\right) = q + 1 + \sum_{\alpha \in \mathbb{F}_q} \left(\frac{\alpha^3 + A\alpha + B}{\mathbb{F}_q}\right)$$

Algorithm 3.1 Naive algorithm for counting points on $E(\mathbb{F}_q)$

Input: The elliptic curve E defined by $y^2 = x^3 + Ax + B$ with $A, B \in \mathbb{F}_q$
Output: The number of points on $E(\mathbb{F}_q)$

1. $n \leftarrow q + 1$
2. **for all** $\alpha \in \mathbb{F}_q$ **do**
3. $\quad n \leftarrow n + \left(\frac{\alpha^3+A\alpha+B}{\mathbb{F}_q}\right)$
4. **end for**
5. **return** n

We have $\left(\frac{\alpha^3+A\alpha+B}{\mathbb{F}_q}\right) = (\alpha^3 + A\alpha + B)^{(q-1)/2}$. So the Legendre symbol can be computed in $O(\log q)$ multiplications in $\mathbb{F}_q$. Therefore, the running time of Algorithm 3.1 is $O(q \log q)$ operations in $\mathbb{F}_q$.

3.4.2 The Baby-step Giant-step

Assume we know how to compute the order of an arbitrary point $P \in E(\mathbb{F}_q)$. By the Hasse's theorem, $q+1-2\sqrt{q} \leq \#E(\mathbb{F}_q) \leq q+1+2\sqrt{q}$. For large enough q there is exactly one multiple of $\#E(\mathbb{F}_q)$ in this interval. The order of any point $P \in E(\mathbb{F}_q)$ divides the order of the group $E(\mathbb{F}_q)$. So, for a randomly selected point $P_1 \in E(\mathbb{F}_q)$, if $N_1 = \text{ord}(P_1)$ has only one multiple in the above interval then $\#E(\mathbb{F}_q) = N_1$, otherwise select another point P_2 and let $N_2 = \text{lcm}(N_1, \text{ord}(P_2))$ and do the same for N_2. This process continues, by selecting further random points and taking least common multiples, until N_k has a unique multiple in the Hasse interval for some k.
To compute the order of a point $P \in E(\mathbb{F}_q)$, we can first find an integer ℓ such that $[\ell]P = \infty$. Then Algorithm 3.2 computes the order of P from ℓ.

Algorithm 3.2 Compute the order of a point from a given annihilator

Input: A point $P \in E(\mathbb{F}_q)$ and an integer ℓ such that $[\ell]P = \infty$
Output: The order of P

1. $n \leftarrow \ell$
2. **for all** prime divisors p of n **do**
3. **while** $[n]P = \infty$ **do**
4. $n \leftarrow n/p$
5. **end while**
6. **end for**
7. **return** n

To find an integer $q+1-2\sqrt{q} \leq \ell \leq q+1+2\sqrt{q}$ such that $[\ell]P = \infty$, one can try all elements of this interval which takes around $4\sqrt{q}$ steps. But, using an adaptation of the Shanks's algorithm [82], the number of steps can be reduced to around $4\sqrt[4]{q}$ as follows. Let $m > \sqrt[4]{q}$ be an integer. Compute the sequences of points $B = \{[j]P,\ j = 0, \pm 1, \ldots, \pm m\}$ and $G = \{[q+1+2mk]P,\ k = -m, -m+1, \ldots, m-1, m\}$. Then there is an element occurring in both sequences, because: in the identity $\#E(\mathbb{F}_q) = q+1-t$ we have $|t| \leq 2m^2$, then there are always $-m < t_0 \leq m$ and $-m \leq t_1 \leq m$ such that $t = 2mt_1 + t_0$. Now, letting $k = -t_1$ we have $G \ni [q+1-2mt_1]P = [q+1-t+t_0]P = [\#E(\mathbb{F}_q) + t_0]P = [t_0]P \in B$.

Algorithm 3.3 baby-step giant-step point counting

Input: An elliptic curve E over $\mathbb{F}_q$
Output: The number of point on $E(\mathbb{F}_q)$

1. $m \leftarrow \lceil \sqrt[4]{q} \rceil$, $d \leftarrow 1$
2. select a random point $P \in E(\mathbb{F}_q)$
3. $P_1 \leftarrow [2m]P$
4. $B_0 \leftarrow \infty$
5. $G \leftarrow [q+1]P - [m]P_1$,, $j \leftarrow 0$
6. **for** $i = 1$ to m **do**
7. $B_i \leftarrow B_{i-1} + P$
8. **end for**
9. **while** $G \neq \pm B_i$, $0 \leq i \leq m$ **do**

10. $G \leftarrow G + P_1$, $j \leftarrow j + 1$
11. **end while**
12. $\ell \leftarrow q + 1 + 2mj \mp i$
13. compute $\mathrm{ord}(P)$ using Algorithm 3.2 with input ℓ
14. $d \leftarrow \mathrm{lcm}(d, \mathrm{ord}(P))$
15. **if** d has only one multiple in the range $[q + 1 - 2\sqrt{q}, q + 1 + 2\sqrt{q}]$ **then**
16. **return** d
17. **end if**
18. go to Step 2

It is not hard to show that the expected running time of Algorithm 3.3 is $O(\sqrt[4]{q}\log^3 q)$ group operations, and it needs storage for $O(\sqrt[4]{q})$ group elements, see [22] for details.

3.4.3 Schoof's Algorithm

The first polynomial time algorithm for counting points on $E(\mathbb{F}_q)$ was introduced by Schoof [80]. By the Hasse's theorem $\#E(\mathbb{F}_q) = q+1-t$ with $|t| \leq 2\sqrt{q}$. The idea of the algorithm is to compute t modulo many small primes, and then recombine the results using the Chinese remaindering theorem to obtain t. Let ϕ_q be the Frobenius map on E. By Theorem 3.4, $\phi_q^2 - t\phi_q + q = 0$. Let $\gamma > 2$ be a prime not equal to $\mathrm{char}(\mathbb{F}_q) = p$. Assume $P = (x, y) \neq \infty$ is a γ-torsion point. Then $\phi_q^2(P) - [t_\gamma]\phi_q(P) + [q_\gamma]P = 0$ where $t_\gamma \equiv t \pmod{\gamma}$, and $q_\gamma \equiv q \pmod{\gamma}$. In other words,

$$(x^{q^2}, y^{q^2}) - [t_\gamma](x^q, y^q) + [q_\gamma](x, y) = 0 \tag{3.1}$$

hence $[t_\gamma](x^q, y^q) = (x^{q^2}, y^{q^2}) + [q_\gamma](x, y)$. Since we know the right side, and γ is small, we can try all values in the range $[0, \gamma - 1]$ to find the t_γ satisfying the above equation. The problem is how to obtain a point $P \in E[\gamma]$. As we saw in Section 2.3, P is a γ-torsion if and only if $\psi_\gamma(P) = 0$, where ψ_γ is the γ-th division polynomial. Since γ is odd, $\psi_\gamma \in \mathbb{F}_q[x]$. Therefore, we can compute a root of ψ_γ to obtain the x-coordinate of a γ-torsion point P, and then compute the y-coordinate using the equation of E. But, the roots of the division polynomials ψ_γ usually occupy in extensions $K/\mathbb{F}_q$ of fairly large degree. The crucial observation is that Equation (3.1) holds for all γ-torsion points so that we can work modulo ψ_γ, i.e. work with all γ-torsion points simultaneously.
In other words, if f and g are the x-coordinates of $[\alpha](x^q, y^q)$ and $(x^{q^2}, y^{q^2}) + [q_\gamma](x, y)$, for some integer α, respectively, then α is the desired value for t_γ if $f - g \equiv 0 \pmod{\psi_\gamma}$. Therefore, we shall do all computations in the quotient ring $A_\gamma = \mathbb{F}_q[x, y]/\langle \psi_\gamma(x), y^2 - f(x) \rangle$ where $y^2 = f(x)$ is the equation of E. Let $\{\gamma_i\}_{i \leq k}$ be the set of primes required by the Chinese remaindering theorem to uniquely reconstruct the value of t. Then we should have

$$\prod_{i=1}^{k} \gamma_i \geq 4\sqrt{q} \tag{3.2}$$

Algorithm 3.4 Schoof's point counting

Input: An elliptic curve E over $\mathbb{F}_q$

Output: The number of point on $E(\mathbb{F}_q)$

1. $M \leftarrow 1$, $\gamma \leftarrow 3$
2. **while** $M \leq 4\sqrt{q}$ **do**
3. $\quad P \leftarrow (x^q, y^q) \mod \langle \psi_\gamma(x), y^2 - f(x) \rangle$
4. $\quad Q \leftarrow (x^{q^2}, y^{q^2}) + [q_\gamma](x, y) \mod \langle \psi_\gamma(x), y^2 - f(x) \rangle$
5. $\quad$ **for** $n = 0$ to $\gamma - 1$ **do**
6. $\quad\quad$ **if** $Q \equiv 0 \pmod{\langle \psi_\gamma(x), y^2 - f(x) \rangle}$ **then**
7. $\quad\quad\quad t_\gamma \leftarrow n$, go to Step 12
8. $\quad\quad$ **else**
9. $\quad\quad\quad Q \leftarrow Q - P$
10. $\quad\quad$ **end if**
11. $\quad$ **end for**
12. $\quad M \leftarrow M.\gamma$
13. $\quad \mathcal{P} \leftarrow \mathcal{P} \cup \gamma$
14. $\quad \gamma \leftarrow$ the next largest prime
15. **end while**
16. use the Chinese remaindering theorem to solve the system of congruences $t \equiv t_\gamma \pmod{\gamma}$, $\gamma \in \mathcal{P}$.
17. **return** $q + 1 - t$

The prime number theorem says that $\lim_{x\to\infty} \pi(x) \log(x)/x = 1$ where $\pi(x)$ is the number of primes not larger than x. It can be shown that this is equivalent to $\lim_{x\to\infty} \vartheta(x)/x = 1$ where

$$\vartheta(x) = \sum_{\substack{\gamma \leq x \\ \gamma \text{ prime}}} \log \gamma$$

is the Chebyshev's ϑ-function. This implies

$$1 = \lim_{x\to\infty} \frac{\vartheta(x)}{x} = \lim_{x\to\infty} \frac{1}{x} \sum_{\substack{\gamma \leq x \\ \gamma \text{ prime}}} \log \gamma = \lim_{x\to\infty} \frac{1}{x} \log \prod_{\substack{\gamma \leq x \\ \gamma \text{ prime}}} \gamma$$

which means that $\prod_{\substack{\gamma \leq x \\ \gamma \text{ prime}}} \gamma \approx e^x$. So if we take $\gamma_k \approx \frac{1}{2}\log(16q) \in O(\log q)$ then the set of primes $\{\gamma_i\}_{i \leq k}$ satisfies condition (3.2). Therefore, for any prime γ produced in Step 14 of Algorithm 3.4, we have $\gamma \in O(\log q)$. The outer loop in Algorithm 3.4 iterates $O(\log q)$ times. Steps 3 and 4 require $O(\log q)$ multiplications of polynomials of order $O(\gamma^2)$ in A_γ which can be accomplished in $O(M(\gamma^2) \log q) = O(M(\log^2 q) \log q)$ operations in $\mathbb{F}_q$. The total cost of the execution of the inner loop is $O(\gamma M(\gamma^2)) = O(M(\log^2 q) \log q)$ operations in $\mathbb{F}_q$. Therefore, the running time of Algorithm 3.4 is $O(M(\log^2 q) \log^2 q)$ operations in $\mathbb{F}_q$. There have been many theoretical and practical improvements on the Schoof's algorithm, see, for example, [7, 21, 81]. An improvement suggested by Atkin and Elkis, called Schoof-Elkies-Atkin (SEA) algorithm, is currently the fastest known algorithm for general characteristics [7]. There are other polynomial time algorithm, that work for small characteristics, like Satoh's algorithm [75]. For refinements of Satoh's algorithm see [77, 76, 97].

Chapter 4

Hyperelliptic Curves

By the increasing applications of hyperelliptic curves in various areas of computational computer science, the theory of these curves has been advanced significantly during recent years. They have mainly been used in areas such as public-key cryptography [44, 17], primality testing [2], integer factorization [53, 54], and error-correcting codes [51]. In this chapter, we give an introduction to the basic theory of hyperelliptic curves. We will discuss rational functions on the curve by following a bit of more general theory to give a clear description of concepts like uniformizers. Then, we will give a brief treatment on divisors, and their basic properties. The Mumford representation, and how to add divisors in the set of divisor classes of degree zero, which is a group called the jacobian of the curve, will be discussed next. At the last section, we summarize some basic facts about hyperelliptic curves over finite fields.

4.1 Basic definitions

Let k be field and $\overline{k}$ be its algebraic closure, and let $g \geq 2$ be an integer. A hyperelliptic curve of genus g over k is a nonsingular plane curve $\mathcal{H} \subset \mathbb{A}_k^2$ of the form

$$\mathcal{H} : y^2 + h(x)y = f(x) \tag{4.1}$$

together with a point at infinity, where $f(x) \in k[x]$ is monic with $\deg f = 2g + 1$, and $h(x) \in k[x]$ with $\deg h \leq g$. Here, nonsingularity means there is no point $P = (x, y) \in \mathbb{A}_{\overline{k}}^2$ such that $y^2 + h(x)y - f(x) = 2y + h(x) = h'(x)y - f'(x) = 0$. When $g = 1$, Equation (4.1) is the generalized Weierstrass equation of an elliptic curve, see Section 2.1. Therefore, hyperelliptic curves can be thought of as a generalization of elliptic curves. If $\mathrm{char}(k) > 2$ then the change of variables $y \mapsto y - \frac{1}{2}h(x)$ in (4.1) gives

$$\mathcal{H} : y^2 = f(x) \tag{4.2}$$

for some monic polynomial $f(x)$ of degree $2g+1$. This implies that $\mathcal{H}$ is nonsingular if there is no point $P = (x, y) \in \mathbb{A}_{\overline{k}}^2$ satisfying $y^2 - f(x) = 2y = f'(x) = 0$ which is simply equivalent to saying that $f(x)$ has no repeated roots. Throughout this chapter, we assume $\mathrm{char}(k) > 2$, unless otherwise specified. For an extension L/k, as in the case of elliptic curves, the set of points in $\mathbb{A}_L^2$ satisfying (4.2) is denoted by $\mathcal{H}(L)$.

Definition 4.1. Let $P = (x, y)$ be a finite point on $\mathcal{H}$. The **involution** of P is defined to be $\tilde{P} = (x, -y)$. We also define $\tilde{\infty} = \infty$. The point P is special if $y = 0$, it is called ordinary otherwise.

Therefore, there are only a finite number of special points on $\mathcal{H}$, namely the points with roots of $f(x)$ as their first coordinate. Throughout this chapter, except for Section 4.6, we assume that $k = \overline{k}$, i.e. k is algebraically closed.

4.2 Rational functions

Let $C \subset \mathbb{A}^2_k$ be an affine plane curve, defined over the field k, and given by the equation $F = 0$ where $F \in k[x, y]$ is irreducible. Therefore, the ideal $\langle F \rangle \subset k[x, y]$ is a prime ideal. The coordinate ring of C, denoted by $\Gamma_k(C)$, is defined to be the quotient ring

$$\Gamma_k(C) = k[x, y]/\langle F \rangle$$

which is an integral domain. Elements of $\Gamma_k(C)$ are called polynomial functions on C. We simply write $\Gamma(C)$ when k is clear from the context. Since $\Gamma(C)$ is an integral domain, we can form its field of fractions denoted by $k(C)$. Elements of $k(C)$ are called **rational functions** on C, and $k(C)$ itself is called the field of rational functions on C. Every element $f \in k(C)$ is of the form g/h where g and h are polynomial functions on C. For a rational function $f \in k(C)$, and a point $P \in C$, we say that f is defined at P if there are some polynomial functions $g, h \in \Gamma(C)$ such that $f = g/h$, and $h(P) \neq 0$. Otherwise, f is said to have a *pole* at P, or P is called a pole of f, and we write $f(P) = \infty$.

Definition 4.2. A ring R is called a local ring if it has a unique maximal ideal.

Assume the set of non-units of a ring R form an ideal $\mathfrak{m}$. Then R is clearly a local ring with $\mathfrak{m}$ as its maximal ideal. This is sometime used as a definition of local rings. For a point $P \in C$, let $\mathcal{O}_P(C)$ denote the set of elements of $k(C)$ defined a P. Then $\mathcal{O}_P(C)$ is clearly a ring such that $\Gamma(C) \subset \mathcal{O}_P(C) \subset k(C)$. Let $f \in \mathcal{O}_P(C)$, and write $f = g/h$ with $g, h \in \Gamma(C)$, $h(P) \neq 0$. The value of f at P is defined as $f(P) = g(P)/h(P)$ which is independent of the choice of g and h.

Lemma 4.3. *$\mathcal{O}_P(C)$ is a local ring.*

Proof. An element $f \in \mathcal{O}_P(C)$ is a non-unit if and only if $f(P) = 0$. Let $\mathfrak{m} = \{f \in \mathcal{O}_P(C) \mid f(P) = 0\}$. Let $\phi : \mathcal{O}_P(C) \to k$ be the map $f \mapsto f(P)$. Then ϕ is surjective, and $\ker \phi = \mathfrak{m}$. Therefore, $\mathfrak{m}$ is a maximal ideal hence $\mathcal{O}_P(C)$ is a local ring. □

A partial order on a set $\mathcal{A}$ is a reflexive, antisymmetric, and transitive relation on $\mathcal{A}$. Let $\mathcal{A}$ be a set partially ordered by a relation $\leq$. Then the following statements are equivalent: i) Every non-decreasing sequence in $\mathcal{A}$ is stationary. ii) Every non-empty subset of $\mathcal{A}$ has a maximal element. Assume that i) is true. If ii) is false, then there is a non-empty subset S of $\mathcal{A}$ with no maximal element. So, one can construct an infinite strictly increasing sequence in T, a contradiction. Conversely, for any non-decreasing sequence $x_1 \leq x_2 \leq \cdots$ let $S = \{x_i, i \in \mathbb{N}\}$. Then S has a maximal element x_n for some integer n, hence $x_1 \leq$

$x_2 \leq \cdots \leq x_n = x_{n+1} = \cdots$. Now, for a ring R, let $\mathscr{I}$ be the set of ideals of R ordered by inclusion $\subseteq$. Then the condition i) on $\mathscr{I}$ is called the *ascending chain condition* (acc). The ring R is said to be Noetherian if it satisfies either of i) or ii).

Proposition 4.4. *A ring R is Noetherian if and only if every ideal of R is finitely generated.*

Proof. '$\Rightarrow$'. Let I be an ideal of R, and let $\mathscr{I}$ be the set of all finitely generated ideals contained in I. Then, by Zorn's lemma, $\mathscr{I}$ has a maximal element, say J. If $J \neq I$ then let $a \in I$, and $a \notin J$. Then J is a proper ideal of the finitely generated ideal $J + Ra$ which is a contradiction.
'$\Leftarrow$'. Let $I_1 \subseteq I_2 \subseteq \cdots$ be an ascending chain of ideals of R. Then $J = \bigcup_{n=1}^{\infty} I_n$ is an ideal of R hence finitely generated, say by $\{a_1, a_2, \ldots, a_r\}$. We have $a_i \in I_{n_i}$ for some n_i, $i = 1, \ldots, r$. Let $k = \max_{i=1}^{r} n_i$, then $J = I_k$, and hence we have $I_1 \subseteq I_2 \subseteq \cdots \subseteq I_k = I_{k+1} = \cdots$. □

Proposition 4.5. *$\mathcal{O}_P(C)$ is a Noetherian ring.*

Proof. It can easily be seen that $\Gamma(C)$ is a Noetherian ring. Let I be an ideal of $\mathcal{O}_P(C)$, and let $\{a_1, a_2, \ldots, a_k\}$ be a set of generators for the ideal $I \cap \Gamma(C)$ of $\Gamma(C)$. Now, let $a \in I$, then there is an element $c \in \Gamma(C)$ with $c(P) \neq 0$ such that $ca \in \Gamma(C)$. So, $ca \in I \cap \Gamma(C)$ hence $ca = \sum_{n=1}^{k} d_n a_n$ where $d_n \in \Gamma(C)$. Thus, $a = \sum_{n=1}^{k} \frac{d_n}{c} a_n$ which means that a_i generate I in $\mathcal{O}_P(C)$. This completes the proof. □

Remark. If a commutative ring R is Noetherian, and $\mathfrak{p}$ is a prime ideal of R, then the localization of R with respect to $\mathfrak{p}$, denoted by $R_{\mathfrak{p}}$, is also Noetherian. The ring $\mathcal{O}_P(C)$ is indeed the localization of the ring $\Gamma(C)$ with respect to the prime ideal $\mathfrak{p} = \{f \in \Gamma(C) \mid f(P) = 0\}$.

Proposition 4.6. *Let R be a Noetherian local domain that is not a field, and let $\mathfrak{m}$ be its maximal ideal. If $\mathfrak{m}$ is principal then there is an element $\alpha \in R$ such that every nonzero element $a \in R$ can be uniquely expressed in the form $a = \beta\alpha^n$ for some unit $\beta \in R$ and nonnegative integer n.*

Proof. Let $\mathfrak{m} = (\alpha)$ for some non-unit $\alpha \in R$. We may assume that a is not a unit. Then $(a) \subseteq \mathfrak{m}$. We have $a = b_1\alpha$ for some $b_1 \in R$. If b_1 is a unit we are done; otherwise let $b_1 = b_2\alpha$. The same thing holds for b_2 and so on. If this process does not terminate then we have chain of ideals $(b_1) \subseteq (b_2) \subseteq \cdots$. Since R is Noetherian, this chain is stationary so that $(b_n) = (b_{n+1}) = \cdots$ for some n. So, $b_{n+1} = cb_n = c\alpha b_{n+1}$ hence $c\alpha = 1$, and α is a unit which is a contradiction. Therefore, $a = \beta\alpha^k$ for some unit β and some integer $k \geq 1$. For the uniqueness let $\beta_1\alpha^m = a = \beta_2\alpha^n$ with $m \geq n$, and β_1, β_2 units. Then $\beta_1\alpha^{m-n} = \beta_2$ which implies $m = n$, and hence $\beta_1 = \beta_2$. □

Remark. For an ideal $\mathfrak{m} \neq (1)$ of a Noetherian domain R we have $\bigcap_{n=1}^{\infty} \mathfrak{m}^n = 0$. This means that there is an integer $k \geq 1$ such that $(a) \subseteq \mathfrak{m}^k$ and $(a) \nsubseteq \mathfrak{m}^{k+1}$. Then $a = \beta\alpha^k$ for some unique $\beta \in R$, and β should be a unit because $(a) \nsubseteq \mathfrak{m}^{k+1}$.

A ring R satisfying the conditions of Proposition 4.6 is called a *discrete valuation ring* (DVR), and the element α is called a *uniformizer* for R. For any element $a \in R$ write $a = \beta\alpha^d$ for some unit $\beta \in R$ and some integer $d \geq 0$. Then the order of a, denoted by $\mathrm{ord}(a)$, is defined

to be d. There maybe more than one uniformizer for a discrete valuation ring R. Let α and λ be two uniformizers for R. Then $\lambda = \beta_1\alpha^{n_1}$, and $\alpha = \beta_2\lambda^{n_2}$. Thus, $\alpha = \beta_1^{n_2}\beta_2\alpha^{n_1n_2}$ which implies that $\alpha^{n_1n_2-1}$ is a unit. So, $n_1n_2 = 1$ hence $n_1 = n_2 = 1$. Therefore, the ring R has a unique, up to a unit, uniformizer. In particular, the order of an element a is independent of the choice of the uniformizer.

Lemma 4.7. *Let R be a DVR, and let $a, b \in R$. Then*

1. $\operatorname{ord}(ab) = \operatorname{ord}(a) + \operatorname{ord}(b)$.
2. $\operatorname{ord}(a + b) \geq \min(\operatorname{ord}(a) + \operatorname{ord}(b))$.

Proof. Let α be a uniformizer for R, and let $a = \beta_1\alpha^{n_1}$, and $b = \beta_2\alpha^{n_2}$ with $\beta_1, \beta_2 \in R$ units. Also let $n_1 \geq n_2$. Then $ab = \beta_1\beta_2\alpha^{n_1+n_2}$ which proves 1. Also $a+b = \alpha^{n_2}(\beta_1\alpha^{n_1-n_2}+\beta_2)$. We can write $(\beta_1\alpha^{n_1-n_2} + \beta_2) = \beta_3\alpha^{n_3}$ for some $n_3 \geq 0$, and some unit β_3. So, $a + b = \beta_3\alpha^{n_1+n_3}$ which proves 2. □

Let K be the field of fractions of R. The definition domain of the order function on R can be extended to K in a natural way as follows. Let $f \in K$, and let $f = a/b$ for some $a, b \in R$. Then $\operatorname{ord}(f) = \operatorname{ord}(a) - \operatorname{ord}(b)$. This is clearly independent of choices of a and b, and also Lemma 4.7 remains true for elements of K.

Theorem 4.8. *Let P be a nonsingular point on C. Then $\mathcal{O}_P(C)$ is a discrete valuation ring. Moreover, any line L intersecting C nontangentially at P is a uniformizer for $\mathcal{O}_P(C)$.*

Proof. Let $P = (a, b)$. It can easily be seen that $\mathfrak{m} = \langle x - a, y - b\rangle$ is the maximal ideal of $\mathcal{O}_P(C)$. We can write

$$F = (x - a)\frac{\partial F}{\partial x}(P) + (y - b)\frac{\partial F}{\partial y}(P) + g(x, y)$$

for some $g \in k[x, y]$. Since F is nonsingular at P, either $\frac{\partial F}{\partial x}(P) \neq 0$ or $\frac{\partial F}{\partial y}(P) \neq 0$. Assume $\frac{\partial F}{\partial x}(P) \neq 0$. Then, grouping together the terms with $(x - a)$ we have $F = (x - a)r(x, y) - (y - b)^\ell s(x, y)$ where $r(P) \neq 0$, and $s(P) \neq 0$, and $\ell \geq 1$ is the largest integer such that $\frac{\partial^i F}{\partial y^i}(P) = 0$ for all $1 \leq i \leq \ell - 1$. Let $\bar{r}, \bar{s}$ be the images of r and s in $\mathcal{O}_P(C)$ respectively. Then $(x-a)\bar{r}(x, y) = (y-b)^\ell\bar{s}(x, y)$ hence $(x-a) = (y-b)^\ell\bar{s}(x, y)\bar{r}^{-1}(x, y) \in \langle y-b\rangle$. Thus, $\mathfrak{m} = \langle y - b\rangle$ is principal, and so, by Proposition 4.6, $\mathcal{O}_P(C)$ is a discrete valuation ring. For the second part of the theorem, let L' be the tangent to F at P. Since the line L is distinct from L', there is always an affine transformation taking L, L', P to $y-b, x-a, P$ respectively. By the first part, $y - b$ is a uniformizer for $\mathcal{O}_P(C)$, hence also L. □

Now, let $\mathcal{H}$ be the hyperelliptic curve (4.2). The polynomial $F(x, y) = y^2-f(x)$ is irreducible over $\overline{k}$; For the only nontrivial factorization of F is of the form $F(x, y) = (y-a(x))(y-b(x))$ which implies that $a(x) + b(x) = 0$ hence $\deg a = \deg b$. Thus, $2\deg a = \deg a + \deg b = \deg f = 2g + 1$ which is a contradiction. We denote by ord_P the order function on $k(\mathcal{H})$ defined by the discrete valuation ring $\mathcal{O}_P(\mathcal{H})$. For an ordinary point $P = (a, b)$ on $\mathcal{H}$, $\frac{\partial F}{\partial y}(P) = 2b \neq 0$. So, $L : x - a$ is not a tangent to $\mathcal{H}$ at P, and hence it is a uniformizer for

$\mathcal{O}_P(\mathcal{H})$ by Theorem 4.8. If $P=(a,0)$ is a special point then $\frac{\partial F}{\partial y}(P)=2b=0$ hence $L:y$ is not a tangent, and so a uniformizer for $\mathcal{O}_P(\mathcal{H})$.
To find a uniformizer at $P=\infty$ we need to use the projective equation of $\mathcal{H}$. Let $f(x)=x^{2g+1}+a_{2g}x^{2g}+\cdots+a_1x+a_0$. Then the projective equation is $\mathcal{H}: z^{2g-1}y^2=x^{2g+1}+a_{2g}zx^{2g}+\cdots+a_1z^{2g}x+a_0z^{2g+1}$, and the point at infinity is $T=(0:1:0)$. Changing to the coordinates $w=z/y, v=x/y$ gives the affine curve

$$\mathcal{G}: \quad w^{2g-1}=v^{2g+1}+a_{2g}wv^{2g}+\cdots+a_1w^{2g}v+a_0w^{2g+1} \tag{4.3}$$

with $Q=(0,0)$ correspond to the point at infinity. Let $\alpha=1+a_{2g}(w/v)+\cdots+a_0(w/v)^{2g+1}$. From (4.3) we have $(w/v)^{2g+1}/\alpha=w^2$ which implies that $w/v=0$ at Q hence $\alpha\in\mathcal{O}_Q(\mathcal{G})$ is a unit. Again from (4.3) we have $w^{2g-1}=v^{2g+1}\alpha$. Let $u=v^g/w^{g-1}$. It can easily be seen that $\mathfrak{m}=\langle w,v\rangle$ is the maximal ideal of $\mathcal{O}_Q(\mathcal{G})$. We have $u^2=\alpha^{-1}w/v=0$ at Q hence $u\in\mathfrak{m}$. It can be readily verified that $v=\alpha^{g-1}u^{2g-1}$, and $w=\alpha^g u^{2g+1}$. So, u is a uniformizer for $\mathcal{O}_Q(\mathcal{G})$. Therefore, $u=v^g/w^{g-1}=x^g/yz^{g-1}$ is a uniformizer for the projective local ring $\mathcal{O}_T(\mathcal{H})$, and hence x^g/y is a uniformizer for the affine local ring $\mathcal{O}_\infty(\mathcal{H})$. From the above we have $\mathrm{ord}_\infty(x)=\mathrm{ord}_T(x/z)=\mathrm{ord}_Q(v/w)=2g-1-2g-1=-2$, and $\mathrm{ord}_\infty(y)=\mathrm{ord}_T(y/z)=\mathrm{ord}_Q(1/w)=-2g-1$.

Corollary 4.9. *Let $P=(a,0)$ be a special point on $\mathcal{H}$. Then $\mathrm{ord}_P(x-a)=2$. In other words, $x-a=y^2g(x,y)$ where $g(P)\neq 0,\infty$.*

Proof. It is clear from the proof of Theorem 4.8. □

Corollary 4.10. *Let $f\in k(\mathcal{H})^\times$ be a rational function. Then f has a finite number of zeros and poles, and $\sum_{P\in\mathcal{H}}\mathrm{ord}_P(f)=0$.*

Proof. It suffices to prove the statement for polynomial functions. Let $l(x)=x-a\in k[x]$, and let $P\in\mathcal{H}$ be a point with a as its x-coordinate. Then $l(x)$ has only one pole of order 2 at ∞. If P is an ordinary point then $l(x)$ has a simple zero at P, and a simple zero at $\tilde{P}$; otherwise $l(x)$ has a double zero at P. Consequently, any polynomial $l(x)\in k[x]$ has a finite number of zeros and poles such that if $\deg l(x)=n$ then $\sum_{P\in\mathcal{H}\setminus\{\infty\}}\mathrm{ord}_P(f)=2n$, and $\mathrm{ord}_\infty(f)=-2n$.
Let $g\in\Gamma(\mathcal{H})$ be a nonzero polynomial function. Then we can write $g=a(x)+yb(x)$ for some polynomials $a,b\in k[x]$. Let $\bar{g}=a(x)-yb(x)$. Since the mapping $\varphi:\mathcal{O}_P(\mathcal{H})\to\mathcal{O}_{\tilde{P}}(\mathcal{H})$, $\varphi(f)=\bar{f}$ is an isomorphism, we have $\mathrm{ord}_P(g)=\mathrm{ord}_{\tilde{P}}(\bar{g})$ for all $P\in\mathcal{H}$, and hence $\sum_{P\in\mathcal{H}}\mathrm{ord}_P(g)=\sum_{P\in\mathcal{H}}\mathrm{ord}_{\tilde{P}}(\bar{g})=\sum_{P\in\mathcal{H}}\mathrm{ord}_P(\bar{g})$. But $g\bar{g}\in k[x]$, and by above, both g and $\bar{g}$ have finite number of zeros and poles, and

$$\sum_{P\in\mathcal{H}}\mathrm{ord}_P(g)=\frac{1}{2}\left(\sum_{P\in\mathcal{H}}\mathrm{ord}_P(g)+\sum_{P\in\mathcal{H}}\mathrm{ord}_P(\bar{g})\right)=\frac{1}{2}\sum_{P\in\mathcal{H}}\mathrm{ord}_P(g\bar{g})=0. \qquad \square$$

4.3 Divisors

A *divisor* D on a hyperelliptic curve $\mathcal{H}$ is a formal sum $D=\sum_{P\in\mathcal{H}}n_pP$ where $n_P\in\mathbb{Z}$, and $n_P=0$ for almost all $P\in\mathcal{H}$. Therefore, the set $\mathbf{D}$ of all divisors D on $\mathcal{H}$ is a free $\mathbb{Z}$-module,

i.e. a free abelian group. The *degree* of a divisor D is defined to be $\deg(D) = \sum_{p\in\mathcal{H}} n_P \in \mathbb{Z}$. For divisors D_1 and D_2, we clearly have $\deg(D_1 + D_2) = \deg(D_1) + \deg(D_2)$. We denote by $\mathbf{D}^0$ the set of all divisors of degree zero, which is clearly a subgroup of $\mathbf{D}$.

Definition 4.11. For divisors $D_1 = \sum_{P\in\mathcal{H}} m_P P$ and $D_2 = \sum_{P\in\mathcal{H}} n_P P$ we say $D_1 \geq D_2$ if $m_P \geq n_P$ for all $P \in \mathcal{H}$. We also define the greatest common divisor of D_1 and D_2 as

$$\gcd(D_1, D_2) = \sum_{P\in\mathcal{H}} \min(n_P, m_P)(P - \infty)$$

Let $f \in k(\mathcal{H})^\times$ be a rational function. Then define the divisor of f to be $\operatorname{div}(f) = \sum_{P\in\mathcal{H}} \operatorname{ord}_P(f)P$, which is well defined, and has degree zero by Corollary 4.10. For example, for a finite point $P = (a, b) \in \mathcal{H}$, $\operatorname{div}_P(x-a) = P+\tilde{P}-2\infty$; Because if P is special then $x-a$ has a double zero at $P = \tilde{P}$, otherwise, it has a simple zero at P, and simple zero at $\tilde{P}$. For every $f_1, f_2 \in k(\mathcal{H})$ we have $\operatorname{div}(f_1 f_2) = \operatorname{div}(f_1)+\operatorname{div}(f_2)$, and $\operatorname{div}(f_1/f_2) = \operatorname{div}(f_1)-\operatorname{div}(f_2)$. These are simply inherited from the order function. A divisor $D \in \mathbf{D}$ is said to be a *principal divisor* if there is a rational function $f \in k(\mathcal{H})$ such that $D = \operatorname{div}(f)$.

Proposition 4.12. *Let $f_1, f_2 \in k(\mathcal{H})^\times$ be rational functions. Then*

1. $\operatorname{div}(f_1) \geq 0 \Leftrightarrow f_1 \in k$.
2. $\operatorname{div}(f_1) = \operatorname{div}(f_2) \Leftrightarrow f_1 = cf_2$ *for some* $c \in k$.

Proof. Since $\operatorname{div}(f_1) \geq 0$, $f_1 \in \mathcal{O}_P(\mathcal{H})$ for all $P \in \mathcal{H}$. Let $f_1(P) = c \in k$ for some $P \in \mathcal{H}$. Then we still have $\operatorname{div}(f_1 - c) \geq 0$, but $\deg(\operatorname{div}(f_1 - c)) > 0$ which is a impossible by Corollary 4.10 unless $f_1 - c = 0$ hence $f_1 = c$. This proves part 1. For part 2, we have $\operatorname{div}(f_1) = \operatorname{div}(f_2) \Leftrightarrow \operatorname{div}(f_1/f_2) = 0 \Leftrightarrow f_1/f_2 \in k$ by part 1. □

Definition 4.13. The set of all principal divisors is a subgroup of $\mathbf{D}^0$ denoted by $\mathbf{P}$. The quotient group $J(\mathcal{H}) = \mathbf{D}^0/\mathbf{P}$ is called the **jacobian** of $\mathcal{H}$. An element $D \in J(\mathcal{H})$ is called a divisor class of degree zero.

Let $D \in \mathbf{D}$, and define

$$\mathcal{L}(D) = \{f \in k(\mathcal{H}) \mid \operatorname{div}(f) + D \geq 0\} \cup \{0\}$$

It can easily be verified that $\mathcal{L}(D)$ is a vector space over k. Let $\ell(D) = \dim_k \mathcal{L}(D)$.

Lemma 4.14. *Let $D \in \mathbf{D}$ be a divisor, and let $P \in \mathcal{H}$. Then $\dim_k(\mathcal{L}(D + P)/\mathcal{L}(D)) \leq 1$.*

Proof. It clear that $\mathcal{L}(D) \subseteq \mathcal{L}(D+P)$. Let α be a uniformizer for $\mathcal{O}_P(\mathcal{H})$, and let n_P be the coefficient of P in D. Define the mapping $\phi : \mathcal{L}(D + P) \to k$ by $\phi(f) = (\alpha^{n_P+1} f)(P)$. Since $f \in \mathcal{L}(D+P)$, $\operatorname{ord}_P(f) \geq -n_P-1$ hence ϕ is well-defined. It can easily be verified that ϕ is a k-linear map with $\ker\phi = \mathcal{L}(D)$. Therefore, there exists an embedding $\mathcal{L}(D+P)/\mathcal{L}(D) \to k$, and the result follows. □

Proposition 4.15. *Let $D, D_1, D_2 \in \mathbf{D}$ be divisors on $\mathcal{H}$. Then*

1. $\mathcal{L}(0) = k$. *Also* $\mathcal{L}(D) = 0$ *if* $\deg(D) < 0$.

2. *If* $D_1 \leq D_2$ *then* $\mathcal{L}(D_1) \subseteq \mathcal{L}(D_2)$, *and* $\dim_k(\mathcal{L}(D_2)/\mathcal{L}(D_1)) \leq \deg(D_2 - D_1)$.

3. *If* $D_1 \equiv D_2 \pmod{\mathbf{P}}$ *then* $\mathcal{L}(D_1) \cong \mathcal{L}(D_2)$.

4. $\ell(D) < \infty$. *Moreover, if* $\deg(D) \geq 0$ *then* $\ell(D) \leq \deg(D) + 1$.

Proof. 1. We have $f \in \mathcal{L}(0) \Leftrightarrow \operatorname{div}(f) \geq 0 \Leftrightarrow f \in k$ By Proposition 4.12.(1). Let $\deg(D) < 0$, and $f \in \mathcal{L}(D)$. Then $\operatorname{div}(f) + D \geq 0$, so $0 \leq \deg(\operatorname{div}(f) + D) = \deg(\operatorname{div}(f)) + \deg(D) = \deg(D) < 0$ which impossible unless $f = 0$.
2. If $f \in \mathcal{L}(D_1)$ then $\operatorname{div}(f) + D_2 \geq \operatorname{div}(f) + D_1 \geq 0$, so $f \in \mathcal{L}(D_2)$ hence $\mathcal{L}(D_1) \subseteq \mathcal{L}(D_2)$. Let $D_2 = D_1 + P_1 + P_2 + \cdots + P_n$ where P_i are not necessarily distinct. Using induction and Lemma 4.14 we have $\dim_k(\mathcal{L}(D_2)/\mathcal{L}(D_1)) \leq n = \deg(D_2 - D_1)$.
3. Assume $D_1 = D_2 + \operatorname{div}(f)$ for some $f \in k(\mathcal{H})^\times$. Then the mapping $\phi : \mathcal{L}(D_1) \to \mathcal{L}(D_2)$, $\phi(g) = fg$ is an isomorphism of vector spaces.
4. By part 1, we my assume $\deg(D) \geq 0$. Let $P \in \mathcal{H}$ be an arbitrary point, and let $D_3 = D - (\deg(D)+1)P$. Then $\mathcal{L}(D_3) = 0$ by part 1, and hence $\ell(D) = \dim_k(\mathcal{L}(D)/\mathcal{L}(D_3)) \leq \deg(D) + 1$ by part 2. □

Definition 4.16. Let $D = \sum_{P \in \mathcal{H}} n_P P$ be a divisor. Then the *support* of D is defined to be $\operatorname{supp}(D) = \{P \in \mathcal{H} \mid n_P \neq 0\}$. A *semi-reduced* divisor is a divisor of the form $\sum_{i=1}^{k} n_i(P_i - \infty)$ such that

(i) $n_i \geq 0$ for all i,

(ii) if P_i is a special point then $n_i \leq 1$,

(iii) if $P_i \in \operatorname{supp}(D)$ then $\tilde{P}_i \notin \operatorname{supp}(D)$.

If we also have $\sum_{i=1}^{k} n_i \leq g$, where g is the genus of $\mathcal{H}$, then D is said to be a *reduced* divisor.

Lemma 4.17. *For every divisor* $D = \sum_{i=1}^{k} n_i P_i \in \mathbf{D}^0$ *there is a semi-reduced divisor* $D_1 = \sum_{i=1}^{\ell} m_i(P_i - \infty)$ *such that* $D_1 \equiv D \pmod{\mathbf{P}}$. *Moreover,* $\sum_{i=1}^{\ell} m_i \leq \sum_{P_i \in \operatorname{supp}(D) \setminus \infty} |n_i|$.

Proof. Let $D = \sum_{P \in \mathcal{H}} n_P P$. For every finite point $P = (a, b) \in \operatorname{supp}(D)$, if $n_P < 0$ then the term $n_p P$ can be eliminated by subtracting the principal divisor $\operatorname{div}((x - a)^{n_P}) = n_P(P + \tilde{P} - 2\infty)$ from D. So, we obtain a divisor $D_1 = \sum m_i(P - \infty) \in \mathbf{D}^0$ such that $m_i \geq 0$ for all i, and $D_1 \equiv D \pmod{\mathbf{P}}$. Now, for every finite point $P = (a, b) \in \operatorname{supp}(D_1)$ we do the following. If P is a special point then we subtract $\operatorname{div}((x - a)^{(m_P - \delta)/2})$, where $\delta \equiv m_P \pmod 2$, from D_1. Otherwise, if $\tilde{P} \in \operatorname{supp}(D_1)$ then we subtract $\operatorname{div}((x - a)^r)$, where $r = \min(m_{\tilde{P}}, m_P)$, from D_1. This way, we obtain a divisor $D_2 \equiv D_1 \pmod{\mathbf{P}}$ which is clearly a semi-reduced divisor. The second part is clear by the construction. □

Proposition 4.18. *For every polynomial* $g(x) \in k[x]$ *the divisor* $\operatorname{div}(g(x) - y)$ *is semi-reduced. Moreover, if* $g(x)^2 - f(x) = \prod_{i=1}^{n}(x - a_i)^{m_i}$ *then* $\operatorname{div}(g(x) - y) = \sum_{i=1}^{n} m_i(P_i - \infty)$ *where* $P_i = (a_i, g(a_i))$.

Proof. We have

$$\begin{aligned}\operatorname{div}(g(x)-y)+\operatorname{div}(g(x)+y) &= \operatorname{div}(g(x)^2-f(x)) = \operatorname{div}(\prod_{i=1}^{n}(x-a_i)^{m_i}) \\ &= \sum_{i=1}^{n}\operatorname{div}((x-a_i)^{m_i}) = \sum_{i=1}^{n} m_i \operatorname{div}(x-a_i) \\ &= \sum_{i=1}^{n} m_i(P_i+\tilde{P}_i-2\infty)\end{aligned}$$

where $P_i=(a_i,g(a_i))$. If P_i is an ordinary point then P_i is a zero of $g(x)-y$ if and only if $\tilde{P}_i$ is a zero of $g(x)+y$, and P_i or $\tilde{P}_i$ can not be a zero of both $g(x)-y$ and $g(x)+y$. If P_i is a special point, i.e. $P_i=(a_i,0)$, then $g(a_i)=f(a_i)=0$ so that $g(x)$ has a double zero at P_i. Since y has a simple zero at P_i, $g(x)-y$ has a simple zero at P_i. Also we have $(x-a_i)^2 \nmid g(x)^2-f(x)$, because otherwise, since $(x-a_i)^2 \mid g(x)^2$, we have $(x-a_i)^2 \mid f(x)$ which is a contradiction because $f(x)$ has no multiple roots. Thus, $m_i=1$. Putting all this together yields $\operatorname{div}(g(x)-y)=\sum_{i=1}^{n} m_i(P_i-\infty)$, and $\operatorname{div}(g(x)+y)=\sum_{i=1}^{n} m_i(\tilde{P}_i-\infty)$ where $m_i=1$ if P_i is a special point. □

Theorem 4.19 (**Riemann-Roch**). *For any algebraic curve C, there exists a divisor ω and an integer g such that for any divisor D on C,*

$$\ell(D)=\deg(D)-g+1+\ell(\omega-D)$$

Proof. See [49, ch. 1] or [10, ch. 12]. □

Remark. The divisor ω is the divisor of a differential on C, and g is called the genus of C.

Theorem 4.20. *For any divisor $D \in \mathbf{D}^0$, there exists a unique reduced divisor D_1 such that $D_1 \equiv D \pmod{\mathbf{P}}$.*

Proof. (Existence) By Theorem 4.19, $\ell(D) \geq \deg(D)-g+1$ for any divisor D.[1] Replacing D by $D+g\infty$ we have $\ell(D+g\infty) \geq g-g+1=1$. This means that there is a rational function $f \in k(\mathcal{H})^{\times}$ such that $\operatorname{div}(f)+D+g\infty \geq 0$. Let $D_1=\operatorname{div}(f)+D$. Then $D_1+g\infty \geq 0$ which means that D_1 is of the form $\sum_i n_i(P_i-\infty)$ with $n_i \geq 0$, and $\sum_i n_i = g$. The result now follows from Lemma 4.17.
(Uniqueness) Letting $D=0$ in Theorem 4.19, we have $\ell(0)=0-g+1+\ell(\omega)$. By Proposition 4.15.(1), $\ell(0)=1$ hence $\ell(\omega)=g$. Similarly, $\ell(\omega)=\deg(\omega)-g+2$ by setting $D=\omega$. Therefore, $\deg(\omega)=2g-2$. Now, let $D_1 \equiv D_2 \pmod{\mathbf{P}}$ be two reduced divisors. Assume that $D_1 \neq D_2$, and let $D=D_1-D_2$ be a principal divisor. By Lemma 4.17, there is a principal divisor $D_3 \equiv D \pmod{\mathbf{P}}$ with $D_3+2g\infty \geq 0$. It can easily be verified that $D_3 \neq 0$ by the proof of Lemma 4.17. Let $\operatorname{div}(f)=D_3$ for some $f \in k(\mathcal{H})^{\times}$. Then $f \in \mathcal{L}(2g\infty)$. Letting $D=2g\infty$ in Theorem 4.19, we have $\ell(2g\infty)=g+1-\ell(\omega-2g\infty)$. Since $\deg(\omega-2g\infty)=-2$ by above, $\ell(\omega-2g\infty)=0$ by Proposition 4.15.(1), and hence $\ell(2g\infty)=g+1$. But, $x^i \in \mathcal{L}(2g\infty)$ for all $i=0,\ldots,g$. Since $1,x,\ldots,x^g$ have poles of

[1] This is called Riemann inequality.

different order, they are linearly independent. So, they form a basis for $\mathcal{L}(2g\infty)$. This means that f is a function in x. If f in nonconstant then it has a root $a \in k$. Let $P = (a, b)$ be a point on $\mathcal{H}$. If P is ordinary then $\tilde{P}$ is also a zero of f hence $P, \tilde{P} \in \text{supp}(D_3)$ which is a contradiction, since D_3 is semi reduced. If P is special then f has a double zero at P hence the coefficient of P in D_3 is at least 2, contradiction again. Thus, f is constant hence $D_3 = 0$, a contradiction. □

Remark. The group $J(\mathcal{H})$ can be considered as an algebraic variety. It is, indeed, an abelian variety of dimension g called the jacobian variety. For any point $P \in \mathcal{H}$, the divisor $P - \infty$ is reduced. Therefore, by Theorem 4.20, the mapping

$$\begin{aligned} \varphi : \mathcal{H} &\longrightarrow J(\mathcal{H}) \\ P &\longmapsto P - \infty \end{aligned}$$

is an embedding of $\mathcal{H}$ into its jacobian. In the case of elliptic curves, this is clearly an isomorphism, hence an elliptic curve is an abelian variety of dimension 1. If ℓ is an integer such that $\text{char}(k) \nmid \ell$; then the ℓ-torsion subgroup of $J(\mathcal{H})$, denoted by $J(\mathcal{H})[\ell]$, is isomorphic to $(\mathbb{Z}/\ell\mathbb{Z})^{2g}$, see [17, ch. 4, 5]. When $g = 1$, i.e. $\mathcal{H}$ is an elliptic curve, this is a special case of Theorem 2.10. Analogous to the case of elliptic curves, there are *division polynomials* for hyperelliptic curves closely related to the torsion elements of $J(\mathcal{H})$. In 1994, Cantor [13] gave these polynomials defined by efficiently computable recurrences.

4.4 Mumford Representation

By Theorem 4.20, the elements of the group $J(\mathcal{H})$ are indeed reduced divisors. So, theoretically, given two reduced divisors D_1, D_2, we can add them, and reduce the result to obtain another reduced divisor. However, representing elements of $J(\mathcal{H})$ by divisor classes, i.e by formal sums of points on $\mathcal{H}$, is not computationally very useful. In this section, we present a concrete representation of the elements of $J(\mathcal{H})$ by pairs of polynomials, which was proposed by Mumford [67].

Let $P = (a, b)$ be a point on $\mathcal{H}$, and let $g \in k(\mathcal{H})$ be a rational function. Let $\mathcal{O}_P(\mathcal{H})$ be the local ring at P, and let t be a uniformizer for it. Then $g = t^m h$ for a unique, not necessarily positive, integer m, and rational function $h \in k(\mathcal{H})$ such that $h(P) \neq 0, \infty$. If h is not a constant then let $h(P) = c \in k$. Then $h(x, y) - c$ has a zero at P, so $h(x, y) - c = t^{m_1} h_1(x, y)$ for a unique integer $m_1 \geq 1$, and a rational function h_1 such that $h_1(P) \neq 0, \infty$. Hence $h(x, y) = c + t h_1(x, y)$. The same thing can be done to h_1, and so on. Therefore, for any given $k \geq m$, we can uniquely write

$$g(x, y) = \sum_{n=m}^{k-1} c_n t(x, y)^n + t(x, y)^k h_k(x, y) \tag{4.4}$$

where $m \in \mathbb{Z}$, and $h_k \in k(\mathcal{H})$ with $h_k(P) \neq 0, \infty$.

Proposition 4.21. *Let f be the polynomial in 4.2. For any ordinary point $P = (a, b) \in \mathcal{H}$, and any integer $k \geq 1$, there exists a unique polynomial $g \in k[x]$ such that $g(a) = b$, and $g(x)^2 \equiv f(x) \mod (x - a)^k$.*

Proof. Since P is ordinary, and y is a polynomial function, $m = 0$, and $t = (x - a)$ in (4.4). Thus, $y = \sum_{n=0}^{k-1} b_n(x-a)^n + (x-a)^k h(x,y)$ where $b_0 = b$. Let $g(x) = \sum_{n=0}^{k-1} b_n(x-a)^n$. Then $g(a) = b_0 = b$ and obviously $g(x)^2 \equiv y^2 \equiv f(x) \mod (x-a)^k$. □

Theorem 4.22. *Let $\mathcal{S}$ be the set of all semi-reduced divisors on $\mathcal{H}$, and let $\mathcal{P}$ be the set of pairs of of polynomials $(u, v) \in k[x] \times k[x]$ such that* (i) *u is monic, and* $\deg v < \deg u$ (ii) *$u \mid v^2 - f$. Then the following mapping is a bijection.*

$$\begin{aligned} \psi : \mathcal{P} &\longrightarrow \mathcal{S} \\ (u, v) &\longmapsto \gcd(\mathrm{div}(u), \mathrm{div}(v - y)) \end{aligned} \tag{4.5}$$

Proof. Let (u, v) be a pair of polynomials as in the theorem. By Proposition 4.18, $\mathrm{div}(v-y)$ is semi-reduced hence $\gcd(\mathrm{div}(u), \mathrm{div}(v-y))$ is semi-reduced.
ψ is **surjective**: Let $D = \sum_{i=1}^{k} m_i(P_i - \infty) \in \mathcal{S}$, where $P_i = (a_i, b_i)$, be a semi-reduced divisor. Let $u(x) = \prod_{i=1}^{k}(x - a_i)^{m_i}$. We will find a set of polynomials $v_i(x)$, $i = 1, \ldots, k$ such that $v_i(x)^2 \equiv f(x) \mod (x-a_i)^{m_i}$, and $v_i(a_i) = b_i$. If P_i is a special point then $m_i = 1$. Set $v_i(x) = 0$. Then we have $v_i(x)^2 = 0 \equiv f \mod (x - a_i)$, and $v_i(a_i) = 0 = b_i$. If P_i is ordinary then let $v_i(x)$ be the polynomial $g(x)$ in Proposition 4.21 with $k = m_i$. By the Chinese remaindering theorem, there exists a a unique polynomial $v(x) \in k[x]$ such that $v(x) \equiv v_i(x) \mod (x-a_i)^{m_i}$ for all i, and $\deg v(x) < \sum_{i=1}^{k} m_i = \deg u(x)$. We clearly have $u \mid v^2 - f$. Let $v(x)^2 - f(x) = \prod_{i=1}^{k}(x-a_i)^{m_i} \prod_{j=1}^{\ell}(x-c_j)^{n_i}$. Then by Proposition 4.18, $\mathrm{div}(v(x) - y) = D + \sum_{j=1}^{\ell} n_i(P_j - \infty)$ where $P_j = (c_j, v(c_j))$. Therefore, $D = \gcd(\mathrm{div}(u), \mathrm{div}(v-y))$ hence $\psi(u, v) = D$.
ψ is **injective**: Let $\psi(u_1, v_1) = \psi(u_2, v_2)$. By construction $u_1 = u_2$. We have $v_1(x) - v_2(x) = (v_1(x) - y) - (v_2(x) - y)$. So, $v_1(x) - v_2(x)$ vanishes at least to order m_i at P_i hence it has at least $\sum_{i=1}^{k} m_i = \deg u(x)$ zeros. This is a contradiction since $\deg(v_1(x) - v_2(x)) < \deg u(x)$. Therefore $v_1(x) - v_2(x)$ is the zero polynomial. □

A direct corollary of Theorem 4.20, and the proof of Theorem 4.22 is that every reduced divisor D, i.e. every element of $J(\mathcal{H})$, corresponds to a pair (u, v) as in (4.5), with $\deg u \leq g$. This pair is called the **Mumford Representation** of D.

4.5 Addition on the jacobian $J(\mathcal{H})$

Assume the elements of $J(\mathcal{H})$, i.e the reduced divisors, are given by their Mumford representations. In this section, we present an algorithm due to Cantor [12] for addition on $J(\mathcal{H})$ based on the Mumford representation. Let us first see how to obtain the reduced divisor corresponding to a given semi-reduced divisor.

Algorithm 4.5 Reduction of semi-reduced divisors

Input: A pair (u, v) representing a semi-reduced divisor D.
Output: A pair representing the reduced divisor $D_1 \equiv D \pmod{\mathbf{P}}$.

1. **while** $\deg(u) > g$ **do**
2. $\quad u_1 \leftarrow (v^2 - f)/u$
3. $\quad v_1 \leftarrow -v \mod u_1$

4. make u_1 monic by dividing it by its leading coefficient.
5. $u \leftarrow u_1$, $v \leftarrow v_1$.
6. **end while**
7. **return** (u, v)

Theorem 4.23. *Algorithm 4.5 works correctly.*

Proof. We first show that the terminates. Assume $\deg u \geq g+1$. Then $\deg(v^2 - f) \leq \max(\deg v^2, \deg f) < \max(\deg u^2, \deg u^2) = 2\deg u$. Thus, $\deg u_1 = \deg(v^2 - f) - \deg u < 2\deg u - \deg u = \deg u$. Therefore, the degree of u decreases strictly by each iteration, hence the while loop iterates finitely many times.
Let $D_1 = \gcd(\mathrm{div}(u_1), \mathrm{div}(v_1 - y))$ where u_1, v_1 are as in steps 2 and 3 of the algorithm repectively. Then u_1 is monic, and $\deg v_1 < \deg u_1$ by steps 3 and 4. Also $v_1^2 - f \equiv v^2 - f \equiv 0 \pmod{u_1}$. Therefore, D_1 is semi-reduced by Theorem 4.22.
For a divisor $D = \sum_{P \in \mathcal{P}} n_P P$, let $\widetilde{D} = \sum_{P \in \mathcal{P}} n_P \tilde{P}$. Then $\mathrm{div}(u) = D + \widetilde{D}$. By Proposition 4.18, $\mathrm{div}(v - y) = D + K$, and $\mathrm{div}(v + y) = \widetilde{D} + \widetilde{K}$, where $K = \sum_{i=1}^{\ell} k_i(P_i - \infty)$, for some $P_i \in \mathcal{H}$, and $k_i \geq 0$. Thus, $\mathrm{div}(u_1) = \mathrm{div}((v^2 - f)/u)) = \mathrm{div}(v^2 - f) - \mathrm{div}(u) = D + K + \widetilde{D} + \widetilde{K} - D - \widetilde{D} = K + \widetilde{K}$. Since $\mathrm{div}(v + y)$ is semi-reduced, $\mathrm{supp}\{\widetilde{D} \backslash \infty\} \cap \mathrm{supp}\{K \backslash \infty\} = \varnothing$, which implies that $\gcd(\mathrm{div}(u_1), \mathrm{div}(v + y)) = \widetilde{K}$ hence $D_1 = \gcd(\mathrm{div}(u_1), \mathrm{div}(v_1 + y)) = \widetilde{K}$. On the other hand, $D - D_1 = D - \widetilde{K} = \mathrm{div}(v - y) - K - \mathrm{div}(u_1) + K = \mathrm{div}(v - y) - \mathrm{div}(u_1)$ which means that $D_1 \equiv D \pmod{\mathbf{P}}$. □

Algorithm 4.6 Cantor's algorithm for adding reduced divisors

Input: Pairs (u_1, v_1), (u_2, v_2) representing reduced divisors D_1, and D_2 respectively.
Output: A pair (u, v) representing the reduced divisor $D \equiv D_1 + D_2 \pmod{\mathbf{P}}$.

1. compute d_1, r_1, r_2 such that $d_1 = \gcd(u_1, u_2)$, and $d_1 = r_1 u_1 + r_2 u_2$ using the extended Euclidean algorithm.
2. compute d, s_1, s_2 such that $d = \gcd(d_1, v_1 + v_2)$, and $d = s_1 d_1 + s_2(v_1 + v_2)$ using the extended Euclidean algorithm.
3. let $g_1 = s_1 r_1$, $g_2 = s_2 r_2$, and $g_3 = s_2$ so that $d = g_1 u_1 + g_2 u_2 + g_3(v_1 + v_2)$.
4. $u \leftarrow u_1 u_2 / d^2$.
5. $v \leftarrow (u_1 v_2 g_1 + u_2 v_1 g_2 + (v_1 v_2 + f) g_3)/d \mod u$
6. use Algorithm 4.5 to reduce (u, v)
7. **return** (u, v)

Theorem 4.24. *Algorithm 4.6 works correctly.*

Proof. The algorithm first generates a pair (u, v) representing a semi-reduced divisor $D \equiv D_1 + D_2 \pmod{\mathbf{P}}$, and then uses Algorithm 4.5 to obtain the desired reduced divisor. Here, we omit the lengthy proof, and refer the reader to [102, ch. 13] or [46, appendix]. □

4.6 Hyperelliptic curves over $\mathbb{F}_q$

In this section we assume the hyperelliptic curve $\mathcal{H}$, Equation (4.2), is defined over the finite field $k = \mathbb{F}_q$ where $q = p^n$ with $p \geq 3$ prime. Let ϕ_q be the q-th power Frobenius map

on $\mathcal{H}$. For a divisor $\sum_{P\in\mathcal{H}} n_P P$, we define $\phi_q(D) = \sum_{P\in\mathcal{H}} n_P \phi_q(P)$. Also for a rational function $g \in \overline{\mathbb{F}_q}(\mathcal{H})$, let g^{ϕ_q} be g with ϕ_q applied to its coefficients. A divisor D is said to be defined over $\mathbb{F}_q$ if $\phi_q(D) = D$. Consequently, a divisor class $D \in J(\mathcal{H})$ is defined over $\mathbb{F}_q$ if $\phi_q(D) \equiv D \pmod{\mathbf{P}}$. We denote by $J_{\mathbb{F}_q}(\mathcal{H})$, the elements of $J(\mathcal{H})$ defined over $\mathbb{F}_q$. The following is a consequence of Theorem 4.22 for the restriction $\mathbb{F}_q$ of $k = \overline{\mathbb{F}_q}$.

Proposition 4.25. *Let $\mathcal{P}_{\mathbb{F}_q}$ be the set of pairs of of polynomials $(u, v) \in \mathbb{F}_q[x] \times \mathbb{F}_q[x]$ such that* (i) *u is monic, and* $\deg v < \deg u \leq g$ (ii) $u \mid v^2 - f$. *Then the following mapping is a bijection.*

$$\begin{aligned} \psi_{\mathbb{F}_q} : \mathcal{P}_{\mathbb{F}_q} &\longrightarrow J_{\mathbb{F}_q}(\mathcal{H}) \\ (u, v) &\longmapsto \gcd(\operatorname{div}(u), \operatorname{div}(v - y)) \end{aligned} \tag{4.6}$$

Proof. Let $D \in J_{\mathbb{F}_q}(\mathcal{H})$, and let E be the unique reduced divisor in the class of D. Then $D - E = \operatorname{div}(g)$ for some function g, and hence $\phi_q(D) - \phi_q(E) = \operatorname{div}(\phi_q(F))$, which implies that $\phi_q(E)$ is in the class of $\phi_q(D)$. By uniqueness, $E = \phi_q(E)$. Let (u, v) be the Mumford representation of E. Then (u^{ϕ_q}, v^{ϕ_q}) is the Mumford representation of $\phi_q(E)$, hence $(u, v) = (u^{\phi_q}, v^{\phi_q})$. Therefore, $u, v \in \mathbb{F}_q[x]$. Conversely, let $u, v \in \mathbb{F}_q$, and let $D = \psi_{\mathbb{F}_q}(u, v)$. Then

$$\begin{aligned} \phi_q(D) &= \phi_q(\gcd(\operatorname{div}(u), \operatorname{div}(v - y))) = \gcd(\phi_q(\operatorname{div}(u)), \phi_q(\operatorname{div}(v - y))) \\ &= \gcd(\operatorname{div}(\phi_q(u)), \operatorname{div}(\phi_q(v - y))) = \gcd(\operatorname{div}(u), \operatorname{div}(v - y)) = D \end{aligned}$$ □

Since there are finitely many polynomials in $\mathbb{F}_q[x]$ of degree less than or equal to g, $J_{\mathbb{F}_q}(\mathcal{H})$ is a finite set. It is clear that $J_{\mathbb{F}_q}(\mathcal{H})$ is closed under addition and inversion, hence it is a group. As in the case of elliptic curves, ϕ_q is an inseparable endomorphism of $J(\mathcal{H})$ of degree q^g. We have $J_{\mathbb{F}_q}(\mathcal{H}) = \ker(\phi_q - 1)$, and hence $\#J_{\mathbb{F}_q}(\mathcal{H}) = \#\ker(\phi_q - 1) = \deg(\phi - 1)$. Denote by $\chi_q(t)$ the characteristic polynomial of ϕ_q. It can be shown that $\chi_q(t) \in \mathbb{Z}[t]$ is monic of degree $2g$, and that $\deg(\phi_q - 1) = \chi_q(1)$. For an integer n prime to p, the restriction $\phi_q|_{J(\mathcal{H})[n]}$ has the characteristic polynomial $\chi_q(t) \pmod{n}$; see [17, ch. 4, 5] or [66].
Let N_k denote the number of $\mathbb{F}_{q^k}$-rational points of $\mathcal{H}$. Define the generating function

$$Z(\mathcal{H}, t) = \exp\left(\sum_{k=1}^{\infty} \frac{N_k}{k} t^k\right) \tag{4.7}$$

which is called the **zeta function** of $\mathcal{H}$ over $\mathbb{F}_q$. The function $Z(\mathcal{H}, t)$ is of fundamental importance in the theory of hyperelliptic curves over finite fields. The following theorem states some known properties of this function.

Theorem 4.26 (**Weil Conjectures**). *Let $\mathcal{H}$ be a hyperelliptic curve of genus g over $\mathbb{F}_q$, and $Z(\mathcal{H}, t)$ be the zeta function of $\mathcal{H}$.*

1. *$Z(\mathcal{H}, t) \in \mathbb{Q}[[t]]$ is a rational function.*
2. *$Z(\mathcal{H}, t)$ satisfies the functional equation*

$$Z\left(\mathcal{H}, \frac{1}{qt}\right) = q^{1-g} t^{2-2g} Z(\mathcal{H}, t)$$

3. *We have*

$$Z(\mathcal{H},t)=\frac{L(t)}{(1-t)(1-qt)} \tag{4.8}$$

where $L(t)\in\mathbb{Z}[t]$ is of degree $2g$ such that $L(t)=\prod_{i=1}^{2g}(1-\alpha_i t)$ where α_i are algebraic integers such that $\alpha_{g+i}=\overline{\alpha}_i$ and $|\alpha_i|=\sqrt{q}$ for $i=1,\dots,g$.

Proof. See [89, ch. 5] or [96, part II]. □

Corollary 4.27. *$L(t)$ is of the form $L(t)=a_0+a_1t+\cdots+a_{2g}t^{2g}$ where $a_0=1, a_{2g}=q^g$, and $a_{2g-i}=q^{g-i}a_i$ for $i=0,\dots,g$.*

Proof. The functional equation of $Z(\mathcal{H},t)$, Theorem 4.26.(2), gives

$$L(t)=q^gt^{2g}L\left(\frac{1}{qt}\right)=\frac{a_{2g}}{q^g}+\frac{a_{2g-1}}{q^{g-1}}t+\cdots+q^ga_0t^{2g}$$

and by Theorem 4.26.(3), $a_0=1$. □

Corollary 4.28. *Let α_i, $i=1,\dots,2g$ be as in Theorem 4.26.(3). Then $\#\mathcal{H}(\mathbb{F}_{q^n})=q^n+1-\sum_{i=1}^{2g}\alpha_i^n$.*

Proof. Taking the logarithm of both sides of (4.8), gives

$$\begin{aligned}\ln(Z(\mathcal{H},t))&=\sum_{i=1}^{2g}\ln(1-\alpha_it)-\ln(1-t)-\ln(1-qt)\\&=-\sum_{i=1}^{2g}\sum_{k=1}^{\infty}\frac{\alpha_i^kt^k}{k}+\sum_{k=1}^{\infty}\frac{t^k}{k}+\sum_{k=1}^{\infty}\frac{q^kt^k}{k}\\&=\sum_{k=1}^{\infty}\frac{1}{k}(q^k+1-\sum_{i=1}^{2g}\alpha_i^k)t^k\end{aligned}$$

Comparing this to (4.7) yields the result. □

It can be shown that $\chi_q(t)=t^{2g}L(1/t)$. Therefore, when $g=1$, Corollary 4.28 gives Theorem 3.7. By Corollary 4.27,

$$\chi_q(t)=t^{2g}+a_1t^{2g-1}+\cdots+a_{g-1}t^{g-1}+a_gt^g+a_{g-1}qt^{g-1}+\cdots+a_2q^{g-2}t^2+a_1q^{q-1}t+q^g \tag{4.9}$$

In particular $\chi_q(t)=\prod_{i=1}^{2g}(t-\alpha_i)$ which means that the eigenvalues of ϕ_q are α_i, $i=1,\dots,2g$. Therefore, the eigenvalues of $\phi_{q^k}=\phi_q^k$ are α_i^k, $i=1,\dots,2g$. Hence $\chi_{q^k}(t)=\prod_{i=1}^{2g}(t-\alpha_i^k)$. As mentioned earlier, $\#J_{\mathbb{F}_q}(\mathcal{H})=\chi_q(1)$, consequently,

$$\#J_{\mathbb{F}_{q^k}}(\mathcal{H})=\chi_{q^k}(1)=\prod_{i=1}^{2g}(1-\alpha_i^k) \tag{4.10}$$

Theorem 4.29 (**Weil bounds**). *Let $\mathcal{H}$ be a hyperelliptic curve of genus g over $\mathbb{F}_q$. Then*

$$|\#\mathcal{H}(\mathbb{F}_{q^n}) - (q^n + 1)| \leq 2g\sqrt{q^n}\,,$$
$$(\sqrt{q^n} - 1)^{2g} \leq \#J_{\mathbb{F}_{q^n}}(\mathcal{H}) \leq (\sqrt{q^n} + 1)^{2g}$$

Proof. By Corollary 4.28 and Theorem 4.26.(3),

$$|\#\mathcal{H}(\mathbb{F}_{q^n}) - (q^n + 1)| = \left|\sum_{i=1}^{2g} \alpha_i^n\right| \leq \sum_{i=1}^{2g} |\alpha_i^n| = \sum_{i=1}^{2g} |\alpha_i|^n = 2g\sqrt{q^n}$$

which proves the first part. For the second part, by (4.10) and Theorem 4.26.(3),

$$\#J_{\mathbb{F}_{q^k}}(\mathcal{H}) = \left|\prod_{i=1}^{2g}(1 - \alpha_i^k)\right| = \prod_{i=1}^{2g}\left|(1 - \alpha_i^k)\right| \leq \prod_{i=1}^{2g}(1 + \left|\alpha_i^k\right|) = (\sqrt{q^n} + 1)^{2g}$$

which establishes the upper bound. The lower bound is established the same way. □

Chapter 5

Fast Integer Matrix Multiplication

In this chapter, we describe and implement an algorithm for fast multiplication of matrices with integer entries of arbitrary length. Assume that we are able to multiply matrices with machine level integer entries efficiently. Then the idea is to find a multiplication preserving mapping that enables us to uniquely represent matrices with big entries by matrices with machine level entries. This mapping and its inverse should be efficiently computable. To this end, we present the modular representation technique for matrices. This is not a new idea, e.g. see [40] for the modular representation technique for arithmetic on large integers. We first describe the method, and then propose some practical improvements to it. In the last section, we describe two computational problems: modular composition, and power projection. We show how the new implementations of the solutions to these problems, based on the fast matrix multiplication, lead to major speed-ups.

5.1 Modular representation

Let R be a commutative ring with 1 and let $I_1, I_2, \ldots, I_k$ be ideals of R. Then there is a natural homomorphism

$$\begin{aligned} \varphi : R &\longrightarrow \textstyle\bigoplus_{m=1}^{k} R/I_m \\ a &\longmapsto (a + I_1, a + I_2, \cdots, a + I_k) \end{aligned}$$

If the ideals I_m are pairwise coprime i.e. $I_i + I_j = R$ whenever $i \neq j$ then φ is surjective with kernel $\prod_{m=1}^{k} I_m$ so that

$$\psi : R/\prod_{m=1}^{k} I_m \longrightarrow \bigoplus_{m=1}^{k} R/I_m$$

is an isomorphism [5]. Therefore, every element a in the left side ring can be uniquely represented as a k-tuple $(a_1, a_2, \cdots, a_k)$ in the right side ring. The k-tuple $(a_1, a_2, \cdots, a_k)$ is called the modular representation of a. Computing ψ is not usually computationally hard. One can compute ψ^{-1} as follows. Let $M_i = \prod_{j \neq i} I_j$. Then I_i and M_i are coprime. So, there are $x_i \in I_i, y_i \in M_i$ such that $x_i + y_i = 1$. Then $1 - y_i \in I_i$ and $y_i \in M_i \subseteq I_j$ for all $j \neq i$.

Now let $a = \sum_{m=1}^{k} a_m y_m$. It can be easily seen that $\varphi(a) = (a_1, a_2, \cdots, a_k)$. Therefore, multiplication in $R/\prod_{m=1}^{k} I_m$ can be carried out by multiplication in $\bigoplus_{m=1}^{k} R/I_m$ via the mapping ψ. This is done by Algorithm 5.7.

Algorithm 5.7 Multiplication in $R/\prod_{m=1}^{k} I_m$

Input: $a, b \in R/\prod_{m=1}^{k} I_m$
Output: the product ab
1. $(a_1, \cdots, a_k) \leftarrow \psi(a)$
2. $(b_1, \cdots, b_k) \leftarrow \psi(b)$
3. compute $(a_1 b_1, \cdots, a_k b_k)$
4. **return** $\psi^{-1}(a_1 b_1, \cdots, a_k b_k)$

Let $M_n(R)$ be the ring of $n \times n$ square matrices over R. The mapping ψ induces the isomorphism

$$\begin{array}{rcl} \Psi : M_n(R/\prod_{m=1}^{k} I_m) & \longrightarrow & \bigoplus_{m=1}^{k} M_n(R/I_m) \\ (a_{ij}) & \longmapsto & (([a + I_1]_{ij}), ([a + I_2]_{ij}), \cdots, ([a + I_k]_{ij})) \end{array}$$

where $([a + I_r]_{ij})$ is the matrix (a_{ij}) with entries reduced modulo I_r. This means we can have a multiplication algorithm similar to Algorithm 5.7 in $M_n(R/\prod_{m=1}^{k} I_m)$.

Algorithm 5.8 Multiplication in $M_n(R/\prod_{m=1}^{k} I_m)$

Input: $A, B \in M_n(R/\prod_{m=1}^{k} I_m)$
Output: the product AB
1. $(A_1, \cdots, A_k) \leftarrow \Psi(A)$
2. $(B_1, \cdots, B_k) \leftarrow \Psi(B)$
3. compute $(A_1 B_1, \cdots, A_k B_k)$
4. **return** $\Psi^{-1}(A_1 B_1, \cdots, A_k B_k)$

Now, consider the special case $R = \mathbb{Z}$. In this case, the ideals are principal and computing ψ, and ψ^{-1} reduces to the usual modular integer operations. Let $I_i = (p_i)$, $i = 1, \ldots, k$ with p_i prime; Then computing ψ^{-1} is equivalent to solving the system of congruences $x \equiv a_i b_i \pmod{p_i}$, $i = 1, \ldots, k$ which can be done as follows. Let $M = \prod_{i=1}^{k} p_i$, and $M_i = M/p_i$. Compute $t_i = M_i^{-1} \pmod{p_i}$. Then $x = \sum_{i=1}^{k} a_i t_i M_i \pmod{M}$ (see Algorithm 5.9). Since $\mathbb{Z}$ is a normed ring, we can talk about the sizes of the elements so that Algorithm 5.7 can be used in the following way. Let $a, b \in \mathbb{Z}$ be of arbitrary length. Chose p_i such that a_i and b_i fit into machine words. Compute machine level products $(a_1 b_1, \cdots, a_k b_k)$, and finally compute $\psi^{-1}(a_1 b_1, \cdots, a_k b_k)$ by Algorithm 5.9.
Accordingly, Algorithm 5.8 can be used in a similar way to multiply integer matrices. It first reduces matrices A, B to tuples $(A_1, \cdots, A_k), (B_1, \cdots, B_k)$ of matrices with entries that can fit into machine words. It then computes the matrix products $(A_1 B_1, \cdots, A_k B_k)$ and at last $\Psi^{-1}(A_1 B_1, \cdots, A_k B_k)$ by applying Algorithm 5.9 to corresponding entries of the matrices $A_i B_i$. In the case of matrices, the values M and M_i can be precomputed for the entire process. We shall discuss some optimization techniques in Section 5.2.

Algorithm 5.9 Computing ψ^{-1} in $\mathbb{Z}$

Input: $(a_1, \cdots, a_k) \in \bigoplus_{i=1}^{k} \mathbb{Z}_{p_i}$
Output: $a \in \mathbb{Z}$

1. $M \leftarrow \prod_{i=1}^{k} p_i$
2. **for** i from 1 to k **do**
3. $\quad M_i \leftarrow \prod_{j \neq i} p_i$
4. **end for**
5. **for** i from 1 to k **do**
6. $\quad t_i \leftarrow M_i^{-1} \mod p_i$
7. $\quad x \leftarrow x + a_i t_i M_i$
8. **end for**
9. **return** $x \mod M$

5.2 Implementation

The algorithm described in Section 5.1 for multiplication of integer matrices has been implemented and embedded into the NTL library [85]. It is a low level implementation with high level interfaces. For both compatibility and handling low level fast big integer arithmetic, the GMP library [92] is used. For multiplication of matrices with machine level entries, ATLAS library [91] is used. A simple parallelism scheme is used, which results in an essentially linear speed up. The number of processors the algorithm is allowed to use can be set for each call; otherwise, it uses all processors of the system. Various techniques, with different performances depending on the shapes of the inputs, for computing Ψ^{-1} have been implemented so one can set the computation mode for each call. Figure 5.1 compares the running times of the proposed matrix multiplication, in sequential mode, and the one built in Magma [93] on a Core 2 Quad machine. [1]

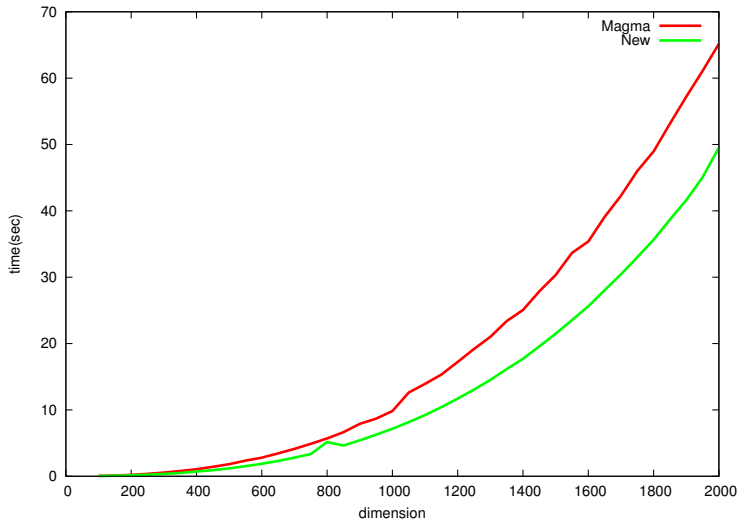

Figure 5.1: Matrix multiplication with bitsize 200

[1] Magma, which probably uses the same idea for matrix multiplication, is apparently the fastest computer algebra system in this respect.

A natural way of computing Ψ is to simply iterate over all entries of the input matrix and reduce each entry module each prime p_i by a fast reduction function. But a more clever way is to do reduction by matrix multiplication as follows. Let $v = (t_1, t_2, \cdots, t_n)$ be a row of the matrix A. Write $t_i = t_{i1}t_{i2}\cdots t_{ir}$ in a basis of the form $\omega = 2^d$. Then compute

$$C = \begin{pmatrix} \omega^r \pmod{p_1} & \omega^{r-1} \pmod{p_1} & \cdots & 1 \\ \omega^r \pmod{p_2} & \omega^{r-1} \pmod{p_2} & \cdots & 1 \\ \vdots & \vdots & \ddots & \vdots \\ \omega^r \pmod{p_k} & \omega^{r-1} \pmod{p_k} & \cdots & 1 \end{pmatrix} \begin{pmatrix} t_{11} & t_{21} & \cdots & t_{n1} \\ t_{12} & t_{22} & \cdots & t_{n2} \\ \vdots & \vdots & \ddots & \vdots \\ t_{1r} & t_{2r} & \cdots & t_{nr} \end{pmatrix}$$

The ith, $1 \leq i \leq k$, row of the matrix C is the row v reduced mod p_i. The same thing can be done for all rows of A to compute the tuple $(A_1, \ldots, A_k)$. The Vandermonde like matrix of ω^i is precomputed.
If the entries of the input matrices are very large then the value $x = \sum_{i=1}^{k} a_i t_i M_i$ in Step 5 of Algorithm 5.9 can be computed by a divide and conquer algorithm [99]: Let $\alpha = p_{k/2+1}p_{k/2+2}\cdots p_k$ and $\beta = p_1 p_2 \cdots p_{k/2}$ then

$$\begin{aligned} x &= \alpha(a_1 t_1 M_1/\alpha + \cdots + a_{k/2}t_{k/2}M_{k/2}/\alpha) \\ &\quad +\beta(a_{k/2+1}t_{k/2+1}M_{k/2+1}/\beta + \cdots + a_k t_k M_k/\beta) \end{aligned}$$

All the values α and β are precomputed. Another way of computing $x = \sum_{i=1}^{k} a_i t_i M_i$ is using a similar matrix multiplication technique as above with t_i replaced by M_i. Figure 5.2 shows a timing for different phases of Algorithm 5.8 where `reduction`, `residue mul`, and `inverse` are steps {1, 2}, 3, and 4 of the algorithm respectively.

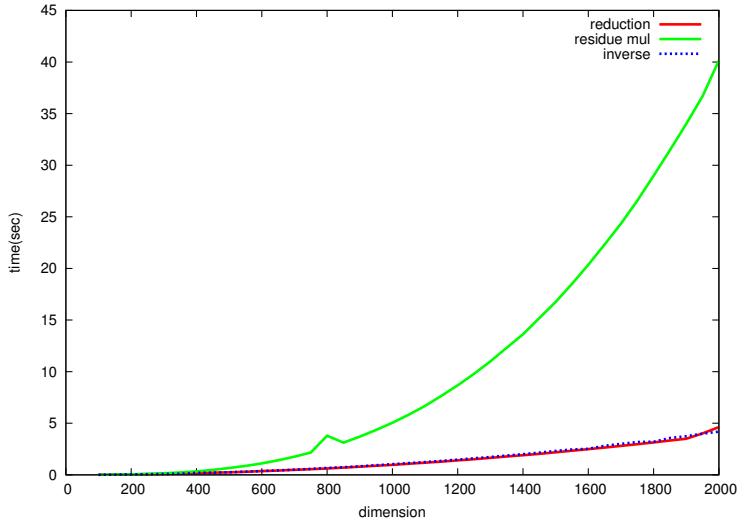

Figure 5.2: Multiplication phases with bitsize 200

5.3 Some computational speed-ups

Matrix multiplication is the building block of many computational algorithms. Therefore, having a fast matrix multiplication library results in a speed-up in many computational

areas. In this section, we present some specific problems, required in subsequent chapters, and show how to have major speed-ups by giving new implementations of the solutions based on matrix multiplication.

Modular Polynomial Composition. Let F be a field and let $p(x) = p_0+p_1x+\cdots+p_nx^n$, $q(x) = q_0+q_1x+\cdots+q_nx^n$, and $f(x)$ be polynomials in $F[x]$. Then the problem is to compute $p(q) \bmod f$. Here, we implement an algorithm by Brent and Kung [9] which was proposed for computation of the first $n \in \mathbb{N}_{\geq 1}$ coefficients of the composition of two formal power series. But it also works for polynomial composition modulo an arbitrary polynomial $f(x)$.

Algorithm 5.10 Modular Polynomial Composition

Input: polynomials $p, q, f \in F[x]$
Output: $q(p) \bmod f$

1. $k \leftarrow \lceil \sqrt{n+1} \rceil$
2. write $q(x) = Q_0(x) + Q_1(x)x^k + \cdots + Q_{k-1}(x)(x^k)^{k-1}$
3. compute $p^i(x)$, $i = 2, \ldots, k$
4. let $T(s) = p^k(x)$ and compute $T^i(x)$, $i = 2, \ldots, k-1$
5. compute $Q_i(p(x))$, $i = 0, \ldots, k-1$ from the Step 3.
6. compute $Q_i(p(x))T^i(x)$, $i = 1, \ldots, k-1$ from the steps 4 and 5.
7. compute $h(x) = \sum_{i=0}^{k-1} Q_i(p(x))T^i(x)$ from Step 6.
8. **return** $h(x)$

Note that all computations of Algorithm 5.10 are done modulo $f(x)$. Let $p^j(x) = \sum_{l=0}^{n} p_l^{(j)} x^l$, $j = 0, \ldots, k$. Then

$$Q_i(p(x)) = \sum_{j=0}^{k-1} q_{ik+j} \sum_{l=0}^{n} p_l^{(j)} x^l = \sum_{l=0}^{n} \left(\sum_{j=0}^{k-1} q_{ik+j} p_l^j \right) x^l$$

which is essentially the following matrix multiplication.

$$\begin{pmatrix} p_0^{(0)} & p_0^{(1)} & \cdots & p_0^{(k-1)} \\ p_1^{(0)} & p_1^{(1)} & \cdots & p_1^{(k-1)} \\ \vdots & \vdots & \ddots & \vdots \\ p_n^{(0)} & p_n^{(1)} & \cdots & p_n^{(k-1)} \end{pmatrix} \begin{pmatrix} q_0 & q_k & \cdots & q_{(k-1)k} \\ q_1 & q_{k+1} & \cdots & q_{(k-1)k+1} \\ \vdots & \vdots & \ddots & \vdots \\ q_{k-1} & q_{2k-1} & \cdots & q_{k^2-1} \end{pmatrix}$$

Therefore, Step 5 of Algorithm 5.10 can be done using matrix multiplication. Let assume that $O(n^\omega)$ is the achievable bound for multiplication of two $n \times n$ matrices. For the classical matrix multiplication $\omega = 3$, and for the best known algorithm $\omega = 2.37$ [19]. Except Step 5, all steps of the algorithm can be done in $O(kM(n)) = O(\sqrt{n}M(n))$ multiplications in F. Multiplication of the matrices of size $n \times \sqrt{n}$ and $\sqrt{n} \times \sqrt{n}$ in Step 5 is indeed equivalent to $\sqrt{n}$ multiplications of square matrices of dimension $\sqrt{n} \times \sqrt{n}$, which can be done in $O(\sqrt{n}n^{\omega/2}) = O(n^{(\omega+1)/2})$ operations in F. Therefore, the running time of Algorithm 5.10 is $O(\sqrt{n}M(n) + n^{(\omega+1)/2})$ operations in F. So, assuming $M(n) = O(n \log n \log\log n)$, and $\omega = 2.37$, the best running time for modular composition is $O(n^{1.69})$.[2] A new implementation

[2]Huang and Pan [35] showed that for the special dimensions $\sqrt{n} \times \sqrt{n}$ times $\sqrt{n} \times n$, $\omega \leq 1.67$. So for their algorithm $C(n) = O(n^{1.67})$.

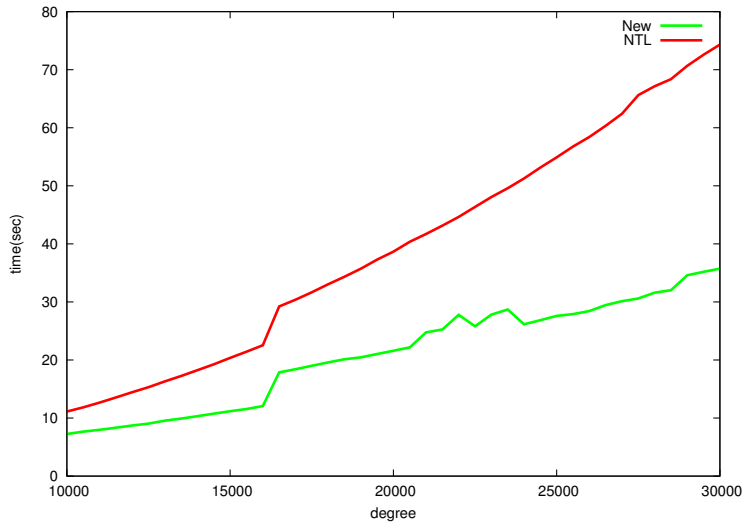

Figure 5.3: Modular Polynomial Composition

of the polynomial composition using our matrix multiplication algorithm has been embedded into NTL. Figure 5.3 compares the new and the old algorithm for modular polynomial composition in NTL.

Now that we have a fast modular polynomial composition we briefly show how to use it in polynomial factorization over finite fields. Let us first briefly review a factorization method. Let $f(x) \in \mathbb{F}_q[x]$ be of degree n, where $q = p^m$ and $\mathbb{F}_q$ is represented as the quotient $\mathbb{F}_p[y]/g(y)$ with $g(y) \in \mathbb{F}_p[y]$ monic irreducible of degree m. An efficient way of factoring f over $\mathbb{F}_q$ is breaking the factoring into three steps:

1. *squarefree factorization*: taking the square-free part of f;
2. *distinct-degree factorization*: splitting the square-free polynomial into polynomials whose irreducible factors have the same degree;
3. *equal-degree factorization*: completely factoring the square-free polynomial whose irreducible factors have the same degree.

The cost of Step 1, which is done by Yun's algorithm [107], is $O(M(n)\log n + n\log(q/p))$ operations in $\mathbb{F}_q$ which is dominated by the cost of other steps. So, let assume f is squarefree. The distinct-degree factorization is based on the fact that $x^{q^d} - x$ is the product of all monic irreducible polynomials of degree dividing d. So, taking $\gcd(x^{q^d} - x, f)$ isolates all factors of degree d of f. This idea is attributed to Arwin [4].

Algorithm 5.11 Distinct-degree factorization

Input: A squarefree monic polynomial $f \in \mathbb{F}_q[x]$ of degree n
Output: The distinct-degree decomposition of f

1. $f_0 \leftarrow f$, $s \leftarrow \lceil \frac{n}{2} \rceil$
2. **for** i from 1 to s **do**
3. $\quad h \leftarrow x^{q^i} \bmod f$
4. $\quad g_i \leftarrow \gcd(h - x, f_i)$

5. $f_i \leftarrow \frac{f_{i-1}}{g_i}$
6. **end for**
7. **return** $(g_1, \cdots, g_s)$

Each g_i produced by Algorithm 5.11 is a squarefree product of monic irreducible factors of degree i of f. The equal-degree factorization takes g_i and splits it into its irreducible factors. This is usually done recursively, i.e. the polynomial is split into two parts then the same is done for each of the two parts and so on. The idea, which is due to [11], is as follows. Let $f \in \mathbb{F}_q[x]$ be a squarefree monic polynomial of degree n with $\ell = n/d$ irreducible factors f_i, $i = 1, \ldots, \ell$ of degree d. Then

$$\mathbb{F}_q[x]/\langle f \rangle \cong \prod_{i=1}^{\ell} \mathbb{F}_q[x]/\langle f_i \rangle \cong \mathbb{F}_{q^d}^{\ell}$$

which induces the mapping

$$\begin{array}{rcl} \sigma : \mathbb{F}_q[x]/\langle f \rangle & \longrightarrow & \{-2, 0\}^{\ell} \\ a & \longmapsto & (a_1, \ldots, a_{\ell}) \end{array}$$

where $a_i = a^{(q^d-1)/2} - 1 \bmod f_i$. Assume $\beta \in \mathbb{F}_q[x]/\langle f \rangle$ is selected at random, and $\sigma(\beta) = (\beta_1, \ldots, \beta_{\ell})$. If $g = \gcd(\beta, f) \neq 1$ then g is a nontrivial factor of f; otherwise $\gcd(\beta, f) = 1$, and $\gcd(\beta^{(q^d-1)/2} - 1, f)$ is a nontrivial factor of f unless $\beta_1 = \beta_2 = \ldots, = \beta_{\ell}$ [3].

Algorithm 5.12 Equal-degree splitting

Input: A squarefree monic polynomial $f \in \mathbb{F}_q[x]$ of degree n and a divisor d of n such that all irreducible factors of f have degree d.

Output: a proper monic factor of f or "failure"

1. $\beta \leftarrow$ a nonconstant random element of $\mathbb{F}_q[x]$ of degree less than n
2. $g \leftarrow \gcd(\beta, f)$
3. **if** $g \neq 1$ **then**
4. **return** g
5. **end if**
6. $g \leftarrow \gcd(\beta^{(q^d-1)/2} - 1 \bmod f, f)$
7. **if** $g \neq 1, f$ **then**
8. **return** g
9. **else**
10. **return** "failure"
11. **end if**

The costliest part of Algorithm 5.12 is Step 6. We show how to compute $\beta^{(q^d-1)/2} - 1 \bmod f$ efficiently. Let $\lambda \in \mathbb{F}_q[x]$, and assume we are asked to compute $\alpha_k = \lambda^{1+p+p^2+\cdots+p^k} \bmod f$ for

[3] If β is selected uniformly at random then $\beta_i = -1$ or 1 with probability 1/2. So, $\beta_1 = \beta_2 = \ldots, = \beta_{\ell}$ occurs with probability $2^{-\ell+1}$.

a given integer $k > 0$. Let $\delta_i = \lambda^{p+p^2+\cdots+p^i} \bmod f$, $\zeta_i = x^{p^i} \bmod f$, and $\xi_i = y^{p^i} \bmod g(y)$. Then we have the following recurrence relations

$$\delta_i = \begin{cases} \delta_{i/2}\delta_{i/2}^{p^{i/2}} & \text{if } i \stackrel{2}{\equiv} 0 \\ \delta_1\delta_{i-1}^{p} & \text{if } i \stackrel{2}{\equiv} 1 \\ \delta_1 = \lambda^p \end{cases} \qquad \zeta_i = \begin{cases} \zeta_{i/2}^{p^{i/2}} & \text{if } i \stackrel{2}{\equiv} 0 \\ \zeta_{i-1}^{p} & \text{if } i \stackrel{2}{\equiv} 1 \\ \zeta_1 = x^p \end{cases} \qquad \xi_i = \begin{cases} \xi_{i/2}^{p^{i/2}} & \text{if } i \stackrel{2}{\equiv} 0 \\ \xi_{i-1}^{p} & \text{if } i \stackrel{2}{\equiv} 1 \\ \xi_1 = y^p \end{cases}$$

The initial values δ_1, ζ_1, and ξ_1 are computed using the usual exponentiation algorithm at a total cost of $O(M(n)M(m)\log p)$ operations in $\mathbb{F}_p$. Assume, inductively, that we have computed δ_r, ζ_r, and ξ_r. Then

$$\begin{aligned} \delta_r^{p^r} = \left(\sum_{j=0}^{n-1} a_j(y)x^j\right)^{p^r} &= \sum_{j=0}^{n-1} a_j(y)^{p^r}(x^j)^{p^r} = \sum_{j=0}^{n-1} a_j(y^{p^r})(x^{p^r})^j \\ &= \sum_{j=0}^{n-1} a_j(\xi_r)\zeta_r^j \bmod \langle f(x), g(y)\rangle \end{aligned}$$

This means that δ_{2r} and δ_{2r+1} can be computed using $O(n)$ modular compositions over $\mathbb{F}_p$, which totally costs $O(nC(m))$ operations in $\mathbb{F}_p$, and $O(1)$ modular multiplications and modular compositions over $\mathbb{F}_q$, which totally costs $O(C(n)M(m))$ operations in $\mathbb{F}_p$. The values ζ_{2r} and ζ_{2r+1} are computed similarly. The cost of computing ξ_{2r} and ξ_{2r+1} is dominated by the above. Therefore, δ_k and hence $\alpha_k = \lambda\delta_k$ can be computed using $O(\log k(M(n)M(m)\log p + nC(m) + C(n)M(m)))$ operations in $\mathbb{F}_p$. Now, let $\lambda = \beta^{(p-1)/2}$ then

$$\beta^{(q^d-1)/2} = \beta^{(p^{md}-1)/2} = \left(\beta^{(p-1)/2}\right)^{1+p+p^2+\cdots+p^{md-1}} = \lambda^{1+p+p^2+\cdots+p^{md-1}} \bmod \langle f(x), g(y)\rangle$$

which can be computed at a cost of $O(\log(md)(M(n)M(m)\log p + nC(m) + C(n)M(m)))$ operations in $\mathbb{F}_p$. The above idea of using modular composition for raising to the powers of the characteristic is essentially due to Kaltofen and Shoup [36].
To speed the distinct-degree factorization up, Algorithm 5.11, the values $x^q, x^{q^2}, \ldots, x^{q^s}$ can be computed at once. For this, we first compute $x^q = x^{p^m}$ using the above algorithm and then compute $x^{q^2}, \ldots, x^{q^s}$ using the algorithm introduced in [98].

Power Projection. Let $f \in \mathbb{F}_p[x]$ with $\deg f = n$. Given $g \in \mathbb{F}_p[x]/(f)$, and the vector $v \in \mathbb{F}_p^n$, the problem is to compute the sequence

$$\langle v, 1\rangle, \langle v, g\rangle, \ldots, \langle v, g^{m-1}\rangle \tag{5.1}$$

for a positive integer m; Here, $\langle ., .\rangle$ is the inner product operator. Let $h \circ v \in \mathbb{F}_p^n$ be the unique vector such that $\langle h \circ v, \alpha\rangle = \langle v, h\alpha\rangle$ for all $\alpha \in \mathbb{F}_p[x]/(f)$. Sequence (5.1) can be computed by Algorithm 5.13 due to Shoup [86]. Step 2 of the algorithm can be replaced by a matrix multiplication. As in the case of polynomial composition, this new power projection algorithm has been embedded into NTL. Figure 5.4 compares the performance of the old and the new algorithm.

Algorithm 5.13 Power Projection

Input: $v \in \mathbb{F}_p^n, f \in \mathbb{F}_p[x], g \in \mathbb{F}_p[x]/(f)$
Output: The sequence $\langle v, 1\rangle, \langle v, g\rangle, \ldots, \langle v, g^{m-1}\rangle$

1. $k \leftarrow \lfloor l^{1/2} \rfloor$, $k' \leftarrow \lceil l/k \rceil$
2. **for** i from 0 to $k' - 1$ **do**
3. $\quad c_{ik+j} \leftarrow \langle v, g^j \rangle \quad (0 \le j < k)$
4. $\quad v \leftarrow g^k \circ v$
5. **end for**
6. **return** $c_0, \ldots, c_{m-1}$

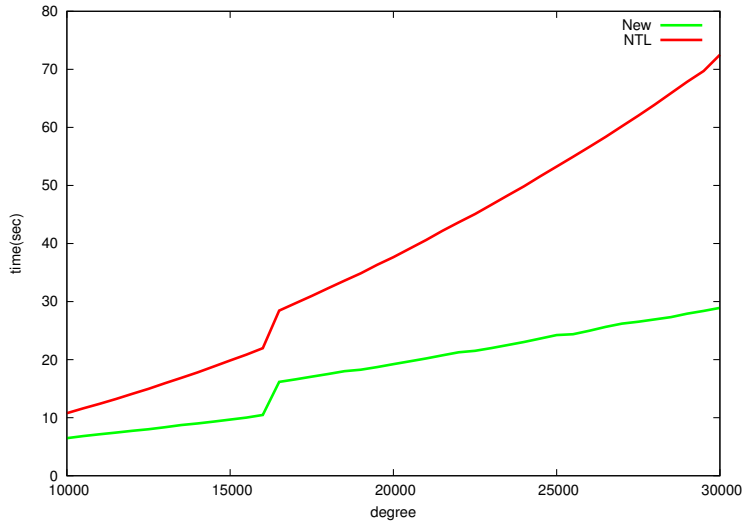

Figure 5.4: Power Projection

Chapter 6

Computing Roots Over Finite Fields

Let R be a ring and let $a \in R$. Computing $a^{1/n}$, where n is an integer, beside its intrinsic interest, is of great importance in many areas of mathematics and computer science. This problem can equivalently be considered as finding a root of $x^n - a \in R[x]$ in some extension of R. In this chapter, we focus on the cases where R is a finite field, and devote some more space to the case $n = 2$ which is of special interest in point counting and cryptography. The first two sections are preliminary for subsequent sections. We first present some square root algorithms, and then extend them to compute higher roots.

6.1 Discrete logarithm in cyclic p-groups

A finite p-group G is a group of order $n = p^m$ where p is a prime. A special case of the discrete logarithm problem occurring in root computation is the one in finite cyclic p-groups. The algorithm we present here is due to Pohlig and Hellman [71]. Let $g, a \in G$ with g a generator of G. The problem is to find the integer $0 \leq x \leq n-1$ such that $g^x = a$. Suppose that x has the expansion $x = \sum_{i=0}^{m-1} x_i p^i$, $0 \leq x_i \leq p-1$ in base p. Then

$$a^{\frac{n}{p}} = g^{x\frac{n}{p}} = (g^{\frac{n}{p}})^x = (g^{\frac{n}{p}})^{\sum_{i=0}^{m-1} x_i p^i} = (g^{\frac{n}{p}})^{x_0} = \zeta^{x_0} \tag{6.1}$$

where $\zeta = g^{\frac{n}{p}}$ is a primitive p-th root of unity. Therefore, x_0 is uniquely determined by (6.1). Let assume $x_1, \ldots, x_k$, $k < m$ are determined. Then

$$(ag^{-\sum_{i=0}^{k} x_i p^i})^{n/p^{k+2}} = (g^{\frac{n}{p}})^{x_{k+2}} = \zeta^{x_{k+2}}$$

so that x_{k+1} is uniquely determined, and hence x is determined by induction. When p is small, the values $0 \leq x_i \leq p-1$ can be found by a brute force search. But when p is large, one can use techniques like Shank's baby-step giant-step [82] and Pollard's rho method [72]. The baby-step giant-step algorithm needs memory for $O(\sqrt{p})$ group elements, and its running time is $O(\sqrt{p})$ group multiplications; while the running time of the Pollard's rho method is the same, but it requires a negligible amount of memory. The expected running time of the above algorithm, which requires m exponentiations and applications of, say Pollard's method, for discrete logarithm in G is $O(m(\log_2 n + \sqrt{p}))$ group multiplications.

6.2 Randomized search for irreducible polynomials

In this section, we briefly discuss the asymptotic probability for a randomly selected polynomial, with prescribed constraints, to be irreducible over a finite field. Let $N(n,q)$ denote the number of monic irreducible polynomials of degree n over $\mathbb{F}_q$. It is not hard to prove that

$$N(n,q) = \frac{1}{n}\sum_{d|n}\mu(d)q^{\frac{n}{d}}$$

where μ is the Möbius function. This formula was discovered by Gauss [31] for the case $q = p$. There are q^n monic polynomials of degree n over $\mathbb{F}_q$. So, the probability $P(n,q)$ for a uniformly random monic polynomial of degree n to be irreducible is $N(n,q)/q^n$. The following lemma shows that $P(n,q) \in \Theta(\frac{1}{n})$.

Lemma 6.1. *The number $N(n,q)$ of monic irreducible polynomials of degree n over $\mathbb{F}_q$ satisfies*

$$\frac{1}{n}q^n - \frac{q}{n(q-1)}(q^{\frac{n}{2}}) \leq N(q,n) \leq \frac{1}{n}(q^n - q) \qquad n \geq 2$$

with equality on the right if and only if n is prime.

Proof. The equality on the right is trivial when n is prime. By the Möbius inversion

$$q^n = \sum_{d|n} dN(d,q) = nN(n,q) + q + \sum_{\substack{d|n \\ d\neq 1,n}} dN(d,q) \geq nN(n,q) + q$$

which establishes the upper bound. Once we proved the upper bound it can be used to prove the lower bound:

$$\begin{aligned} q^n = \sum_{d|n} dN(d,q) &= nN(n,q) + \sum_{\substack{d|n \\ d\neq n}} dN(d,q) \\ &\leq nN(n,q) + \sum_{d=1}^{n/2} q^d = nN(n,q) + q\frac{q^{\frac{n}{2}}-1}{q-1} \end{aligned}$$ □

The coefficient of x^{n-1} of the monic polynomial f of degree n is called the trace of f. Let $N_\gamma(n,q)$ denote the number of polynomials of degree n and trace γ over $\mathbb{F}_q$. Carlitz [14] showed that

$$N_\gamma(n,q) = \frac{1}{qn}\sum_{\substack{d|n \\ p\nmid d}}\mu(d)q^{\frac{n}{d}} \qquad \gamma \neq 0$$

which means that $N_\gamma(n,q)$ does not depend on the trace γ. Let $n = p^k m$ with $p \nmid m$. Then

$$N_0(n,q) = \frac{1}{qn}\sum_{d|m}\mu(d)q^{\frac{n}{d}} - \frac{\varepsilon}{n}\sum_{d|m}\mu(d)q^{\frac{n}{dp}} \qquad \gamma \neq 0$$

where $\varepsilon = 1$ if $k > 0$ and $\varepsilon = 0$ if $k = 0$ [106]. Let $N(n,c,q)$ denote the number of monic irreducible polynomial of degree n and constant term c. Let $D_n = \{r : r \mid q^n - 1, r \nmid$

$q^m - 1$ for $m < n\}$, and let λ be the order of c. For each $r \in D_n$ let $r = m_r d_r$ where $d_r = \gcd(r, (q^n-1)/(q-1))$. An explicit formula for $N(n,c,q)$ was obtained in [106] as follows.

$$N(n,c,q) = \frac{1}{n\varphi(\lambda)} \sum_{\substack{r \in D_n \\ m_r = \lambda}} \varphi(r) \tag{6.2}$$

where φ is the Euler's function. Following the same notation as above let $N_\gamma(n,c,q)$ denote the number of irreducible polynomials of degree n, trace γ, and constant term c. The following bound was established by Wan [101]. See [64] for an improvement to this bound.

$$\left| N_\gamma(n,c,q) - \frac{q^{n-1}}{n(q-1)} \right| \leq \frac{3}{n} q^{n/2} \tag{6.3}$$

Let $P(n,c,q)$ denote the probability for a uniformly random monic polynomial of degree n and constant term c to be irreducible. Summing both sides of (6.2) over all elements of $\mathbb{F}_q$ we have

$$\begin{aligned} \frac{3}{n} q^{(n+2)/2} = \sum_{\gamma \in \mathbb{F}_q} \frac{3}{n} q^{n/2} &\geq \sum_{\gamma \in \mathbb{F}_q} \left| N_\gamma(n,c,q) - \frac{q^{n-1}}{n(q-1)} \right| \\ &\geq \left| \sum_{\gamma \in \mathbb{F}_q} N_\gamma(n,c,q) - \sum_{\gamma \in \mathbb{F}_q} \frac{q^{n-1}}{n(q-1)} \right| \\ &= \left| N(n,c,q) - \frac{q^n}{n(q-1)} \right| \end{aligned}$$

Since the number of all polynomials of degree n and a prescribed constant term is q^{n-1}, the above bound shows that $P(n,c,q) \in \Theta(\frac{1}{n})$. This means that for a monic polynomial $f \in \mathbb{F}_q[x]$ of degree n, if the constant term is fixed and all other coefficients are selected in a uniformly random way then there still is a reasonable chance for f to be irreducible. The surprising fact is that the above asymptotic results hold for some polynomials of very special form. Extensive research has been done on the number of irreducible binomials and trinomials over finite fields. These polynomials are very computationally useful, and result in simpler representations of extensions of finite fields.

Let $T_n(p)$ be the number of irreducible polynomials of the form $x^n + x + a \in \mathbb{F}_p[x]$ over $\mathbb{F}_p$. Then $T_n(p)$ is asymptotic to p/n for a fixed n and $p \to \infty$. This was first conjectured by Chowla [15]. The following more general result was proved by Cohen [18] and Ree [73].

Theorem 6.2. *For an integer n such that $p \nmid n(n-1)$, let $T_n(q)$ denote the number of trinomials $x^n + x + a \in \mathbb{F}_q[x]$ that are irreducible over $\mathbb{F}_q$. Then*

$$\left| T_n(q) - \frac{q}{n} \right| \leq C_n q^{\frac{1}{2}}$$

where C_n is a constant depending only on n.

6.3 General approaches

There are many polynomial factorization algorithms that can be used as general algorithms to find an nth root of an element of a finite field. See [100] for a survey of polynomial factorization over finite fields and [57] for special root finding algorithms based on factorization. For an element $a \in \mathbb{F}_q$ if $\sqrt[n]{a} \notin \mathbb{F}_q$ then there is a finite extension $\mathbb{F}_{q^m}/\mathbb{F}_q$ containing $\sqrt[n]{a}$. Therefore, given $a \in \mathbb{F}_q$, we can always assume that an n-th root of a is contained in $\mathbb{F}_q$, since a polynomial over $\mathbb{F}_q$ can always be viewed as a polynomial over $\mathbb{F}_{q^m}$. Let $f \in \mathbb{F}_q[x]$ be an arbitrary polynomial. To find the zeros of f in $\mathbb{F}_q$, it is sufficient to apply the equal degree factorization algorithm to $\gcd(x^q - x, f)$.

Algorithm 6.14 Root finding using factorization

Input: A nonconstant $f \in \mathbb{F}_q[x]$
Output: Zeros of f in $\mathbb{F}_q$

1. $h \leftarrow x^q \mod f$.
2. $g \leftarrow \gcd(h - x, f), d \leftarrow \deg g$
3. **if** $d = 0$ **then**
4. **return** $\varnothing$
5. **end if**
6. factorize g using equal degree factorization to get the linear factors $x - u_i$, $i = 1, \ldots, d$
7. **return** u_i, $i = 1, \ldots, d$

Algorithm 6.14 can be applied to the polynomial $x^n - a$ to compute an n-th root of a. It is essentially due to Legendre [52]. He suggested, in the case $q = p$, splitting $\gcd(f, x^{p-1} - 1)$ by computing $\gcd(f, x^{(p-1)/2} \pm 1)$ and substituting x by $x + a$ for a random $a \in \mathbb{F}_p$ in the case of a trivial split. The dominant steps of the algorithm are steps 1 and 2 which take $O(M(n) \log q)$ and $O((\log q + \log n) M(n) \log n)$ operations in $\mathbb{F}_q$ respectively. Therefore, the running time is $O(M(n) \log n \log(nq))$ or $\tilde{O}(n \log q)$ operations in $\mathbb{F}_q$.

6.4 Computing square roots

Let G be a group with an odd order n. Then the mapping $f : G \to G$, $f(x) = x^2$ is an automorphism of G, hence every element $x \in G$ has a unique square root which is $x^{(n+1)/2}$. For the cyclic group $\mathbb{F}_q^*$ if $q = 2^m$ then the square root of $x \in \mathbb{F}_q^*$ is $x^{2^{n-1}}$. In fact, this is true in general, i.e. for $q = p^n$ the p-th root of $x \in \mathbb{F}_q^*$ is $x^{p^{n-1}}$. If $q \equiv 3 \pmod 4$ then for any $x \in (\mathbb{F}_q^*)^2$ the square root is $x^{(q+1)/4}$. The latter is because $(\mathbb{F}_q^*)^2$ is a subgroup of odd order $(q-1)/2$.

An interesting field theoretic approach to computing square roots in $\mathbb{F}_q$ was introduced by Cipolla [16]. Let $K = \mathbb{F}_{q^m}$ be a finite extension of $\mathbb{F}_q$, and let $N_{K/\mathbb{F}_q} : K \to \mathbb{F}_q$ be the norm function $N_{K/\mathbb{F}_q}(\alpha) = \prod_{i=1}^{m-1} \alpha^{q^i} = \alpha^{(q^m-1)/(q-1)}$. Then $N_{K/\mathbb{F}_q}$ is surjective. The idea of Cipolla's algorithm is as follows. Let $a \in \mathbb{F}_q$, and assume we find a quadratic extension $K = \mathbb{F}_{q^2}$ of $\mathbb{F}_q$ by adjoining a quadratic nonresidue to $\mathbb{F}_q$. Then, there is an element $x \in K$ such that $N_{K/\mathbb{F}_q}(x) = a$. But $N_{K/\mathbb{F}_q}(x) = x^{q+1}$ hence $\sqrt{a} = x^{(q+1)/2}$.

Algorithm 6.15 Cipolla's square root

Input: A nonzero $a \in \mathbb{F}_q$
Output: Square root of a in $\mathbb{F}_{q^2}$

1. choose a random $b \in \mathbb{F}_q$
2. **if** $b^2 - 4a$ is a square **then**
3. **return** failure.
4. **end if**
5. $c \leftarrow x^{(q+1)/2} \mod x^2 - bx + a$
6. **return** c

If $\left(\frac{b^2-4a}{\mathbb{F}_q}\right) = -1$ then the polynomial $f(x) = x^2 - bx + a$ is irreducible over $\mathbb{F}_q$ hence $\mathbb{F}_q[x]/(f)$ is a field. Since f is the minimal polynomial of x over $\mathbb{F}_q$, $c^2 = N_{K/\mathbb{F}_q}(x) = a$. According to Section 6.2, finding a quadratic nonresidue of the form $b^2 - 4a$ by choosing random $b \in \mathbb{F}_q$, which is equivalent to choosing a uniformly random polynomial of degree 2 and constant term a, does not require too many trials. More precisely [6, page 158],

Lemma 6.3. *The probability of* $\left(\frac{b^2-4a}{\mathbb{F}_q}\right) = -1$ *for a randomly chosen* $b \in \mathbb{F}_q$ *is* $(q-1)/2q$.

Algorithm 6.15 fails with probability $(q+1)/2q$. The quadratic residue test, and Step 5 take $O(\log q)$ and $O(\log q)$ multiplications in $\mathbb{F}_q$ respectively. Therefore, its expected complexity is $O(\log q)$ multiplications in $\mathbb{F}_q$.

Another algorithm for computing square roots is the algorithm of Tonelli [94]. The algorithm, which is more group theoretic, was improved by Shanks [83] and is known as Tonelli-Shanks algorithm. The idea of the algorithm is to use discrete logarithm to reduce the problem to a subgroup of $\mathbb{F}_q^*$ of an odd order. Let $q - 1 = 2^r\ell$ with $(\ell, 2) = 1$. Let H be the unique subgroup of $\mathbb{F}_q^*$ of order ℓ. Then we have a chain of subgroups

$$H = H_0 \subset H_1 \subset \cdots \subset H_r = \mathbb{F}_q^*$$

where H_i/H_{i-1} is a simple group of order 2 for $i = 1, \ldots, r$. The natural homomorphism $\mathbb{F}_q^* \to \mathbb{F}_q^*/H$ sends any quadratic nonresidue of $\mathbb{F}_q^*$ to a generator of $\mathbb{F}_q^*/H$. Assume we find a quadratic nonresidue $g \in \mathbb{F}_q^*$. Then the square root of an element $a \in \mathbb{F}_q^*$ can be computed as follows. We can be express a as $g^t h \in g^t H$ by solving a discrete logarithm in $\mathbb{F}_q^*/H$. Now, t is necessarily even, so that $\sqrt{a} = g^{t/2}h^{(\ell+1)/2}$. Here is the algorithm.

Algorithm 6.16 Tonelli-Shanks square root

Input: A nonzero $a \in \mathbb{F}_q$ with q odd
Output: Square root of a in $\mathbb{F}_q$

1. choose a random $g \in \mathbb{F}_q$
2. **if** g is a square **then**
3. **return** failure.
4. **end if**
5. let $q - 1 = 2^r\ell$ with $2 \nmid \ell$.
6. let H be the subgroup of $\mathbb{F}_q^*$ of order ℓ
7. $t \leftarrow$ the discrete logarithm of aH in base gH
8. $h \leftarrow ag^{-t}$

9. **return** $g^{t/2}h^{(\ell+1)/2}$

According to Section 6.1, Step 7 of Algorithm 6.16 requires $O(r^2)$ multiplications in $\mathbb{F}_q$ where r is the highest power of 2 dividing $q-1$. All other steps take $O(\log q)$ multiplications in $\mathbb{F}_q$. Hence, the expected running time of the algorithm is $O(r^2+\log q)$ multiplications in $\mathbb{F}_q$. Despite Algorithm 6.15, the efficiency of this algorithm depends on the structure of $\mathbb{F}_q^*$. Let r and ℓ be as in Step 5 of the algorithm. For most q, r is fairly small [1], and Algorithm 6.16 requires few exponentiations in $\mathbb{F}_q^*$, and hence preferred over Algorithm 6.15 which requires exponentiation in $\mathbb{F}_{q^2}$. If 2^r is comparable to ℓ then the running time of Algorithm 6.16 is $O((\log q)^2)$. In this case, Algorithm 6.15 is preferable. The latter case can happen quite naturally:

Theorem 6.4. *Let a and b be positive coprime integers. Then there are infinitely many primes p such that $p \overset{b}{\equiv} a$.*

This is due to Dirichlet [20], and known as *Dirichlet's theorem on arithmetic progressions*; Because it equivalently says that if a and b are positive coprime integers then the arithmetic progression $a, a+b, a+2b, \ldots$ contains infinitely many primes. Let $p(a,b)$ be the least prime in this arithmetic progression. Linnik [58] proved that there is a constant $L>0$ such that $p(a,b) < b^L$. This constant is not too large, e.g. it is shown in [34] that $L \leq 11/2$. By Theorem 6.4, for any given integer $r>0$, there are infinitely many primes in the progression $1, 1+2.2^r, 1+3.2^r, \ldots, 1+k.2^r, \ldots$. Let q be the least prime in this sequence. Then $2^r \mid q-1$, and $q \leq 2^{11r/2}$, and hence $2\log q/11 \leq r$. This shows the bound $O((\log q)^2)$ for Algorithm 6.16 is tight.

Let $\mathcal{P} = \bigcup_{i\in\mathbb{N}} \{\text{the least prime } p_i \text{ such that } p_i \equiv 2^i+1 \bmod 2^{i+1}\}$. Then $\mathcal{P}$ is clearly not finite. Let $\{q_n\}_{n\in\mathbb{N}}$ be an increasing sequence of elements of $\mathcal{P}$, and let $C(q)$ and $T(q)$ be the expected complexity of algorithms 6.15 and 6.16 respectively, averaged over all quadratic residue and non-residue inputs. Then it is not hard to prove, see [95], that

$$\lim_{n\to\infty} \frac{T(q_n)}{C(q_n)} = \infty.$$

which means we have found an infinite sequence of primes for which Algorithm 6.15 is asymptotically better.

The last square root algorithm we present is a new algorithm based on the trace function. Assume that the field $\mathbb{F}_q$, where $q=p^n$, is represented, as usual, by a quotient $\mathbb{F}_p[x]/(f(x))$ with $f(x) \in \mathbb{F}_p[x]$ a monic irreducible polynomial of degree n. Let $T_{\mathbb{F}_q/\mathbb{F}_p} : \mathbb{F}_q \to \mathbb{F}_p$ be the trace function $T_{\mathbb{F}_q/\mathbb{F}_p}(\alpha) = \sum_{i=0}^{n-1} \alpha^{p^i}$ where $\alpha \in \mathbb{F}_q$. Given $a \in \mathbb{F}_q^\times$, let $\gamma \in \mathbb{F}_q$ be a square root of it. Then

$$\begin{aligned} \mathbb{F}_p \ni \beta = T_{\mathbb{F}_q/\mathbb{F}_p}(\gamma) = \sum_{i=1}^{n-1} \gamma^{p^i} &= \gamma(1+\gamma^{p-1}+\gamma^{p^2-1}+\cdots+\gamma^{p^{n-1}-1}) \\ &= \gamma(1+a^{(p-1)/2}+a^{(p^2-1)/2}+\cdots+a^{(p^{n-1}-1)/2}) \end{aligned} \tag{6.4}$$

[1] For example, primes used in public-key cryptography, see [60, 59].

Let $b = 1 + a^{(p-1)/2} + a^{(p^2-1)/2} + \cdots + a^{(p^{n-1}-1)/2}$. We may assume $b \neq 0$; because otherwise, we can start by ac^2 for different random elements $c \in \mathbb{F}_q^\times$ until we get a nonzero b, and at the end multiply the result by c^{-1}. Squaring both side of the Equation (6.4) results in the quadratic equation $\beta^2 = ab^2$ over $\mathbb{F}_p$ from which β can be determined. Then $\gamma = \beta b^{-1}$. Computing β from the above quadratic equation takes $O(\log p)$ operation in $\mathbb{F}_p$. Therefore, efficient computation of γ needs an efficient computation of b. For this, we use a recursive technique, similar to the one used in Section 5.3, as follows. Let $\lambda \in \mathbb{F}_q$, and assume we are asked to compute

$$\alpha_k = \lambda^{1+p} + \lambda^{1+p+p^2} + \cdots + \lambda^{1+p+p^2+\cdots+p^k}$$

for a given integer $k > 0$. Let $\delta_i = \lambda^p + \lambda^{p+p^2} + \cdots + \lambda^{p+p^2+\cdots+p^i}$, $\zeta_i = \lambda^{p+p^2+\cdots+p^i}$, and $\xi_i = x^{p^i}$. Then we have the following recurrence relations

$$\delta_i = \begin{cases} \delta_{i/2} + \zeta_{i/2}\delta_{i/2}^{p^{i/2}} & \text{if } i \overset{2}{\equiv} 0 \\ \delta_1 + \zeta_1\delta_{i-1}^p & \text{if } i \overset{2}{\equiv} 1 \\ \delta_1 = \lambda^p \end{cases} \qquad \zeta_i = \begin{cases} \zeta_{i/2}\zeta_{i/2}^{p^{i/2}} & \text{if } i \overset{2}{\equiv} 0 \\ \zeta_1\zeta_{i-1}^p & \text{if } i \overset{2}{\equiv} 1 \\ \zeta_1 = \lambda^p \end{cases} \qquad \xi_i = \begin{cases} \xi_{i/2}^{p^{i/2}} & \text{if } i \overset{2}{\equiv} 0 \\ \xi_{i-1}^p & \text{if } i \overset{2}{\equiv} 1 \\ \xi_1 = x^p \end{cases}$$

Assume, inductively, that we have computed δ_r, ζ_r, and ξ_r. Then $\delta_r^{p^r} = \left(\sum_{j=0}^{n-1} a_j x^j\right)^{p^r} = \sum_{j=0}^{n-1} a_j (x^j)^{p^r} = \sum_{j=0}^{n-1} a_j (x^{p^r})^j = \sum_{j=0}^{n-1} a_j \xi_r^j$ which means that raising δ_r to the power of p^r is indeed computing the modular polynomial composition $\delta_r \circ \xi_r$ over $\mathbb{F}_p$. Thus, ignoring the additions, computing δ_{2r} and δ_{2r+1} costs $O(1)$ polynomial multiplications and modular polynomial compositions over $\mathbb{F}_p$. The same can be done for computing ζ_{2r}, ζ_{2r+1}, ξ_{2r}, and ξ_{2r+1}. Therefore, δ_k and hence $\alpha_k = \lambda\delta_k$ can by computed using $O(C(n)\log k)$ operations in $\mathbb{F}_p$. Now, let $\lambda = a^{(p-1)/2}$, then $b = 1 + \lambda + \alpha_{n-2}$. Computing λ needs $O(M(n)\log p)$ operations in $\mathbb{F}_p$, and hence computing b needs $O(M(n)\log p + C(n)\log n)$ operations in $\mathbb{F}_p$. Thus, the expected running time of the above algorithm for computing a square of an element $a \in \mathbb{F}_q$ is $O(M(n)\log p + C(n)\log n)$ operations in $\mathbb{F}_p$.

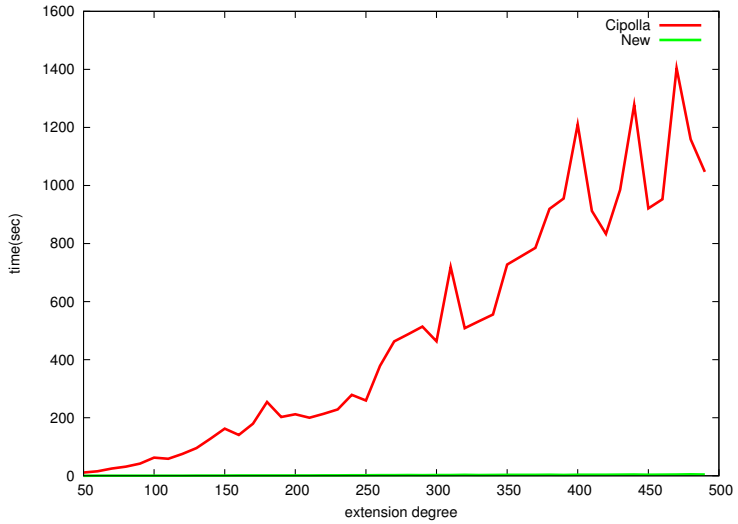

Figure 6.1: The new square root algorithm

We have implemented the above algorithm, and Algorithm 6.15 in NTL. Figure 6.1 compares the two algorithms in $\mathbb{F}_q$ with $q = p^n$, for a randomly selected prime $p = 348975609381470$

925634534573457497, and different values of the extension degree n. Since Figure 6.1 does not reveal the behaviour of the new algorithm, a better view of the asymptotic description of the algorithm for extension degrees ≤ 10000 is provided in Figure 6.2.

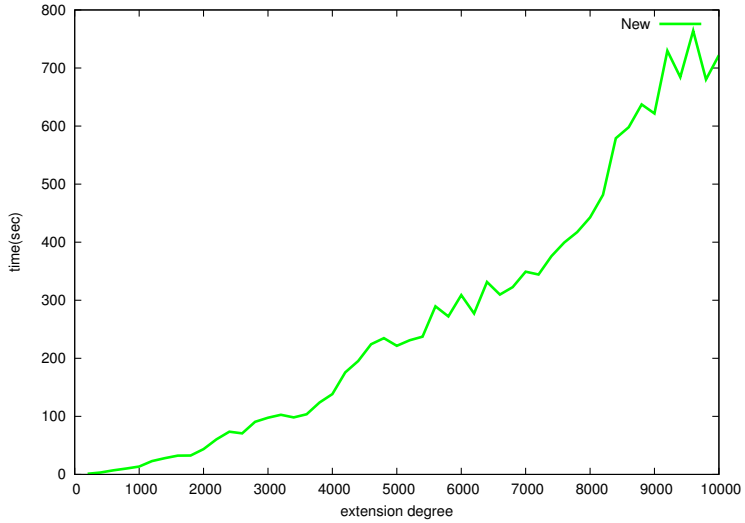

Figure 6.2: The new square root algorithm for high extensions

6.5 Computing higher roots

All of the algorithms presented in Section 6.4 for computing square roots can somehow be extended to compute m-th roots where $m \geq 3$ is an arbitrary integer. In this section, we present such extensions, but let us first make the following observation. Let G be a group of order n and let k be an integer such that $(n, k) = 1$. Then every $a \in G$ has a unique k-th root $b = a^{k^{-1} \bmod n}$ in G; for if c is another k-th root of a then $b^k = c^k$ so $(cb^{-1})^k = 1$ which implies $\mathrm{ord}(cb^{-1}) \mid k$ hence $\mathrm{ord}(cb^{-1}) \mid (k, n) = 1$. Therefore $cb^{-1} = 1$ hence $c = b$. Suppose we can find a t-th root of $a \in \mathbb{F}_q$ when t is a prime divisor of $q - 1$. Then computing an m-th root of a for an arbitrary m is as follows. Let $m = m_1 m_2$ with $(m_2, q - 1) = 1$, and $t \mid q-1$ for every prime divisor t of m_1. Then we can compute $a_0 = \sqrt[m_2]{a}$ by simply inverting $m_2 \bmod q - 1$. Let $m_1 = \prod_{i=1}^{s} p_i^{\alpha_i}$ be the prime factorization of m_1. Then we can compute $a_k = \sqrt[p_1]{a_{k-1}}$, $k = 1, \ldots, \alpha_1$ and hence $a_{\alpha_1} = \sqrt[p_1^{\alpha_1}]{a_0}$. The same process can be applied to compute $\sqrt[p_2^{\alpha_2}]{a_{\alpha_1}}$ and so on. Therefore, the problem is reduced to computing t-th roots when t is a prime divisor of $q - 1$, and so the algorithms we present in this section will compute t-th roots for such a t.

A natural extension of Algorithm 6.16 was introduced in [1]. Let t be a prime divisor of $q-1$ and let $q - 1 = t^r \ell$ such that $t \nmid \ell$. As before, let H be the unique subgroup of $\mathbb{F}_q^*$ of order ℓ. Then we have a chain of subgroups

$$H = H_0 \subset H_1 \subset \cdots \subset H_r = \mathbb{F}_q^*$$

where H_i/H_{i-1} is a simple group of order t for $i = 1, \ldots, r$. If g is not a t-th power then gH generates $\mathbb{F}_q^*/H$ so that a can be represented as $g^s h$ with $h \in H$. Since a is a t-th power, it

can easily be seen that $t \mid s$. On the other hand $(|H|, t) = 1$. Therefore, a t-th root of a is $g^{s/t} h^{t^{-1} \bmod \ell}$.

Algorithm 6.17 Tonelli-Shanks t-th root when t is a prime divisor of $q-1$

Input: A nonzero $a \in \mathbb{F}_q$ with q odd
Output: a t-th root of a in $\mathbb{F}_q$

1. choose a random $g \in \mathbb{F}_q$
2. **if** g is a t-th power **then**
3. **return** failure.
4. **end if**
5. let $q - 1 = t^r \ell$ with $t \nmid \ell$.
6. let H be the subgroup of $\mathbb{F}_q^*$ of order ℓ
7. $s \leftarrow$ the discrete logarithm of aH in base gH
8. $h \leftarrow a g^{-s}$
9. $u \leftarrow t^{-1} \bmod \ell$
10. **return** $g^{s/t} h^u$

By the following lemma, a randomly chosen $g \in \mathbb{F}_q$ is a t-th power with probability $1/t$. Therefore, Algorithm 6.17 fails with probability $1/t < 1/2$.

Lemma 6.5. *Let G be a cyclic group of order n. Then $a \in G$ is a d-th power if and only if $a^{n/(d,n)} = 1$.*

Proof. '$\Rightarrow$' is trivial. For the converse let g be a generator of G, and $a = g^\ell$. Then $1 = a^{n/(d,n)} = g^{\ell n/(d,n)}$. So, $n \mid \ell \frac{n}{(d,n)}$ and hence $(d,n) \mid \ell$. Therefore $a = g^\ell = g^{\ell_1 (d,n)} = g^{\ell_1 (r_1 n + s_1 d)} = (g^{\ell_1 s_1})^d$ as desired. □

The expected cost of finding a non t-th power is $O(t \log q)$ multiplications. Step 7 is done in $O(r(\log t^r + \sqrt{t}))$ operations, see Section 6.1, and the rest of the algorithm is accomplished in $O(\log q)$ operations. Therefore, the expected running time of Algorithm 6.17 is $O(t \log q + r^2 \log t + r\sqrt{t})$ multiplications in $\mathbb{F}_q$.
Next, we extend Algorithm 6.15 to compute t-th roots where t is a prime divisor of $q-1$. Given an element $a \in \mathbb{F}_q$ we can find a monic irreducible polynomial $f(x) \in \mathbb{F}_q[x]$ of degree t and constant term a by a random search. According to Section 6.2, this needs $\Theta(t)$ trials in average. The primality of t, and the following theorem result in a simple irreducibility test.

Lemma 6.6. *A monic polynomial $f \in \mathbb{F}_q[x]$ of degree $n \geq 1$ is irreducible if and only if*

I. f divides $x^{q^n} - x$

II. $(x^{q^{n/t}} - x, f) = 1$ for all prime divisors t of n.

Proof. It is well known that $x^{q^n} - x$ is the product of all monic irreducible polynomials over $\mathbb{F}_q$ of degree dividing n. In other words, a monic irreducible polynomial $f \in \mathbb{F}_q[x]$ divides $x^{q^n} - x$ if and only if $\deg(f) \mid n$. So, If f is irreducible then it clearly satisfies the conditions. Conversely, let h be an irreducible factor of f of degree $d < n$. Since $h \mid x^{q^n} - x$, we have $d \mid n$, and so $d \mid \frac{n}{t}$ for some prime divisor t of n. This means $h \mid x^{q^{n/t}} - x$ which contradicts (II). Therefore, $d = n$ and hence f is irreducible. □

Therefore, the polynomial f is irreducible if and only if $\gcd(x^q - x, f) = 1$ and $f \mid x^{q^t} - x$. Once we found an irreducible f, the norm of $x \in \mathbb{F}_q[x]/(f(x))$ is a, hence $x^{(q^t-1)/(q-1)} = a$. It can easily be seen that $(q^t - 1)/(q - 1)$ is divisible by t so that $x^{(q^t-1)/t(q-1)}$ is a t-th root of a.

Algorithm 6.18 Cipolla's t-th root when t is a prime divisor of $q - 1$

Input: A nonzero $a \in \mathbb{F}_q$
Output: a t-th root of a in $\mathbb{F}_{q^t}$

1. choose a random $f \in \mathbb{F}_q[x]$ with constant term a
2. **if** $\gcd(x^q - x, f) \neq 1$ **then**
3. **return** failure.
4. **end if**
5. **if** $f \nmid x^{q^t} - x$ **then**
6. **return** failure.
7. **end if**
8. $c \leftarrow x^{(q^t-1)/t(q-1)} \bmod f$
9. **return** c

Step 2 requires $O(M(t) \log q)$ multiplications, and Step 5 requires $O(C(t) \log t)$ multiplications [98], assuming we have x^q from Step 2. Thus, the expected number of operations for finding an irreducible polynomial of degree t is $O(M(t)t \log q + C(t)t \log t)$. Step 8 also needs $O(M(t)t \log q)$ multiplications. Therefore, the expected complexity of Algorithm 6.18 is $O(M(t)t \log q + C(t)t \log t)$ operations in $\mathbb{F}_q$.

Finally, we extend the new square root algorithm proposed at the end of Section 6.4 to compute t-th roots where t is a prime divisor of $q - 1$. Given $a \in \mathbb{F}_q$, let $\gamma \in \mathbb{F}_q$ be a t-th root of it. Since $t \mid q - 1 = p^n - 1 = (p - 1)(p^{n-1} + \cdots + p + 1)$, we consider two cases:

Case 1: Assume that $t \mid p - 1$. Then

$$\begin{aligned} \mathbb{F}_p \ni \beta = T_{\mathbb{F}_q/\mathbb{F}_p}(\gamma) = \sum_{i=1}^{n-1} \gamma^{p^i} &= \gamma(1 + \gamma^{p-1} + \gamma^{p^2-1} + \cdots + \gamma^{p^{n-1}-1}) \\ &= \gamma(1 + a^{(p-1)/t} + a^{(p^2-1)/t} + \cdots + a^{(p^{n-1}-1)/t}) \end{aligned} \tag{6.5}$$

Analogous to the square root case, letting $b = 1 + a^{(p-1)/t} + a^{(p^2-1)/t} + \cdots + a^{(p^{n-1}-1)/t}$, and raising both side of the Equation (6.5) to the power of t result in the equation $\beta^t = ab^t$ over $\mathbb{F}_p$. Computing β from the above equation takes $O(t \log p)$ operations in $\mathbb{F}_p$. Computing b and then b^t needs $O(M(n) \log p + C(n) \log n)$ and then $O(M(n) \log t)$ operations in $\mathbb{F}_p$ respectively. Therefore, the expected running time of the algorithm in this case is $O((t + M(n)) \log p + C(n) \log n)$ operations in $\mathbb{F}_p$.

Case 2: If $t \nmid p - 1$ then $(t, p - 1) = 1$. Let $(q - 1)/(p - 1) = t^r \ell$ with $t \nmid \ell$. Then

$$\mathbb{F}_p \ni \gamma^{\frac{q-1}{p-1}} = \gamma^{t^r \ell} = (\gamma^\ell)^{t^r} = a^{t^{r-1}\ell} \tag{6.6}$$

which gives us an equation of degree t^r over $\mathbb{F}_p$ from which γ^ℓ can be computed as follows. Let k be the order of p in $\mathbb{Z}/t^r\mathbb{Z}$, then $k \mid \varphi(t^r) = t^r - t^{r-1}$ where φ is the Euler function. Let $f(x) \in \mathbb{F}_p[x]$ be an irreducible polynomial of degree k and constant term

$a^{t^{r-1}\ell}$. Then $x^{(p^k-1)/(p-1)} = N_{\mathbb{F}_{p^k}/\mathbb{F}_p}(x) = a^{t^{r-1}\ell}$, where $N(\cdot)$ is the norm function. Thus, $x^{(p^k-1)/(p-1)t^r} = \gamma^\ell$. There exist integers u, v such that $ut + v\ell = 1$, and hence $a^u(\gamma^\ell)^v = (\gamma^t)^u(\gamma^\ell)^v = \gamma^{ut+v\ell} = \gamma$. Finding $f(x)$ requires an average $\Theta(k)$ applications of irreducibility test. Each irreducibility test takes $O(C(k)\log k + M(k)\log p)$ operation in $\mathbb{F}_p$, see [84]. So, finding f takes $O(C(k)k\log k + M(k)k\log p)$ operations in $\mathbb{F}_p$. Computing $a^{t^{r-1}\ell}$, x^ℓ, a^u, and $(\gamma^\ell)^v$ requires $O(M(n)\log q) = O(M(n)n\log p)$ multiplications in $\mathbb{F}_p$. Therefore, the expected complexity of the algorithm in this case is $O(C(k)k\log k + M(k)k\log p + M(n)n\log p)$ operations in $\mathbb{F}_p$.

Chapter 7

Point Counting on Genus 2 Curves

By counting points on a genus 2 curve over a finite field we mean computing the order of its jacobian. For cryptographic purposes, the order of the jacobian should be a non-smooth number. A curve is called a secure curve if it is also defined over a large enough base field. In this chapter, as mentioned in Chapter 1, we present a generalization of the genus 1 Schoof algorithm for point counting on genus 2 curves. We first present a general picture of the work of Gaudry and Schost, without going into details, we refer the reader to the original reference [30] for great detail. Then we report the practical improvements on their work by applying the contributions presented in previous chapters.

7.1 Preliminaries

Let $p > 2$ be a fixed prime and let $\mathbb{F}_p$ be a finite field of p elements. A hyperelliptic curve of genus 2 over $\mathbb{F}_p$ is a curve $\mathcal{H}$ defined by Equation (4.2) by setting $g = 2$. Let simply denote $J_{\mathbb{F}_p}(\mathcal{H})$ by $J(\mathcal{H})$ in this chapter. For a divisor $D \in J(\mathcal{H})$ with Mumford representation (u, v), the *weight* of D is defined to be the degree of u. Let Θ denote the set of divisors of weight smaller than 2. Then the representation of an element $D \in J(\mathcal{H})\backslash\Theta$ is of the form $(x^2 + u_1x + u_0, v_1x + v_0)$. By Equation (4.9), the characteristic polynomial of the Frobenius endomorphism ϕ_p of $J(\mathcal{H})$ is $\chi_p(t) = t^4 - a_1t^3 + a_2t^2 - a_1pt + p^2$ with $a_1 = \alpha_1 + \alpha_2 + \alpha_3 + \alpha_4$, and $a_2 = \alpha_1\alpha_2 + \alpha_1\alpha_3 + \cdots + \alpha_3\alpha_4$ where α_i are as in Theorem 4.26.(3). Therefore, $|a_1| \leq 4\sqrt{p}$, and $|a_2| \leq 6p$. Also by Equation (4.10), $\#J(\mathcal{H}) = \chi_p(1) = p^2 + 1 - a_1(p + 1) + a_2$. For a positive integer ℓ prime to p, the ℓ-torsion subgroup $J(\mathcal{H})[\ell]$ is isomorphic to $(\mathbb{Z}/\ell\mathbb{Z})^4$. Since $\chi_p(D) = 0$ for all $D \in J(\mathcal{H})$, we have

$$\phi_p^4(D) - [a_1 \bmod \ell]\phi_p^3(D) + [a_2 \bmod \ell]\phi_p^2(D) - [a_1p \bmod \ell]\phi_p(D) + [p^2 \bmod \ell]D = 0 \quad (7.1)$$

for all $D \in J(\mathcal{H})[\ell]$. According to a result of [37], for any odd prime power ℓ, $J(\mathcal{H})\backslash\Theta$ contains a $\mathbb{Z}/\ell\mathbb{Z}$-basis of $J(\mathcal{H})[\ell]$. Hence, $\phi_p|_{J(\mathcal{H})[\ell]}$ is completely determined by its action on $J(\mathcal{H})[\ell]\backslash\Theta$. The goal is to imitate the elliptic version of the Schoof's algorithm by computing $\chi_p(t) \mod \ell$ for small primes or prime powers ℓ, and then recombine the results using Chinese remaindering theorem to get $\chi_p(t)$. By the above, we can always assume that D is a divisor of weight 2.

7.2 Representing ℓ-torsion divisors

Let ℓ be a prime or prime power such that $\gcd(\ell, p) = 1$. In the case of elliptic curves, i.e. curves of genus 1, a divisor D, which is a indeed a point on the curve, is an ℓ-torsion divisor if and only if $\psi_\ell(D) = 0$ where ψ_ℓ is the ℓ-th division polynomial. Therefore, an ℓ-torsion divisor can be obtained by extracting a root of ψ_ℓ. A similar situation holds for genus 2 curves. Let D be a divisor of weight 2 given by $(x^2 + u_1x + u_0, v_1x + v_0)$. Then there exist a radical ideal $I_\ell \subset \mathbb{F}_p[U_1, U_0, V_1, V_0]$ such that $D \in J(\mathcal{H})[\ell]$ if and only if $s(u_1, u_0, v_1, v_0) = 0$ for all $s \in I_\ell$. The ideal I_ℓ is called the *ℓ-th division ideal*. See [37] for an explicit set of generators for I_ℓ.
It can be shown that the Gröbner basis of the ideal I_ℓ has the form

$$I_\ell \left| \begin{array}{ll} V_0 & - V_1 Z(U_1) \\ V_1^2 & - W(U_1) \\ U_0 & - S(U_1) \\ & R(U_1) \end{array} \right.$$

where $R \in \mathbb{F}_p[U_1]$ is a squarefree monic polynomial of degree $(\ell^4-1)/2$, and $Z, W, S \in \mathbb{F}_p[U_1]$ are polynomials of degree less than $(\ell^4-1)/2$. Therefore, we can use a hyperelliptic analogous of the Schoof's algorithm by working in the quotient algebra $\mathbb{F}_p[U_1, U_0, V_1, V_0]/I_\ell$. This algorithm has polynomial time. But since there is no computationally efficient recurrence relations, like for division polynomials of elliptic curves, for the Gröbner bases of the division ideals, the algorithm requires computation of Gröbner bases, which is time consuming in practice. A more efficient approach is to use Cantor's division polynomials. Let $P = (x - x_P, y_P)$ be a divisor of weight 1. Then there are polynomials $d_0, d_1, d_2, e_0, e_1, e_2 \in \mathbb{F}_p[X]$, depending on ℓ, such that

$$[\ell]P = \left(x^2 + \frac{d_1(x_P)}{d_0(x_P)}x + \frac{d_2(x_P)}{d_0(x_P)}, y_P \frac{e_1(x_P)}{e_0(x_P)}x + y_P \frac{e_2(x_P)}{e_0(x_P)} \right). \tag{7.2}$$

For ℓ odd, the degrees of the above polynomials are $2\ell^2 - 1, 2\ell^2 - 2, 2\ell^2 - 3, 3\ell^2 - 2, 3\ell^2 - 2$, and $3\ell^2 - 3$ respectively, and for ℓ even, these degrees are reduced by 5. These division polynomials can be easily computed using recurrence relations. Now, let $D = (x^2 + U_1x + U_0, V_1x + V_0) = (u, v) \in J(\mathcal{H})[\ell] \backslash \Theta$ be a generic divisor. Then we can write $D = P_1 + P_2$ where $P_1 = (x - X_1, Y_1)$, and $P_2 = (x - X_2, Y_2)$ such that X_1, X_2 are roots of u, and $Y_i = v(X_i), i = 1, 2$. Therefore, $[\ell]D = 0$ if and only if $[\ell]P_1 = -[\ell]P_2$. Rewriting this equation using (7.2) results in the following system of equations

$$\mathbf{E} \left| \begin{array}{ll} E_1(X_1, X_2) & = (d_1(X_1)d_2(X_2) - d_1(X_2)d_2(X_1)/(X_1 - X_2) = 0, \\ E_2(X_1, X_2) & = (d_0(X_1)d_2(X_2) - d_0(X_2)d_2(X_1)/(X_1 - X_2) = 0, \\ F_1(X_1, X_2, Y_1, Y_2) & = Y_1e_1(X_1)e_0(X_2) + Y_2e_1(X_2)e_0(X_1) = 0, \\ F_2(X_1, X_2, Y_1, Y_2) & = Y_1e_2(X_1)e_0(X_2) + Y_2e_2(X_2)e_0(X_1) = 0, \end{array} \right.$$

which encodes the ℓ-torsion divisors in $J(\mathcal{H})[\ell] \backslash \Theta$. It can be shown that the division ideal I_ℓ can be reconstructed from the system $\mathbf{E}$.

7.3 A Schoof algorithm for genus 2

Assume we have the ℓ-th division ideal reconstructed from the system **E** of Section 7.2. Then we can extract a root r of $R(U_1)$ and obtain the coordinates of a weight 2 divisor D by substituting r into equations of **E** or I_ℓ. The divisor D can then be plugged into the characteristic polynomial $\chi_p(t)$ for computing $a_1 \bmod \ell$, and $a_2 \bmod \ell$. With this descriptions, the sketch of a genus 2 Schoof algorithm is the following.

Algorithm 7.19 A genus 2 Schoof algorithm

Input: A genus 2 hyperelliptic curve $\mathcal{H}$ over $\mathbb{F}_p$
Output: The number $\#J(\mathcal{H})$

1. $\mathcal{A} \leftarrow \varnothing$
2. **for** enough number of small primes or prime powers ℓ **do**
3. Let $L = \{(a_1, a_2);\ a_1, a_2 \in [0, \ell - 1]\}$
4. **while** $\#L > 1$ **do**
5. construct a new ℓ-torsion divisor D
6. eliminate the pairs (a_1, a_2) in L such that
$$\phi_p^4(D) - [a_1]\phi_p^3(D) + [a_2]\phi_p^2(D) - [a_1 p \bmod \ell]\phi_p(D) + [p^2 \bmod \ell]D \neq 0$$
7. **end while**
8. use the elements of L to deduce $\chi_p(t) \bmod \ell$
9. $\mathcal{A} \leftarrow \mathcal{A} \cup \{(\chi_p(t) \bmod \ell, \ell)\}$
10. **end for**
11. deduce $\chi_p(t)$ from the elements of $\mathcal{A}$ by Chinese remaindering
12. **return** $\chi_p(1)$

An efficient way of extracting a root of $R(U_1)$ is to extract an irreducible factor of it, and then construct an extension $\mathbb{F}_q$ of $\mathbb{F}_p$ using this factor. Since factoring is rather time consuming, when ℓ is prime, we may avoid factoring $R(U_1)$ as follows. Define the quotient algebra

$$\mathbb{D} = \mathbb{F}_p[U_1, U_0, V_1, V_0]/\langle V_0 - V_1 Z(U_1), V_1^2 - W(U_1), U_0 - S(U_1), R(U_1)\rangle$$

Then we may define divisors with coordinates in $\mathbb{D}$, although it may not a field. In particular, we let $D_\ell = (x^2 + U_1 x + U_0, V_1 x + V_0) = (x^2 + U_1 x + S(U_1), V_1 x + V_0)$. We can also use the standard addition formulae to add such divisor. Since $\mathbb{D}$ may contain zero divisors, and the addition law of the jacobian involves inversions, we may encounter a division by zero. In that case, we can instead factor $R(U_1)$, and work modulo all factors separately. Computing $\phi_p^i(D_\ell)$ for a positive integer i is also straightforward. Now, since Equation (7.1) holds for all $D \in J(\mathcal{H})[\ell]$, we have the equality

$$\phi_p^4(D_\ell) - [a_1 \bmod \ell]\phi_p^3(D_\ell) + [a_2 \bmod \ell]\phi_p^2(D_\ell) - [a_1 p \bmod \ell]\phi_p(D_\ell) + [p^2 \bmod \ell]D_\ell = 0$$

over $\mathbb{D}$. Therefore, to find the pairs $(a_1, a_2) \in [0, \ell - 1]^2$ satisfying this relation, we can proceed as in Algorithm 7.19.

7.4 Lifting ℓ^k-torsion divisors

Let ℓ be a prime different from p. Since the $[\ell] : J(\mathcal{H}) \to J(\mathcal{H})$ is surjective, for any $D \in J(\mathcal{H})$, there is a divisor $D_1 \in J(\mathcal{H})$ such that $[\ell]D_1 = D$. This is a division by ℓ in the jacobian. In the following, we show how to obtain a sequence of ℓ^k-torsion divisors P_k, such that $[\ell]P_{k+1} = P_k$, and $P_1 \in J(\mathcal{H})[\ell]$. For $k = 1$, an ℓ-torsion divisor P_1 is obtained by factoring the polynomial R in the Gröbner basis of the division ideal I_ℓ (see Section 7.2). Now, assume we have computed an ℓ^k-torsion divisor $P_k = (x^2 + u_1x + u_0, v_1x + v_0)$. Suppose that $P_{k+1} = (x^2 + U_1x + U_0, V_1x + V_0)$ such that $[\ell]P_{k+1} = P_k$. Using the addition law on the jacobian, this equality yields the system of equations

$$\mathcal{F}_k \left| \begin{array}{l} H_1(U_1, U_0, V_1, V_0) = u_1, \\ H_2(U_1, U_0, V_1, V_0) = u_0, \\ H_3(U_1, U_0, V_1, V_0) = v_1, \\ H_4(U_1, U_0, V_1, V_0) = v_0, \end{array} \right.$$

where H_i are rational functions. Clearing the denominators results in a system of polynomial equations $\mathcal{P}_k$ in four variables U_1, U_0, V_1, V_0. We may assume that the ideal $\langle \mathcal{P}_k \rangle$ admits a Gröbner basis of the form

$$\mathcal{P}_k \left| \begin{array}{rl} V_0 & - G_1(U_1) \\ V_1 & - G_2(U_1) \\ U_0 & - G_3(U_1) \\ & M(U_1) \end{array} \right.$$

Let $\mathbb{F}_q$ be the field of definition of P_k, and let e_k be the degree of the extension $\mathbb{F}_q/\mathbb{F}_p$. Let $F \in \mathbb{F}_p[T]$ be a monic irreducible polynomial of degree e_k so that $\mathbb{F}_q \cong \mathbb{F}_p[T]/F$. Then u_1, u_0, v_1, v_0 are expressed as polynomials in $\mathbb{F}_p[T]$ of degree less than e_k, and hence P_{k+1} is described by a system of the form

$$K_{\ell^k} \left| \begin{array}{rl} V_0 & = Z(T) \\ V_1 & = W(T) \\ U_0 & = S(T) \\ U_1 & = R(T) \\ F(T) & = 0 \end{array} \right.$$

where $Z, W, S, R \in \mathbb{F}_p[T]$. At each step of computing the sequence $P_1, P_2, \ldots$, we can use P_i to obtain modular information on the polynomial $\chi_p(t)$ modulo ℓ^i as in Algorithm 7.19.

7.5 Experimental results

In this section, we compare timings for lifting 2^k-torsion divisors. For the case of $\ell = 2$, it is more efficient to work on the Kummer surface associated to the curve than the jacobian. The Kummer surface $\mathcal{K} \in \mathbb{P}^3$ is the quotient of the jacobian $J(\mathcal{H})$ by the hyperelliptic involution. More precisely, there is a surjective mapping $\varphi : J(\mathcal{H}) \to \mathcal{K}$ where $\varphi^{-1}(\kappa)$ is a set of two opposite divisors for every $\kappa \in \mathcal{K}$. Because of the simple doubling formulae, division by 2 in $\mathcal{K}$ is done efficiently by taking four square roots and doing a few multiplications or divisions.

Therefore, for halving an element $P \in J(\mathcal{H})$, we can halve $\kappa = \varphi(P)$ in $\mathcal{K}$, and then compute $\varphi^{-1}(\kappa)$ which can be done rather efficiently [28].
Table 7.1 compares the timings (in seconds) obtained for lifting 2^k-torsion for the sample curve

$$\begin{aligned} y^2 \;=\; & x^5 + 168757993785992721416148486985004362096\, x^4 \\ & +226947768353809748194485150253252104 63\, x^3 \\ & +777412357385137042338766696068626751 28\, x^2 \\ & +150617856041609651434793310038133555411\, x \\ & +143282909778412049875859459912573485378. \end{aligned}$$

over $\mathbb{F}_p$ with $p = 2^{127} - 1 = 170141183460469231731687303715884105727$. The degree e_k is the degree of the field extension defined in Section 7.4. There are two main rows: the first one gives the timings for computing all required square roots, and the second row gives the timing for computing the Frobenius and searching for the pair (a_1, a_2) as in Algorithm 7.19. For a more precise profiling, each of the two main rows is divided into three subrows labelled with I, II, and III: I denotes the original Guadry and Schost implementation; II denotes the same implementation but with the new NTL containing the new modular composition and power projection (Section 5.3); III is the same as II except that the new square root algorithm (Section 6.4) has been used.

index 2^k		2^6	2^7	2^8	2^9	2^{10}	2^{11}	2^{12}	2^{13}	2^{14}	2^{15}	2^{16}	2^{17}
degree e_k		2^5	2^6	2^7	2^8	2^9	2^{10}	2^{11}	2^{12}	2^{13}	2^{14}	2^{15}	2^{16}
square roots	I	0.2	0.4	1.2	3.5	11	33	109	365	1262	4466	16246	60689
	II	0.3	0.6	1.5	4.0	12	36	114	360	1140	3610	11507	36938
	III	0.2	0.5	1.2	2.9	8	23	73	232	734	2309	7368	23604
Frobenius + finding (a_1, a_2)	I	0.5	1.1	2.8	6.5	14	32	73	164	368	816	2020	4827
	II	0.5	1.1	2.7	6.5	14	32	72	162	366	813	2011	4827
	III	0.4	1.0	2.3	5.4	12	27	62	139	309	657	1609	3740

Table 7.1: Timings for lifting 2^k-torsion

Bibliography

[1] Adleman, Manders, and Miller. On taking roots in finite fields. In *IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 175–178, 1977.

[2] L. M. Adleman and M.-D. A. Huang. *Primality Testing and Abelian Varieties over Finite Fields*, volume 1512 of *Lecture Notes in Mathematics*. Springer-Verlag, Berlin-Heidelberg-New York, 1992.

[3] Leonard M. Adleman, Jonathan DeMarrais, and Ming-Deh A. Huang. A subexponential algorithm for discrete logarithms over the rational subgroup of the jacobians of large genus hyperelliptic curves over finite fields. In *Algorithmic Number Theory*, volume 877 of *Lecture Notes in Computer Science*, pages 28–40. Springer-Verlag, 1994.

[4] A. Arwin. Über Kongruenzen von dem fünften und höheren Graden nach einem Primzahlmodulus. *Ark. fr Mat., Astron. och Fys.*, 14:1–46, 1918.

[5] M. F. Atiyah and J. G. MacDonald. *Introduction to Commutative Algebra*. Addison-Wesley, Reading, Massachusetts, 1969.

[6] Eric Bach and Jeffrey Shallit. *Algorithmic Number Theory; Volume I: Efficient Algorithms*. The MIT Press, 1996.

[7] Ian F. Blake, G. Seroussi, and Nigel P. Smart. *Elliptic curves in cryptography*, volume 265 of *London Mathematical Society lecture note series*. Cambridge University Press, 1999.

[8] Ian F. Blake, G. (Gadiel) Seroussi, and Nigel P. (Nigel Paul) Smart, editors. *Advances in elliptic curve cryptography*, volume 317 of *London Mathematical Society lecture note series*. Cambridge University Press, 2004.

[9] Brent and Kung. Fast algorithms for manipulating formal power series. *JACM: Journal of the ACM*, 25, 1978.

[10] Daniel Bump. *Algebraic geometry.* World Scientific Publishing Co., 1998.

[11] D. G. Cantor and H. Zassenhaus. A new algorithm for factoring polynomials over finite fields. *Mathematics of Computation*, 36:587–592, 1981.

[12] David G. Cantor. Computing in the Jacobian of a hyperelliptic curve. *Mathematics of Computation*, 48(177):95–101, January 1987.

[13] David G. Cantor. On the analogue of the division polynomials for hyperelliptic curves. *J. Reine Angew. Math.*, 447:91–145, 1994.

[14] Leonard Carlitz. A theorem of dickson on irreducible polynomials. *Proc. Am. Math. Soc.*, 3:693–700, 1952.

[15] Sarvadaman Chowla. A note on the construction of finite Galois fields $GF(p^n)$. *J. Math. Anal. Appl.*, 15:53–54, 1966.

[16] M. Cipolla. Un metodo per la risoluzione della congruenza di secondo grado. *Napoli Rend.*, 9:153–163, 1903.

[17] Henri Cohen and Gerhard Frey, editors. *Handbook of elliptic and hyperelliptic curve cryptography.* Discrete Mathematics and its Applications. Chapman & Hall/CRC, Boca Raton, 2006.

[18] S.D. Cohen. The distribution of polynomials over finite fields. *Acta Arith.*, 17:255–271, 1970.

[19] Don Coppersmith and Shmuel Winograd. Matrix multiplication via arithmetic progressions. *J. Symb. Comput.*, 9(3):251–280, 1990.

[20] P. G. L. Dirichlet. Beweis des satzes, dass jede unbegrenzte arithmetische progression, deren erstes glied und differenz ganze zahlen ohne gemeinschaftlichen factor sind, unendlich viele primzahlen enthält. *Abhandlungen der Königlich Preussischen Akademie der Wissenschaften*, pages 45–81, 1837.

[21] N. D. Elkies. Elliptic and modular curves over finite fields and related computational issues. In *Studies in Advanced Mathematics*, volume 7, pages 21–76. American Mathematical Society, International Press, 1998.

[22] Andreas Enge. *Elliptic curves and their applications to cryptography: an introduction.* Kluwer Academic Publishers, Norwell, MA, USA, 1999.

[23] Andreas Enge. Computing discrete logarithms in high-genus hyperelliptic jacobians in provably subexponential time. *Math. Comput.*, 71:729–742, 2002.

[24] Gerhard Frey and Hans-Georg Rück. A remark concerning m-divisibility and the discrete logarithm in the divisor class group of curves. *Mathematics of Computation*, 62(206):865–874, April 1994.

[25] William Fulton. *Algebraic curves. An introduction to algebraic geometry.* Advanced Book Classics. Redwood City, CA: Addison-Wesley Publishing Company, Inc., 2008.

[26] Gaudry and Harley. Counting points on hyperelliptic curves over finite fields. In *ANTS: 4th International Algorithmic Number Theory Symposium (ANTS)*, pages 313–332, 2000.

[27] Gaudry and Schost. Construction of secure random curves of genus 2 over prime fields. In *Advances in Cryptology, Eurocrypt'04*, volume 3027 of *Lecture Notes in Computer Science*, pages 239–256, 2004.

[28] P. Gaudry. Fast genus 2 arithmetic based on theta functions. *J. Math. Cryptol.*, 1(3):243–265, 2007.

[29] Pierrick Gaudry. An algorithm for solving the discrete log problem on hyperelliptic curves. In *Advances in Cryptology–Eurocrypt '2000*, volume 1807 of *Lecture Notes in Computer Science*, pages 19–34. Springer-Verlag, 2000.

[30] Pierrick Gaudry and Éric Schost. Genus 2 point counting over prime fields. preprint `http://hal.inria.fr/inria-00542650/en/`, 2010.

[31] C. F. Gauss. *Untersuchungen Über Höhere Arithmetik*. Chelsea publishing company, New York, second edition, 1981.

[32] Darrel Hankerson, Alfred J. Menezes, and Scott Vanstone. *Guide to Elliptic Curve Cryptography*. Springer-Verlag, 2004.

[33] H. Hasse. Abstrakte Begründung der komplexen multiplication und Riemannsche Vermutung in Funktionenkörpern. *Abh. Math. Sem. Hamburg Univ.*, 10:325–347, 1934.

[34] D.R. Heath-Brown. Zero-free regions for Dirichlet L-functions and the least prime in an arithmetic progression. *Proc. Lond. Math. Soc., III. Ser.*, 64(2):265–338, 1992.

[35] Xiaohan Huang and Victor Y. Pan. Fast rectangular matrix multiplications and improving parallel matrix computations. In *Proceedings of the second international symposium on Parallel symbolic computation–PASCO '97*, pages 11–23. ACM, 1997.

[36] Erich Kaltofen and Victor Shoup. Fast polynomial factorization over high algebraic extensions of finite fields. In *ISSAC*, pages 184–188, 1997.

[37] W. Kampkötter. *Explizite Gleichungen für Jacobische Varietäten hyperelliptischer Kurven*. PhD thesis, Gesamthochschule Essen, 1991.

[38] Jonathan Katz and Yehuda Lindell. *Introduction to modern cryptography*. Chapman & Hall/CRC, 2008.

[39] Kiran S. Kedlaya. Counting points on hyperelliptic curves using Monsky-Washnitzer cohomology. *Journal of the Ramanujan Mathematical Society*, 16(4):323–338, 2001.

[40] Donald E. Knuth. *The Art of Computer Programming: volume 2 / Seminumerical Algorithms*. Addison-Wesley, Reading, Mass., third edition, 1988.

[41] N. Koblitz. Elliptic curve cryptosystems. *Mathematics of Computation*, 48(177):203–209, 1987.

[42] N. Koblitz. A family of jacobians suitable for discrete log cryptosystems. In *Proc. Advances in Cryptology – CRYPTO '88*, volume 403 of *Lect. Notes Comput. Sci*, pages 94–99, 1988.

[43] N. Koblitz and A. J. Menezes. A survey of public-key cryptosystems. *SIAM Review*, 46(4):599–634, 2004.

[44] Neal Koblitz. Hyperelliptic cryptosystems. *J. Cryptology*, 1(3):139–150, 1989.

[45] Neal Koblitz. CM-curves with good cryptographic properties. In *Advances in Cryptology–Crypto '91*, volume 576 of *Lecture Notes in Computer Science*, pages 279–287. Springer-Verlag, 1991.

[46] Neal Koblitz. *Algebraic Aspects of Cryptography*. Springer-Verlag, Berlin Heidelberg, 1998.

[47] Neal Koblitz. An elliptic curve implementation of the finite field digital signature algorithm. In *Advances in Cryptology–Crypto '98*, volume 1462 of *Lecture Notes in Computer Science*, pages 327–337. Springer-Verlag, 1998.

[48] Serge Lang. *Elliptic curves: Diophantine analysis*. Grundlehren der Mathematischen Wissenschaften. 231. Berlin-Heidelberg-New York: Springer-Verlag, 1978.

[49] Serge Lang. *Introduction to algebraic and abelian functions*. Graduate Texts in Mathematics; 89. Springer-Verlag, second edition, 1982.

[50] Tanja Lange. Formulae for arithmetic on genus 2 hyperelliptic curves. *Appl. Algebra Eng. Commun. Comput*, 15(5):295–328, 2005.

[51] D. Le Brigand. Decoding of codes on hyperelliptic curves. In *EUROCODE '90. International Symposium on Coding Theory and Applications*, volume 514 of *Lecture Notes in Computer Science*, pages 126–134. 1991.

[52] A. M. Legendre. Recherches d'analyse indéterminée. *Mém. Acad. R. Sci*, pages 465–559, 1785.

[53] H.W.jun. Lenstra, J. Pila, and Carl Pomerance. A hyperelliptic smoothness test. I. *Philos. Trans. R. Soc. Lond., Ser. A*, 345(1676):397–408, 1993.

[54] H.W.jun. Lenstra, J. Pila, and Carl Pomerance. A hyperelliptic smoothness test. II. *Proc. Lond. Math. Soc., III. Ser.*, 84(1):105–146, 2002.

[55] L.A. Levin. The tale of one-way functions. *Problems of Information Transmission*, 39(1):92–103, 2003.

[56] S. Lichtenbaum. Duality theorems for curves over p-adic fields. *Invent. Math.*, 7:120–136, 1969.

[57] Rudolf Lidl and Harald Niederreiter. *Introduction to finite fields and their applications*. Cambridge University Press, Cambridge, 1994.

[58] U. V Linnik. On the least prime in an arithmetic progression. I. the basic theorem. *Rec. Math.*, 15:139–178, 1944.

[59] Ueli M. Maurer. Fast generation of secure RSA-moduli with almost maximal diversity. In *Advances in CryptologyEUROCRYPT '89 (LNCS 434)*, pages 636–647, 1989.

[60] Ueli M. Maurer. *Some number-theoretic conjectures and their relation to the generation of cryptographic primes*, pages 173–191. Oxford University Press, 1992. Cryptography and Coding, II.

[61] Alfred J. Menezes, Paul C. van Oorschot, and Scott A. Vanston, editors. *Handbook of Applied Cryptography*. CRC Press, 1996.

[62] V. Miller. Use of elliptic curves in cryptography. In *Advances in Cryptology–CRYPTO '85*, volume 218 of *Lect. Notes in Comp. Sci.*, pages 417–426. Springer-Verlag, 1985.

[63] Victor S. Miller. The weil pairing, and its efficient calculation. *J. Cryptology*, 17(4):235–261, 2004.

[64] Marko Moisio. Kloosterman sums, elliptic curves, and irreducible polynomials with prescribed trace and norm, November 21 2007. Comment: 21 pages; revised version with somewhat more clearer proofs; to appear in Acta Arithmetica.

[65] Volker Müller, Andreas Stein, and Christoph Thiel. Computing discrete logarithms in real quadratic congruence function fields of large genus. *Math. Comput.*, 68:807–822, 1999.

[66] David Mumford. *Abelian varieties. With appendices by C. P. Ramanujam and Yuri Manin*. Tata Institute of Fundamental Research Studies in Mathematics, second edition, 1974.

[67] M. Nori E. Previato M. Stillman Mumford, David With the collaboration of C. Musili and H. Umemura. *Tata lectures on theta. II: Jacobian theta functions and differential equations*, volume 43. Progress in Mathematics, 1984.

[68] Andrew M. Odlyzko. Discrete logarithms: The past and the future. *Des. Codes Cryptography*, 19(2/3):129–145, 2000.

[69] Jan Pelzl, Thomas Wollinger, Jorge Guajardo, and Christof Paar. Hyperelliptic curve cryptosystems: Closing the performance gap to elliptic curves. *Lecture Notes in Computer Science*, 2779:351–365, 2003.

[70] Jonathan Pila. Frobenius maps of abelian varieties and finding roots of unity in finite fields. *Mathematics of Computation*, 55:745–763, 1990.

[71] S. C. Pohlig and M. E. Hellman. An improved algorithm for computing logarithms over $GF(p)$ and its cryptographic significance. *IEEE Transactions on Information Theory*, IT-24:106–110, 1978.

[72] J. M. Pollard. Monte carlo methods for index computation mod p. *Mathematics of Computation*, 32:918–924, 1978.

[73] R. Ree. Proof of a conjecture of S. Chowla. *J. Number Theory*, 3:210–212, 1971.

[74] Hans-Georg Rück. A note on elliptic curves over finite fields. *Mathematics of Computation*, 49(179):301–304, 1987.

[75] T. Satoh. The canonical lift of an ordinary elliptic curve over a finite field and its point counting. *J. Ramanujan Math. Soc.*, 15:247–270, 2000.

[76] Takakazu Satoh. On p-adic point counting algorithms for elliptic curves over finite fields. In *5th International Algorithmic Number Theory Symposium (ANTS)*, volume 2369 of *Lecture Notes in Comput. Sci.*, pages 43–66. SpringerVerlag, 2002.

[77] Takakazu Satoh, Berit Skjernaa, and Yuichiro Taguchi. Fast computation of canonical lifts of elliptic curves and its application to point counting. *Finite Fields Appl.*, 9(1):89–101, 2003.

[78] Susanne Schmitt and Horst G. Zimmer. *Elliptic curves. A computational approach. With an appendix by Attila Pethö.* de Gruyter Studies in Mathematics 31. Berlin: Walter de Gruyter, 2003.

[79] A. Schönhage and V. Strassen. Schnelle multiplikation grosser zahlen. *Computing*, 7:281–292, 1971.

[80] René Schoof. Elliptic curves over finite fields and the computation of square roots mod p. *Math. Comput.*, 44:483–494, 1985.

[81] René Schoof. Counting points on elliptic curves over finite fields. *Journal de Théorie des Nombres de Bordeaux*, 7(1):219–254, 1995.

[82] Daniel Shanks. Class number, a theory of factorization, and genera. In *Analytic Number Theory*, volume 20 of *Proceedings of Symposia in Pure Mathematics*, pages 415–440. American Mathematical Society, 1971.

[83] Daniel Shanks. Five number-theoretic algorithms. In *Proceedings of the Second Manitoba Conference on Numerical Mathematics*, pages 51–70, 1972.

[84] V. Shoup. Fast Construction of Irreducible Polynomials over Finite Fields". *Journal of Symbolic Computation*, 17(5):371–391, May 1994.

[85] V. Shoup. A library for doing number theory (NTL). `http://www.shoup.net/ntl/`, 2009.

[86] Victor Shoup. Efficient computation of minimal polynomials in algebraic extensions of finite fields. In *ISSAC*, pages 53–58, 1999.

[87] J. H. Silverman and J. Suzuki. Elliptic curve discrete logarithms and the index calculus. In *Advances in Cryptology–ASIACRYPT '98*, volume 1514 of *Lecture Notes in Computer Science*, pages 110–125. Springer-Verlag, 1998.

[88] Joseph H. Silverman. *The arithmetic of elliptic curves*. Graduate Texts in Mathematics 106. New York, NY: Springer, second edition, 2009.

[89] Henning Stichtenoth. *Algebraic function fields and codes*. Graduate Texts in Mathematics 254. Berlin: Springer, second edition, 2009.

[90] Andrew V. Sutherland. A generic approach to searching for Jacobians. *Math. Comput.*, 78(265):485–507, 2009.

[91] ATLAS Team. Automatically tuned linear algebra software (ATLAS). `http://math-atlas.sourceforge.net/`, 2010.

[92] GMP Team. The gnu multiple precision arithmetic library (GMP). `http://gmplib.org/`, 2010.

[93] Magma Team. Magma computational algebra system. `http://magma.maths.usyd.edu.au/`, 2010.

[94] A. Tonelli. Bemerkung über die auflösung quadratischer congruenzen. *Göttinger Nachrichten*, pages 344–346, 1891.

[95] G. Tornaría. Square roots modulo p. In *LATIN 2002: Theoretical Informatics*, pages 430–434, 2002.

[96] Jacobus H. van Lint and Gerard van der Geer. *Introduction to coding theory and algebraic geometry*. Birkhäuser Verlag, Basel, Germany, 1988.

[97] Vercauteren, Preneel, and Vandewalle. A memory efficient version of satoh's algorithm. In *Advances in Cryptology: EUROCRYPT 2001*, volume 2045 of *Lecture Notes in Comput. Sci.*, pages 1–13. SpringerVerlag, 2001.

[98] von zur Gathen and Shoup. Computing frobenius maps and factoring polynomials. *CMPCMPL: Computational Complexity*, 2:187–224, 1992.

[99] J. von zur Gathen and J. Gerhard. *Modern Computer Algebra*. Cambridge University Press, second edition, 1999.

[100] Joachim von zur Gathen and Daniel Panario. Factoring polynomials over finite fields: A survey. *J. Symb. Comput*, 31(1/2):3–17, 2001.

[101] Daqing Wan. Generators and irreducible polynomials over finite fields. *Math. Comput.*, 66:1195–1212, July 1997.

[102] Lawrence C. Washington. *Elliptic curves Number theory and cryptography*. Boca Raton, FL: Chapman and Hall/CRC, second edition, 2008.

[103] William C. Waterhouse. Abelian varieties over finite fields. *Ann. Sci. Ec. Norm. Sup.*, 2(4):521–560, 1969.

[104] André Weil. Sur les fonctions algebriques á corps de constantes finis. *C. R. Acad. Sci. Paris*, 210:592–594, 1940.

[105] Annegret Weng. Constructing hyperelliptic curves of genus 2 suitable for cryptography. *Math. Comput*, 72(241):435–458, 2003.

[106] Joseph L. Yucas. Irreducible polynomials over finite fields with prescribed trace / prescribed constant term. *Finite Fields and Their Applications*, 12(2):211–221, 2006.

[107] David Y. Y. Yun. On square-free decomposition algorithms. In *Proc. ACM Symp. Symbolic and Algebraic Comp.*, pages 26–35, 1976.