AIX-MARSEILLE UNIVERSITÉ
ÉCOLE DOCTORALE DE MATHÉMATIQUES ET
D'INFORMATIQUE (ED184)
INSTITUT DE MATHÉMATIQUES DE MARSEILLE

Thèse présentée pour obtenir le grade universitaire de docteur
Discipline : Mathématiques
par

Thanh-Hung DANG

Titre de la thèse :

Complexité scalaire des algorithmes de type Chudnovsky de multiplication dans les corps finis

---

Soutenue le 25 mai 2020 devant le jury composé de :

Laurent-Stéphane DIDIER	*Professeur des universités* USTV/IMATH, Toulon	Rapporteur
Sihem MESNAGER	*Maîtresse de conférence, HDR* Univ. Paris 8/ Télécom ParisTech, Paris	Rapporteuse
Serge VLADUTS	*Professeur des universités* AMU/I2M, Marseille	Examinateur
Nadia EL-MRABET	*Professeure assistante* École des Mines/SAS, Saint-Étienne	Examinatrice
Daniel AUGOT	*Directeur de recherche* Inria Saclay – île-de-France & LIX, Paris	Examinateur
Stéphane BALLET	*Maître de conférence, HDR* AMU/I2M, Marseille	Co-directeur de thèse
Alexis BONNECAZE	*Professeur des universités* AMU/I2M/Polytech, Marseille	Directeur de thèse

Abstract

Complexité scalaire des algorithmes de type Chudnovsky de multiplication dans les corps finis

by Thanh-Hung DANG

L'algorithme de type évaluation-interpolation sur des courbes algébriques, introduit par D.V et G.V Chudnovsky en 1987, est à la base des techniques algorithmiques fournissant actuellement les meilleures bornes de la complexité bilinéaire de la multiplication dans les corps finis. En particulier, ces algorithmes sont connus pour avoir asymptotiquement une complexité bilinéaire linéaire ou quasi linéaire. Mais jusqu'à présent aucun travaux ne s'était attaqué à l'analyse de leur complexité scalaire. Aussi, s'intéresse-t-on dans cette thèse à la complexité scalaire de ces algorithmes.

Plus précisément, nous présentons une stratégie générique permettant d'obtenir des algorithmes de type Chudnovsky ayant une complexité scalaire optimisée. Cette complexité est directement liée à une représentation des espaces de Riemann-Roch sous-jacents visant à l'obtention de matrices creuses.

Les résultats théoriques et numériques obtenus suggèrent que notre stratégie d'optimisation est indépendante du choix du diviseur permettant de construire les espaces de Riemann-Roch.

En utilisant cette stratégie, nous améliorons de 27% la complexité scalaire de la construction de Baum-Shokrollahi (1992) sur le corps $\mathbb{F}_{256}/\mathbb{F}_4$. De plus, pour ce corps, notre construction est la meilleure connue en terme de complexité totale. Les sources des programmes Magma utilisés dans cette thèse sont données en appendice.

--- English version ---

Scalar complexity of Chudnovsky-type algorithms of multiplication in finite fields

The evaluation-interpolation algorithm on algebraic curves, introduced by D.V. and G.V. Chudnovsky in 1987, is the basis of algorithmic techniques currently providing the best bounds of the bilinear complexity of multiplication in finite fields. In particular, these algorithms are known to have asymptotically linear or quasi-linear bilinear complexity. But until now no work has been done on the analysis of their scalar complexity. Therefore, in this thesis we are interested in the scalar complexity of these algorithms.

More precisely, we present a generic strategy to obtain Chudnovsky-type algorithms with optimized scalar complexity. This complexity is directly related to a representation of the underlying Riemann-Roch spaces aimed at obtaining sparse matrices.

The theoretical and numerical results obtained suggest that our optimization strategy is independent of the choice of the divisor used to construct the Riemann-Roch spaces.

Using this strategy, we improve by 27% the scalar complexity of the construction of Baum-Shokrollahi (1992) on the field $\mathbb{F}_{256}/\mathbb{F}_4$. Moreover, for this field, our construction is the best known in terms of total complexity. The sources of the Magma programs used in this thesis are given in appendix.

Acknowledgements

First and foremost, I would like to thank my supervisors Alexis BONNECAZE and Stéphane BALLET for the guidance and the support that they provided me during my Ph.D. study. They suggested the topics of research problems for my Ph.D thesis and taught me how to choose the appropriate methods to work on. I found them always encouraging, supportive and patient with me. I sometimes had some difficulties in my research, they helped and explained to solve and find new ideas for my problem. Without their mentoring, enthusiasm and endless support, my Ph.D. study and this thesis would not have been possible. I count myself as being the most fortunate to be able to work under Alexis and Stéphane's supervision.

I would like to thank the *Institute of Mathematics of Marseille* (I2M) and École doctorale de mathématiques and d'informatique (ED184) for helping me in the administrative procedures and facilities during my Ph.D. study. Sincerely thanks to the secretaries, Mme. Sonia ASSEUM at ED184 and Mme. Corinne ROUX at I2M - campus Luminy for their enthusiastic help and supports in the administrative affairs for a doctoral student. I also express many thanks to all members of *Arithmetic and Theory of Information* group. I greatly benefited from conversations, communications and research activities in the weekly seminars of the group. I really have passed my study in a warm, friendly and joyful atmosphere here together with other Ph.D. students, Bastien, Alejandro, Elena, Leonardo. My sincere thanks to Bastien Pacifico who is my nice office mate. I really appreciate him for providing valuable comments for my works and talks.

I also would like to thank Professors: Sihem MESNAGER and Laurent-Stéphane DIDIER, for spending their time to review my thesis and providing valuable comments to improve the quality of the thesis submission.

I am grateful to the programme cooperated by *Ministry of Education and Training (MOET) of Vietnam Government* and the *Embassy of France in Hanoï* provided a scholarship for my study in France. I also want to thank my colleagues at the *Faculty of Informatics and Mathematics of Phuong Dong University* in Vietnam for supporting and encouraging me when I have studied in the Ph.D program in France.

Finally, I am immensely grateful to my wife Thúy Quỳnh, my daughter Hà Chi and all members in my family in Vietnam for a tremendous amount of love, support and sharing that they have given me during my study in France. This thesis is dedicated to them.

Contents

List of Tables

Introduction

In order to keep pace with the development of digital technology and to meet the growing need for security, it is becoming increasingly important to be able to manage digital data quickly. These data are sometimes large numbers and it is essential that the basic operations on these large numbers are optimized. Arithmetic operations such as additions, multiplications and exponentiations are an important part of these basic operations. For example, asymmetric cryptography uses these operations on numbers up to several thousand bits in size. This is the case of the well-known algorithms RSA, El Gamal and DSA which use modular exponentiations. The efficiency of these algorithms (and the protocols that use them) depends on how quickly these operations are carried out.

In this thesis, we are interested in the efficiency of multiplication in a finite field. Let q be a power of a prime number p and n a positive integer, the multiplication of two elements of $\mathbb{F}_{q^n}$ can be seen as a multiplication of two polynomials f and g of degree $n-1$ with coefficients in $\mathbb{F}_q$ followed by a modular reduction. This reduction is done with the irreducible polynomial of degree n used to define the field extension. The classical method to multiply two polynomials costs $O(n^2)$ operations and a method due to Karatsuba allows the cost to be reduced to $O(n^{1.59})$. In many practical cases, Karatsuba's method is sufficient, but when the elements to be multiplied belong to a large field, it is imperative to use more efficient algorithms. Currently, the fastest algorithms are derived from the Discrete Fourier Transform and its efficient implementation of the Fast Fourier Transform. These methods are based on evaluation and interpolation and consist in:

- Evaluating f and g on some set of points $\alpha_i \in \mathbb{F}_q$,
- Multiplying the corresponding values "pointwise",
- Interpolating the corresponding values to recover fg.

With FFT, the multiplication of f and g can be computed with $O(n \log n)$ operations. However, this method works only when the algebraic structure (FFT is generally defined over a ring) contains special roots of unity. Thus, FFT cannot be applied with just any ring. Schonhage-Strassen's algorithm solves the problem by using "virtual" roots of unity for a cost of $O(n \log n \log\log n)$. All multiplication algorithms perform addition, scalar multiplication and bilinear multiplication. These operations have different costs. The bilinear multiplication (i.e. multiplying two elements of $\mathbb{F}_q$ which depend on f and g, the elements of $\mathbb{F}_{q^n}$ being multiplied) is heavier than the scalar multiplication (i.e. multiplying a constant of $\mathbb{F}_q$ by an element of $\mathbb{F}_q$ which depends on the elements being multiplied) which is itself heavier than the addition. Thus, algorithms with low bilinear complexity are particularly interesting. In 1988, generalizing interpolation algorithms on the projective line over $\mathbb{F}_q$ to algebraic curves of higher genus over $\mathbb{F}_q$, D.V. and G.V. Chudnovsky provided a method [CC88] which enabled to prove the *linearity* [Bal99] of the bilinear complexity of multiplication in finite extensions of a finite field. This is the so-called Chudnovsky-Chudnovsky multiplication algorithm (or CCMA). Many studies, including recently

[Bal+19], focused on the qualitative improvement of CCMA, but very little is known about the scalar complexity of this method and therefore its overall complexity. The problem of its scalar complexity was only addressed in 2015 by Atighehchi, Ballet, Bonnecaze and Rolland [Ati+17]. They proposed a new construction which slightly improved the scalar complexity eventhough the main objective of this work was not to optimize scalar complexity. The objective of their article was mainly to compare the total complexity of an algorithm due to Couveignes and Lercier [CL09] with that of a method using parallel calculations on an improvement of CCMA. Thus, there was no strategy dedicated to scalar optimization and in fact, the number of scalar multiplications was not significantly reduced in finite distance. Therefore, it has to be noted that so far, practical implementations of multiplication algorithms of type Chudnovsky over finite fields have failed to simultaneously optimize the number of scalar multiplications and bilinear multiplications.

The objective of this thesis is to *analyse this scalar complexity and possibly find Chudnowsky-type algorithms with an improved scalar complexity*. It would make it possible to compare the efficiency of Chudnovsky-type multiplication algorithms with that of the other well-known algorithms.

The thesis is structured as follows. The objective of Chapter 1 is to present an overview of the main known multiplication algorithms in finite fields. Two polynomial basis representations are considered: the standard basis and the normal one. In both cases, their complexity is compared in finite distance and asymptotic. Moreover and contrary to existing analyses, scalar and bilinear complexities are clearly identified in the complexity of the multiplication.

Chapter 2 presents the notions of algebraic geometry necessary for the understanding of the following chapters. It is inspired by Stichtenoth's book [Sti93] and is illustrated with examples to better understand the theory.

Chapter 3 focuses on Chudnovsky's method and its improvements. This is an interpolation method based on algebraic curves which can be roughly explained as follows (we use some notations that will be used and mathematically defined in the rest of the thesis). The elements to be multiplied are seen as the evaluation of some functions f and g of a Riemann-Roch space of a divisor D on a certain point (also called a place) Q. Since this evaluation is designed to be an isomorphism, multiplying two elements of the field $\mathbb{F}_{q^n}$ is the same as multiplying two elements of the Riemann-Roch space. A second evaluation function evaluates f and g on enough points P_i so that, by interpolation, the product fg can be calculated. This second evaluation function can be represented by a matrix denoted T_D in the thesis. This is the first phase of the algorithm. The second phase, which involves an other matrix (denoted T_{2D}^{-1} in the thesis), consists of transforming fg into an element of the field $\mathbb{F}_{q^n}$. Indeed, both matrices are linked and their coefficients depend on the choice of different parameters (the place Q, the Riemann-Roch space, the points used on the second evaluation, etc.). As with any interpolation, the number of points on which evaluations can be made is particularly important. To increase this number, it is possible either to increase the genus or to increase the degree of the points. It is also possible to use derivated evaluations. The chapter describes in details the method along with its known improvements and provides explicit examples.

It is known that Chudnovsky's method is competitive in terms of bilinear complexity but what about its total complexity? Is it possible to control the scalar complexity of Chudnovsky's method in such a way as to make it competitive with the best multiplication algorithms? Analyzing the scalar complexity of Chudnovsky's method and then trying to optimize it is a complex problem. This is the purpose of Chapter 4 which represents the heart of the thesis. It is easy to see that the scalar

complexity depends on the coefficients of the two matrices involved in the method since each matrix will have to be multiplied by a vector which depends on the elements to multiply. If a coefficient is equal to 0 or 1 the multiplication (in $\mathbb{F}_q$) does not have to be done. In fact $0 \cdot x = 0$ and $1 \cdot x = x$. Thus, in order to control the number of scalar multiplications, the number of 0 and 1 in each matrix should be controlled. As a first step, we decided to focus on the number of zeros. This means that we want to make the two matrices as sparse as possible. If we denote by $N_{zero}(T_D)$ and $N_{zero}(T_{2D,n}^{-1})$ respectively the number of zeros in the first matrix and the number of zeros in the second one (actually, we just need to consider the first n rows of the matrix), we seek to maximize

$$N_z = 2N_{zero}(T_D) + N_{zero}(T_{2D,n}^{-1}).$$

Since these matrices are linked, the best value of $N_{zero}(T_D)$ may not lead to the best solution.

The first task was to define a strategy to optimize N_z. We chose to fix the divisor D as well as the points Q and P_i and to determine the set of bases of the Riemann-Roch space which lead to the best value $N_{zero}(T_D)$. We note that fixing a basis of the Riemann-Roch space also fixes the basis of F_{q^n}. This strategy is generic in the sense that when the divisor and points are fixed, there is no other possible strategy. We obtained an upper-bound on $N_{zero}(T_D)$:

$$N_{zero}(T_D) \leq n \cdot \deg D.$$

Among the best bases of the Riemann-Roch space $\mathcal{L}(D)$, we searched which ones gave the best value of $N_{zero}(T_{2D,n}^{-1})$. However, there is no evidence to suggest that an optimal basis of the Riemann-Roch space for the "sparsity" of T_D is also optimal for the global algorithm. The theoretical and numerical results obtained, which depend only on the degree of D, suggest that this optimization strategy is also independent of the choice of the divisor D.

We used our strategy to improve the scalar complexity of the Baum-Shokrollahi construction on $\mathbb{F}_{256}/\mathbb{F}_4$. We obtained $N_z = 52$ which corresponds to a gain of 27% over Baum-Shokrollahi's method. Moreover, for this field, our construction is better than any other known algorithm.

The thesis finishes by an appendix giving the source in Magma of all the built algorithms.

Chapter 1

State of the Art

1.1 Algebraic complexity of multiplication algorithm in finite fields

1.1.1 Complexity of multiplication in $\mathbb{F}_{q^n}$

In this thesis, we assume that p is a prime number, q is a power of p, and $\mathbb{F}_q$ denotes the finite field of q elements with its characteristic p. The field $\mathbb{F}_{q^n}$ is always considered as an n-dimensional extension of $\mathbb{F}_q$ and is thus a vector space of dimension n over $\mathbb{F}_q$.

We consider the multiplication of two elements A and B in the field $\mathbb{F}_{q^n}$.
Let $\mathcal{B} = \{e_1, \cdots, e_n\}$ be a basis of $\mathbb{F}_{q^n}$ over $\mathbb{F}_q$. We have

$$A = \sum_{i=1}^{n} a_i e_i \text{ and } B = \sum_{i=1}^{n} b_i e_i,$$

then

$$AB = \sum_{h=1}^{n} (\sum_{i,j=1}^{n} t_{ijh} a_i b_j) e_h, \tag{1.1}$$

where $e_i e_j = \sum_{h=1}^{n} t_{ijh} e_h$, $t_{ijh} \in \mathbb{F}_q$ being some constants.

In the multiplication of A and B, there are 3 types of elementary operations as follows:

Addition	$(a, b) \mapsto a + b$	with $a, b \in \mathbb{F}_q$,
Scalar multiplication	$a \mapsto c.a$	with $c, a \in \mathbb{F}_q$,
Bilinear multiplication	$(a, b) \mapsto a.b$	with $a, b \in \mathbb{F}_q$.

We define the complexity of the multiplication in $\mathbb{F}_{q^n}$ over $\mathbb{F}_q$ as the minimal number of elementary operations which is necessary to compute the product AB in $\mathbb{F}_{q^n}$. Therefore, in order to obtain the product AB we need
(i) $n^3 - n$ additions,
(ii) n^3 scalar multiplications,
(iii) n^2 bilinear multiplications.

Definition 1.1.1. *i. The total number of scalar multiplications in $\mathbb{F}_q$ used in an algorithm $\mathcal{U}$ of multiplication in $\mathbb{F}_{q^n}$ is called scalar complexity and noted $\mu^s(\mathcal{U})$.*

ii. The total number of bilinear multiplications in $\mathbb{F}_q$ used in an algorithm $\mathcal{U}$ of multiplication in $\mathbb{F}_{q^n}$ is called bilinear complexity and noted $\mu^b(\mathcal{U})$.

The study of the bilinear complexity has been closely associated to the notion of "tensor rank". In fact, in the pioneering article [STV92], Shparlinski, Tsafsman and

Vladut established the link between the bilinear complexity of the multiplication in $\mathbb{F}_{q^n}$ over $\mathbb{F}_q$ and the tensor rank of the multiplication in $\mathbb{F}_{q^n}$.

Let $\mathrm{Bil}(\mathbb{F}_{q^n} \times \mathbb{F}_{q^n}, \mathbb{F}_{q^n})$ be $\mathbb{F}_q$-vector space of bilinear forms of $\mathbb{F}_{q^n} \times \mathbb{F}_{q^n}$ in $\mathbb{F}_{q^n}$. This space is isomorphic to the tensor product $\mathbb{F}_{q^n}^\star \otimes \mathbb{F}_{q^n}^\star \otimes \mathbb{F}_{q^n}$, where $\mathbb{F}_{q^n}^\star$ as $\mathbb{F}_q$-vector space is the dual of $\mathbb{F}_{q^n}$ (cf [BCS97, Proposition 14.16]):

Proposition 1.1.1. $\mathrm{Bil}(\mathbb{F}_{q^n} \times \mathbb{F}_{q^n}, \mathbb{F}_{q^n}) \cong \mathbb{F}_{q^n}^\star \otimes \mathbb{F}_{q^n}^\star \otimes \mathbb{F}_{q^n}$.

More generally, we will present here the notion of tensor rank of the multiplication and its link to the bilinear complexity in the framework of algebras.

Let $\mathcal{A}$ be an F-algebra, where F is a field. Denote $m_{\mathcal{A}}$ the multiplication in $\mathcal{A}$. Since $m_{\mathcal{A}}$ is a bilinear maps from $\mathcal{A} \times \mathcal{A}$ in $\mathcal{A}$, it implies that $m_{\mathcal{A}}$ corresponds to a linear maps M from the tensor product $\mathcal{A} \times \mathcal{A}$ in $\mathcal{A}$ and we therefore can represent M by a tensor $t_{\mathcal{A}} \in \mathcal{A}^\star \otimes \mathcal{A}^\star \otimes \mathcal{A}$ where $\mathcal{A}^\star$ is the dual of $\mathcal{A}$ over F. Hence, the product of two elements x and y of $\mathcal{A}$ is obtained by convolution of tensor $t_{\mathcal{A}}$ with the tensor $x \otimes y \in \mathcal{A} \otimes \mathcal{A}$.

The rank of the tensor $t_{\mathcal{A}}$ is defined as the minimal number λ of elementary tensors (of rank 1) of the form $a_i \otimes b_i \otimes c_i$ such that $t_{\mathcal{A}}$ can be represented as follows:

$$t_{\mathcal{A}} = \sum_{i=1}^{r} a_i \otimes b_i \otimes c_i \tag{1.2}$$

where $a_i, b_i \in \mathcal{A}^\star, c_i \in \mathcal{A}$. In this case, for any $x, y \in \mathcal{A}$, we have

$$xy = \sum_{i=1}^{r} a_i(x) b_i(y) c_i. \tag{1.3}$$

Definition 1.1.2. *An algorithm $\mathcal{U}$ of bilinear multiplication x and y in $\mathcal{A}$ has complexity λ if there are $a_1, ..., a_\lambda, b_1, ..., b_\lambda \in \mathcal{A}^\star$ and $c_1, ..., c_\lambda \in \mathcal{A}$ satisfying (1.3). The complexity λ is denoted by $\mu^b(\mathcal{U})$.*

The bilinear complexity of the multiplication in $\mathcal{A}$ over F, denoted $\mu^b(\mathcal{A}/F)$, is the minimal complexity on F of all algorithms of bilinear complexity in $\mathcal{A}$, namely:

$$\mu^b(\mathcal{A}/F) := \min_{\mathcal{U}} \mu^b(\mathcal{U})$$

where $\mathcal{U}$ belongs to set of bilinear multiplication algorithms in $\mathcal{A}$ over F.

Definition 1.1.3. *A bilinear multiplication algorithm of type (1.3) is called symmetric if for all $i \in 1, ..., \lambda$, we have $a_i = b_i$.*

The symmetric bilinear complexity of the multiplication in $\mathcal{A}$ over F, denoted $\mu^b_{sym}(\mathcal{A}/F)$, is the minimal complexity of all the symmetric bilinear multiplication algorithms in $\mathcal{A}$ over F, namely:

$$\mu^b_{sym}(\mathcal{A}/F) := \min_{\mathcal{U}_{sym}} \mu^b(\mathcal{U}_{sym})$$

where $\mathcal{U}^{sym}$ belongs to set of symmetric bilinear multiplication algorithms in $\mathcal{A}$ on F.

It is easy to see that

$$\mu^b(\mathcal{A}/F) \leq \mu^b_{\mathrm{sym}}(\mathcal{A}/F).$$

The tensor rank of the bilinear multiplication in an extension of degree n of finite field $\mathbb{F}_q$ corresponds to particular case in where $F = \mathbb{F}_q$ and $\mathcal{A} = \mathbb{F}_{q^n}$. Then we set

$$\mu^b_q(n) := \mu^b(\mathbb{F}_{q^n}/\mathbb{F}_q)$$

and

$$\mu^b_{q,\mathrm{sym}}(n) := \mu^b_{\mathrm{sym}}(\mathbb{F}_{q^n}/\mathbb{F}_q),$$

respectively to be the bilinear complexity and symmetric bilinear complexity of bilinear multiplication algorithm in $\mathbb{F}_{q^n}$ over $\mathbb{F}_q$.

For $n, m \geq 2$, $\mathbb{F}_{q^{nm}}$ is an extension field of $\mathbb{F}_{q^n}$. We have the following inequalities:

$$\mu^b_q(n) \leq \mu^b_q(nm) \leq \mu^b_q(n).\mu^b_{q^n}(m).$$

Definition 1.1.4. *The total number of operations in $\mathbb{F}_{q^n}$ used in an algorithm $\mathcal{U}$ of multiplication in $\mathbb{F}_{q^n}$ is called total complexity of the algorithm $\mathcal{U}$ and denoted $M_{\mathcal{U}}(\mathbb{F}_{q^n})$.*

The *total complexity* of the multiplication in $\mathbb{F}_{q^n}$, denoted $M(\mathbb{F}_{q^n})$, is the minimal complexity of all the multiplication algorithms in $\mathbb{F}_{q^n}$, namely:

$$M(\mathbb{F}_{q^n}) := \min_{\mathcal{U}} M_{\mathcal{U}}(\mathbb{F}_{q^n}).$$

1.1.2 Complexity of polynomial multiplication in $\mathbb{F}_q[x]$

For the complexity of polynomial multiplication over a finite field $\mathbb{F}_q$, we consider the total number of arithmetic operations, i.e. multiplications and additions/-subtractions that are needed to multiply two polynomials of degree less than n in $\mathbb{F}_q[X]$. Since the resulting polynomials can be computed without divisions, it seems natural to consider only *division-free* algebraic algorithms.

For an algorithm $\mathcal{V}$ which computes the product of two polynomials of degree $n-1$ over $\mathbb{F}_q$, we define $m^b_{\mathcal{V}}(n)$ and $m^s_{\mathcal{V}}(n)$ as respectively the number of non-scalar (bilinear) and scalar multiplication used in $\mathcal{V}$. We also define $m_{\mathcal{V}}(n)$ to be the number of multiplications used in $\mathcal{V}$. It is obvious that

$$m_{\mathcal{V}}(n) = m^b_{\mathcal{V}}(n) + m^s_{\mathcal{V}}(n).$$

We denote $a_{\mathcal{V}}(n)$ the number of additions and subtractions used in $\mathcal{V}$. Then we set

$$M_{\mathcal{V}}(n) := m_{\mathcal{V}}(n) + a_{\mathcal{V}}(n),$$

the total algebraic complexity of $\mathcal{V}$ computing the product of two polynomials of degree $n-1$ over $\mathbb{F}_q$.

In what follows, $\mathcal{A}^n_{\mathbb{F}_q}$ is set of division-free algorithms computing the product of two degree $n-1$ polynomials over $\mathbb{F}_q$, setting

$$M_q(n) := \min_{\mathcal{V}\in\mathcal{A}^n_{\mathbb{F}_q}} M_{\mathcal{V}}(n), \quad m_q(n) := \min_{\mathcal{V}\in\mathcal{A}^n_{\mathbb{F}_q}} m_{\mathcal{V}}(n), \quad a_q(n) := \min_{\mathcal{V}\in\mathcal{A}^n_{\mathbb{F}_q}} a_{\mathcal{V}}(n)$$

and

$$m^b_q(n) := \min_{\mathcal{V}\in\mathcal{A}^n_{\mathbb{F}_q}} m^b_{\mathcal{V}}(n), \quad m^s_q(n) := \min_{\mathcal{V}\in\mathcal{A}^n_{\mathbb{F}_q}} m^s_{\mathcal{V}}(n).$$

$M_q(n)$ is called the *complexity of multiplications* of two polynomials of degree $n-1$ in $\mathbb{F}_q[x]$.

$a_q(n)$, $m_q(n)$, $m^b_q(n)$ and $m^s_q(n)$ are respectively called *additive*, *multiplicative*, *bilinear* and *scalar* complexity of polynomial multiplication of degree $n-1$ in $\mathbb{F}_q[x]$.

Remark 1.1.1. *$M_q(n)$ needs not to be equal to $m_q(n) + a_q(n)$, since the minimal number of multiplications and the minimal number of additive operations can be achieved by different algorithms. Moreover, $M(\mathbb{F}_{q^n})$ is different from $M_q(n)$ since an element in the finite*

field $\mathbb{F}_{q^n}$ can be represented as a polynomial of degree $n-1$ in $\mathbb{F}_q[x]$ modulo an irreducible polynomial $f(x)$ of degree n over $\mathbb{F}_q$, therefore the multiplication of two elements in $\mathbb{F}_{q^n}$ can be proceeded by multiplying two corresponding polynomials in $\mathbb{F}_q[x]$, afterward doing a reduction modulo $f(x)$.

Let $M_{q,f}(n)$ denote the *total number of operations* over $\mathbb{F}_q$ (or the complexity over $\mathbb{F}_q$) required for multiplication of polynomials modulo f with $\deg f = n$.

Then, we have

$$M(\mathbb{F}_{q^n}) \leq M_{q,f}(n)M(\mathbb{F}_q), \tag{1.4}$$

where $M(\mathbb{F}_q)$ is defined as in the Definition 1.1.4 in case $n = 1$.

1.2 A review of known multiplication algorithms in finite fields

Gashkov and Sergeev in [GS13] made a complete overview of the complexity of implementation of arithmetic operations in finite fields. In this subsection, we focus on the multiplication algorithms in finite fields, in particular we will specify the complexities of the existing multiplication algorithms in terms of bilinear and scalar complexity.

Before presenting the multiplication algorithms of two elements in finite fields, it is important to point out that there are several possibilities to represent elements of a finite field.

Basis representation

- *Standard polynomial basis* representation: Let $p(x)$ be an irreducible polynomial over $\mathbb{F}_q$, q a prime

$$p(x) = \sum_{i=0}^{n} g_i x^i$$

where $g_i \in \mathbb{F}_q$.

Let α be a root of $p(x)$, then one can represent an element $a \in \mathbb{F}_{q^n}$, as a polynomial in α as

$$a = a_{n-1}\alpha^{n-1} + \cdots + a_1\alpha + a_0,\ a_i \in \mathbb{F}_q, i = 1, ..., n-1. \tag{1.5}$$

The set $\{1, \alpha, \alpha^2, ..., \alpha^{n-1}\}$ is then said to be a polynomial basis (or standard basis) of the finite field $\mathbb{F}_{q^n}$ over $\mathbb{F}_q$.

- *Normal basis* representation: Let $g(x)$ be an irreducible polynomial of degree n. If β is a root of $g(x)$, then the n distinct roots of $g(x)$ in $\mathbb{F}_{q^n}$ is given by $N = \{\beta, \beta^q, \beta^{q^2}, ..., \beta^{q^{n-1}}\}$. If the elements of N are linearly independent, then N is called a *normal basis* for $\mathbb{F}_{q^n}$ over $\mathbb{F}_q$, $g(x)$ is called a normal polynomial and element β is called a normal element of $\mathbb{F}_{q^n}$.

It can be shown that for any field $\mathbb{F}_q$ and any extension field $\mathbb{F}_{q^n}$, there always exists a normal basis of $\mathbb{F}_{q^n}$ over $\mathbb{F}_q$ (see [LN97, Theorem 2.35]). Given any element $a \in \mathbb{F}_{q^n}$, one can write

$$a = a_{n-1}\beta^{q^{n-1}} + a_{n-2}\beta^{q^{n-2}} + \cdots + a_1\beta^q + a_0\beta, \tag{1.6}$$

where $a_i \in \mathbb{F}_q, i = 1, ..., n-1$.

The polynomial basis is frequently used in cryptographic applications that are based on the discrete logarithm problem such as elliptic curve cryptography. The advantage of the polynomial basis is that multiplication is relatively easy.

For contrast, the normal basis is an alternative to the polynomial basis and it has more complex multiplication but squaring, exponentiation is very simple. Hardware implementations of polynomial basis arithmetic usually consume more power than their normal basis counterparts.

1.2.1 Polynomial Basis Multiplication

Since

$$\mathbb{F}_{q^n} \cong \mathbb{F}_q[x]/\langle p(x)\rangle$$

with $p(x) \in \mathbb{F}_q[x]$ be an irreducible polynomial of degree n, in the polynomial basis a field element a of $\mathbb{F}_{q^n}$ is represented as a polynomial of degree less than or equal to $n-1$ with coefficient in $\mathbb{F}_q$, i.e. $a \in \mathbb{F}_{q^n}$ can be written as

$$a = \sum_{i=0}^{n-1} a_i \alpha^i = a_{n-1}\alpha^{n-1} + \cdots + a_2\alpha^2 + a_1\alpha + a_0 \text{ with } a_i \in \mathbb{F}_q \tag{1.7}$$

where α is the root of $p(x)$. Consequently, we can restore the n coefficients of $a \in \mathbb{F}_{q^n}$ in a vector $(a_{n-1}, ..., a_1, a_0)$.

Let $a = \sum_{i=0}^{n-1} a_i\alpha^i$ and $b = \sum_{i=0}^{n-1} b_i\alpha^i$ be two elements of $\mathbb{F}_{q^n}$.
The addition of two elements a and b is accomplished by adding the coefficients a_i and b_i in $\mathbb{F}_q$. On the other hand, the multiplication $c = a \cdot b$ can be realized in two steps:

- First we perform polynomial multiplication and obtain a product polynomial $c'(x) = a(x)b(x)$ of degree less than 2n; then

- The degree of this product polynomial is reduced to the proper degree range by applying the polynomial modular operation in order to obtain the polynomial $c(x)$ of degree at most $n-1$.

That means

$$c(x) = a(x)b(x) \mod p(x),$$

and we therefore yield the corresponding element in $\mathbb{F}_{q^n}$ as the result of the multiplication of a and b.

In this section, we will make a review of various polynomial basis multiplication algorithms in finite fields. First of all, we begin with a naive method for multiplication of two field elements that is taught in primary school. We usually call it as a *school-book* multiplication method.

School-book multiplication method

Given two polynomials of degree $n-1$ in $\mathbb{F}_q[x]$:

$$f = \sum_{i=0}^{n-1} a_i x^i = a_{n-1}x^{n-1} + a_{n-2}x^{n-2} + \cdots + a_1 x + a_0$$

$$g = \sum_{i=0}^{n-1} b_i x^i = b_{n-1}x^{n-1} + b_{n-2}x^{n-2} + \cdots + b_1 x + b_0.$$

We write:

$$\begin{aligned} f \cdot g &= (\sum_{i=0}^{n-1} a_i x^i) \cdot (\sum_{i=0}^{n-1} b_i x^i) \\ &= (\sum_{i=0}^{n-2} a_i x^i) \cdot (\sum_{i=0}^{n-2} b_i x^i) \quad + (a_{n-1}b_0 + a_0 b_{n-1})x^{n-1} + (a_{n-1}b_1 + a_1 b_{n-1})x^n + \cdots + \\ &\qquad + (a_{n-2}b_{n-1} + a_{n-1}b_{n-2})x^{2n-3} + a_{n-1}b_{n-1}x^{2n-2}. \end{aligned}$$

The steps of the algorithm are:
- Recursively multiply $\sum_{i=0}^{n-2} a_i x^i$ and $\sum_{i=0}^{n-2} b_i x^i$;
- Compute $(a_{n-1}b_0 + a_0 b_{n-1})x^{n-1} + (a_{n-1}b_1 + a_1 b_{n-1})x^n + \cdots + a_{n-1}b_{n-1}x^{2n-2}$.

This takes $2n-1$ multiplications and $n-1$ additions over $\mathbb{F}_q$;
- Add the former to the latter. This takes $n-1$ additions for the coefficients in $\mathbb{F}_q$ of $x^{n-1}, ..., x^{2n-4}$, the other coefficients do not overlap.

We get the recursion formula $M_q(n) \leq M_q(n-1) + 4n$ and the obtained estimation

$$M_q(n) \leq 2n^2 - 2n + 1.$$

In other words, the asymptotic complexity of the naive multiplication algorithm of polynomials of degree less than n is $O(n^2)$.

This algorithm is efficient only in low degrees, the cost of the school-book algorithm is too high from degree 14 on.

The earliest method for reducing the complexity of an integer multiplication is due to A. Karatsuba [KO62] (1962). He was the first to observe that multiplication of large integers could be done in complexity less than $O(n^2)$. Indeed, he observed that the product $(a_1 x + a_0)(b_1 x + b_0)$ can be computed using 3 *multiplications and* 4 *additions* instead of 4 multiplications and 1 addition as in the school-book algorithm. The product is re-written as follows:

$$(a_1 x + a_0)(b_1 x + b_0) = a_1 b_1 x^2 + ((a_1 + a_0)(b_1 + b_0) - a_1 b_1 - a_0 b_0)x + a_0 b_0. \quad (1.8)$$

We call this the 2-segment Karatsuba method or K_2. We here present 2-segment Karatsuba's method for multiplication of two polynomials in $\mathbb{F}_q[x]$.

Karatsuba multiplication algorithm

The basis idea of Karatsuba's algorithm is as follows.

Given two polynomials $f = \sum_{i=0}^{n-1} a_i x^i$ and $g = \sum_{i=0}^{n-1} b_i x^i$ in $\mathbb{F}_q[x]$ where $a_i, b_i \in \mathbb{F}_q$, $i = 0, ..., n-1$. Setting $m = \lceil \frac{n}{2} \rceil$.

Then, we can write f and g as:

$$f = \sum_{i=0}^{n-1} a_i x^i = f_1 x^m + f_0$$
$$g = \sum_{i=0}^{n-1} b_i x^i = g_1 x^m + g_0$$

where $f_0 = \sum_{i=0}^{m-1} a_i x^i$, $f_1 = \sum_{i=m}^{n-1} a_i x^{i-m}$ and similarly for g.

Two polynomials can be multiplied using the following formula:

$$(f_1 x^m + f_0)(g_1 x^m + g_0) = f_1 g_1 x^{2m} + ((f_1 + f_0)(g_1 + g_0) - f_1 g_1 - f_0 g_0)x^m + f_0 g_0, \quad (1.9)$$

where f_0, f_1, g_0, g_1 are the polynomials over $\mathbb{F}_q$ of degree less than $m = \lceil \frac{n}{2} \rceil$.

The algorithm becomes recursive if it is applied again to multiply these polynomial halves. The next iteration step splits these polynomials again in half. The algorithm eventually terminates after t steps. In the final step the polynomials degenerate into single coefficients. Since every step exactly halves the number of coefficients, the algorithm terminates after $t = \log_2 n$ steps.

Algorithm 1: Karatsuba Algorithm

Input: $f = \sum_{i=0}^{n-1} a_i x^i, g = \sum_{i=0}^{n-1} b_i x^i \in \mathbb{F}_q[x]$, for $n \in \mathbb{N}_{\geq 2}$
Output: $h = f \cdot g = \sum_{i=0}^{2n-1} c_i x^i$
1 Set $m \leftarrow \lceil \frac{n}{2} \rceil$
2 Write $f = f_1 x^m + f_0, g = g_1 x^m + g_0$ with $f_0 = \sum_{i=0}^{m-1} a_i x^i, f_1 = \sum_{i=m}^{n-1} a_i x^{i-m}$ and $g_0 = \sum_{i=0}^{m-1} b_i x^i, g_1 = \sum_{i=m}^{n-1} b_i x^{i-m}$
3 $v_0 \leftarrow$ **Karatsuba Algorithm**(f_0, g_0)
4 $v_\infty \leftarrow$ **Karatsuba Algorithm**(f_1, g_1)
5 $v_1 \leftarrow$ **Karatsuba Algorithm**$(f_0 + f_1, g_0 + g_1)$
6 $u \leftarrow v_1 - v_0 - v_\infty$
7 $s \leftarrow v_1 x^n + u x^{n/2} + v_0$
8 **return** s

Complexity analysis of Karatsuba algorithm:

Recall that $m_q(n)$ denote the number of multiplications in $\mathbb{F}_q$ required by the Karatsuba Algorithm when multiplying two polynomials of degree less than n and let $a_q(n)$ be the number of additions in $\mathbb{F}_q$ required. If $n = 1$, then $m_q(1) = 1$ and $a_q(1) = 0$.

In lines 4, 5, 6 the procedure calls itself three times on $n/2$-term polynomials. In line 6, there are 2 additions of two polynomials of size $n/2$. Each polynomial addition costs no multiplication and $n/2$ additions. In lines 7, two polynomial subtractions are needed to compute u. Each of these computations requires no multiplications and $n-1$ subtractions. Moreover, in line 8, $n-2$ elements of the middle term overlap with those of the first and last terms in formula $v_1 x^n + u x^{n/2} + v_0$. These elements need to be combined as well.

Combining these results, the number of multiplication and additions needed to compute a product of degree less than $2n-1$ is given respectively by

$$m_q(n) = 3m_q(\frac{n}{2}), \tag{1.10}$$

$$a_q(n) \leq 3a_q(\frac{n}{2}) + 4n - 4 \tag{1.11}$$

where $m_q(1) = 1$ and $a_q(1) = 0$.

Then, multiplicative and additive complexity of the multiplication of two polynomials with n coefficients over $\mathbb{F}_q$ is as follows:

$$m_q(n) \leq n^{\log_2 3}, \tag{1.12}$$

$$a_q(n) \leq 6n^{\log_2 3} - 8n + 2, \tag{1.13}$$

and total complexity:

$$M_q(n) \leq 7n^{\log_2 3} - 8n + 2. \tag{1.14}$$

Therefore, the *asymptotic complexity* of Karatsuba multiplication algorithm of two polynomials of degree less than n in $\mathbb{F}_q[x]$ is $M_q(n) = O(n^{\log_2 3}) = O(n^{1.5849})$.

We summarize the resulting complexities into the following table:

Complexity	$m_q^b(n)$	$a_q(n)$	$M_q(n)$
Finite Distance	$n^{\log_2 3}$	$6n^{\log_2 3} - 8n + 2$	$7n^{\log_2 3} - 8n + 2$
Asymptotic	$O(n^{\log_2 3})$	$O(n^{\log_2 3})$	$O(n^{\log_2 3})$

TABLE 1.1: Complexity of Karatsuba multiplication algorithm of degree $n-1$ polynomials in $\mathbb{F}_q[x]$

We recall that $m_q^b(n)$ stands for the bilinear complexity of the polynomial multiplication in $\mathbb{F}_q[x]$. Bernstein (2009) proposed the formula:

$$(f_1x^m + f_0)(g_1x^m + g_0) = (1 - x^m)(f_0g_0 - f_1g_1x^n) + (f_1 + f_0)(g_1 + g_0)x^m \quad (1.15)$$

and improved the Karatsuba algorithm defining the so-called Refined Karatsuba algorithm [Ber09].

A recursive complexity estimation of Karatsuba's method was given as follows:

$$M_q(2n) \leq 3M_q(n) + 7n - 3,$$

for $n = 2^k, k \geq 3$, and the resulting:

$$M_q(n) \leq \frac{103}{18}n^{\log_2 3} - 7n + \frac{3}{2}. \quad (1.16)$$

This is a slight improvement of complexity in finite distance compared to the result (1.14) in the original Karatsuba's algorithm. Next, we consider a multiplication algorithm which is a generalization of Karatsuba's algorithm.

Toom-Cook multiplication algorithm

Toom Cook is a faster generalisation of the Karatsuba method. Unlike Karatsuba, it deals with 3 parts rather than 2 parts which makes it even more complex. Toom-Cook algorithm [Too63] is also referred as Toom-Cook 3-way which is the collective name for all Toom-Cook based algorithms.

Let us consider two degree $n-1$ polynomials $A = A(x) = \sum_{i=0}^{n-1} a_i x^i$ and $B = B(x) = \sum_{i=0}^{n-1} b_i x^i$ in $\mathbb{F}_q[x]$ for $q \geq 4$ and n a power of 3. The Toom-Cook 3-way algorithm can be performed in the following steps:

- Step 1: Splitting A and B in three parts:

$$A = A_0 + A_1x^{n/3} + A_2x^{2n/3} \quad (1.17)$$

$$B = B_0 + B_1x^{n/3} + B_2x^{2n/3} \quad (1.18)$$

where A_i and B_i are degree $n/3 - 1$ polynomials in $\mathbb{F}_q[x]$.

We replace $x^{n/3}$ by the indeterminate y in these expressions of A and B resulting in

$$A = A_0 + A_1y + A_2y^2 \quad (1.19)$$

$$B = B_0 + B_1y + B_2y^2 \quad (1.20)$$

Thus, each of A and B is a polynomial of degree 2 in $\mathbb{F}_q[y]$. The product C of A and B is then a polynomial in y of degree 4. Polynomial C can be determined if its values are known at 5 distinct points.

Let $\alpha_1 = 0, \alpha_2 = 1, \alpha_3, \alpha_4 \in \mathbb{F}_q$. Now, we define a new element α_5 outside $\mathbb{F}_q$, namely $\alpha_5 := \infty$ as follows:

We set $\mathbb{F}_q^\infty := \mathbb{F}_q \cup \{\infty\}$. When we "evaluate" a polynomial $p(x) \in \mathbb{F}_q[x]$ at element $x = \infty \in \mathbb{F}_q^\infty$ actually means to take the limit of $p(x)/x^{\deg p}$ as x goes to infinity. Consequently, $p(\infty)$ is always the value of its highest-degree coefficient. In this context, it means that $A(\infty) = A_2$ and $B(\infty) = B_2$.

- Step 2: we evaluate $A(y)$ and $B(y)$ at these five elements and then multiply term by term $A(\alpha_i)$ and $B(\alpha_i)$ for $i = 1, ..., 5$, which provide the evaluations $C(\alpha_i)$ of $C(y) = A(y) \cdot B(y)$ at these five elements. These five multiplications can be computed by recursively applying the same process for degree $\frac{n}{3} - 1$ polynomial in x.

- Step 3: we interpolate $C(y)$ to obtain its polynomial expression in y. Specifically, we can use the Lagrange interpolation method to recover the product $C(y)$ as follows:

$$C(y) = \sum_{i=1}^{4} C(\alpha_i) L_i(y) + C(\infty) L_\infty(y), \tag{1.21}$$

where

$$L_i(y) = \prod_{j=1, j\neq i}^{4} \frac{y - \alpha_j}{\alpha_i - \alpha_j}, \text{ for } i = 1, ..., 4$$

and

$$L_\infty(y) = \prod_{i=1}^{4} (y - \alpha_i).$$

We reconstruct C as a polynomial in $\mathbb{F}_q[x]$ by replacing y by x^3 in $C(y)$. These steps are presented in the following algorithm:

Algorithm 2: ToomCook3

Input: $A = \sum_{i=0}^{n-1} a_i x^i, B = \sum_{i=0}^{n-1} b_i x^i \in \mathbb{F}_q[x]$, for $q \geq 4$
Output: $C = A \cdot B$

1 Write $A = A_0 + A_1 x^{n/3} + A_2 x^{2n/3}$ and $B = B_0 + B_1 x^{n/3} + B_2 x^{2n/3}$, where A_i, B_i degree $\frac{n}{3} - 1$ polynomials $\in \mathbb{F}_q[x]$
2 $A(y) \leftarrow A_0 + A_1 y + A_2 y^2$, $B(y) \leftarrow B_0 + B_1 y + B_2 y^2$
3 Choose $\alpha_i \in \mathbb{F}_q$ for $i = 1, ..., 4$
4 **for** $i = 1, ..., 4$ **do**
5 $\quad u_i \leftarrow A(\alpha_i)$; $v_i \leftarrow B(\alpha_i)$; $u_\infty \leftarrow A_2$; $v_\infty \leftarrow B_2$
6 $\quad c_i \leftarrow$ **ToomCook3**(u_i, v_i); $c_\infty \leftarrow$ **ToomCook3**(u_∞, v_∞)
7 **for** $i = 1, ..., 4$ **do**
8 $\quad L_i(y) \leftarrow \prod_{j=1, j\neq i}^{4} \frac{y - \alpha_j}{\alpha_i - \alpha_j}$; $L_\infty(y) \leftarrow \prod_{i=1}^{4} (y - \alpha_i)$
9 $C(y) \leftarrow \sum_{i=1}^{4} c_i L_i(y) + c_\infty L_\infty(y)$
10 **return** $C(x^{n/3})$.

Complexity analysis of ToomCook3 algorithm: The following table shows the cost in the number of operations (bilinear, scalar multiplication and addition) of essential lines in the above algorithm.

	# {bilinear mul.}	# {scalar mul.}	# {addition}
Line 2;6: compute $A(\alpha_i)$	0	$\frac{4n}{3}$	$\frac{4n}{3}$
Line 3;6: compute $B(\alpha_i)$	0	$\frac{4n}{3}$	$\frac{4n}{3}$
Line 7: compute c_i, c_∞	$5m_q^b(\frac{n}{3})$	$5m_q^s(\frac{n}{3})$	$5a_q(\frac{n}{3})$
Line 8-11: reconstruction	0	$\frac{38n}{3} - 19$	$\frac{34n}{3} - 19$
Total	$5m_q^b(\frac{n}{3})$	$5m_q^s(\frac{n}{3}) + \frac{50n}{3} - 19$	$5a_q(\frac{n}{3}) + \frac{46n}{3} - 19$

TABLE 1.2: Complexity analysis of ToomCook3 multiplication algorithm of degree $n-1$ polynomials in $\mathbb{F}_q[x]$

In summary, we thus obtain the following recursion for $m_q^b(n)$, $m_q^s(n)$, $a_q(n)$ and $M_q(n)$ needed in order to multiply two polynomials of degree less than n in $\mathbb{F}_q[x]$ by using the Toom-Cook 3-way Algorithm:

$$\begin{aligned} m_q^b(n) &= 5m_q^b(\frac{n}{3}), \\ m_q^s(n) &= 5m_q^s(\frac{n}{3}) + \frac{50n}{3} - 19, \\ a_q(n) &= 5a_q(\frac{n}{3}) + \frac{46n}{3} - 19, \\ M_q(n) &= 5M_q(\frac{n}{3}) + \frac{96n}{3} - 38. \end{aligned}$$

The following lemma gives us a general solution to such recursive equations.

Lemma 1.2.1. *Let a, b and i be positive integers and assume that $a \neq b$. Let $n = b^i$, $a \neq 1$. The solution to the inductive relation* $\begin{cases} r_1 &= e, \\ r_n &= ar_{n/b} + cn + d, \end{cases}$ *is as follows*

$$r_n = (e + \frac{bc}{a-b} + \frac{d}{a-1})n^{\log_b(a)} - \frac{bc}{a-b}n - \frac{d}{a-1}. \tag{1.22}$$

We apply (1.22) to the above recursive equations with the initial condition $m_q^b(1) = 1, m_q^s(1) = 0, a_q(1) = 0$ and $M_q(1) = 1$, we obtain:

$$\begin{aligned} m_q^b(n) &= n^{\log_3 5}, \\ m_q^s(n) &= \frac{81}{4}n^{\log_3 5} - 25n + \frac{19}{4}, \\ a_q(n) &= \frac{73}{4}n^{\log_3 5} - 23n + \frac{19}{4}, \\ M_q(n) &= \frac{79}{2}n^{\log_3 5} - 48n + \frac{19}{2}. \end{aligned}$$

Remark 1.2.1. *The original Toom-Cook 3-way does not work in the multiplication of degree less than n polynomials in $\mathbb{F}_q[x]$ for $q = 2, 3$ since there are not enough evaluation elements to apply the method in Algorithm 2.*

For example in $\mathbb{F}_2[x]$, to overcome this problem, Winograd in [Win+80] and Sunar in [Sun04] proposed to replace the two missing products $A(\alpha_3) \cdot B(\alpha_3)$ and $A(\alpha_4) \cdot B(\alpha_4)$ by a multiplication modulo the degree 2 polynomial $y^2 + y + 1$. It means that

$$C(0) = A(0)B(0) = A_0B_0,$$

$$C(1) = A(1)B(1) = (A_0 + A_1 + A_2)(B_0 + B_1 + B_2),$$

$$C(y) \mod (y^2 + y + 1) = (A_0 + A_2 + (A_1 + A_2)y)(B_0 + B_2 + (B_1 + B_2)y) \mod (y^2 + y + 1),$$

$$C(\infty) = A(\infty)B(\infty) = A_2B_2.$$

Then, the complexities are obtained as follows:

$$\begin{aligned} m_q^b(n) &= n^{\log_3 6}, \\ a_q(n) &= \frac{16}{3}n^{\log_3 6} - \frac{22n}{3} + 2, \\ M_q(n) &= \frac{19}{3}n^{\log_3 6} - \frac{22n}{3} + 2. \end{aligned}$$

Bernstein in [Ber09] modified the Toom-Cook 3-way algorithm for the case of polynomial multiplication in $\mathbb{F}_2[x]$ by evaluating polynomials at five elements $0, 1, x, x+1$ and ∞. This modification allows obtaining the following complexities in finite distance

$$\begin{aligned} m_q^b(n) &= 7n^{\log_3 5} - 4n - 2, \\ a_q(n) &= \frac{37}{2}n^{\log_3 5} - \frac{43n}{2} + 3, \\ M_q(n) &= \frac{51}{2}n^{\log_3 5} - \frac{51n}{2} + 1. \end{aligned}$$

Moreover, Bernstein's approach gains the asymptotic complexity of $O(n^{\log_3 5})$ that is really better than Winograd and Sunar's one.

In conclusion, the asymptotic complexity of Toom-Cook 3-way algorithm can be given as in the following theorem.

Theorem 1.2.1. *The asymptotic complexity of Toom-Cook 3-way multiplication algorithm of two degree less than n polynomials in $\mathbb{F}_q[x]$ is $O(n^{\log_3 5})$. In general, the asymptotic complexity of Toom-Cook k-way algorithm is $O(n^{\log_k(2k-1)})$.*

In the next section, we will discuss a method due to Schönhage and Strassen (1971) that yields an asymptotic complexity of $O(n \log_2 n \log\log_2 n)$. The method is similar to the Toom-Cook approach in the sense that it considers *the input numbers as polynomials and then computes the product of the polynomials using an evaluation/interpolation approach.* The crucial point is that evaluating and interpolating a polynomial at the roots of unity can be done in a very efficient way.

The Discrete Fourier Transform and the Fast Fourier Transform

Definition 1.2.1. *The element ω is a primitive nth root of unity in a finite field $\mathbb{F}_q$ if the multiplicative order of ω is n.*

This notion is crucial for the application of the Discrete Fourier Transform (DFT) method. We see that the existence of nth primitive roots of unity is not sure in a given finite field. The following lemma (see [GG13, Lemma 8.8], [Dum+15, Theorem 17]) enables one to determine the finite field which contains an nth primitive root of unity.

Lemma 1.2.2. *Let q be a power of some prime number and $n \in \mathbb{N}$. The finite field $\mathbb{F}_q$ contains an nth primitive root of unity if and only if n divides $q-1$.*

Definition 1.2.2 (Discrete Fourier Transform (DFT))**.** *Let ω be an nth root of unity in $\mathbb{F}_q$ where $n \in \mathbb{N}_{\geq 1}$. The Discrete Fourier Transform of a polynomial $f \in \mathbb{F}_q[x]$ is defined as*

$$DFT_\omega(f) := (f(1), f(\omega), ..., f(\omega^{n-1})).$$

One says that a field **supports the DFT** at order n if there exist nth primitive roots of unity in this field.

Now we consider the Discrete Fourier Transform (DFT) approach for polynomial multiplication. This approach is a special case of the evaluation/interpolation strategy. In order to multiply two polynomials $f(x)$ and $g(x)$ in $\mathbb{F}_q[x]$ of degree less than k, we first multi-evaluate both of them in $m \geq 2n-1$ elements of $\mathbb{F}_q$. Then we deduce the evaluation of $h(x) = f(x) \cdot g(x)$ by computing term by term the evaluation of f and g. Finally, we perform a Lagrange interpolation to get the polynomial form of $h(x)$.

In the DFT approach, we use the evaluation set to be the set of n-th roots of unity. Let $\omega \in \mathbb{F}_q$ be a primitive root of unity, then the DFT approach works as follow.

Let

$$f(x) = a_0 + a_1 x + \cdots + a_{n-1}x^{n-1} \in \mathbb{F}_q[x]$$

and

$$g(x) = b_0 + b_1 x + \cdots + b_{n-1}x^{n-1} \in \mathbb{F}_q[x].$$

- *Multi-evaluation*: The DFT evaluation of f at order m:

$$\hat{f} := DFT_\omega = \big(f(1), f(\omega), ..., f(\omega^{m-1})\big).$$

can be computed as follow.

$$\hat{f} = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & \omega & \omega^2 & \cdots & \omega^{n-1} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & \omega^{m-1} & \omega^{2(m-1)} & \cdots & \omega^{(n-1)(m-1)} \end{pmatrix} \cdot \begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_{n-1} \end{pmatrix}.$$

The same is done for g.

- *Term by term multiplications*:

$$\begin{aligned} \hat{h} &= (\hat{f}_1 \cdot \hat{g}_1, \hat{f}_2 \cdot \hat{g}_2, ..., \hat{f}_m \cdot \hat{g}_m) \\ &= (\hat{h}_1, ..., \hat{h}_m) \end{aligned}$$

- *Interpolaltion*: We interpolate on $\hat{h}_1, ..., \hat{h}_m$ to compute the polynomial h. We obtain:

$$h = \frac{1}{m} \begin{pmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \omega^{m-1} & \omega^{2(m-1)} & \cdots & \omega^{(m-1)^2} \\ 1 & \omega^{2(m-1)} & \omega^{4(m-1)} & \cdots & \omega^{2(m-1)^2} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & \omega^{(m-1)(2n-2)} & \omega^{2(m-1)(2n-2)} & \cdots & \omega^{(2n-2)(m-1)^2} \end{pmatrix} \cdot \begin{pmatrix} \hat{h}_1 \\ \hat{h}_2 \\ \vdots \\ \hat{h}_m \end{pmatrix}.$$

Remark 1.2.2. *We can avoid the division by m in the interpolation process by using a Montgomery representation (cf. [Mon85]) of f and g:*

$$\tilde{f} := \frac{1}{m} f \text{ and } \tilde{g} := \frac{1}{m} g.$$

If we perform DFT approach without the division by m to multiply $\tilde{f}$ and $\tilde{g}$ we get

$$m\tilde{f}\tilde{g} = m(\frac{1}{m} f) \cdot (\frac{1}{m} g) = \tilde{h},$$

where $h = f \cdot g$.

We evaluate the complexity of the polynomial multiplication approach using DFT. Through three above steps, the total of operations in the multiplication of two degree $n-1$ polynomials f and g in $\mathbb{F}_q[x]$ includes m bilinear multiplications, $m(4n-1)$ scalar multiplications by powers of ω and $m(4n-1)$ additions/substractions.

We see that in a classical representation of the field $\mathbb{F}_q$, multiplication by a root of unity would have as same complexity as the multiplication by a random element of $\mathbb{F}_q$. Consequently, the $m(4n-1)$ scalar multiplications by powers of ω, where $m \geq 2n-1$ and ω is a primitive nth root of unity in $\mathbb{F}_q$, would be expensive. El Mrabet and Nègre in [EN09] used the *AMNS representation* of finite field $\mathbb{F}_q$ which was introduced by Bajard et al. in [BIP04] in order to improve the multiplication in an extension $\mathbb{F}_{q^n}$ for several values of n. They combined the DFT multiplication with the AMNS representation. The advantage of the AMNS representation is that *all the multiplication by a root of unity correspond to cyclic shifts*, and become very cheap. Recently, in [EG12] El Mrabet and Gama improved the multiplication in $\mathbb{F}_{q^n}$ by an efficient construction of AMNS basis compared to the multiplication in $\mathbb{F}_{q^n}$ using the DFT approach in the classical representation of $\mathbb{F}_q$.

Next, we consider the following notation.

Definition 1.2.3 (Convolution)**.** *Let $f(x) = a_0 + \cdots + a_{n-1}x^{n-1}$ and $g(x) = b_0 + \cdots + b_{n-1}x^{n-1}$ be two polynomials of degree less than n in $\mathbb{F}_q[x]$. We define*

$$f \star_n g := \sum_{k=0}^{n-1} c_k x^k := \sum_{k=0}^{n-1} \Big(\sum_{i,j:i+j=k \bmod n}^{a_i b_j} \Big) x^k$$

as the convolution of f and g.

If n is not mentioned, we write the convolution of f and g as $f \star g$. Notice that, this notion of convolution is equivalent to polynomial multiplication in the ring $\mathbb{F}_q[x]/\langle x^n - 1\rangle$. This means the lth coefficient of $f \cdot g$ is $\sum_{j+k \equiv l \bmod n} f_j g_k$, and hence $f \star_n g = f \cdot g \mod (x^n - 1)$. In particular, if we consider two polynomials f and g of degree less than n as polynomials of degree less than $2n-1$ (by setting $a_n = \cdots = a_{2n-1} = b_n = \cdots = b_{2n-1} = 0$), then it holds that $f \star_{2n} g = f \cdot g$.

Next, we will exploit this relationship between convolution and multiplication to obtain a fast polynomial multiplication algorithm. We have the following lemma.

Lemma 1.2.3. *Let $f, g \in \mathbb{F}_q[x]$ be two plynomials of degree less than n. Then,*

$$\mathrm{DFT}_\omega(f \star g) = \mathrm{DFT}_\omega(f) \cdot \mathrm{DFT}_\omega(g),$$

where $\cdot$ denotes the pointwise multiplication of vectors.

Proof. We have $f \star g = fg + q \cdot (x^n - 1)$ for some $q \in \mathbb{F}_q[x]$, so that

$$(f \star g)(\omega^j) = f(\omega^j)g(\omega^j) + q(\omega^j)(\omega^{jn} - 1) = f(\omega^j)g(\omega^j) \text{ for } 0 \leq j < n.$$

□

We consider the map $\mathbb{F}_q[x] \to \mathbb{F}_q^n$ that evaluates f at $1, \omega, ..., \omega^{n-1}$ with kernel $\langle x^n - 1 \rangle$. Then, the lemma says that $\text{DFT}_\omega : \mathbb{F}_q[x]/\langle x^n - 1 \rangle \to \mathbb{F}_q^n$ is a homomorphism of $\mathbb{F}_q$-algebras, where multiplication in $\mathbb{F}_q^n$ is pointwise multiplication of vectors. This is illustrated by the following diagram:

$$\begin{array}{ccc} (\mathbb{F}_q[x]/\langle x^n - 1 \rangle)^2 & \xrightarrow{\text{DFT}_\omega \times \text{DFT}_\omega} & \mathbb{F}_q^n \times \mathbb{F}_q^n \\ \text{convolution} \downarrow & & \downarrow \text{pointwise multiplication} \\ \mathbb{F}_q[x]/\langle x^n - 1 \rangle & \xrightarrow{\text{DFT}_\omega} & \mathbb{F}_q^n \end{array}$$

We now present an important algorithm - the *Fast Fourier Transform* algorithm, or FFT for short, that computes the DFT quickly. It was (re)discovered by Cooley and Tukey in 1965 ([CT65]).

We have the following algorithm.

Algorithm 3: Fast Fourier Transform (FFT)

Input: $n = 2^k \in \mathbb{N}$ with $k \in \mathbb{N}$, $f = \sum_{0 \leq j < n} f_j x^j \in \mathbb{F}_q[x]$, and the powers $\omega, \omega^2, \cdots, \omega^{n-1}$ of a primitive nth root of unity $\omega \in \mathbb{F}_q$.

Output: $\text{DFT}_\omega(f) = (f(1), f(\omega), \cdots, f(\omega^{n-1})) \in \mathbb{F}_q^n$.

1 **if** $n = 1$ **do**
2 **return** (f_0)
3 $r_0 \leftarrow \sum_{0 \leq j < n/2} (f_j + f_{j+n/2}) x^j$
4 $r_1^* \leftarrow \sum_{0 \leq j < n/2} (f_j - f_{j+n/2}) \omega^j x^j$
5 **call** the algorithm recursively to evaluate r_0 and r_1^* at the powers of ω^2
6 **return** $r_0(1), r_1^*(1), r_0(\omega^2), r_1^*(\omega^2), \cdots, r_0(\omega^{n-2}), r_1^*(\omega^{n-2})$

Let $a_q(n)$ and $m_q^s(n)$ denote the number of additions and scalar multiplications in $\mathbb{F}_q$, respectively, that the algorithm uses for input size n. The cost for the algorithm consists in:

- n additions and $n/2$ scalar multiplications in line 3,4,
- $2a_q(n/2)$ additions and $2m_q^s(n/2)$ scalar multiplications in line 5.

This yields

$$a_q(n) = 2a_q(n/2) + n,$$

and

$$m_q^s(n) = 2m_q^s(n/2) + n/2,$$

with $a_q(1) = m_q^s(1) = 0$ and by solving the recursions we find that

$$a_q(n) = n \log_2 n \text{ and } m_q^s(n) = \frac{1}{2} n \log_2 n.$$

In summary, let n be a power of 2 and $\omega \in \mathbb{F}_q$ be a primitive nth root of unity. Then Algorithm 3 correctly computes $(DFT)_\omega$ using $n \log_2 n$ additions in $\mathbb{F}_q$, $\frac{1}{2} n \log_2 n$ scalar multiplications by powers of ω. In total, it uses a total of $\frac{3}{2} n \log_2 n$ operations. Hence, we have the following theorem (see [BCS97, p. 2.7], [GG13, p. 8.15]).

Theorem 1.2.2. *Let n be a power of 2 and $\omega \in \mathbb{F}_q$ be a primitive nth root of unity. The asymptotic complexity of the Fast Fourier Transform algorithm is $O(n \log_2 n)$.*

We can now directly derive an efficient algorithm for computing the convolution $f \star_n g$ of two polynomials $f, g \in \mathbb{F}_q[x]$ of degree less than n.

Algorithm 4: FFT-based multiplication

Input: Two polynomials $f, g \in \mathbb{F}_q[x]$ of degree less than $n = 2^k$, with $k \in \mathbb{N}_{\geq 1}$, and a primitive nth root of unity $\omega \in \mathbb{F}_q$.
Output: $f \star_n g$.
1 compute $\omega^2, \cdots, \omega^{n-1}$
2 $\alpha \leftarrow \mathrm{DFT}_\omega(f)$, $\beta \leftarrow \mathrm{DFT}_\omega(g)$
3 $\gamma \leftarrow \alpha \cdot \beta$ { pointwise product }
4 $E \leftarrow \frac{1}{n}\mathrm{DFT}_{\omega^{-1}}(\gamma)$
5 **return** E

Complexity analysis of the FFT-based multiplication:
The cost for the individual steps of above FFT-based multiplication algorithm is:
- $n - 2$ scalar multiplications by ω,
- $2n \log_2 n$ additions and $n \log_2 n$ scalar multiplications by powers of ω,
- n bilinear multiplications,
- $n \log_2 n$ additions, $\frac{1}{2} n \log_2 n$ scalar multiplications by powers of ω, and n divisions by n.

In conclusion, let $\mathbb{F}_q$ be a finite field that **supports the FFT**, the multiplication of polynomials $f, g \in \mathbb{F}_q[x]$ of degree less than n, with $n = 2^k$ or the convolution $f \star_n g$ can be performed using:
- $3n \log_2 n$ additions in $\mathbb{F}_q$,
- $(\frac{3}{2} n \log_2 n + n - 2)$ scalar multiplications by powers of ω,
- n bilinear multiplications in $\mathbb{F}_q$,
- n divisions by n,

in total $\frac{9}{2} n \log_2 n + 3n - 2$ arithmetic operations.
These are summarized in Table 1.3:

Complexity	$m_q^b(n)$	$m_q^s(n)$	$a_q(n)$	$M_q(n)$
Finite Distance	n	$\frac{3}{2} n \log_2 n + 2n - 2$	$3n \log_2 n$	$\frac{9}{2} n \log_2 n + 3n - 2$
Asymptotic	$O(n)$	$O(n \log_2 n)$	$O(n \log_2 n)$	$O(n \log_2 n)$

TABLE 1.3: Complexity of FFT multiplication algorithm of polynomials of degree $n - 1$ in $\mathbb{F}_q[x]$

We have the following result (see [BCS97], [GG13]).

Theorem 1.2.3. *If $\mathbb{F}_q$ supports the FFT of order n, then asymptotic complexity of FFT algorithm of the multiplication of two polynomials of degree less than n in $\mathbb{F}_q[x]$ is $O(n \log_2 n)$.*

Schönhage-Strassen's multiplication algorithm

In this section, we discuss the problem of polynomial multiplication in $\mathbb{F}_q[x]$ when $\mathbb{F}_q$ does not support FFT. The basic idea is to use an algebra of the form $\mathbb{F}_q[x]/\langle x^n+1\rangle$ which contains primitive root of unity such that the FFT-based approach can be worked on it.

Let $n = 2^k$ for some $k \in \mathbb{N}$.

Case 1: $\mathrm{char}(\mathbb{F}_q) \neq 2$.

In this case, x is a $2n$-th principal root of unity in $A = \mathbb{F}_q[x]/\langle x^n+1\rangle$, which is a $\mathbb{F}_q$-algebra of dimension n ([BCS97], 2.11). To multiply two polynomials $f, g \in \mathbb{F}_q[x]$ of degree less than n, it is sufficient to compute fg modulo x^n+1.

Let $m = 2^{\lfloor k/2 \rfloor}, t = n/m = 2^{\lceil k/2 \rceil}$. The coefficients of f and g are divided into t blocks of size m:

$$f = \sum_{0 \leq j < t} f_j x^{mj}, \quad g = \sum_{0 \leq j < t} g_j x^{mj},$$

with $f_j, g_j \in R[x]$ of degree less than m for $0 \leq j < t$.

Set $f' = \sum_{0 \leq j < t} f_j y^j$ and $g' = \sum_{0 \leq j < t} g_j y^j \in R[x,y]$. We then have

$$f = f'(x, x^m) \quad g = g'(x, x^m).$$

We write

$$f'g' = h' + q'(y^t+1) \equiv h' \mod (y^t+1) \tag{1.23}$$

for some $h', q' \in \mathbb{F}_q[x,y]$. Then

$$fg = h'(x,x^m) + q'(x,x^m)(x^{tm}+1) \equiv h'(x,x^m) \mod (x^n+1).$$

Set $f^* = f' \mod (x^{2m}+1), g^* = g' \mod (x^{2m}+1)$ and $h^* = h' \mod (x^{2m}+1)$ in $D[Y]$. The congruence (1.23) implies

$$f^*g^* \equiv h^* \mod (y^t+1) \text{ in } D[y]. \tag{1.24}$$

We now take the primitive $4m$th root of unity

$$\xi = x \mod (x^{2m}+1) \in D := \mathbb{F}_q[x]/\langle x^{2m}+1\rangle.$$

D contains a primitive $2t$th root of unity η, namely $\eta = \xi$ if $t = 2m$ and $\eta = \xi^2$ if $t = m$. Then (1.24) is equivalent to

$$f^*(\eta y)g^*(\eta y) \equiv h^*(\eta y) \mod ((\eta y)^t+1)),$$

or

$$f^*(\eta y)g^*(\eta y) \equiv h^*(\eta y) \mod (y^t-1), \quad \text{since } \eta^t = -1.$$

We now present the following algorithm.

Algorithm 5: Schönhage-Strassen

Input: Two polynomials $f, g \in \mathbb{F}_q[x]$ of degree less than $n = 2^k$ for some $k \in \mathbb{N}$, where $\mathbb{F}_q$ is a finite field of characteristic $\neq 2$.
Output: $h \in \mathbb{F}_q[x]$ such that $fg \equiv h \mod (x^n + 1)$ and $\deg h < n$.

1 **if** $k \leq 2$ **do**
2 **call** the Karatsuba Algorithm 1 to compute $f \cdot g$
3 **return** $fg \mod (x^n + 1)$.
4 $m \leftarrow 2^{\lfloor k/2 \rfloor}$, $t \leftarrow n/m$. Let $f', g' \in \mathbb{F}_q[x, y]$ with $\deg_x f', \deg_x g' < m$ such that $f = f'(x, x^m)$ and $g = g'(x, x^m)$
5 **if** $t = 2m$ **do**
6 $\eta \leftarrow x \mod (x^{2m} + 1)$
7 **else**
8 $\eta \leftarrow x^2 \mod (x^{2m} + 1)$
9 $f^* \leftarrow f' \mod (x^{2m} + 1)$, $g^* \leftarrow g' \mod (x^{2m} + 1)$
10 **call** the Algorithm 4 with $\omega = \eta^2$ to compute $h^* \in D[y]$ of degree less than t such that

$$f^*(\eta y) g^*(\eta y) \equiv h^*(\eta y) \mod (y^t - 1),$$

using Algorithm 5 recursively for the multiplication in $D := \mathbb{F}_q[x]/\langle x^{2m} + 1 \rangle$
11 Let $h' \in \mathbb{F}_q[x, y]$ with $\deg_x h' < 2m$ such that $h^* = h' \mod (x^{2m} + 1)$
12 $h \leftarrow h'(x, x^m) \text{ rem } (x^n + 1)$
13 **return** h

Complexity analysis of Schönhage-Strassen algorithm:

Denote $M_q(k)$ the number of arithmetic operations in $\mathbb{F}_q$ that the algorithm 5 uses on inputs of size $n = 2^k$. We now analyse the cost for each step in the algorithm:

- step 1 (line 1-3), the cost is constant,
- step 2 (line 4) has no arithmetic operations in $\mathbb{F}_q$ performed,
- step 3 (line 5-10), the Algorithm 4 uses
  - $3t \log_2 t$ additions in D
  - $\frac{3}{2} t \log_2 t$ multiplications by powers of $\omega = \eta^2$ in the FFT- steps
  - t bilinear multiplication of two arbitrary elements of D,
  - t divisions by t in D
- step 4 (line 11-13) has cost at most $n = mt$ additions for the computation of h from h'.

Notice that one addition in D costs $2m$ additions in $\mathbb{F}_q$, one scalar multiplication of an element in D by power of η using at most $2m$ scalar multiplications in $\mathbb{F}_q$, and each bilinear multiplication in D is done recursively, using $m_q^b(\lfloor k/2 \rfloor + 1)$ bilinear multiplications in $\mathbb{F}_q$. The computation of $f^*(\eta y), g^*(\eta y)$ from f^*, g^* and of $h^* = h^*(\eta(\eta^{-1} y))$ from $h^*(\eta y)$ amounts to $3t$ scalar multiplications by powers of η. Thus the cost of step 3 is at most

$$9mt \log_2 t + 8mt + t M_q(\lfloor k/2 \rfloor + 1). \tag{1.25}$$

All divisions in Algorithm 4 are by powers of two. We replace the last line by

4. **return** $n \cdot \mathrm{DFT}_{\omega}^{-}1(\gamma) = \mathrm{DFT}_{\omega^{-1}}(\gamma)$.

Then the Algorithm 4 uses only additions and multiplications, but no divisions in $\mathbb{F}_q$, and return $n \cdot (f * g)$ instead $(f * g)$. Thus in step 3, there are $2mt$ scalar multiplications in $\mathbb{F}_q$. It adds to $6mt$ scalar multiplications in $\mathbb{F}_q$ used to compute $f^*(\eta y), g^*(\eta y)$ from f^*, g^* and of $h^* = h^*(\eta(\eta^{-1}y))$ from $h^*(\eta y)$ as analysed above in order that we have $8mt$ as in the formula (1.25).

Therefore, we have:

$$M_q(k) \leq 2^{\lceil k/2 \rceil} M_q(\lfloor k/2 \rfloor + 1) + 9 \cdot 2^k(\lceil k/2 \rceil + 1) \text{ if } k \geq 2 \tag{1.26}$$

where

$$m_q^s(k) \leq 2^{\lceil k/2 \rceil} m_q^s(\lfloor k/2 \rfloor + 1) + 2^k(3\lceil \frac{k}{2} \rceil + 8) \tag{1.27}$$

$$m_q^b(k) \leq 2^{\lceil k/2 \rceil} m_q^b(\lfloor k/2 \rfloor + 1) \tag{1.28}$$

$$a_q(n) \leq 2^{\lceil k/2 \rceil} a_q(\lfloor k/2 \rfloor + 1) + 2^k(6\lceil \frac{k}{2} \rceil + 1). \tag{1.29}$$

Unfolding this inequality (1.26) gives us the following results (refer to [GG13, p. 8.22] for the detail of proof)

$$M_q(n) \leq \frac{9}{2} n \log_2 n \log_2 \log_2 n + \frac{1}{2} n \big(\varepsilon(\log_2 n - 2) - \frac{45}{2})\big), \tag{1.30}$$

where $\varepsilon := \frac{1}{8} M_q(8) + \frac{45}{2}$.

Similarly, we also have the complexity estimations:

$$m_q^s(n) \leq \frac{3}{2} n \log_2 n \log_2 \log_2 n + \frac{1}{2} n \big(\varepsilon_1(\log_2 n - 2) - \frac{25}{2})\big), \tag{1.31}$$

$$m_q^b(n) \leq \frac{1}{8} m_q^b(8) n (\log_2 n - 2), \tag{1.32}$$

$$a_q(n) \leq 3n \log_2 n \log_2 \log_2 n + \frac{1}{2} n \big(\varepsilon_2(\log_2 n - 2) - 10)\big), \tag{1.33}$$

where $\varepsilon_1 := \frac{1}{8} m_q^s(8) + \frac{25}{2}$ and $\varepsilon_2 := \frac{1}{8} a_q(8) + 10$.

Table 1.4, 1.5 and 1.6 summarize the complexities of the multiplication algorithm in this case.

Complexity	$m_q^b(n)$	$m_q^s(n)$
Finite Distance	$\frac{1}{8} m_q^b(8) n (\log_2 n - 2)$	$\frac{3}{2} n \log_2 n \log_2 \log_2 n + \frac{1}{2} n \big(\varepsilon_1(\log_2 n - 2) - \frac{25}{2})\big)$
Asymptotic	$O(n \log_2 n)$	$O(n \log_2 n \log_2 \log_2 n)$

TABLE 1.4: Multiplicative complexity of Schönhage-Strassen multiplication algorithm of degree $n-1$ polynomials in $\mathbb{F}_q[x]$.

Complexity	$a_q(n)$
Finite Distance	$3n \log_2 n \log_2 \log_2 n + \frac{1}{2} n \big(\varepsilon_2(\log_2 n - 2) - 10)\big)$
Asymptotic	$O(n \log_2 n \log_2 \log_2 n)$

TABLE 1.5: Additive complexity of Schönhage-Strassen multiplication algorithm of degree $n-1$ polynomials in $\mathbb{F}_q[x]$.

Complexity	$M_q(n)$
Finite Distance	$\frac{9}{2}n\log_2 n\log_2\log_2 n+\frac{1}{2}n(\varepsilon(\log_2 n-2)-\frac{45}{2}))$
Asymptotic	$O(n\log_2 n\log_2\log_2 n)$

TABLE 1.6: Total complexity of Schönhage-Strassen multiplication algorithm of degree $n-1$ polynomials in $\mathbb{F}_q[x]$.

In Algorithm 5, we consider the case $\mathrm{char}(\mathbb{F}_q)\neq 2$, the result of asymptotic complexity of polynomial multiplication in $\mathbb{F}_q[x]$ of degree less than n is $O(n\log_2 n\log_2\log_2 n)$.

Case 2: $\mathrm{char}(\mathbb{F}_q)=2$. In this case, Schönhage's algorithm [Sch77] reduces the multiplication of polynomials over $\mathbb{F}_q$ to the multiplication in $D:=\mathbb{F}_q[x]/\langle x^{2m}+x^m+1\rangle$ where x is a $3m$-primitive root of unity, with $m=3^{\lceil\log_3 n/2\rceil}$.

Algorithm 6: Schönhage

Input: Two polynomials $f,g\in\mathbb{F}_q[x]$ of degree less than $2n=2\cdot 3^k$ for some $k\in\mathbb{N}$, where $\mathbb{F}_q$ is a finite field of characteristic 2.

Output: $h\in\mathbb{F}_q[x]$ such that $fg\equiv h \mod (x^{2n}+x^n+1)$ and $\deg h<2n$.

if $k\leq 2$ **do**
  call the Karatsuba Algorithm 1 to compute $f\cdot g$
  return $fg \mod (x^{2n}+x^n+1)$.
$m\leftarrow 3^{\lceil k/2\rceil}$, $t\leftarrow n/m$. Let $f',g'\in\mathbb{F}_q[x,y]$ with $\deg_x f',\deg_x g'<m$ such that $f=f'(x,x^m)$ and $g=g'(x,x^m)$
if $m=t$ **do**
  $\eta\leftarrow x \mod (x^{2m}+x^m+1)$
else
  $\eta\leftarrow x^3 \mod (x^{2m}+x^m+1)$; ($\eta$ is a primitive $3t$th root of unity)
$f^*\leftarrow f' \mod (x^{2m}+x^m+1)$, $g^*\leftarrow g' \mod (x^{2m}+x^m+1)$
for $j=1,2$ **do**
  $f_j\leftarrow f^* \text{ rem } y^t-\eta^{jt}$, $g_j\leftarrow g^* \text{ rem } g^t-\eta^{jt}$
  call the Algorithm 4 with $\omega=\eta^3$ to compute $h_j\in D[y]$ of degree less than t such that

$$f_j(\eta^j y)g_j(\eta^j y)\equiv h_j(\eta^j y) \mod (y^t-1),$$

  using Algorithm 6 recursively for the multiplication in $D:=\mathbb{F}_q[x]/\langle x^{2m}+x^m+1\rangle$
$h^*\leftarrow\frac{1}{3}(y^t(h_2-h_1)+\eta^{2t}h_1-\eta^t h_2)(2\eta^t+1)$
Let $h'\in\mathbb{F}_q[x,y]$ with $\deg_x h'<2m$ such that $h^*=h' \mod (x^{2m}+x^m+1)$
$h\leftarrow h'(x,x^m) \text{ rem } (x^{2n}+x^n+1)$
return h

It has been shown that the Schönhage's multiplication algorithm has a total complexity bounded by $24n\log_3 n\log_2\log_3 n+cn\log_2 n$ ([GG13, Exercise 8.30]).

In conclusion, we have the following asymptotic complexity of Schönhage-Strassen's algorithm.

Theorem 1.2.4. *The asymptotic complexity of Schönhage-Strassen's multiplication algorithm of two polynomials of degree less than n over $\mathbb{F}_q$ is $O(n\log_2 n\log_2\log_2 n)$.*

In summary, among multiplication algorithms of polynomials of degree less than n over finite field $\mathbb{F}_q$, one appreciates Schönhage's trick [Sch77]. Schönhage's multiplication method based on FFT becomes advantageous over Karatsuba and Toom-Cook's one in the case $n \geq 1000$. Schönhage's technique for polynomial multiplication involves FFT in the ring $\mathbb{F}_q[x]/\langle x^n + 1\rangle$ and leads to the asymptotic complexity $O(n \log_2 n \log_2 \log_2 n)$.

Cantor and Kaltofen [CK91] generalized Schönhage-Strassen's algorithm into an algorithm for the problem of multiplication of polynomials over arbitrary algebras A to achieve the same asymptotic complexity $O(n \log_2 n \log \log_2 n)$.

In the Turing machine model, Harvey, van der Hoeven and Lecerf [HHL17] (2017) have given a new algorithm that achieves an upper bound of *bit complexity* $\mathcal{M}_p(n)$:

$$\mathcal{M}_p(n) = O(n \log_2 n \cdot 8^{\log_2^* n} \log_2 p) \tag{1.34}$$

for multiplying two polynomials in $\mathbb{F}_p[x]$ of degree less than n, for n large compared to prime p. This is the first known Fürer-type complexity bound for $\mathbb{F}_p[x]$ (in 2007, Fürer [Für07] gave an algorithm of the integer multiplication with the asymptotic bit complexity $O(n \log_2 n \cdot 2^{O(\log_2^* n)})$), and improves the previously best-known bound $\mathcal{M}_q(n) = O(n \log_2 n \log_2 \log_2 n \log_2 q)$. These are remarkable results in the theoretical interest of studying the polynomial multiplication algorithm over finite fields. However, in the framework of our investigation in the thesis, we just focus on the algebraic complexity of the multiplication of two degree n polynomials over $\mathbb{F}_q$.

Before addressing the problem of the modular reduction of a polynomial $p(x)$ over $\mathbb{F}_q$, we summarize, in Table 1.7, all existing best known multiplication algorithms along with their asymptotic multiplicative complexities

Algorithm	$m_q^b(n)$	$m_q^s(n)$	$M_q(n)$
Karatsuba	$O(n^{\log_2 3})$		$O(n^{\log_2 3})$
Toom-Cook 3-way	$O(n^{\log_3 5})$	$O(n^{\log_3 5})$	$O(n^{\log_3 5})$
FFT multiplication	$O(n)$	$O(n \log_2 n)$	$O(n \log_2 n)$
Schönhage-Strassen	$O(n \log_2 n)$	$O(n \log_2 n \log_2 \log_2 n)$	$O(n \log_2 n \log_2 \log_2 n)$

TABLE 1.7: Asymptotic complexity of the polynomial multiplication of degree $n-1$ in $\mathbb{F}_q[x]$.

1.2.2 Modular Reduction over finite fields

We recall that an element in a finite field $\mathbb{F}_{q^n}$ can be considered as a degree $n-1$ polynomial in $F_q[x]$. In order to multiply two elements in $\mathbb{F}_{q^n}$ we proceed the multiplication of two corresponding polynomials in $\mathbb{F}_q[x]$ and reduction modulo an irreducible polynomial of $M(x)$. In the previous section, we already investigated the first step of the processing. Now, we focus on considering the modular reduction in $\mathbb{F}_q[x]$. Barrett reduction algorithm is one of the most commonly used reduction algorithms to avoid computationally intensive multi-precision division.

Barrett modular reduction

Barrett reduction algorithm for integers was introduced by P.D. Barrett in 1987 [Bar87]. This algorithm computes $r \equiv a \mod m$ for an input a and a modulus m, and uses μ, a pre-computed reciprocal of m that are necessary to compute the quotient q such that $a = qm + r$.

In order to reduce a polynomial $A(x)$ of degree of $n+\alpha$ by a polynomial $M(x)$ of degree n in $\mathbb{F}_q[x]$, i.e. $R(x) = A(x) \mod M(x)$, one could first evaluate the quotient $Q(x)$ defined by the equation $A(x) = Q(x)M(x) + R(x)$, where $R(x)$ and $Q(x)$ are respectively the remainder and the quotient of the Euclidean division of $A(x)$ by $Q(x)$. In [Dhe03], J-F. Dhem showed that Barrett's strategy for modular reduction over integers can be adapted to polynomial modular reduction over finite fields. We can pre-compute $\mu(x) := \lfloor \frac{x^{n+\alpha}}{M(x)} \rfloor$. The quotient $Q(x) := \lfloor \frac{A(x)}{M(x)} \rfloor$ may then be reduced to a multiplication by $\mu(x)$ and an appropriate shift (division by a power of x) as shown in equation (1.35).

$$Q_3(x) = \left\lfloor \frac{\lfloor \frac{A(x)}{x^n} \rfloor \cdot \lfloor \frac{x^{n+\alpha}}{M(x)} \rfloor}{x^\alpha} \right\rfloor = \left\lfloor \frac{Q_1(x) \cdot \mu(x)}{x^\alpha} \right\rfloor \tag{1.35}$$

where $Q_3(x) := \lfloor \frac{A(x)}{x^n} \rfloor$. It has been shown that $Q(x) = Q_3(x)$ (see [Dhe03, p. 2.1]).

We now present the Barrett modular reduction algorithm for the polynomial in $\mathbb{F}_q[x]$.

Algorithm 7: Barrett modular reduction

Input: Polynomial $A(x) \in \mathbb{F}_q[x]$ of degree $n+\alpha$, $M(x) \in \mathbb{F}_q[x]$ of degree n and $\mu(x) = x^{n+\alpha} \text{ div } M(x)$
Output: $R(x) \equiv A(x) \mod M(x)$

1 $Q_1(x) \leftarrow A(x) \text{ div } x^n$
2 $Q_2(x) \leftarrow Q_1(x) \cdot \mu(x)$
3 $Q_3(x) \leftarrow Q_2(x) \text{ div } x^\alpha$
4 $R_1(x) \leftarrow A(x) \mod x^\alpha$
5 $R_2(x) \leftarrow Q_3(x) \cdot M(x) \mod x^\alpha$
6 **return** $R_1(x) - R_2(x)$

In this algorithm, we see that the division by power of x in steps 1, 3, 4, and 5 in order to get the quotient and remainder are just shifts of coefficient in the polynomial. Hence, there is no cost in these operations. The cost of the algorithm includes essentially two polynomial multiplications in step 2, 5 and a subtraction of two polynomials $R_1(x)$ and $R_2(x)$ of degree α in step 6 of the algorithm. For the multiplications of two polynomials in $\mathbb{F}_q[x]$, we can apply some appropriate methods mentioned in previous sections, in particular the algorithms of Karatsuba, Toom-Cook, Fast Fourier Transform, Schönhage-Strassen.

Notice that the Barrett modular reduction algorithm uses pre-computed value $\mu(x) = \lfloor \frac{x^{n+\alpha}}{M(x)} \rfloor$ of modulus $M(x)$. This causes an increasing of computing implementation as well as memory area needed for the storage of this value. For the modular reduction in $\mathbb{F}_{2^n}$ without pre-computational phase, Knežević et al. in [Kne+08] recommend to use a special modulus that of type $M(x) = x^n + T(x)$ where the degree of $T(x)$ is far smaller than n. Indeed, we have Lemma 1.2.4 as follows:

Lemma 1.2.4. *[Kne+08] Let $M(x) = x^n + \sum_{i=0}^{l} m_i x^i$ and $\mu(x) = x^{2n} \text{ div } M(x)$ be polynomials in $\mathbb{F}_2[x]$, where $l = \lfloor \frac{n}{2} \rfloor$. Then it holds:*

$$\mu(x) = M(x).$$

Then, Barrett reduction algorithm over $\mathbb{F}_{2^n}$ without pre-computation is shown in Algorithm 8.

Algorithm 8: Barrett modular reduction in $\mathbb{F}_2[x]$ without pre-computation

Input: Polynomials $A(x)$ of degree $2n$ and $M(x) = x^n + \sum_{i=0}^{l} m_i x^i$ of degree n in $\mathbb{F}_2[x]$ where $l = \lfloor \frac{n}{2} \rfloor$
Output: $R(x) \equiv A(x) \mod M(x)$.

$Q_1(x) \leftarrow A(x)$ div x^n
$Q_2(x) \leftarrow Q_1(x) \cdot M(x)$
$Q_3(x) \leftarrow Q_2(x)$ div x^n
$R_1(x) \leftarrow A(x) \mod x^n$
$R_2(x) \leftarrow Q_3(x) \cdot M(x) \mod x^n$
return $R_1(x) - R_2(x)$

Remark 1.2.3. *Koç and Acar [KA98] proposed Montgomery modular reduction algorithm over $\mathbb{F}_{2^n}$ or for the polynomial in $\mathbb{F}_2[x]$. This algorithm is actually the polynomial analogous to the Montgomery reduction algorithm for integers [Mon85]. Similarly, we present the algorithm in $\mathbb{F}_q[x]$ as follows:*

Algorithm 9: Montgomery modular reduction in $\mathbb{F}_q[x]$

Input: *Polynomials $A(x)$ of degree $2n$, $M(x)$ of degree n in $\mathbb{F}_q[x]$ and* $\eta(x) \equiv -M^{-1}(x) \mod x^n$.
Output: $R(x) \equiv A(x)x^{-n} \mod M(x)$.

$S_1(x) \leftarrow A(x) \mod x^n$
$S_2(x) \leftarrow \eta(x)S_1(x) \mod x^n$
$S_3(x) \leftarrow M(x)S_2(x)$
$R(x) \leftarrow (A(x) + S_3(x))/x^n$
return $R(x)$

In order to perform modular reduction, Algorithm 9 needs a pre-computed value of the inverse of modulus $M(x)$. From the implementation point of view, this requires extra computational cost and memory space to store this pre-computational value. As in Barrett's algorithm, the pre-computed value will be removed in the Montgomery modular reduction of the polynomial in $\mathbb{F}_2[x]$ by a modulus $M(x) = \sum_{i=l}^{n} m_i x^i + 1 \in \mathbb{F}_2[x]$ where $l = \lceil \frac{n}{2} \rceil$ (see [Kne+08, Lemma 2]).

1.2.3 Normal Basis Multiplication

Let $N = \{\beta_0, \beta_1, ..., \beta_{n-1}\}$ be a normal basis of $\mathbb{F}_{q^n}$ where $\beta_i = \beta^{q^i}$ for a fixed normal element $\beta \in \mathbb{F}_{q^n}$. Thus, we have $\beta_i^{q^k} = \beta_{i+k}$ for any integer k, where the subscript of β reduced modulo n.

Let $a = \sum_{i=0}^{n-1} a_i \beta^{q^i}$ and $b = \sum_{j=0}^{n-1} b_i \beta^{q^j}$ be arbitrary elements of $\mathbb{F}_{q^n}$ represented in the normal basis N.

It is realized that there is a computational advantage of raising to the power q of an element in $\mathbb{F}_{q^n}$ for the normal bases. In particular, the operation of raising to the power q (and consequently, to any power q^n) in the normal basis N is the cyclic shift of the coefficients in $\mathbb{F}_{q^n}$, since

$$a^q = a_{n-1}\beta + a_0\beta^q + a_1\beta^{q^2} + \cdots + a_{n-2}\beta^{q^{n-1}}.$$

Thus, the use of a normal basis may accelerate the standard computer algorithms of exponentiation (raising to a power). Note that the exponentiation algorithms play the main role in many cryptographic protocols.

It is more complicated to carry out the multiplication in the normal basis in a finite field. In the following we present a multiplication algorithm for normal bases, which is based on the Massey-Omura's algorithm [OM86].

Their product can be calculated by the formula:

$$c = a \cdot b = \sum_{i,j=0}^{n-1} a_i b_j \beta^{q^i+q^j} = \sum_{i,j=0}^{n-1} a_i b_j \beta^{(q^{i-j}+1)q^j},$$

where $i - j$ is computed modulo n.

Let $T = [t_{i,j}]$ be the matrix whose ith row is the vector of coefficients of the element $\beta\beta^{q^i} \in \mathbb{F}_{q^n}$ represented in the normal basis N, that is

$$\beta\beta^{q^i} = \sum_{j=0}^{n-1} t_{i,j}\beta^{q^j}, \quad \text{for } i = 0, ..., n-1.$$

Definition 1.2.4. *Complexity of the normal basis N, denoted by C_N, is defined to be the number of non-zero entries of the matrix T.*

It is well known that $C_N \geq 2n - 1$ (see [Mul+89]). Normal basis N in $\mathbb{F}_{q^n}$ of the minimal complexity $C_N = 2n - 1$ is called *optimal normal basis* (ONB).

Since

$$\begin{aligned}\beta^{(q^{i-j}+1)q^j} &= (\beta^{q^{i-j}+1})^{q^j} = \Big(\sum_{k=0}^{n-1} t_{i-j,k}\beta^{q^k}\Big)^{q^j} \\ &= \sum_{k=0}^{n-1} t_{i-j,k}\beta^{q^{k+j}} = \sum_{m=0}^{n-1} t_{i-j,m-j}\beta^{q^m},\end{aligned}$$

where $s - j$ and $k + j$ are also calculated modulo n, we have:

$$C = \sum_{m=0}^{n-1} p_m \beta^{q^m},$$

where

$$p_m = \sum_{i,j=0}^{n-1} t_{i-j,m-j} a_i b_j.$$

We see that p_m can be rewritten in the form:

$$\begin{aligned}p_m &= \sum_{i,j=0}^{n-1} t_{i-j,m-j} a_i b_j = \sum_{k,l=0}^{n-1} t_{k-l,-l} a_{k+m} b_{l+m} \\ &= \sum_{k,l=0}^{n-1} t_{k-l,-l} S^m(a_k) S^m(b_l),\end{aligned}$$

where S^m is the operation of the cyclic shift of the coordinates of the vector by m coordinates.

We can define $T(a, b) = \sum_{i,j=0}^{n-1} t_{i-j,-j} a_i b_j$ as the bilinear form associated with the matrix T.

Then, we have the following formula:

$$ab = T(a,b)\beta + T(a^{q^{n-1}}, b^{q^{n-1}})\beta^q + T(a^{q^{n-2}}, b^{q^{n-2}})\beta^{q^2} + \cdots + T(a^q, b^q)\beta^{q^{n-1}}. \quad (1.36)$$

In order to calculate each bilinear form $T(a,b)$, it suffices to execute $2C_N + n - 1$ additions and multiplications over $\mathbb{F}_q$. There is no cost of the operation of cyclic shifts. Therefore, Massey-Omura's scheme [OM86] for the normal basis multiplication in $\mathbb{F}_{q^n}$ requires $n(2C_N + n - 1)$ operations over $\mathbb{F}_q$ (by the formula (1.36)). We see that the arithmetic complexity of multiplication over the normal basis depends only on C_N. The upper bound of the complexity of multiplication over an arbitrary normal basis is cubic.

In [RH00] a more efficient algorithm with the bound $\frac{1}{2}n(C_N + 3n - 2)$ was proposed by Reyhani-Masoleh and Hassan. These algorithms for multiplication in the optimal normal bases which have the asymptotic complexity $O(n^2)$ are slower than the algorithms for multiplication in the standard bases even in fields of small dimension.

It is better to execute multiplication in the standard representation of the field $\mathbb{F}_{q^n}$, but raising to a power (exponentiation) is more effective in the normal basis. An idea of transition from the normal basis to the standard one and back has been exploited. Following this idea, Sergeev [Ser07] gave the asymptotic total complexity of multiplication in $\mathbb{F}_{q^n}$:

$$O(\sqrt{n}C_N + n^{1.667} + n^{1.5}\log_2 q \log_2 n \log_2 \log_2 n).$$

Chapter 2

Background on algebraic geometry

2.1 Introduction

2.1.1 Algebraic function fields

This section introduces the notions and main results concerning algebraic function fields which will be used throughout this thesis. We essentially follow the presentation of [Sti93] and we give several examples in order to illustrate the text.

An algebraic function field over K of one variable is a finite algebraic extension of the rational function field $K(x)$ where K is an arbitrary field. This type of field extension naturally occurs in many branches of mathematics such as algebraic geometry, number theory, theory of complexity and so on. As a result, one can study the algebraic function fields according to different points of view.

The algebraic approach of algebraic functions is more basic than that via algebraic geometry: only basic knowledge of algebraic extension fields, including Galois theory, is assumed. Another advantage is that the main results of the theory (such as the Riemann-Roch theorem) can be obtained very quickly from the function fields on an arbitrary constant field. This facilitates the presentation of certain applications of algebraic functions to the theory of error correcting code, to cryptography and to the theory of complexity which are our main motivations.

In the following, K will denote an arbitrary field (unless otherwise stated).

Definition 2.1.1. *An algebraic function field F/K of one variable over K is an extension field $F \supseteq K$ such that F is a finite extension of $K(x)$ for some element $x \in F$ which is transcendental over K.*

For simplicity, we will refer to F/K as a function field.

When the primitive element theorem can be applied, a function field F/K associated to a curve $\mathcal{C}$ can be represented as a simple algebraic extension of a rational function field $K(x)$, i.e. $F = K(x,y)$ where $\phi_x(y) = 0$ for an irreducible polynomial $\phi_x(y) \in K(x)[y]$. In this case, F can be represented as a rupture field of ϕ_x i.e. $F = K(x)[y]/\langle \phi_x(y) \rangle$. This is strictly the same as considering the polynomial $\phi_y(x) = 0$ where $\phi_y(x)$ is an irreducible polynomial of $K(y)[x]$. This representation also occurs when the extension F/K is a separable extension of $K(x)$ or when F/K is a finite algebraic extension of $K(x)$ where K is a perfect field.

The polynomial ϕ can be seen as a rational fraction of $K(x,y)$. The curve of equation $\phi(x,y) = 0$ is associated to the function field.

In particular, the simplest function field is indeed the rational function field $K(x)$. In this case, the projective smooth curve associated to $K(x)$ is clearly the projective line $\overline{K}$.

Example 2.1.1. *Consider $\mathbb{F}_q[x,y]/\mathbb{F}_q$ the polynomial ring in y with coefficients in $\mathbb{F}_q[x]$ and $\mathcal{C}$ an algebraic curve defined by $f(x,y) = y^2 + y - x^5 \in \mathbb{F}_q[x,y]$. The polynomial*

$f(x,y)$ being irreducible over $\mathbb{F}_q[x]$, the ideal $\langle f\rangle$ generated by f is maximal in $\mathbb{F}_q[x][y]$. So $\frac{\mathbb{F}_q[x,y]}{\langle y^2+y-x^5\rangle}$ denoted by $\mathbb{F}_q(\mathcal{C})$ is an integer domain and the fraction field $Frac\big(\frac{\mathbb{F}_q[x,y]}{\langle y^2+y-x^5\rangle}\big)$ is an algebraic function field defined over $\mathbb{F}_q$ associated to the curve $\mathcal{C}$. This is the rational function field on $\mathcal{C}$.

Definition 2.1.2. *The set $\tilde{K} = \{z \in F | z \text{ is algebraic over } K\}$ is a subfield of F, denoted the field of constants F/K.*

We have $K \subseteq \tilde{\mathrm{K}} \subset F$ and it can be easily checked that $F/\tilde{\mathrm{K}}$ is a function field over $\tilde{\mathrm{K}}$. The field K is said to be algebraically closed in F (or K is the full constant field of F) if $\tilde{\mathrm{K}} = K$.

Proposition 2.1.1. *The field of constants $\tilde{K}$ of F/K is a finite algebraic extension of K.*

Transcendental elements over K can be characterized as followed:

Proposition 2.1.2. *The element $z \in F$ is transcendental over K if and only if $F/K(z)$ is of finite degree i.e. $[F : K(z)] < \infty$.*

Remark 2.1.1. *If the curve $\mathcal{C}$ is absolutely irreducible (i.e. irreducible in every extension of K: $\mathcal{C}$ is also called geometrically irreducible), the full constant field of F/K associated to the curve $\mathcal{C}$ is K. Otherwise, it is possible that the full constant field strictly contains K.*

For example, consider $F/\mathbb{F}_q = Frac\big(\frac{\mathbb{F}_q[X,Y]}{\langle Y^2+1\rangle}\big)$ associated to the curve $\mathcal{C}$ with affine equation $Y^2 + 1 = 0$ such that -1 is not a square in $\mathbb{F}_q$. Then, $\mathcal{C}$ is clearly irreducible over $\mathbb{F}_q$ but not absolutely irreducible and $F/\mathbb{F}_q = \mathbb{F}_{q^2}(X)$.

Definition 2.1.3. *A valuation ring of the function field F/K is a ring $\mathcal{O} \subseteq F$ with the following properties:*

1) $K \subsetneq \mathcal{O} \subsetneq F$.

2) For every $z \in F$, $z \in \mathcal{O}$ or $z^{-1} \in \mathcal{O}$.

Example 2.1.2. *Let us consider the case of a rational function field $K(x)$: let $p(x) \in K[x]$ be an irreducible polynomial, then the set*

$$\mathcal{O}_{p(x)} = \{\frac{f(x)}{g(x)} | f(x), g(x) \in K[x], p(x) \nmid g(x)\}$$

is clearly a valuation ring. Moreover two distinct irreducible polynomials give two distinct valuation rings.

Proposition 2.1.3. *Let $\mathcal{O}$ be a valuation ring of F/K. Then*

(a) $\mathcal{O}$ is a local ring, i.e. $\mathcal{O}$ has a unique maximal ideal $P = \mathcal{O} \setminus \mathcal{O}^\times$, where $\mathcal{O}^\times = \{z \in \mathcal{O} | \ \exists w \in \mathcal{O} \text{ such that } zw = 1\}$ is the group of units of $\mathcal{O}$.

(b) For $0 \neq x \in F$, $x \in P \Leftrightarrow x^{-1} \notin \mathcal{O}$.

(c) For the field $\tilde{K}$ of constants of F/K, we have $\tilde{K} \subseteq \mathcal{O}$ and $\tilde{K} \cap P = \{0\}$.

Let $\mathcal{O}$ be a valuation ring of F/K and P be its unique maximal ideal. Then P is a principal ideal. If $P = t\mathcal{O}$ then every $0 \neq z \in F$ has a unique representation of the form $z = t^n u$ with $n \in \mathbb{Z}$, et $u \in \mathcal{O}^\times$. The element $t \in F$ is called local parameter or uniformizing variable. $\mathcal{O}$ is a principal ring. More precisely, if $P = t\mathcal{O}$ and $\{0\} \neq I \subseteq \mathcal{O}$ is an ideal, then $I = t^n\mathcal{O}$ with $n \in \mathbb{N}$.

2.1.2 Places

Definition 2.1.4. *A place P of F/K is the maximal ideal of a valuation ring $\mathcal{O}$ of F/K. We note $\mathbb{P}_F = \{P \mid P \text{ is a place of } F/K\}$, the set of all places of F/K.*

If $\mathcal{O}$ is a valuation ring of F/K and P its maximal ideal, then $\mathcal{O}$ is uniquely determined by P, i.e. $\mathcal{O} = \{z \in F | z^{-1} \notin P\}$. We note $\mathcal{O}_\mathcal{P}$ the valuation ring of the place P.

For every place $P \in \mathbb{P}_F$, we can define a discrete valuation of F/K, $v_P : F \to \mathbb{Z} \cup \{\infty\}$ as follows: we choose $t_P \in F$ a local parameter of P and we set $v_P(0) := \infty$ and $v_P(z) := n$ for all $0 \neq z \in F$ such that $z = t_P^n u$ where $n \in \mathbb{Z}$ and $u \in \mathcal{O}^\times$. This definition is correct since the integer n only depends on the place P and not on the choice of the local parameter t_P.

Example 2.1.3. *Let $\mathbb{F}_4 = \{0, 1, \omega, \omega^2\}$ where ω the primitive root of the irreducible polynomial $X^2 + X + 1 \in \mathbb{F}_2[X]$. Let $F/\mathbb{F}_4$ be the algebraic function field associated to the elliptic curve $\mathcal{C}$ with plane model $y^2 + y = x^3 + 1$. This model represents the Fermat curve $(u^3 + v^3 = 1)$ used in an article of Baum-Shokrollahi [BS91]. We will mention it in Section 3.2.1 of next chapter. We have $\mathbb{F}_4$-rational points of $\mathcal{C}$:*

$$P_1 = (0, \omega), P_2 = (0, \omega^2), P_3 = (1, 0), P_4 = (1, 1),$$
$$P_5 = (\omega, 0), P_6 = (\omega, 1), P_7 = (\omega^2, 0), P_8 = (\omega^2, 1).$$

For $P_1 = (0, \omega)$, the place (P_1) with respect to the rational point P_1 of $\mathcal{C}$ is the unique maximal ideal in the valuation ring $\mathcal{O}_{P_1}$ generated by x and $y + x + \omega$. We write $(P_1) = (x, y + x + \omega)$. Similarly, we have also the places (P_i) with respect to the rational points P_i for $i = 2, ..., 8$. In particular,

$$\begin{aligned}
(P_2) &= (x, y + x + \omega^2),\\
(P_3) &= (x + 1, y),\\
(P_4) &= (x + 1, y + 1),\\
(P_5) &= (x + \omega, y),\\
(P_6) &= (x + \omega, y + 1),\\
(P_7) &= (x + \omega^2, y),\\
(P_8) &= (x + \omega^2, y + 1).
\end{aligned}$$

Example 2.1.4. *Let K be a field such that $char(K) \neq 2$. Let $e_1, e_2, e_3 \in \bar{K}$ be distinct, and consider the curve:*

$$\mathcal{E} : \quad f(x, y) := y^2 - (x - e_1)(x - e_2)(x - e_3) = 0.$$

For $i = 1, 2, 3$, let $P_i = (e_i, 0) \in \mathcal{E}$.

We consider the function field $F = Frac(\frac{K[x,y]}{\langle f(x,y)\rangle}) = \frac{K(x)}{\langle f(x,y)\rangle}$. The maximal ideal (P_i) in the valuation ring $\mathcal{O}_{P_i}$ of F/K is generated by y and $x - e_i$. We can consider (P_i) for $i = 1, 2, 3$ as the places of F/K.

Now we compute the valuation at P_i of $g := x - e_i \in F/K$. We have

$$v_{P_i}(\prod_{i=1}^{3}(x - e_i)) = \sum_{i=1}^{3} v_{P_i}(x - e_i) = v_{P_i}(x - e_i).$$

Thus,

$$v_{P_i}(y^2) = v_{P_i}(x - e_i).$$

We have $v_{P_i}(y^2) = 2v_{P_i}(y) = 2$. *Here* $v_{P_i}(y) = 1$ *since* y *is an uniformizing variable (local parameter) at* P_i. *Therefore, we have* $v_{P_i}(x - e_i) = 2$.

We have the following characterizations:

$$\mathcal{O}_P = \{z \in F | v_P(z) \geq 0\},$$

$$\mathcal{O}_P^{\times} = \{z \in F | v_P(z) = 0\},$$

$$P = \{z \in F | v_P(z) > 0\}.$$

Definition 2.1.5. *Let* $z \in F$ *and* $P \in \mathbb{P}_F$.

(a) We say that P *is a zero of* z *if and only if* $v_P(z) > 0$;

(b) We say that P *is a pole of* z *if and only if* $v_P(z) < 0$.

(c) If $v_P(z) = m > 0$, P *is a zero of* z *of order* m;

(d) If $v_P(z) = -m < 0$, P *is a pole of* z *of order* m.

Definition 2.1.6. *The field* $F_P := \mathcal{O}_P / P$ *is called the residue class field of* P.
We define the residue class map with respect to P *as*

$$\begin{array}{ccc} \mathcal{O}_P & \longrightarrow & F_P \cup \{\infty\} \\ x & \longmapsto & x(P) \end{array}$$

where $x(P)$ *is called the residue class map with respect to* P *if* $x \in \mathcal{O}_P$ *and* $x(P) := \infty$ *otherwise.*

This map allows us to consider K as a subfield of F_P. Indeed, since $K \in \mathcal{O}_P$ and $K \cap P = \{0\}$, the residue class map $\mathcal{O}_P \longrightarrow \mathcal{O}_P / P$ induces a canonical embedding of K into $\mathcal{O}_P / P$. This gives meaning to the following definition

Definition 2.1.7. *Let* P *a place in* F/K. $\deg P = [F_P : K]$ *is called the degree of* P. *A place of* F/K *of degree* 1 *is also called a rational place of* F/K.

Example 2.1.5. *Let* $F/K(x)$ *be the rational function field over* K. *It is obvious that for any non constant rational function* $r(x) \in F(x)$ *there exists a finite place* P *of* F *such that* $v_P(r(x)) \neq 0$. *Thus, the full constant field of* F *is* K. *The degree of a finite place* $p(x)$ *of* F *is equal to the degree of the polynomial* $p(x)$ *and the degree of the place* ∞ *of* F *is equal to* 1. *If* $K = \mathbb{F}_q$, *then* F *has thus exactly* $q + 1$ *rational places.*

The degree of a place is always finite, more precisely:

Proposition 2.1.4. *If* P *is a place of* F/K *and* $0 \neq x \in P$ *then*

$$\deg P \leq [F : K(x)] < \infty.$$

The set $\mathbb{P}_F$ of places of a function field F/K is non empty. We have the following corollary:

Corollary 2.1.1. *Let* F/K *be a function field,* $z \in F$ *transcendental over* K. *Then* z *has at least one zero and one pole. In particular,* $\mathbb{P}_F \neq \emptyset$.

Note that by this corollary, we have : each element $z \in F$, which is not in the constant field $\tilde{K}$ yields a non-constant function.

2.1.3 Independence of valuations

The independence of valuations is an important result that is used in particular for the problem of support moving of a divisor. It is useful for the practical use of Chudnovsky-type algorithm which is considered more deeply in the next chapters. The Weak Approximation Theorem, also referred to as the Theorem of Independence essentially says the following: If $v_1, ..., v_n$ are pairwise distinct discrete valuations of F/K and $z \in F$, and if we know the $n-1$ first values of z, $v_1(z), ..., v_{n-1}(z)$, then we cannot conclude anything about $v_n(z)$.

This theorem plays an important role in the proof of the two following results.

Theorem 2.1.1. *Let F/K a function field, $P_1, ..., P_n \in \mathbb{P}_F$ pairwise distinct place F/K, $x_1, ..., x_n \in F$, and $r_1, ..., r_n \in \mathbb{Z}$. Then there is an element $x \in F$such that*

$$v_{P_i}(x - x_i) = r_i \text{ for } i = 1, ..., n$$

Corollary 2.1.2. *Every function field has infinitely many places.*

In the next section, we will see that an element $x \in F$ which is transcendental over K has as many zeros as poles.

Proposition 2.1.5. *Let F/K a function field and $P_1, ..., P_r$ be zeros of the element $x \in F$. Then*

$$\sum_{i=1}^{r} v_{P_i}(x).\deg P_i \leq [F : K(x)].$$

Corollary 2.1.3. *In a function field F/K, every element $0 \neq x \in F$ has only a finite number of zeros and poles.*

2.1.4 Divisors

From now on, F/K will denote an algebraic function field of one variable such that K is the full constant field. This hypothesis is not critical to the theory since the field $\tilde{K}$ of constants of an algebraic function field F/K is a finite extension field K by Corollary 2.1.1, and F can be seen as a function field over $\tilde{K}$.

Definition 2.1.8. *The free abelian group (additively written) generated by the places of F/K is denoted $\mathcal{D}_F$ and is called the divisor group of F/K.*

The elements of $\mathcal{D}_F$ are called divisors of F/K. In other words, a divisor is a formal sum

$$D = \sum_{P \in \mathbb{P}_\mathbb{F}} n_P P \text{ with } n_P \in \mathbb{Z}, \text{ almost all } n_P = 0.$$

A divisor of the form $D = P$ with $P \in \mathbb{P}_F$ is called a prime divisor. Two divisors $D = \sum_{P \in \mathbb{P}F} n_P P$, and $D' = \sum_{P \in \mathbb{P}_F} n'_P P$ are added coefficient-wise:

$$D + D' = \sum_{P \in \mathbb{P}_F} (n_P + n'_P) P.$$

The zero element of the divisor group is

$$0 := \sum_{P \in \mathbb{P}_F} r_P P, \text{ with all } r_P = 0.$$

Definition 2.1.9. *We call support of a divisor $D = \sum_{P \in \mathbb{P}_F} n_P P \in \mathcal{D}_F$ the set*

$$supp\ D = \{P \in \mathbb{P}_F | v_P(D) \neq 0\}.$$

Definition 2.1.10. *For $Q \in \mathbb{P}_F$ and $D = \sum n_P P$, we define the valuation of D in Q, denoted $v_Q(D)$, by setting $v_Q(D) := n_Q$.*

Thanks to the previous definition, we can define a partial order on $\mathcal{D}_F$:

$$D_1 \leq D_2 \Longleftrightarrow v_P(D_1) \leq v_P(D_2) \text{ for all } P \in \mathbb{P}_F.$$

A divisor $D \geq 0$ is said positive or effective. The degree of a divisor is defined by

$$\deg D := \sum_{P \in \mathbb{P}_F} v_P(D) \cdot \deg P$$

and yields an homomorphism $\deg : \mathcal{D} \longrightarrow \mathbb{Z}$. A non zero element $x \in F$ has only finitely many zeros and poles. Thus the following definition makes sense:

Definition 2.1.11. *Let $0 \neq x \in F$ and let Z and N respectively the set of zeros and poles of x in $\mathbb{P}_F$. Then we define:*

$$(x)_0 := \sum_{P \in Z} v_P(x)P, \text{ the zero divisor of } x,$$

$$(x)_\infty := \sum_{P \in N} v_P(x)P, \text{ the pole divisor of } x,$$

$$(x) := (x)_0 - (x)_\infty, \text{ the principal divisor of } x.$$

Note that for all $0 \neq x \in F$, we have $(x)_0 \geq 0$, $(x)_\infty \geq 0$ and

$$(x) = \sum_{P \in \mathbb{P}_F} v_P(x)P. \tag{2.1}$$

The field K being algebraically closed in F, the elements $0 \neq x \in F$ which are constant are characterized by:

$$x \in K \Longleftrightarrow (x) = 0.$$

Theorem 2.1.2. *Every principal divisor has a degree equal to zero. More precisely: let $x \in F/K$ and $(x)_0$ resp. $(x)_\infty$ the divisors of the zeros resp. the divisors of the poles of x. Then*

$$\deg(x)_0 = deg(x)_\infty = [F : K(x)].$$

Definition 2.1.12. *The set*

$$\mathcal{P}_F := \{(x) \mid 0 \neq x \in F\}$$

is called the group of principal divisors of F/K. This is a subgroup of the group of divisors $\mathcal{D}_F$, since for $0 \neq x, y \in F$, $(xy) = (x) + (y)$ by (2.1). The factor group

$$\mathcal{C}_F := \mathcal{D}_F/\mathcal{P}_F$$

is called the divisor class group.

For a divisor $D \in \mathcal{D}_F$, the corresponding element in the factor group $\mathcal{C}_F$ is denoted by $[D]$, and is called the divisor class of D. Two divisors D and D' are said to be equivalent, written

$$D \sim D',$$

if $[D] = [D']$, i.e $D = D' + (x)$ for an element $x \in F \setminus \{0\}$. This relation is clearly an equivalence relation.

By Theorem 2.1, equivalent divisors have same degree. Thus, we can set $deg[\mathcal{A}] := deg\mathcal{A}$, for all divisor class $[\mathcal{A}] \in \mathcal{C}_F$.

2.1.5 Riemann-Roch Spaces

The following definition plays a fondamental role in the theory of algebraic function fields.

Definition 2.1.13. *For a divisor $D \in \mathcal{D}_F$, we set:*

$$\mathcal{L}(D) := \{x \in F | (x) \geq -D\} \cup \{0\}.$$

This definition means that if

$$D = \sum_{i=1}^{r} n_i P_i - \sum_{j=1}^{s} m_j Q_j$$

with $n_i > 0$, $m_j > 0$ then $\mathcal{L}(D)$ consists of all elements $x \in F$ such that:
(1) x has zeros of order greater than m_j at Q_j, for $j = 1, ..., s$.
(2) x may have poles only at the places $P_1, ..., P_r$, with the pole order at P_i being bounded by n_i $(i = 1, ..., r)$.

Here are some useful properties:

Properties 2.1.1. *Let $D \in \mathcal{D}_F$. Then:*
(1) $x \in \mathcal{L}(D)$ if and only if $v_P(x) \geq -v_P(D)$ for all $P \in \mathbb{P}_F$.
(2) $\mathcal{L}(D) \neq \{0\}$ if and only if there is a divisor $D' \sim D$ with $D' \geq 0$.
(3) For every divisor $D \in \mathcal{D}_F$, $\mathcal{L}(D)$ is a vector space K of finite dimension. We note $dim\mathcal{L}(D)$, the dimension of $\mathcal{L}(D)$. The integer $\dim D := \mathcal{L}(D)$ is called the dimension of D.
(4) If D' is a divisor equivalent to D then $\mathcal{L}(D)$ is isomorphic, as a K-vector space, to $\mathcal{L}(D')$.
(5) $\mathcal{L}(0) = K$.
(6) If $\deg D < 0$ then $\mathcal{L}(D) = \{0\}$ (in particular, it is true if $D < 0$). Therefore, if $\deg D < 0$ then $\dim D = 0$.
(7) Let D and D' two divisors of F/K with $D \sim D'$. Then, we have $\mathcal{L}(D) \subseteq \mathcal{L}(D')$ and $dim(\mathcal{L}(D')/\mathcal{L}(D)) \leq \deg D' - \deg D$.
(8) Let D and D' two divisors such that $D \sim D'$, then we have $\dim D = \dim D'$ and $\deg D = \deg D'$.
(9) For all zero degree divisors D, we have

$$\dim D = 0 \text{ or } \dim D = 1,$$

$$D \text{ is principal if and only if } \dim D = 1.$$

(10) For all zero degree divisor D, we have $\dim D \leq 1 + \deg D$.

Proposition 2.1.6. *There is a constant $\gamma \in \mathbb{Z}$ such that, for all divisors $D \in \mathcal{D}_F$, we have:*

$$\deg D - \dim D \leq \gamma$$

This means that γ is independent of any divisor. In fact, this constant only depends on the function field F/K.

Definition 2.1.14. *The genus g of F/K is defined by*

$$g := max\{\deg D - \dim D + 1 | D \in \mathcal{D}_F\}.$$

The genus is the most important invariant of a function field. It is an integer greater than or equal to 0 since if we take in the definition the zero divisor, we obtain $\deg 0 - \dim 0 + 1 = 0$, so $g \geq 0$.

The following theorem, called Riemann Theorem is a first important result about the dimension of a divisor.

Theorem 2.1.3. *Let F/K a function field of genus g.*
(a) For all divisor $D \in \mathcal{D}_F$, $\dim D \geq \deg D + 1 - g$.
(b) There is an integer c, dependent of F/K, such that

$$\dim D = \deg D - g + 1$$

as soon as $\deg D \geq c$.

In general, determining the genus of a function field is difficult. However, in certain particular cases, it can be determined by applying the Riemann-Roch theorem which we will see in the next section, or even by applying Riemann's theorem which is a weak version of Riemann-Roch's theorem.

2.1.6 Riemann-Roch Theorem

Definition 2.1.15. *For $D \in \mathcal{D}_F$,*

$$i(D) = \dim D - \deg D + g - 1$$

is called index of specialty of D.

$i(D)$ is a non-negative integer, and $i(D) = 0$ if $\deg D$ is large enough.

Definition 2.1.16. *We say that a divisor $W \in \mathcal{D}_F$ is canonical if* $\deg W = 2g - 2$ *and* $\dim W = g$.

A useful characterization of the canonical divisors is as follows:

Proposition 2.1.7. *A divisor $W \in \mathcal{D}_F$ is canonical if and only if* $\deg W = 2g - 2$ *and* $\dim W \geq g$.

There always exist a canonical divisor and all canonical divisors are equivalent. As a consequence, the canonical divisors of F/K form a whole class $[W]$ in the group of divisor classes $\mathcal{C}_F$. This class of divisors is called the canonical class of F/K.

Moreover, by duality, we have the following theorem.

Theorem 2.1.4. *Let D a divisor and W a canonical divisor of F/K. Then the index of specialty $i(D)$ of D is such that:*

$$i(D) = \dim(W - D).$$

We can now give the Riemann-Roch theorem, without doubt the most important theorem of the theory of the algebraic function fields.

Theorem 2.1.5. *Let W a canonical divisor of F/K. Then, for every divisor $D \in \mathcal{D}_F$,*

$$\dim D = \dim(W - D) - g + 1 + \deg D.$$

An important corollary of Riemann-Roch theorem is:

Theorem 2.1.6. *If D is a divisor of F/K of degree $\geq 2g-1$ then*

$$\dim D = \deg D - g + 1.$$

Note that the bound $2g-1$ is the best possible since for a canonical divisor W, $\dim W > \deg W + 1 - g$ by Definition 2.1.16 and Proposition 2.1.7.

Now, let us give some important consequences of Riemann-Roch theorem.

Proposition 2.1.8. *For a given function field, the following conditions are equivalent:*
(1) F/K is rational, i.e $F = K(x)$ for a transcendental element x on K.
(2) F/K has a genus 0, and there is a divisor $D \in \mathcal{D}_F$ with $\deg D = 1$.

In fact, there exist non-rational function fields of genus 0 but they can not have divisors of degree 1. However, if K is an algebraically closed field or a finite field, there always exists degree 1 divisor. For these cases, we have $g = 0$ if and only if F/K is rational.

Another interesting result concerns an improvement of the low approximation theorem that we call strong approximation theorem.

Theorem 2.1.7. *Let $S \subsetneq \mathbb{P}_F$ a strong sub-set of $\mathbb{P}_F$ and $P_1, ..., P_r \in S$. Consider $x_1, ...x_r \in F$ and $n_1, ..., n_r \in \mathbb{Z}$. Then there exists an element $x \in F$ such that:*

$$v_{P_i}(x - x_i) = n_i \ \ (i = 1, ..., r), \text{ and}$$

$$v_P(x) \geq 0, \text{ for all } P \in S \setminus \{P_1, ..., P_r\}.$$

This theorem is used to state the following Moving Lemma which will be useful for the effective construction of Chudnovsky algorithms.

Lemma 2.1.1. *(Moving Lemma) Let F/K an algebraic function field. Let $T := \{P_1, \ldots, P_N\}$ a set of places of F of arbitrary degrees. Then, for all $D \in \mathcal{D}_F$, there exist $D' \in \mathcal{D}_F$ such that $D \sim D'$ and $supp\ D \cap T = \emptyset$.*

Now, we seek elements of F which have only one pole (with order possibly large).

Proposition 2.1.9. *Let $P \in \mathbb{P}_F$. Then, for all $n \geq 2g$, there exists an element $x \in F$ with pole divisor $(x)_\infty = nP$.*

Definition 2.1.17. *A divisor $D \in \mathcal{D}_F$ is called non-special if $i(D) = 0$, otherwise D is called special.*

Here are some immediate consequences of this definition.

Properties 2.1.2. *(a) D is non-special if and only if $\dim D = \deg D - g + 1$.*
(b) If $\deg D > 2g - 2$ then D is non-special.
(c) The property of a divisor D being special or not only depends on the class $[D]$ of D in the divisor class group.
(d) Canonical divisors are specials.
(e) Every divisor D such that $\dim D > 0$ and $\deg D < g$ is special.
(f) If D is non-special and $D' \leq D$ then D is non-special.

We conclude this section with an inequality for the dimension of an arbitrary divisor. By Riemann-Roch Theorem, we know the dimension of a divisor D such that $\deg D \geq 2g - 1$ according to its degree. Indeed, in this case, we have $i(D) =$

$dim(W - D) = 0$ (where W is a canonical divisor) since $deg(W - D) < 0$. However, when $\deg D \leq 2g - 2$, we only have the inequality

$$\dim D \geq \deg D + 1 - g.$$

In this case, the following theorem, called Clifford's Theorem, gives a bound on the dimension of D.

Theorem 2.1.8. *(Clifford's Theorem) For all divisors D such that $0 \leq \deg D \leq 2g - 2$, we have :*

$$\dim D \leq 1 + \frac{1}{2} \deg D.$$

2.2 Extensions of algebraic function fields

Every function field can be seen as a finite extension of the rational function field. So, it is interesting to consider extensions F'/F of algebraic function fields. In this thesis, we only consider the case where the constant field is a finite field. Thus, F denotes here an algebraic function field of genus g whose the constant field is the finite field $\mathbb{F}_q$.

2.2.1 Algebraic extensions of function fields

Definition 2.2.1. *(a) An algebraic function field F'/K' is called an algebraic extension of F/K if $F' \supseteq F$ is an algebraic field extension and $K' \supseteq K$.*

(b) The algebraic extension F'/K' of F/K is called a constant field extension if $F' = FK'$, the composite field of F and K'.

(c) The algebraic extension F'/K' of F/K is called a finite extension if $[F' : F] < \infty$.

(d) The algebraic extension F'/K' of F/K is of Galois if it is finite and if its automorphism group $Aut(F'/F) = \{\sigma : F' \longrightarrow F | \sigma$ is an isomorphism such that $\sigma(a) = a$ for all $a \in F\}$ has order $[F' : F]$. In this case, we call $Aut(F'/F)$ the Galois group of F'/F and we note: $Gal(F'/F) := Aut(F'/F)$.

Definition 2.2.2. *Consider an algebraic extension F'/K' of F/K. A place $P' \in \mathcal{P}_{F'}$ is said to lie over P if $P \subseteq P'$. We also say that P' is an extension of P or that P lies under P', and we write $P'|P$.*

Definition 2.2.3. *Let F'/K' an algebraic extension F/K and let $P' \in \mathcal{P}_{F'}$ a place of F'/K' lying over $P \in \mathcal{P}_F$.*

(a) The integer $e(P'|P) := e$ such that

$$v_{P'}(x) = e \cdot v_P(x) \text{ for all } x \in F$$

is called the ramification index of P' over P. We say that $P'|P$ is ramified if $e(P'|P) > 1$, and $P'|P$ is non ramified if $e(P'|P) = 1$.

(b) $f(P'|P) := [F'_{P'} : F_P]$ is called the relative degree of P' over P.

We note that the relative degree can be finite or infinite, whereas the ramification index is always a natural number.

Definition 2.2.4. *Let F'/K' an algebraic extension of F/K and let $P \in \mathcal{P}_F$.*

(a) An extension P' of P in F' is tamely ramified (resp. wildly ramified) if $e(P'|P) > 1$ and the characteristic char K of K does not divide $e(P'|P)$ (resp. char K divides $e(P'|P)$).

(b) We say that P is ramified (resp. non ramified) in F'/F if there is at least a place $P' \in \mathbb{P}_F$ lying over P such that $P'|P$ is ramified (resp. if $P'|P$ is non ramified for every place $P'|P$). The place P is tamely ramified in F'/F if it is ramified in F'/F and no extension of P in F' is wildly ramified. If there is at least a place wildly ramified $P'|P$, we say that P is wildly ramified in F'/F.

(c) P is totally ramified in F'/F if there is just one extension $P' \in \mathcal{P}_{F'}$, and the index of ramification is $e(P'|P) = [F' : F]$.

(d) F'/F is ramified (resp. non ramified) if at least one place $P \in \mathcal{P}_F$ is ramified in F'/F (resp. if none of the places $P \in \mathcal{P}_F$ are ramified in F'/F).

(e) F'/F is tame if no place $P \in \mathcal{P}_F$ is wildly ramified.

(f) A place P is totally splitted in F if $e(P'|P) = f(P'|P) = 1$ for all the places P' of this extension which lie over P.

Let us give a geometrical interpretation of the previous definitions. Consider the elliptic curve $\mathcal{C}$ défined over $\mathbb{F}_3$ by the Weierstrass equation $y^2 = x^3 + x^2 + 2$. We associate to this curve the (elliptic) function field $F_\mathcal{C}/\mathbb{F}_3$. When we construct places that lie over the places of the rational function field over $\mathbb{F}_3$, we look for the points of the elliptic curve which are above a point $x_0 \in \mathbb{F}_3$. This means that we have to find y such that the points $P_0 = (x_0, y)$ are on the curve. This corresponds to finding the intersections of the line $x = x_0$ with the curve. The number of such points is equal to the degree of y in the equation of the curve. There are three possible cases.

1. The line $x = x_0$ cuts the curve in exactly two points (note that $[F_\mathcal{C} : \mathbb{F}_3(x)] = 2$). In this case, we say that the place $(x_0) := (x - x_0)\mathcal{O}_{P_0} \subset F_\mathcal{C}$ is totally splitted.

2. The line $x = x_0$ is tangent to the curve. Then, we say that the place (x_0) is ramified.

3. The line $x = x_0$ does not cut the curve. In this case, we first extend the field of constants of $\mathbb{F}_3(x)/\mathbb{F}_3$ to $\mathbb{F}_3(x)/\mathbb{F}_9$. Then, we extend the field of rational fractions to $\mathbb{F}_\mathcal{C}/\mathbb{F}_9$ and we regroup the points to come back to the elliptic function field $\mathbb{F}_\mathcal{C}/\mathbb{F}_3$.

Now, we give some important results related to extensions of algebraic function fields.

Proposition 2.2.1. *Let F'/K' be an algebraic extension of F/K. Then:*

(1) K'/K is algebraic.

(2) F'/K' is a finite extension of F/K if and only if $[K' : K] < \infty$.

(3) Let $F_1 = FK$. Then F_1/K' is an extension of the field of constants of F/K, and F'/K' is a finite extension of F_1/K' having the same constant field.

Proposition 2.2.2. *Let F'/K' be an algebraic extension of F/K. Suppose that P (resp. P') is a place of F/K (resp. F'/K'), and let $\mathcal{O}_P \subseteq F$ (resp. $\mathcal{O}_{P'}$) be the corresponding valuation ring, and v_P (resp.$v_{P'}$) be the corresponding discrete valuation. Then the following assertions are equivalent:*

(1) $P'|P$,

(2) $\mathcal{O}_P \subseteq \mathcal{O}_{P'}$,

(3) There exists an integer $e \geq 1$ such that $v_{P'}(x) = e \cdot v_P(x)$ for all $x \in F$.

Moreover, if $P'|P$ then

$$P = P' \cap F \text{ and } \mathcal{O}_P = \mathcal{O}_{P'} \cap F.$$

This is why P is called the restriction of P' to F.

As a consequence, for $P'|P$, there is a canonical embedding of the residue class field $F_P = \mathcal{O}_P/P$ into the residue class field $F_{P'} = \mathcal{O}_{P'}/P'$, given by

$$x(P) \mapsto x(P') \; for \; x \in \mathcal{O}_P.$$

Thus, F_P can be considered as a subfield of $F'_{P'}$.

Proposition 2.2.3. *Let F'/K' be an algebraic extension of F/K and P' be a place of F'/K' lying over $P \in \mathcal{P}_F$. Then*
(a) $f(P'|P) < \infty \Longleftrightarrow [F' : F] < \infty$.
(b) If F''/K'' is an algebraic extension of F'/K' and $P'' \in \mathbb{P}_F$ is an extension of P', then

$$e(P''|P) = e(P''|P') \cdot e(P'|P),$$

$$f(P''|P) = f(P''|P') \cdot f(P'|P).$$

Now, we focus on the existence of extensions of places in extensions of functions fields.

Proposition 2.2.4. *Let F'/K' be an algebraic extension of F/K.*
(1) For each place $P' \in \mathcal{P}_{F'}$, there is exactly one place $P \in \mathcal{P}_F$ such that $P'|P$, namely $P = P' \cap F$.
(2) Conversely, every place $P \in \mathcal{P}_F$ has at least one, but only finitely many extensions $P' \in \mathcal{P}_{F'}$.

Theorem 2.2.1. *Let F'/K' be a finite extension of F/K, P a place of F/K and $P_1, ..., P_m$ be all the places of F'/K' lying over P. Let $e_i := e(P_i|P)$ be the ramification index and $f_i := f(P_i|P)$ the relative degree of $P_i|P$. Then*

$$\sum_{i=1}^{m} e_i f_i = [F' : F].$$

Corollary 2.2.1. *Let F'/K' be a finite extension of F/K et $P \in \mathcal{P}_F$. Then*
(a) $|\{P' \in \mathcal{P}_{F'}; P' \text{ lies over } P\}| \leq [F' : F]$.
(b) If $P' \in \mathcal{P}_{F'}$ lies over P then

$$e(P'|P) \leq [F' : F] \text{ and } f(P'|P) \leq [F' : F].$$

2.2.2 Galois extensions of function fields

In this section, we consider the particular case of Galois extensions of algebraic function fields. These extensions have many interesting properties not valid for any extensions.

Proposition 2.2.5. *Let F'/K' be a Galois extension of F/K and P be a place of F/K. Then the Galois group $Gal(F'/F)$ acts on the set of all the extensions of P: $\{P' \in \mathcal{P}_{F'} | P' \text{ lies over } P\}$ via $\sigma(P') = \{\sigma(x) | x \in P'\}$ and the corresponding valuation $v_{\sigma(P')}$ is characterized by*

$$v_{\sigma(P')}(y) = v_{P'}(\sigma^{-1}(y)) \text{ for all } y \in F'.$$

Theorem 2.2.2. *Let F'/K' be a Galois extension of F/K and $P_1, P_2 \in \mathbb{P}_{F'}$ extensions of $P \in \mathbb{P}_F$. Then there exists an element $\sigma \in Gal(F'/F)$ such that $P_2 = \sigma(P_1)$. In other words, the Galois group acts transitively on the set of extensions of P.*

Corollary 2.2.2. *Let F'/K' be a Galois extension of F/K. Let $P_1, ..., P_r$ be all the places of F' lying over P. Then we have:*
(a) $e(P_i|P) = e(P_j|P)$ and $f(P_i|P) = f(P_j|P)$ for all i, j. Moreover, we set

$$e(P) := e(P_i|P) \text{ and } f(P) := f(P_i|P)$$

and call $e(P)$ (resp. $f(P)$) the ramification index (resp. the relative degree) of P in F'/F.
(b) $e(P) \cdot f(P) \cdot r = [F' : F]$. In particular, $e(P), f(P)$ and r divide the degree $[F' : F]$.

2.3 Algebraic function fields over finite constant fields

In this section, we consider the case where the constant field is a finite field, $\mathbb{F}_q$.

Proposition 2.3.1. *An algebraic function field $F/\mathbb{F}_q$ has at most finitely many rational places.*

For each integer $n \geq 1$, we consider the composite field

$$F_n := \mathbb{F}_{q^n} \cdot F.$$

This is an algebraic function field over $\mathbb{F}_{q^n}$ which is called a constant field extension of $F/\mathbb{F}_q$ of degree n. Let N_n denote the number of rational places of $F_n/\mathbb{F}_{q^n}$, which is a finite number by Proposition 2.3.1.

Definition 2.3.1. *The zeta function of $F/\mathbb{F}_q$ is defined to be the formal power series over the complex numbers*

$$Z(F,t) = exp\Big(\sum_{n=1}^{\infty} \frac{N_n}{n} t^n\Big) \in \mathbb{C}[[t]].$$

Theorem 2.3.1. *Let $F/\mathbb{F}_q$ be an algebraic function field of genus g. Then:*
(i) $Z(F,t)$ is a ratinal function of the form

$$Z(F,t) = \frac{L(F,t)}{(1-t)(1-qt)},$$

where $L(F,t) \in \mathbb{Z}[t]$, called L-polynomial of $F/\mathbb{F}_q$, that is a polynomial of degree $2g$ with $L(F,0) = 1$ and leading coefficient q^g. Moreover, $L(F,1)$ is equal to the divisor class number $h(F)$ of F.
(ii) Factor $L(F,t)$ into the form

$$L(F,t) = \prod_{i=1}^{2n} (1 - \omega_i t) \in \mathbb{C}[t]$$

where

$$|\omega_i| = q^{1/2} \text{ for } 1 \leq i \leq 2g.$$

This theorem plays an essential role in the establishment of the Hasse-Weil theorem.

2.3.1 Bounds on the number of rational places

Theorem 2.3.2. *(Hasse-Weil Bound) The number $N = N(f)$ of places of $F/\mathbb{F}_q$ of degree one satisfies the inequality*

$$|N - (q+1)| \leq 2g\sqrt{q}.$$

This result is due to Hasse for $g = 1$ and to Weil for the general case. Notice that when applying this bound to the constant field extension $F_r/\mathbb{F}_{q^r}$, we obtain

$$|N_r - (q^r + 1)| \leq 2g\sqrt{q}.$$

An algebraic function field $F/\mathbb{F}_q$ of genus g, which reaches the Hasse-Weil bound, i.e. it has exactly $q + 1 + 2g\sqrt{q}$ places of degree 1, is said maximal. A necessary condition for $F/\mathbb{F}_q$ to be maximal is that q is a square. Moreover the following result due to Ihara show that a function field can be maximal only if its genus is relatively small compare to q.

Proposition 2.3.2. *Let $F/\mathbb{F}_q$ be a maximal algebraic function field of genus g. Then $g \geq \frac{1}{2}(q - \sqrt{q})$.*

Lemma 2.3.1. *Existence of a place of degree n. Let $F/\mathbb{F}_q$ be an algebraic function field of genus g. If n is an integer such that*

$$2g + 1 \leq q^{\frac{n-1}{2}}(\sqrt{q} - 1),$$

Then there is at least a place of degree n in $F/\mathbb{F}_q$.

The Serre bound is an improvement of the Hasse-Weil bound.

Theorem 2.3.3. *(Serre-Weil Bound) The number $N = N(F)$ of places of $F/\mathbb{F}_q$ of degree one satisfies the inequality*

$$|N - (q + 1)| \leq g[2\sqrt{q}],$$

where $[.]$ denotes the integer part.

When the genus is large compare to q, Hasse-Weil bound does not give a good approximation of the number of rational places.

Definition 2.3.2. *(a) For an arbitrary integer g, we define*

$$N_q(g) := max\{N(F) \mid F \text{ is a function field over } \mathbb{F}_q \text{ of genus } g\}.$$

(b) We call Ihara constant the integer

$$A(q) := lim\ sup_{g\to\infty} \frac{N_q(g)}{g}.$$

Notice that the Serre bound gives $0 \leq A(q) \leq [2\sqrt{q}]$. The best upper bound is due to Drinfeld-Vladut.

Theorem 2.3.4. *(Drinfeld-Vladut Bound) The Ihara constant has the following bound*

$$A(q) \leq q^{\frac{1}{2}} - 1.$$

This bound is reached when q is a square.

2.4 Elliptic and hyperelliptic function fields

Now, we will give examples of the application of the basic notions of the theory of algebraic functions. For this purpose, we will establish several important properties of some remarkable algebraic functions.

The rational function field has a zero genus. Conversely, we have seen that if F/K is a function field of genus 0 with a divisor $D \in \mathcal{D}_F$ of degree 1, then F/K is rational. As a result, the simplest non-rational function fields are of genus 1.

Definition 2.4.1. *An algebraic function field (where K is the full constant field) is said elliptic if*
(1) The genus of F/K is $g = 1$.
(2) There is a divisor $D \in \mathcal{D}_F$ such that $\deg D = 1$.

Proposition 2.4.1. *Let F/K be an elliptic function field.*
(a) If $charK \neq 2$, there exist $x, y \in F$ such that $F = K(x, y)$ and

$$y^2 = f(x) \in K[x] \tag{2.2}$$

where $f(x) \in K[x]$ is a square-free polynomial (i.e a polynomial which is not divisible by the square of an irreducible polynomial) of degree 3.
(b) If $charK = 2$, there exist $x, y \in F$ such that $F = K(x, y)$ and

$$y^2 + y = f(x) \in K[x] \text{ with } \deg f = 3 \tag{2.3}$$

or

$$y^2 + y = x + \frac{1}{ax + b} \text{ with } a, b \in K \text{ and } a \neq 0. \tag{2.4}$$

Definition 2.4.2. *An hyperelliptic function field over K is an algebraic function field F/K of genus $g \geq 2$ with rational subfield $K(x) \subseteq F$ with $[F : K(x)] = 2$*

We can notice that without the condition $g \geq 2$, an elliptic function field is a particular case of hyperelliptic function field since every elliptic function field F/K has a rational subfield $K(x) \subseteq F$ with $[F : K(x)] = 2$.

2.5 Example

We consider the elliptic curve $\mathcal{C}/\mathbb{F}_q$ of equation $Y^2 = X^3 + X + 1$ defined over the finite field $\mathbb{F}_q$ with $q = 3$. Set $\mathcal{C}(X, Y) = Y^2 - X^3 - X - 1 = 0$ the definition polynomial of the curve. We will study the decomposition of the place of degree 1 and 2 of the rational function field $\mathbb{F}_q(X)$ in the function field $F/\mathbb{F}_q$ of the curve $\mathcal{C}/\mathbb{F}_q$.

2.5.1 Places of degree one

There are three places of degree 1 in the rational function field $\mathbb{F}_3(X)/\mathbb{F}_3$: (X), $(X - 1)$, and $(X - 2)$. The place in $F/\mathbb{F}_3 = \mathbb{F}_3(\mathcal{C})$ lying above the place (X) are $(X, Y - 1)$ and $(X, Y - 2)$. So the place (X) of $\mathbb{F}_3(X)$ in is totally splitted in $F/\mathbb{F}_3$. The place in $F/\mathbb{F}_3$ lying above the place $(X - 1) \in \mathbb{F}_3(X)$ is ramified. It is the place $(X - 1, Y)$ that is the ideal generated by $(X - 1)$. The place lying above the place $(X - 2)$ is inert since the equation $Y^2 - 2$ does not have a solution in $\mathbb{F}_3$. This is the place $(X - 2, Y^2 - 2) = \langle X - 2 \rangle$, the ideal generated by $(X - 2)$.

2.5.2 Places of degree two

In this case there are three places of degree 2 in the field $\mathbb{F}_q(X)$. These are the maximal ideals of degree 2 (generated by the irreducible polynomials of degree 2) of the polynomial ring $\mathbb{F}_q[X]$.

$P_1 = (X^2 + 1)$, $P_2 = (X^2 + X + 2)$, and $P_3 = (X^2 + 2X + 2)$.

Moreover, these places being of degree 2 in $\mathbb{F}_q(X)$, we have to define the field $\mathbb{F}_{q^2}$ in order to study and explicitly describe the decomposition of these places. Let us represent the field $\mathbb{F}_{q^2}$ by the splitting field of the polynomial $P(X) = X^2 + X + 2$ where α is a primitive element. Then $P(\alpha) = 0$, i.e. $\alpha^2 + \alpha = 1$. Let G_1 be the Galois group of the extension $\mathbb{F}_{q^2}$ over $\mathbb{F}_q$. Recall that the Galois group is independent of the chosen representation of the field, which means that its action will be used to describe each of the places lying above the places of degree 2 of $\mathbb{F}_q(X)$. Thus, $G_1 = Gal(\mathbb{F}_{q^2}/\mathbb{F}_q) = \{id, \phi\}$ where $\phi : x \mapsto x^q$ is the Frobenius, automorphism of $\mathbb{F}_{q^2}$ that leaves invariant $\mathbb{F}_q$.

Moreover, we perfectly know the Galois group G_2 of the function field $F/\mathbb{F}_q(X)$ since $F/\mathbb{F}_q(X)$ is a quadratic extension of Kummer type. This group is

$$G_2 = Gal(F/\mathbb{F}_q(X)) = \{id, \sigma\}$$

where σ is the automorphism of F that leaves invariant $\mathbb{F}_q(X)$ such that $\sigma(Y) = -Y$ (since $\epsilon = -1$ is a 2^{nd} root of unity). Now, we describe the places lying above the places of $\mathcal{P} = \{P_1, P_2, P_3\}$. We know that there will be at least two places lying above each of the places of $\mathcal{P}$.

Decomposition of the place $P_1 = (X^2 + 1)$

Since P_1 is an irreducible polynomial of degree 2 over $\mathbb{F}_3$, the root of $X^2 + 1$ are the conjugate roots (under the action of G_1) α^2 and α^6. Then, since $\mathcal{C}(Y, \alpha) = Y^2 - 1 = 0$, we have $Y = 1$ or $Y = -1$. This proves that there are two places lying above P_1, which is totally splitted in $F/\mathbb{F}_q$.

Let us choose the point $P_0 = (x_0, y_0) = (\alpha^2, 1)$ to describe the places lying above P_1.

Then, the first place is given by the orbit of the point P_0 under the action of G_1, i.e.

$$P_{11} = \{(x_0, y_0), (x_0^q, y_0^q)\} = \{(\alpha^2, 1), (-\alpha^2, 1)\}.$$

The second place (possibly the same) is given by P_{12} the conjugate place of P_{11} under the action of the group G_2. We have

$$P_{12} = \{(x_0, -y_0), (x_0^q, (-y_0)^q)\} = \{(x_0, -y_0), (x_0^q, -(y_0)^q)\} = \{(\alpha^2, -1), (-\alpha^2, -1)\}.$$

Notice that here, we have $(\alpha^2)^q = \alpha^6 = -\alpha^2$ and we obtain two distinct places P_{11} and P_{12} above P_1.

Decomposition of the place $P_2 = (X^2 + X + 2)$

In this case, the roots of $X^2 + X + 2$ are the conjugate roots (under the action of G_1) α and α^3. Then, since $\mathcal{C}(Y, \alpha) = Y^2 = 0$, we have $Y = 0$ and we can see that the place is ramified.

$$P_2' = \{(\alpha, 0), (\alpha^3, 0)\}.$$

Decomposition of the place $P_3 = (X^2 + 2X + 2)$

In this case, the roots of $X^2 + 2X + 2$ are the conjugate roots (under the action of G_1) α^5 and $(\alpha^5)^q = \alpha^7$. We have $\mathcal{C}(Y, \alpha^5) = Y^2 + 1 = 0$, which means that $Y^2 = 2 =$

α^4 and so $Y = \alpha^2$ or $Y = -\alpha^2 = \alpha^6$. Therefore, there are two places lying above P_3 which is totally splitted in $F/\mathbb{F}_q$. Let us choose the point $P_0 = (x_0, y_0) = (\alpha^5, \alpha^2)$ to describe these two places lying above P_3. The first place is given by the orbit of the point P_0 under the action of G_1, i.e:

$$P_{31} = \{(x_0, y_0), (x_0^q, y_0^q)\} = \{(\alpha^5, \alpha^2), ((\alpha^5)^q, (\alpha^2)^q)\} = \{(\alpha^5, \alpha^2), (\alpha^7, \alpha^6)\}.$$

The second place is given by P_{32}, the conjugate place of P_{31} under the action of the group G_2.

$$P_{32} = \{(x_0, -y_0), (x_0^q, (-y_0)^q)\} = \{(x_0, -y_0), (x_0^q, -(y_0)^q)\} = \{(\alpha^5, -\alpha^2), (\alpha^7, \alpha^2)\}.$$

Chapter 3

Chudnovsky-Chudnovsky Multiplication Algorithm

Generalizing interpolation algorithms on the projective line over $\mathbb{F}_q$ to algebraic curves of higher genus over $\mathbb{F}_q$, D.V. Chudnovsky and G.V. Chudnovsky provided a method [CC88] which enabled to prove the linearity ([Bal99; Bal+19]) of the bilinear complexity of multiplication in finite extensions of a finite field. Moreover, they proposed the first known multiplication algorithm of two elements in $\mathbb{F}_{q^n}$ using interpolation to algebraic function fields of one variable defined over $\mathbb{F}_q$. This is the so-called Chudnovsky and Chudnovsky multiplication algorithm (abbreviated CCMA), also called Chudnovsky algorithm to simplify. In the framework of this thesis, we just consider the symmetric version of CCMA.

3.1 Description of algorithm

Let $F/\mathbb{F}_q$ be an algebraic function field over the finite field $\mathbb{F}_q$ of genus $g(F)$. We denote by $N_k(F/\mathbb{F}_q)$ the number of places of degree k of F over $\mathbb{F}_q$. Let D be a divisor and we denote by $\mathcal{L}(D)$ the Riemann-Roch space associated to D.

Recall that $\mathcal{O}_Q$ denotes the valuation ring of the place Q and $F_Q = \mathcal{O}_Q/\langle Q\rangle$ is the residue class field of Q which is isomorphic to $\mathbb{F}_{q^{\deg Q}}$ where $\deg Q$ is the degree of the place Q. The order of a divisor $D = \sum_P a_P P$ in the place P is the number a_P, denoted $\mathrm{ord}_P(D)$. The support of a divisor D is the set $suppD$ of the places P such that $\mathrm{ord}_P(D) \neq 0$.

Let us define the *Hadamard product* in $\mathbb{F}_{q^{l_1}} \times \mathbb{F}_{q^{l_2}} \times \cdots \times \mathbb{F}_{q^{l_N}}$ denoted by $\odot$ where the l_i's denote positive integers, by

$$(u_1, u_2, ..., u_N) \odot (v_1, v_2, ..., v_N) = (u_1v_1, u_2v_2, ..., u_Nv_N).$$

The following theorem describes the original multiplication algorithm of D.V. and G.V. Chudnovsky [CC88].

Theorem 3.1.1. *Let*

- $F/\mathbb{F}_q$ *be an algebraic function field,*
- Q *be a degree n place of* $F/\mathbb{F}_q$*,*
- D *be a divisor of* $F/\mathbb{F}_q$*,*
- $\mathcal{P} = \{P_1, \ldots, P_N\}$ *be a set of places of degree one of* $F/\mathbb{F}_q$*.*

We suppose that $suppD \cap \{Q, P_1, ..., P_N\} = \emptyset$ *and that*

(i) the evaluation map

$$\begin{array}{rccc} Ev_Q: & \mathcal{L}(D) & \to & F_Q \\ & f & \mapsto & f(Q) \end{array}$$

is surjective

(ii) the evaluation map

$$\begin{array}{rccc} Ev_{\mathcal{P}}: & \mathcal{L}(2D) & \to & \mathbb{F}_q^N \\ & f & \mapsto & \left(f(P_1), \ldots, f(P_N)\right) \end{array}$$

is injective.

Then

(1) For any two elements x, y in $\mathbb{F}_{q^n}$, we have:

$$xy = E_Q \circ Ev_{\mathcal{P}}^{-1}|_{ImEv_{\mathcal{P}}} \left(E_{\mathcal{P}} \circ Ev_Q^{-1}(x) \odot E_{\mathcal{P}} \circ Ev_Q^{-1}(y) \right), \tag{3.1}$$

where E_Q denotes the canonical projection from the valuation ring $\mathcal{O}_Q$ of the place Q in its residue class field F_Q, $E_{\mathcal{P}}$ the extension of $Ev_{\mathcal{P}}$ on the valuation ring $\mathcal{O}_Q$ of the place Q, $Ev_{\mathcal{P}}^{-1}|_{ImEv_{\mathcal{P}}}$ the restriction of the inverse map of $Ev_{\mathcal{P}}$ on its image, and $\circ$ the standard composition map.

(2)

$$\mu_{q,sym}^b(n) \leq N.$$

Since Q is a place of degree n, the residue class field F_Q of place Q is an extension of degree n of $\mathbb{F}_q$ and it therefore can be identified to $\mathbb{F}_{q^n}$. Moreover, the evaluation map Ev_Q being onto, one can associate the elements $x, y \in \mathbb{F}_{q^n}$ with elements of $\mathbb{F}_q$-vector space $\mathcal{L}(D)$, denoted respectively f and g. We define $h := fg$ by

$$\big(h(P_1), ..., h(P_N)\big) = E_{\mathcal{P}}(f) \odot E_{\mathcal{P}}(g) = \big(f(P_1)g(P_1), ..., f(P_N)g(P_N)\big). \tag{3.2}$$

We know that such an element h belongs to $\mathcal{L}(2D)$ since the functions f, g lie in $\mathcal{L}(D)$. Moreover, thanks to injectivity of $Ev_{\mathcal{P}}$, the function h is in $\mathcal{L}(2D)$ and is uniquely determined by (3.2). We have

$$xy = Ev_Q(f) \cdot Ev_Q(g) = E_Q(f) \cdot E_Q(g) = E_Q(h)$$

where E_Q is the canonical projection from the valuation ring $\mathcal{O}_Q$ of the place Q in its residue class field F_Q, Ev_Q is the restriction of E_Q over the vector space $\mathcal{L}(D)$.

The multiplication of any x and y in $\mathbb{F}_{q^n}$ can be performed briefly with the following steps numbered in the diagram below:

(1) *lift* x and y to some functions f and g in the Riemann-Roch space $\mathcal{L}(D)$ so that $f(Q) = x$ and $g(Q) = y$;

(2) *evaluate* f and g on each point P_i of the set $\mathcal{P}$;

(3) *compute* the product of these evaluations for each P_i: $u_i := f(P_i) \cdot g(P_i)$. We obtain the vector $(u_1, ..., u_N)$;

(4) *interpolate* this vector to the unique function $h \in \mathcal{L}(D + D)$ satisfying $h(P_i) = u_i$;

(5) *evaluate* function h at Q to find product of x and y.

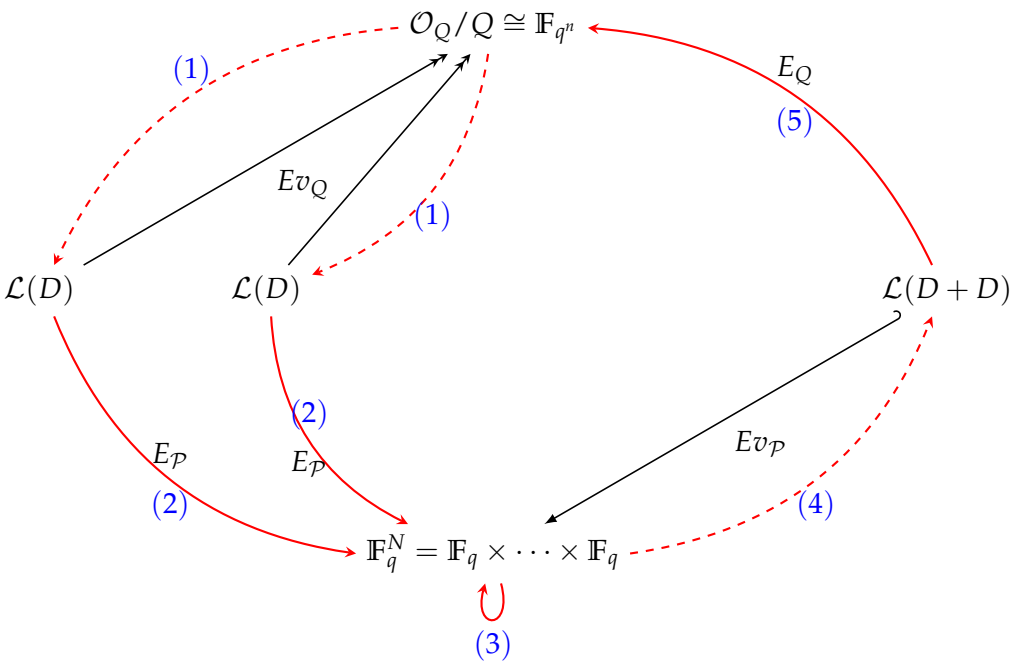

FIGURE 3.1: Diagram of construction of the original Chudnovsky-Chudnovsky multiplication

Definition 3.1.1. *The subroutine from Step (1) to Step (3) in the above diagram is called forward-phase of CCMA. We call return-phase of CCMA the subroutine from Step (4) to Step (5).*

3.2 Construction of algorithm

In order to make the study and the construction of this algorithm easier, we proceed in the following way.

We choose a place Q of degree n and a divisor D of degree $n+g-1$, such that Ev_Q and $Ev_\mathcal{P}$ are isomorphisms. In this aim, S. Ballet in [Bal99] introduced simple numerical conditions on algebraic curves of an arbitrary genus g giving a sufficient condition for the application of CCMA (existence of places of certain degree, of non-special divisors of degree $g-1$) generalizing the result of A. Shokrollahi [Sho92] for the elliptic curves. The following theorem that makes effective the original Chudnovsky-Chudnovsky algorithm groups some results in [Bal99].

Theorem 3.2.1. *Let q be a prime power and let n be an integer > 1. If there exists an algebraic function field $F/\mathbb{F}_q$ of genus g satisfying the conditions:*

1. $N_n > 0$ *(which is always the case if* $2g+1 \leq q^{\frac{n-1}{2}}(q^{\frac{1}{2}}-1)$*),*
2. $N_1(F/\mathbb{F}_q) > 2n+2g-2$,

then there exist a divisor D of degree $n+g-1$ and a place Q of degree n which is not in the support of D such that:

(i) the evaluation map

$$\begin{array}{rccc} Ev_Q: & \mathcal{L}(D) & \to & \mathcal{O}_Q/\langle Q\rangle \\ & f & \mapsto & f(Q) \end{array}$$

is an isomorphism of vector spaces over $\mathbb{F}_q$,

(ii) there exist $2n+g-1$ *places of degree one* $P_1,\ldots,P_{2n+g-1}$ *which are not in the support of* D *such that the evaluation map*

$$\begin{array}{rccc} Ev_{\mathcal{P}}: & \mathcal{L}(2D) & \to & \mathbb{F}_q^{2n+g-1} \\ & f & \mapsto & \left(f(P_1),\ldots,f\left(P_{2n+g-1}\right)\right) \end{array}$$

is an isomorphism of vector spaces over $\mathbb{F}_q$.

Remark 3.2.1. *Note that the divisor* D *is not necessarily effective. It is important to add the property of effectivity for the divisor* D *in a perspective of implementation. Indeed, it is easier to construct the algorithm CCMA with this assumption because in this case* $\mathcal{L}(D) \subseteq \mathcal{L}(2D)$ *and we can directly apply the evaluation map* $Ev_{\mathcal{P}}$ *instead of* $E_{\mathcal{P}}$ *in (3.1) of Theorem 3.1.1, by means of a suitable representation of* $\mathcal{L}(2D)$.

Moreover, in this case, we need to consider simultaneously the assumption that the support of the divisor D *does not contain the rational places and the place* Q *of degree* n *and the assumption of effectivity of the divisor* D. *Indeed, it is known that the support moving technique (cf. [Pie12, Lemma 1.1.4.11]), which is a direct consequence of Strong Approximation Theorem (cf. [Sti93, Proof of Theorem I.6.4]), applied on an effective divisor generates the loss of effectivity of the initial divisor (cf. also [Ati+17, Remark 2.2]). So, let us suppose these two assumptions.*

Remark 3.2.2. *As in [Bal02], in practice, we take as a divisor* D *one place of degree* $n+g-1$. *It has the advantage to solve the problem of the support of divisor* D *(cf. also [Ati+17, Remark 2.2]) as well as the problem of the effectivity of the divisor* D. *However, it is not required to be considered in the theoretical study, but, as we will see, it will have some importance in the strategy of implementation.*

We consider the basis $\mathcal{B}_Q$ of the residue class field F_Q over $\mathbb{F}_q$ as the image of a basis of $\mathcal{L}(D)$ by Ev_Q or equivalently (which is sometimes useful following the considered situation) the basis of $\mathcal{L}(D)$ as the reciprocal image of a basis of the residue class field F_Q over $\mathbb{F}_q$ by Ev_Q^{-1}. Let

$$\mathcal{B}_D := (f_1,\ldots,f_n) \tag{3.3}$$

be a basis of $\mathcal{L}(D)$ and let us denote the basis of the supplementary space of $\mathcal{L}(D)$ in $\mathcal{L}(2D)$ by

$$\mathcal{B}_D^c := (f_{n+1},\ldots,f_N) \tag{3.4}$$

where $N := dim\mathcal{L}(2D) = 2n+g-1$. Then, we choose

$$\mathcal{B}_{2D} := \mathcal{B}_D \cup \mathcal{B}_D^c \tag{3.5}$$

as the basis of $\mathcal{L}(2D)$.

We denote by T_{2D} the matrix of the isomorphism $Ev_{\mathcal{P}} : \mathcal{L}(2D) \to \mathbb{F}_q^N$ in the basis $\mathcal{B}_{2D}$ of $\mathcal{L}(2D)$ (the basis of $\mathbb{F}_q^N$ will always be the canonical basis). Then, we denote by T_D the matrix of the first n columns of the matrix T_{2D}. Therefore, T_D is the matrix of the restriction of the evaluation map $Ev_{\mathcal{P}}$ on the Riemann-Roch vector space $\mathcal{L}(D)$, which is an injective morphism.

Note that the canonical surjection E_Q is the extension of the isomorphism Ev_Q, since Q not belong to $supp(D)$, we have $\mathcal{L}(D) \subseteq \mathcal{O}_Q$. Moreover, as $supp(2D) = supp(D)$, we also have $\mathcal{L}(2D) \subseteq \mathcal{O}_Q$. We can therefore consider the images of elements of the basis $\mathcal{B}_{2D}$ by E_Q and obtain a system of N linear equations as follows:

$$E_Q(f_r) = \sum_{m=1}^{n} c_r^m Ev_Q(f_m), \quad r = 1, ..., N$$

where E_Q denotes the canonical projection from the valuation ring $\mathcal{O}_Q$ of the place Q in its residue class field F_Q, Ev_Q is the restriction of E_Q over the vector space $\mathcal{L}(D)$ and $c_r^m \in \mathbb{F}_q$ for $r = 1, ..., N$.

Let C be the matrix of the restriction of the map E_Q on the Riemann-Roch vector space $\mathcal{L}(2D)$, from the basis $\mathcal{B}_{2D}$ in the basis $\mathcal{B}_Q$:

$$C = \begin{pmatrix} c_1^1 & \cdots & c_N^1 \\ c_1^2 & \cdots & c_N^2 \\ \vdots & \ddots & \vdots \\ c_1^n & \cdots & c_N^n \end{pmatrix} \tag{3.6}$$

We obtain the product $z := xy$ of two elements $x, y \in \mathbb{F}_{q^n}$ by the formula (3.1) in Theorem 3.1.1, where the notation M^t denotes the transpose of some matrix M:

Algorithm 10: Original Chudnovsky-Chudnovsky multiplication algorithm in $\mathbb{F}_{q^n}$

Input: $x = \sum_{i=1}^{n} x_i Ev_Q(f_i)$, and $y = \sum_{i=1}^{n} y_i Ev_Q(f_i)$ $\quad //x_i, y_i \in \mathbb{F}_q$

Output: $z = xy = \sum_{i=1}^{n} z_i Ev_Q(f_i)$

1 $$X := \begin{pmatrix} X_1 \\ \vdots \\ X_N \end{pmatrix} \leftarrow T_D \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} \text{ and } Y := \begin{pmatrix} Y_1 \\ \vdots \\ Y_N \end{pmatrix} \leftarrow T_D \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix}.$$

2 $(Z_1, ..., Z_N)^t \leftarrow (X_1Y_1, \ldots, X_NY_N)^t =: X \odot Y$

3 $$(z_1, \ldots, z_n)^t \leftarrow CT_{2D}^{-1} \begin{pmatrix} Z_1 \\ \vdots \\ Z_N \end{pmatrix}$$

We see that the bilinear multiplications appear only in the second step of the algorithm. Thus we conclude that

$$\mu_{q,sym}^b(n) \leq N.$$

There are several studies focused on the qualitative improvement of this algorithm (for example [Arn06; BR04; CÖ10; Ran12]) as well as the improvements of upper bound (for example [BP11; BR05]) and asymptotic upper bound (for example [Cas+12; STV92]) of the symmetric bilinear complexity. A few works have been done on the efficient construction of the algorithms of type Chudnovky.

3.2.1 An efficient construction using an elliptic curve

First efficient construction of a bilinear algorithm of multiplication which implements CCMA through interpolation on algebraic curves was developed by Shokrollahi and Baum [BS91]. It concerns a multiplication algorithm in the finite field $\mathbb{F}_{256}$ over $\mathbb{F}_4$, namely $q = 4$ and $n = 4$, using the maximal Fermat elliptic curve $x^3 + y^3 = 1$. The *bilinear complexity* $\mu^b_{sym}(U)$ of this symmetric algorithm U is *optimal* and such that

$$\mu^b_{sym}(U) = \mu^b_{q,sym}(n) = 2n = 8.$$

Experiment of Baum-Shokrollahi. The article [BS91] presents the original Chudnovsky - Chudnovsky multiplication in $\mathbb{F}_{4^4}$, for the case $q = 4$ and $n = 4$. The elements of $\mathbb{F}_4$ are denoted by $0, 1, \omega$ and ω^2 where ω is a primitive root of the irreducible polynomial $X^2 + X + 1$ in $\mathbb{F}_2[X]$.

In Theorem 3.2.1, the first necessary condition ($N_n > 0$) for CCMA in $\mathbb{F}_{q^n}$ is easily checked by the inequality $2g + 1 \leq q^{\frac{n-1}{2}}(q^{\frac{1}{2}} - 1)$. For the second one $\big(N_1(F/\mathbb{F}_q) > 2n + 2g - 2\big)$, we have known a result of Chaumine [Cha06] that if $3 \leq n \leq \frac{1}{2}(q + 1 + 2\lfloor q^{\frac{1}{2}} \rfloor)$, then there exists an elliptic curve over $\mathbb{F}_q$ with at least $2n$ $\mathbb{F}_q$-rational points for using CCMA in $\mathbb{F}_{q^n}$. Consequently, in case $\mathbb{F}_{4^4}$, the sufficient condition for the elliptic CCMA is $N_1(F/\mathbb{F}_q) \geq 8$. Moreover, Theorem 2.3.2 (Hasse-Weil bound) shows that $N_1(F/\mathbb{F}_q)$ is always upper bounded by $q + 1 + 2g\sqrt{q} = 9$. In their article, Baum-Shokrollahi chose the well known Fermat curve ($u^3 + v^3 = 1$) and we see that this sufficient condition is satisfied.

By the substitutions $x = 1/(u + v)$ and $y = u/(u + v)$, we get the isomorphic curve $y^2 + y = x^3 + 1$. From now on, $F/\mathbb{F}_q$ denotes the algebraic function field associated to the elliptic curve $\mathcal{C}$ with plane model $y^2 + y = x^3 + 1$, of genus one. The projective coordinates $(x : y : z)$ of $\mathbb{F}_4$-rational points of this elliptic curve are:

$$P_\infty = (0 : 1 : 0), P_1 = (0 : \omega : 1), P_2 = (0 : \omega^2 : 1), P_3 = (1 : 0 : 1),$$
$$P_4 = (1 : 1 : 1), P_5 = (\omega : 0 : 1), P_6 = (\omega : 1 : 1), P_7 = (\omega^2 : 0 : 1), P_8 = (\omega^2 : 1 : 1).$$

Now, we represent $\mathbb{F}_{256}$ as $\mathbb{F}_4[x]/\langle Q(x)\rangle$ with primitive root b, where $Q(x) = x^4 + x^3 + \omega x^2 + \omega x + \omega$.

- For the place Q of degree 4, the authors considered $Q = \sum_{i=1}^{4} \mathfrak{p}_i$ where $\mathfrak{p}_1$ corresponds to the $\mathbb{F}_{4^4}$-rational point with projective coordinates $(b^{16} : b^{174} : 1)$ and $\mathfrak{p}_2, \mathfrak{p}_3, \mathfrak{p}_4$ are its conjugates under the Frobenius map. Since b is a primitive root of $Q(x)$, under the action of the Frobenius map, then b^{16} is also a root of $Q(x)$. Thus, the place Q is a place lying over the place $(Q(x))$ of $\mathbb{F}_4(x)/\mathbb{F}_4$. Note also that the place $((Q(x))$ of $\mathbb{F}_4(x)/\mathbb{F}_4$ is totally splitted in the algebraic function field $F/\mathbb{F}_4$, which means that there exist two places of degree n in $F/\mathbb{F}_4$ lying over the place $(Q(x))$ of $\mathbb{F}_4(x)/\mathbb{F}_4$, since the function field $F/\mathbb{F}_4$ is an extension of degree 2 of the rational function field $\mathbb{F}_4(x)/\mathbb{F}_4$. The place Q is one of the two places in $F/\mathbb{F}_4$ lying over the place $(Q(x))$. Notice that the second place is given by the orbit of the conjugated point $(b^{16} : b^{174} + 1 : 1)$. Therefore, we can represent $\mathbb{F}_{256} = \mathbb{F}_{4^4} = \mathbb{F}_4[x]/\langle Q(x)\rangle$ as the residue class field F_Q of the place Q in $F/\mathbb{F}_4$.

- For the divisor D, authors choose the place described as $\sum_{i=1}^{4} \mathfrak{d}_i$ where $\mathfrak{d}_1$ corresponds to the $\mathbb{F}_{4^4}$-rational point $(b^{17} : b^{14} : 1)$ and $\mathfrak{d}_2, \mathfrak{d}_3, \mathfrak{d}_4$ are its conjugates under the Frobenius map. By computation we see that b^{17} is a root of irreducible polynomial $\mathcal{D}(x) = x^2 + x + \omega$ and $\deg D = 4$ because $\mathfrak{d}_1$, $\mathfrak{d}_2, \mathfrak{d}_3, \mathfrak{d}_4$ are all distinct. Therefore, D is the only place in $F/\mathbb{F}_4$ lying over the place $(\mathcal{D}(x))$ of $\mathbb{F}_4(x)$ since

the residue class field F_D of the place D (in F) is a quadratic extension of the residue class field $F_{\mathcal{D}}$ of the place $\mathcal{D}$ (in $\mathbb{F}_q(x)$), which is an inert place of $\mathbb{F}_4(x)$ in $F/\mathbb{F}_4$.

By using a Magma program, we can check the choices of D and Q satisfying that $D - Q$ is not a principal divisor, so $D - Q$ is not a special divisor.

Using the algorithm in [Hes02] implemented in Magma program, we can determine a basis $\mathcal{B}_{2D} = \{f_1, ..., f_8\}$ of Riemann-Roch space $L(2D)$:

$$\begin{aligned} f_1(x,y) &= \frac{1}{x^2+x+\omega}, \quad f_2(x,y) = \frac{x}{x^2+x+\omega}, \\ f_3(x,y) &= \frac{y}{x^2+x+\omega}, \quad f_4(x,y) = \frac{x^2}{x^2+x+\omega}, \\ f_5(x,y) &= \frac{1}{x^4+x^2+\omega^2}, \quad f_6(x,y) = \frac{xy}{x^4+x^2+\omega^2}, \\ f_7(x,y) &= \frac{y}{x^4+x^2+\omega^2}, \quad f_8(x,y) = \frac{x}{x^4+x^2+\omega^2}. \end{aligned}$$

The matrix T_{2D} of the second evaluation map $Ev_{\mathcal{P}}$, where $\mathcal{P} := \{P_\infty, P_1, P_2, P_7, P_8, P_3, P_4, P_5\}$, is

$$T_{2D} = \begin{pmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ \omega^2 & 0 & 1 & 0 & \omega & 0 & \omega^2 & 0 \\ \omega^2 & 0 & \omega & 0 & \omega & 0 & 1 & 0 \\ \omega^2 & \omega^2 & 0 & \omega^2 & \omega & 0 & 0 & \omega \\ \omega^2 & \omega^2 & \omega^2 & \omega^2 & \omega & \omega & \omega & \omega \\ \omega & \omega^2 & 0 & 1 & \omega^2 & 0 & 0 & 1 \\ \omega & \omega^2 & \omega & 1 & \omega^2 & 1 & \omega^2 & 1 \\ \omega & 1 & 0 & \omega^2 & \omega^2 & 0 & 0 & \omega \end{pmatrix}.$$

We have the matrix

$$T_D = \begin{pmatrix} 0 & 0 & 0 & 1 \\ \omega^2 & 0 & 1 & 0 \\ \omega^2 & 0 & \omega & 0 \\ \omega^2 & \omega^2 & 0 & \omega^2 \\ \omega^2 & \omega^2 & \omega^2 & \omega^2 \\ \omega & \omega^2 & 0 & 1 \\ \omega & \omega^2 & \omega & 1 \\ \omega & 1 & 0 & \omega^2 \end{pmatrix}. \tag{3.7}$$

Then, the computation gives:

$$C = \begin{pmatrix} 1 & 0 & 0 & 0 & \omega & 0 & \omega^2 & \omega \\ 0 & 1 & 0 & 0 & 0 & \omega^2 & \omega & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & \omega & 0 & \omega \end{pmatrix},$$

and

$$CT_{2D}^{-1} = \begin{pmatrix} 1 & \omega & 1 & \omega & 1 & 1 & \omega & 0 \\ 1 & 0 & \omega^2 & \omega & 1 & \omega^2 & 1 & \omega \\ 1 & \omega & \omega & \omega^2 & 1 & \omega^2 & \omega & \omega \\ 0 & \omega & \omega^2 & \omega & 1 & \omega^2 & 0 & 0 \end{pmatrix}. \tag{3.8}$$

An application of CCMA in $\mathbb{F}_{256}/\mathbb{F}_4$. We give a concrete example of the multiplication in $\mathbb{F}_{256}/\mathbb{F}_4$ whose basis is determined by:

$$\mathcal{B}_Q := (Ev_Q(f_1), ..., Ev_Q(f_8)) = (b^{147}, b^{148}, b^{126}, b^{149})$$

where b is primitive root of the irreducible polynomial $Q(x) = x^4 + x^3 + \omega x^2 + \omega x + \omega$ in $\mathbb{F}_4[x]$.

In this basis, let us consider two elements in $\mathbb{F}_{4^4}$ over $\mathbb{F}_4$:

$$x = b^{244} = (\omega, 1, 0, \omega^2)_{\mathcal{B}_Q} = \omega \cdot b^{147} + 1 \cdot b^{148} + 0 \cdot b^{126} + \omega^2 \cdot b^{149}$$

and

$$y = b^{72} = (1, \omega^2, \omega, \omega^2)_{\mathcal{B}_Q} = 1 \cdot b^{147} + \omega^2 \cdot b^{148} + \omega \cdot b^{126} + \omega^2 \cdot b^{149}.$$

We have the computations at each step in Algorithm 10,
-Step 1:

$$\bar{x} := T_D \cdot \begin{pmatrix} \omega \\ 1 \\ 0 \\ \omega^2 \end{pmatrix} = \begin{pmatrix} \omega^2 \\ 1 \\ 1 \\ 0 \\ 0 \\ \omega^2 \\ \omega^2 \\ 0 \end{pmatrix} \text{ and } \bar{y} := T_D \cdot \begin{pmatrix} 1 \\ \omega^2 \\ \omega \\ \omega^2 \end{pmatrix} = \begin{pmatrix} \omega^2 \\ 1 \\ 0 \\ \omega^2 \\ \omega \\ \omega^2 \\ 0 \\ \omega^2 \end{pmatrix}.$$

- Step 2:

$$u := \bar{x} \odot \bar{y} = \begin{pmatrix} \omega \\ 1 \\ 0 \\ 0 \\ 0 \\ \omega \\ 0 \\ 0 \end{pmatrix}.$$

- Step 3:

$$z := CT_{2D}^{-1} \cdot u = \begin{pmatrix} \omega \\ \omega^2 \\ 1 \\ \omega^2 \end{pmatrix}.$$

Hence, we obtain the product of x and y:

$$z = (\omega, \omega^2, 1, \omega^2)_{\mathcal{B}_Q} = \omega \cdot b^{147} + \omega^2 \cdot b^{148} + 1 \cdot b^{126} + \omega^2 \cdot b^{149} = b^{61}.$$

In conclusion, we have the correct result of the multiplication of two elements x, y in $\mathbb{F}_{256}/\mathbb{F}_4$:

$$x \cdot y = b^{244} \cdot b^{72} = b^{61} = z.$$

Remark 3.2.3. *Baum and Shokrollahi [BS91] had constructed the orginal Chudnovsky algorithm in $\mathbb{F}_{256}/\mathbb{F}_4$ with the basis $\mathcal{B}_{2D} := \{f_1, f_2, ..., f_8\}$ of $\mathcal{L}(2D)$, and the parameter matrices $T_D, T_{2D}, C^t T_{2D}^{-1}$ of the algorithm were computed as above. Then, the basis $\mathcal{B}_Q$ of*

residue class field F_Q is

$$\mathcal{B}_Q = (Ev_Q(f_1), ..., Ev_Q(f_8)) = (b^{147}, b^{148}, b^{126}, b^{149}).$$

If we want to use the canonical basis $\mathcal{B}_Q^c := (1, b, b^2, b^3)$ of $F_Q \cong \mathbb{F}_{4^4}/\mathbb{F}_4$ where b is the primitive root of the irreducible polyomial $Q(x) = x^4 + x^3 + \omega x^2 + \omega x + \omega$ in $\mathbb{F}_4[x]$, then we have to compute a new basis $\mathcal{B}'_D$ of $\mathcal{L}(D)$ as follows:

$$\mathcal{B}'_D := \left(Ev_Q^{-1}(1), Ev_Q^{-1}(b), Ev_Q^{-1}(b^2), Ev_Q^{-1}(b^3)\right) = (g_1, g_2, g_3, g_4),$$

where

$$\begin{aligned} g_1 &= 1, \\ g_2 &= \frac{\omega^2 y}{x^2 + x + \omega} + \omega, \\ g_3 &= \frac{\omega x + \omega}{x^2 + x + \omega}, \\ g_4 &= \frac{\omega x^2 + \omega x}{x^2 + x + \omega}. \end{aligned}$$

Moreover, by using a Magma program, we can compute the basis $\mathcal{B}_D^c := (g_5, g_6, g_7, g_8)$ of the complementary space of $\mathcal{L}(D)$ in $\mathcal{L}(2D)$:

$$\begin{aligned} g_5 &= \frac{x^3}{x^4 + x^2 + \omega^2}, \\ g_6 &= \frac{x^4}{x^4 + x^2 + \omega^2}, \\ g_7 &= \frac{xy}{x^4 + x^2 + \omega^2}, \\ g_8 &= \frac{x^2 y}{x^4 + x^2 + \omega^2}. \end{aligned}$$

Therefore, the basis $\mathcal{B}'_{2D}$ of $\mathcal{L}(2D)$ which is appropriate to $\mathcal{B}'_Q$ is

$$\mathcal{B}'_{2D} = (g_1, g_2, ..., g_8).$$

3.3 Kernel-type construction of CCMA

In [Ati+17], the authors proposed an effective method of construction of the symmetric CCMA from Theorem 3.2.1. We can name it as *kernel-type* construction, for which we use the kernel M of the restriction map of E_Q over $\mathcal{L}(2D)$ where E_Q is the canonical projection from the evaluation ring of the place Q onto its residue class field F_Q. Then the kernel M is represented as a complementary subspace of $\mathcal{L}(D)$ in $\mathcal{L}(2D)$. This construction is based on the choice of the place Q of degree n, the divisor D of degree $n + g - 1$, the bases of spaces $\mathcal{L}(D)$ and $\mathcal{L}(2D)$ and the basis of the residue class field F_Q of the place Q.

3.3.1 Finding places D and Q

In order to build the good places, we draw them at random and we check that they satisfy some conditions as follows:

(1) We choose at random an irreducible polynomial $Q(x)$ of degree n in $\mathbb{F}_q[x]$ and check that this polynomial is:

   (a) *Primitive*

   (b) *Totally decomposed* in the algebraic function field $F/\mathbb{F}_q$ (which implies that there exists a place Q of degree n above the polynomial $Q(x)$.

   Then, we choose a place Q of degree n among the n places lying above the polynomial $Q(x)$

(2) We draw at random a place D of degree $n+g-1$ and check that $D-Q$ is a *non-special* divisor of degree $g-1$, i.e. $\dim \mathcal{L}(D-Q) = 0$.

Remark 3.3.1. *Recall that as in Remark 3.2.2, we take a divisor D to be a place of degree $n+g-1$ since it has the advantage to solve the problem of the support of divisor D (condition of (i) of Theorem 3.2.1) as well as the problem of the effectivity of the divisor D.*

A sufficient condition for the existence of at least one place of degree n is given by the following inequality in Lemma 2.3.1:

$$2g(F) + 1 \leq q^{\frac{n-1}{2}}\left(q^{\frac{1}{2}} - 1\right).$$

We are sure of the existence of a non-special divisor of degree $g-1$ when $q \geq 4$ and $g \geq 2$ by [BL06, Theorem 11]. In practice, it is easy to find Q and D satisfying these properties in the case of algebraic function fields having many rational places since there exist many such places, in particular such that $D-Q$ is non-special. Such divisors are easily found by experimental results.

3.3.2 Choice of bases of spaces

As the residue class field, F_Q of the place Q is isomorphic to the finite field $\mathbb{F}_{q^n}$, we can identify $\mathbb{F}_{q^n}$ to F_Q. The place Q of degree n lies above an irreducible polynomial $Q(x)$ in $\mathbb{F}_q[x]$, we choose a basis of $\mathbb{F}_{q^n}$ as the basis $\mathcal{B}_Q$ of the residue class field F_Q.

We can normally take into account the *standard polynomial basis*

$$\mathcal{B}_Q^c := \{1, b, b^2, ..., b^{n-1}\}$$

be the basis of $\mathbb{F}_{q^n}$, where b be a primitive root of the irreducible polynomial $Q(x)$. Moreover, one can use the *normal basis*

$$\mathcal{B}_Q^n := \{\beta, \beta^q, \beta^{q^2}, ..., \beta^{q^{n-1}}\}$$

as a basis of $\mathbb{F}_{q^n}$, where β be a root (so-called *normal element*) of a certain normal polynomial $Q(x)$ (see Section 1.2.1 in Chapter 1). In fact, authors in [Ati+17] chose the normal basis $\mathcal{B}_Q^n$ to construct CCMA of kernel-type because they were also interested in exponentiation in finite fields.

Notice that the choice of places D and Q chosen as above are such that the map Ev_Q of Theorem 3.2.1 is an isomorphism. Therefore, there are two possibilities for choosing the basis $\mathcal{B}_D$ of $\mathcal{L}(D)$ and the basis $\mathcal{B}_Q$ of F_Q as follows:

(i) fixing $\mathcal{B}_D = \{f_1, ..., f_n\}$ and then choose

$$\mathcal{B}_Q = Ev_Q(\mathcal{B}_D) = \big(Ev_Q(f_1), ..., Ev_Q(f_n)\big); \tag{3.9}$$

(ii) fixing $\mathcal{B}_Q = \{\phi_1, ..., \phi_n\}$ and choose

$$\mathcal{B}_D = Ev_Q^{-1}(\mathcal{B}_Q) = \big(Ev_Q^{-1}(\phi_1), ..., Ev_Q^{-1}(\phi_n)\big). \tag{3.10}$$

Let P be the map from $\mathcal{L}(2D)$ to $\mathcal{L}(2D)$ defined as follows:

$$\begin{array}{rcl} P : \mathcal{L}(2D) & \to & \mathcal{L}(2D) \\ f & \mapsto & I \circ Ev_Q^{-1}\big(f(Q)\big) \end{array}$$

where I is the embedding map from $\mathcal{L}(D)$ into $\mathcal{L}(2D)$. Then P is a linear map from $\mathcal{L}(2D)$ into $\mathcal{L}(2D)$ whose image is $\mathcal{L}(D)$. More precisely, P is a projection from $\mathcal{L}(2D)$ onto $\mathcal{L}(D)$. Let M be the kernel of P. We note that the divisor D is an effective divisor so that we have $\mathcal{L}(D) \subset \mathcal{L}(2D)$. Then

$$\mathcal{L}(2D) = \mathcal{L}(D) \oplus M.$$

We remark that

$$M = \{f \in \mathcal{L}(2D) | f(Q) = 0\},$$

and

$$f(Q) = P(f)(Q) \text{ for any } f \in \mathcal{L}(2D).$$

As $\deg 2D > 2g - 2$ and by Riemann-Roch theorem, we have the divisor $2D$ is non-special and

$$\dim \mathcal{L}(2D) = 2n + g - 1.$$

Therefore, the basis $\mathcal{B}_{2D}$ of Riemann-Roch space $\mathcal{L}(2D)$ is written as follows:

$$\mathcal{B}_{2D} = (f_1, ..., f_n, f_{n+1}, ..., f_{2n+g-1})$$

where $\mathcal{B}_D = (f_1, ..., f_n)$ is the basis of $\mathcal{B}_D$ and $\mathcal{B}_M = (f_{n+1}, ..., f_{2n+g-1})$ is a basis of M.

If

$$x = (x_1, ..., x_n, x_{n+1}, ..., x_{2n+g-1}) \in \mathcal{L}(2D)$$

then

$$P(x) = (x_1, ..., x_n, 0, ..., 0).$$

We now present the setup algorithm for the kernel-type construction of CCMA that is only done once.

Algorithm 11: Setup algorithm

Input: $F/\mathbb{F}_q, Q, D, \{P_1, ..., P_{2n+g-1}\}$.

Output: $\mathcal{B}_{2D}$, T_{2D} and T_{2D}^{-1}.

1. Check the function field $F/\mathbb{F}_q$, the place Q, the divisors D and the points $P_1, ..., P_{2n+g-1}$ are as in Theorem 3.2.1.
2. Represent $\mathbb{F}_{q^n}$ as the residue class field of the place Q.
3. Construct a basis $(f_1, ..., f_n, f_{n+1}, ..., f_{2n+g-1})$ of $\mathcal{L}(2D)$ where $(f_1, ..., f_n)$ is the basis of $\mathcal{L}(D)$ defined as in (3.10) and $(f_{n+1}, ..., f_{2n+g-1})$ a basis of M.
4. Compute the matrices T_{2D} and T_{2D}^{-1}.

We use the representation basis of spaces $F_Q, \mathcal{L}(D), \mathcal{L}(2D)$ performed above to compute the product of two elements in $\mathbb{F}_{q^n}$ by the kernel-type CCMA.

Let $x = (x_1, ..., x_n)$ and $y = (y_1, ..., y_n)$ be two elements of $\mathbb{F}_{q^n}$ given by their components over $\mathbb{F}_q$ relative to the chosen basis $\mathcal{B}_Q$. Then, we can consider that x and y are identified to the following elements of $\mathcal{L}(D)$:

$$f_x = \sum_{i=1}^{n} x_i f_i \quad and \quad f_y = \sum_{i=1}^{n} y_i f_i.$$

The product of the two elements f_x and f_y of $\mathcal{L}(D)$ is computed in the valuation ring $\mathcal{O}_Q$. This product lies in $\mathcal{L}(2D)$. We will consider that x and y are respectively the elements f_x and f_y of $\mathcal{L}(2D)$ where the $n + g - 1$ last components are 0. It is clear that knowing x or f_x by their coordinates is the same thing.

Hence, the product of x by y is such that

$$f_{xy} = P\Big(T_{2D}^{-1}\big(T_{2D}(f_x) \odot T_{2D}(f_y)\big)\Big). \tag{3.11}$$

We note that in (3.11), all the bilinear multiplications which are made to compute the product of two elements exactly correspond to the $2n + g - 1$ multiplications in $\mathbb{F}_q$ of the Hadamard product $T_{2D}(f_x) \odot T_{2D}(f_y)$. The operations performed in the evaluation of the map $Ev_{\mathcal{P}}$ (with respect to the representation matrix T_{2D}) on the element f and in the evaluation of map $P \circ Ev_{\mathcal{P}}^{-1}$ (with respect to the representation matrix $P \cdot T_{2D}^{-1}$) on any element of $\mathbb{F}_q^{2n+g-1}$ only involve *additions* and *scalar multiplications*.

We have the multiplication algorithm as follows:

Algorithm 12: Chudnovsky-type multiplication algorithm

Input: $x = (x_1, ..., x_n)$ and $y = (y_1, ..., y_n)$.
Output: xy.

1. Compute

$$\begin{pmatrix} a_1 \\ \vdots \\ a_{2n+g-1} \end{pmatrix} = T_{2D} \cdot \begin{pmatrix} x_1 \\ \vdots \\ x_n \\ 0 \\ \vdots \\ 0 \end{pmatrix} \quad \text{and} \quad \begin{pmatrix} b_1 \\ \vdots \\ b_{2n+g-1} \end{pmatrix} = T_{2D} \cdot \begin{pmatrix} y_1 \\ \vdots \\ y_n \\ 0 \\ \vdots \\ 0 \end{pmatrix}.$$

2. Compute $u = (u_1 \cdots u_{2n+g-1})^t$ where $u_i = a_i \cdot b_i$ for $i = 1, 2, ..., 2n + g - 1$.
3. Compute $v = (v_1 \cdots v_{2n+g-1})^t = T_{2D}^{-1} \cdot u$
4. Return $xy = (v_1, ..., v_n)$.

Now, we consider again the example of the multiplication in $\mathbb{F}_{4^4}/\mathbb{F}_4$ as in Section 3.2.1 of Chapter 3 for the kernel-type construction of Algorithm 12.

3.3.3 Example of the kernel-type construction of CCMA

Set $q = 4$ and $n = 4$. We present $\mathbb{F}_4$ as the field $\mathbb{F}_2(\omega) = \mathbb{F}_2[x]/(p(x))$, where $p(x)$ is the irreducible polynomial $p(x) = x^2 + x + 1$ and ω denotes a primitive root of $p(x)$. We denote $F/\mathbb{F}_q$ the algebraic function field associated to the elliptic curve $\mathcal{C}$ with plane model $y^2 + y = x^3 + 1$, of genus one. This curve has 9 rational points, which is maximal over F_q according to the Hasse-Weil bound. Let us give the projective coordinates $(x : y : z)$ of the rational points of the curve $\mathcal{C}$ as follows:

$$P_\infty = (0:1:0), P_1 = (0:\omega:1), P_2 = (0:\omega^2:1), P_3 = (1:0:1),$$
$$P_4 = (1:1:1), P_5 = (\omega:0:1), P_6 = (\omega:1:1), P_7 = (\omega^2:0:1), P_8 = (\omega^2:1:1).$$

Construction of the required places and divisors:

- *A place Q of degree n*: we take Q as in Section 3.2.1. In particular, we consider Q is a place in $F/\mathbb{F}_q$ lying over the place $(\mathcal{Q}(x))$ of $\mathbb{F}_q(x)/\mathbb{F}_q$, namely the orbit of the $\mathbb{F}_{4^4}$-rational point $(b^{16} : b^{174} : 1)$ where b is a primitive root of the following irreducible polynomial:

$$\mathcal{Q}(x) = x^4 + x^3 + \omega x^2 + \omega x + \omega.$$

- *A divisor D of degree $n + g - 1$*: similarly, we choose D be the place of degree 4 according to the method used for the place Q. In particular, we consider the orbit of the $\mathbb{F}_{4^4}$-rational point $(b^{17} : b^{14} : 1)$ where b^{17} is a root of the following normal irreducible polynomial:

$$\mathcal{D}(x) = x^2 + x + \omega.$$

The place Q and the divisor D satisfy the good property: the dimension of the divisor $D - Q$ is zero which means that the divisor $D - Q$ is non-special of degree $g - 1$.

- *The basis of the residue class field* F_Q: we choose a basis of the residue class field F_Q as the standard basis $\mathcal{B}_Q^c$ associated to the place Q

$$\mathcal{B}_Q^c = \{1, b, b^2, b^3\},$$

where b is a primitive root of the irreducible polynomial $Q(x) = x^4 + x^3 + \omega x^2 + \omega x + \omega$.

- *The basis of* $\mathcal{L}(D)$: we choose a basis of the Riemann-Roch space $\mathcal{L}(D)$ the basis $\mathcal{B}_D = (f_1, ..., f_n)$ such that $Ev_Q(\mathcal{B}_D) = \mathcal{B}_Q^n$. By computing in Magma program, we can give the elements of $\mathcal{B}_D$:

$$\begin{aligned} f_1(x,y) &= 1, \\ f_2(x,y) &= \frac{\omega^2 y}{x^2 + x + \omega} + \omega, \\ f_3(x,y) &= \frac{\omega x + \omega}{x^2 + x + \omega}, \\ f_4(x,y) &= \frac{\omega x^2 + \omega x}{x^2 + x + \omega}. \end{aligned}$$

We can check that the evaluations of this basis $\mathcal{B}_D$ at the place Q choosen above is

$$(f_1(Q), f_2(Q), f_3(Q), f_4(Q)) = (1, b, b^2, b^3).$$

- *The basis of* $\mathcal{L}(2D)$: let $\mathcal{B}_{2D} = (g_1, ..., g_{2n+g-1})$ be a basis of $\mathcal{L}(2D)$ as defined in Section 3.3.2. Since the divisor D is effective, we can complete the basis f_i of $\mathcal{L}(D)$ in $\mathcal{L}(2D)$ for $i = 1, ..., n$. Then any element g_i of g is such that $g_i(x,y) = f_i(x,y)$ for $i = 1, ..., n$ and $g_j(x,y) = \frac{g_{j1}y + g_{j2}}{(D(x))^2}$ for $j = n+1, ..., 2n+g-1$. Moreover, we require the completion of $\mathcal{L}(D)$ by the kernel of the map P. Therefore, the set $(g_j)_{j=n+1,...,2n+g-1}$ is a basis of the kernel of the restriction map over $\mathcal{L}(2D)$ of the canonical projection from the evaluation ring of the place Q onto its residue class field F_Q.

By the computation in Magma, we give the elements g_j for $j = n+1, ..., 2n+g-1$:

$$\begin{aligned} g_5(x,y) &= \frac{(\omega x^2 + \omega x)y + (\omega^2 x^4 + \omega x^3 + x^2 + x + \omega)}{x^4 + x^2 + \omega^2}, \\ g_6(x,y) &= \frac{(\omega^2 x^2)y + \omega x^4 + \omega x^3 + x^2 + \omega x}{x^4 + x^2 + \omega^2}, \\ g_7(x,y) &= \frac{(x^2 + \omega^2 x)y + \omega x^4 + \omega x^2}{x^4 + x^2 + \omega^2}, \\ g_8(x,y) &= \frac{(\omega x + \omega)y + \omega x^4}{x^4 + x^2 + \omega^2}. \end{aligned}$$

Then, we compute the matrix T_{2D} of the second evaluation map with respect to the basis $\mathcal{B}_{2D} := \{g_1, ..., g_8\}$ and a set $\mathcal{P} := \{P_\infty, P_1, P_2, P_7, P_8, P_3, P_4, P_5\}$ of the rational places to implement the kernel-type construction given in Algorithm 12 for the multiplication in $\mathbb{F}_{4^4}/\mathbb{F}_4$:

$$T_{2D} = \begin{pmatrix} 1 & \omega & 0 & \omega & \omega^2 & \omega & \omega & \omega \\ 1 & 1 & 1 & 0 & \omega^2 & 0 & 0 & 1 \\ 1 & \omega^2 & 1 & 0 & \omega^2 & 0 & 0 & \omega \\ 1 & \omega & 0 & 0 & 1 & 1 & 0 & \omega^2 \\ 1 & 0 & 0 & 0 & 1 & 0 & \omega^2 & \omega^2 \\ 1 & \omega & \omega & \omega^2 & 0 & \omega^2 & 1 & \omega \\ 1 & \omega^2 & \omega & \omega^2 & 1 & \omega & 0 & 1 \\ 1 & \omega & 1 & \omega^2 & \omega & 0 & 1 & \omega^2 \end{pmatrix}$$

and

$$T_{2D}^{-1} = \begin{pmatrix} \omega^2 & 1 & \omega & \omega^2 & 0 & 1 & \omega^2 & 0 \\ \omega & \omega^2 & \omega^2 & 1 & \omega & 1 & \omega^2 & \omega^2 \\ \omega^2 & 1 & 0 & 0 & 0 & 0 & \omega^2 & 1 \\ 0 & 1 & 0 & \omega & 1 & 0 & 1 & \omega^2 \\ 1 & \omega^2 & \omega & \omega & 1 & 1 & \omega & 0 \\ \omega & \omega^2 & 1 & 0 & 1 & 0 & 1 & 0 \\ \omega & \omega^2 & 0 & 1 & 1 & \omega^2 & 0 & \omega \\ 1 & 0 & 0 & \omega^2 & 1 & \omega^2 & \omega & \omega \end{pmatrix}.$$

Testing the multiplication of two concrete elements in $\mathbb{F}_{4^4}/\mathbb{F}_4$: We consider again $x = b^{244}$ and $y = b^{72}$ in $\mathbb{F}_{4^4}/\mathbb{F}_4$ calculated by Baum-Shokrollahi's method in the end of Section 3.2.1 for testing the multiplication of x and y following the kernel-type construction of CCMA.

We first represent the two elements in the standard basis $\mathcal{B}_Q^c = \{1, b, b^2, b^3\}$ of $\mathbb{F}_{4^4}/\mathbb{F}_4 \cong F_Q$. In particular,

$$x = b^{244} = (0, 0, 1, \omega)_{\mathcal{B}_Q^c}$$

and

$$y = b^{72} = (\omega, \omega^2, 0, 0)_{\mathcal{B}_Q^c}.$$

We have the computations at each step in Algorithm 12 with the matrices T_{2D} and T_{2D}^{-1} determined as above.

Step 1:

$$X := T_{2D} \cdot (0, 0, 1, \omega, 0, 0, 0, 0)^t = (\omega^2, 1, 1, 0, 0, \omega^2, \omega^2, 0)^t$$

$$Y := T_{2D} \cdot (\omega, \omega^2, 0, 0, 0, 0, 0, 0)^t = (\omega^2, 1, 0, \omega^2, \omega, \omega^2, 0, \omega^2)^t$$

Step 2: $u := X \odot Y = (\omega, 1, 0, 0, 0, \omega, 0, 0)^t$
Step 3: $Z := T_{2D}^{-1} \cdot u = (\omega, \omega, 0, 1, 0, 0, 0, 0)^t$
Step 4: $z := x \cdot y = (\omega, \omega, 0, 1)$.

We note that $z = (\omega, \omega, 0, 1)$ in the standard basis $\mathcal{B}_Q^c$ corresponds to the element b^{61} which is correct as in the result of the multiplication of x and y in last section.

We add a Magma program in Appendix A.1.1 for the reference of above example.

Remark 3.3.2. *In the studies related to CCMA, some authors have developed different construction methods of the algorithm. In particular, S. Ballet in [Bal02] gave the first effective symmetric construction of CCMA using an algebraic curve of genus $g > 1$, e.g. hyperelliptic curve. He considered an explicit example concerning the multiplication algorithm in the finite field $\mathbb{F}_{16^n}$ over $\mathbb{F}_{16}$, more precisely $q = 16$ and $n = 13, 14, 15$ by using the maximal hyperelliptic curve $y^2 + y = x^5$. An example of an effective symmetric construction using higher degree places and derivated evaluations on rational places on the elliptic curve were*

developed by M. Cenk and F. Özbudak in [CÖ10]. It is the first effective construction of a bilinear algorithm of multiplication which implements the combination of the generalizations of CCMA introduced in [BR04] using places of degree one and two and in [Arn06] using derivated evaluations. Note that in this example, the derivated evaluations are only used on rational places of the order one. More precisely, it concerns a multiplication algorithm in the finite field $\mathbb{F}_{3^9}$ over $\mathbb{F}_3$ using the non-optimal elliptic curve $y^2 = x^3 + x + 2$.

Moreover, S. Ballet, N. Baudru, A. Bonnecaze and M. Tukumuli in [Bal+] (announced in [Bal+17]) developed the first effective construction of bilinear algorithms of multiplication which implements the asymmetric generalization of CCMA introduced in [Ran12]. Note that these examples use two distinct Riemann-Roch spaces $\mathcal{L}(D_1)$ and $\mathcal{L}(D_2)$ without derivated evaluations.

Chapter 4

Optimized construction of Chudnovsky-type multiplication algorithm

The objective of this chapter is to build a Chudnovsky-type algorithm with improved scalar complexity. Improving the scalar complexity leads to a better total complexity, and thus allows comparisons with the best-known algorithms. Optimization by "brute force" is very expensive. In fact, it is necessary to vary the bases $\mathcal{B}_D$, $\mathcal{B}_D^c$, $\mathcal{B}_Q$ and $\mathcal{B}_{2D}$, for each divisor D and each place Q, and find the ones that minimize the number of scalar operations. Therefore, to achieve our goal, we need to define a dedicated strategy.

In the original construction of CCMA defined in the previous chapter, the scalar operations are done when the two matrices T_D and CT_{2D}^{-1} (or T_D and $T_{2D,n}^{-1}$ for the kernel-type construction) are multiplied by vectors depending on the elements of $\mathbb{F}_{q^n}$ that must be multiplied. The greater the number of zeros in these matrices, the more the number of scalar multiplications will be reduced. So our strategy is to maximize the number of zeros in these matrices. To achieve this, we choose to fix the divisor D and the place Q and vary the basis of $\mathcal{L}(D)$ and then the basis of the complementary of $\mathcal{L}(D)$ in $\mathcal{L}(2D)$.

In Section 4.1.2, we show that we actually just needs to vary the basis $\mathcal{B}_D$ in order to minimize the number of zeros of the algorithm denoted by N_s. In other words, we seek a basis of $\mathcal{L}(D)$, denoted by $\mathcal{B}_{D,max}$, such that N_s reaches the best value. We prove that for each basis $\mathcal{B}_D$, the change of $\mathcal{B}_D^c$ does not affect the numbers of zeros of the two matrices. To summarize, we just need to change $\mathcal{B}_D$ until obtaining $\mathcal{B}_{D,max}$, while leaving $\mathcal{B}_D^c$ fixed, in order to minimize N_s.

For the first matrix T_D, we obtain an upper-bound on its number of zeros, which does not depend on the choices of the divisor D and the place Q, but only on their degrees. So we compute the set of bases of $\mathcal{L}(D)$ which reach this bound. Among all these bases, we choose one which gives the largest number of zeros in the second matrix. This strategy drastically reduces the complexity of the optimization method compared to the "brute force" optimization.

Furthermore, we also prove that the order of places $P_1, ..., P_N$ of CCMA on which we interpolate does not affect the number of zeros in the matrices.

We use our strategy on the Baum-Shokrollahi's example already considered in Section 3.2.1. In this example, the authors proposed a construction of CCMA which has an optimal bilinear complexity. By using our method we get a better result on the scalar complexity for elliptic CCMA in $\mathbb{F}_{256}$ over $\mathbb{F}_4$. In particular, we give a construction of CCMA with $N_s = 52$ which is better than Baum-Shokrollahi's result

($N_s = 71$). Moreover, for this specific field, our construction is better than any of the algorithms introduced in Chapter 1.

4.1 Improving the scalar complexity

4.1.1 Parameters to evaluate scalar complexity

We first consider an analysis of the scalar complexity of the *original* CCMA (Algorithm 10). In this case, we observe that the number of the scalar multiplications, denoted by N_s, depends directly on the number of zeros in the matrices T_D and CT_{2D}^{-1}, respectively denoted by $N_{zero}(T_D)$ and $N_{zero}(CT_{2D}^{-1})$. Indeed, all the involved matrices being constructed once, the multiplication by a zero coefficient in a matrix has not to be taken into account. Thus, we get the formula to compute the number of scalar multiplications of this algorithm with respect to the number of zeros of the involved matrices as follows:

$$\begin{aligned} N_s &= 2\big(n(2n+g-1) - N_{zero}(T_D)\big) + \big(n(2n+g-1) - N_{zero}(CT_{2D}^{-1})\big) \\ &= 3n(2n+g-1) - N_z, \end{aligned} \tag{4.1}$$

where

$$N_z = 2N_{zero}(T_D) + N_{zero}(CT_{2D}^{-1}). \tag{4.2}$$

In order to reduce the number of scalar operations, we seek an algebraic function field $F/\mathbb{F}_q$ of genus g and a suitable triplet of divisors and places $(D, Q, \mathcal{P})$ with a good representation of the associated Riemann-Roch spaces such that the matrices T_D and CT_{2D}^{-1} are as *sparse* as possible. Therefore, we seek the best possible representations of Riemann-Roch space $\mathcal{L}(2D)$ along with F_Q and $\mathbb{F}_q^N$ to maximize the parameter N_z.

Secondly, we consider the kernel-type construction of CCMA (Algorithm 12 of Chapter 3). It involves the matrice T_{2D}, where

$$T_{2D} = \begin{pmatrix} f_1(P_1) & \cdots & f_{2n+g-1}(P_1) \\ f_1(P_2) & \cdots & f_{2n+g-1}(P_2) \\ \vdots & \vdots & \vdots \\ f_1(P_n) & \cdots & f_{2n+g-1}(P_n) \\ \vdots & \vdots & \vdots \\ f_1(P_{2n+g-1}) & \cdots & f_{2n+g-1}(P_{2n+g-1}) \end{pmatrix} = \left(\begin{array}{c|c} U_1 & U_3 \\ \hline U_2 & U_4 \end{array}\right)$$

where U_1 is a $n \times n$-matrix, U_2 is a $(n+g-1) \times n$-matrix, U_3 is a $n \times (n+g-1)$-matrix and U_4 is a $(n+g-1) \times (n+g-1)$-matrix over $\mathbb{F}_q$.

Le us denote by P the matrix

$$\left(\begin{array}{c|c} I_n & 0 \\ \hline 0 & 0 \end{array}\right)$$

where I_n is the unit matrix of size $n \times n$. In the same way, we can write T_{2D}^{-1} in the following form:

$$T_{2D}^{-1} = \left(\begin{array}{c|c} V_1 & V_3 \\ \hline V_2 & V_4 \end{array}\right).$$

Then

$$PT_{2D}^{-1} = \left(\begin{array}{c|c} V_1 & V_3 \\ \hline 0 & 0 \end{array}\right).$$

We thus conclude that the number of scalar multiplications in Algorithm 12 depends on the number of zeros in the sub-matrices U_1 and U_2 of T_{2D}, i.e. *the first n columns of matrix* T_{2D}, and the number of zeros in the sub-matrices V_1 and V_3 of PT_{2D}^{-1}, i.e. *the first n rows of matrix* T_{2D}^{-1}. We denote by $T_{2D,n}^{-1}$ the first n rows of T_{2D}^{-1}.

Therefore, when considering the kernel-type construction of CCMA, the formula (4.1) becomes:

$$\begin{array}{rcl} N_s & = & 2\big(n(2n+g-1) - N_{zero}(T_D)\big) + \big(n(2n+g-1) - N_{zero}(T_{2D,n}^{-1})\big) \\ & = & 3n(2n+g-1) - N_z, \end{array} \tag{4.3}$$

where

$$N_z = 2N_{zero}(T_D) + N_{zero}(T_{2D,n}^{-1}). \tag{4.4}$$

4.1.2 Optimization strategy of the scalar complexity

In this section, we consider the optimization for a fixed suitable triplet of divisor and places $(D, Q, \mathcal{P})$ for a given algebraic function field $F/\mathbb{F}_q$ of genus g.

For the *original CCMA of type (3.1)*, let us give the following definition:

Definition 4.1.1. *We call* $\mathcal{U}_{D,Q,\mathcal{P}}^{F,n} := (\mathcal{U}_{D,Q,\mathcal{P}}^{A}, \mathcal{U}_{D,Q,\mathcal{P}}^{R})$ *a Chudnovsky-Chudnovsky multiplication algorithm of type (3.1) where* $\mathcal{U}_{D,Q,\mathcal{P}}^{A} := E_{\mathcal{P}} \circ Ev_Q^{-1}$ *and* $\mathcal{U}_{D,Q,\mathcal{P}}^{R} := E_Q \circ {Ev_{\mathcal{P}}|_{ImEv_{\mathcal{P}}}}^{-1}$, *satisfying the assumptions of Theorem 3.1.1. We will say that two algorithms are equal, and we will note:* $\mathcal{U}_{D,Q,\mathcal{P}}^{F,n} = \mathcal{U}_{D',Q',\mathcal{P}'}^{F,n}$, *if* $\mathcal{U}_{D,Q,\mathcal{P}}^{A} = \mathcal{U}_{D',Q',\mathcal{P}'}^{A}$ *and* $\mathcal{U}_{D,Q,\mathcal{P}}^{R} = \mathcal{U}_{D',Q',\mathcal{P}'}^{R}$.

For *CCMA of the kernel-type construction*, we can modify the definition $\mathcal{U}_{D,Q,\mathcal{P}}^{F,n}$ in Definition 4.1.1 as follows:

$$\mathcal{U}_{D,Q,\mathcal{P}}^{F,n} = (\mathcal{U}_{D,Q,\mathcal{P}}^{A}, \mathcal{U}_{D,Q,\mathcal{P}}^{R}) = (Ev_{\mathcal{P}} \circ Ev_Q^{-1}, E_Q \circ Ev_{\mathcal{P}}^{-1}). \tag{4.5}$$

The notations $\mathcal{U}_{D,Q,\mathcal{P}}^{A}, \mathcal{U}_{D,Q,\mathcal{P}}^{R}$ are respectively demonstrated in the forward-phase and the return-phase of CCMA (cf. Definition 3.1.1)

Note that these definitions makes sense only if the bases of implied vector-spaces are fixed. So, we denote respectively by $\mathcal{B}_Q$, $\mathcal{B}_D$, and $\mathcal{B}_{2D}$ the basis of the residue class field F_Q, and of Riemann-Roch vector spaces $\mathcal{L}(D)$, and $\mathcal{L}(2D)$ associated to

$\mathcal{U}^{F,n}_{D,Q,\mathcal{P}}$. Note that the basis of the $\mathbb{F}_q$-vector space $\mathbb{F}_q^N$ is always the canonical basis. Then, we obtain the following result:

Proposition 4.1.1. *Let us consider an algorithm $\mathcal{U}^{F,n}_{D,Q,\mathcal{P}}$ such that the divisor D is an effective divisor, $D-Q$ a non-special divisor of degree $g-1$, and such that the cardinality of the set $\mathcal{P}$ is equal to the dimension of the Riemann-Roch space $\mathcal{L}(2D)$. Then we can choose the basis $\mathcal{B}_{2D}$ as (3.5) and for any σ in $GL_{\mathbb{F}_q}(2n+g-1)$, where $GL_{\mathbb{F}_q}(2n+g-1)$ denotes the linear group, we have*

$$\mathcal{U}^{F,n}_{\sigma(D),Q,\mathcal{P}} = \mathcal{U}^{F,n}_{D,Q,\mathcal{P}},$$

where $\sigma(D)$ denotes the action of σ on the basis $\mathcal{B}_D$ of $\mathcal{L}(D)$ in $\mathcal{U}^{F,n}_{D,Q,\mathcal{P}}$, with a fixed basis $\mathcal{B}_Q$ of the residue class field of the place Q and $\mathcal{B}^c$ the canonical basis of $\mathbb{F}_q^{2n+g-1}$. In particular, the quantity $N_{zero}(CT_{2D}^{-1})$ is constant under this action.

Proof. Let E, F and H be three vector spaces of finite dimension on a field K respectively equipped with the basis $\mathcal{B}_E$, $\mathcal{B}_F$ and $\mathcal{B}_H$. Consider two morphisms f and h respectively defined from E into F and from F into H and consider respectively their associated matrix $M_f(\mathcal{B}_E,\mathcal{B}_F)$ and $M_h(\mathcal{B}_F,\mathcal{B}_H)$. Then it is obvious that the matrix $M_{h\circ f}(\mathcal{B}_E,\mathcal{B}_H)$ of the morphism $h\circ f$ is independant from the choice of the basis $\mathcal{B}_F$ of F.

As the divisor D is effective, we have $\mathcal{L}(D)\subset\mathcal{L}(2D)$ and then $\mathcal{U}^{A}_{D,Q,\mathcal{P}} := E_{\mathcal{P}}\circ Ev_Q^{-1} = Ev_{\mathcal{P}}\circ Ev_Q^{-1}$ and as $D-Q$ a non-special divisor of degree $g-1$, Ev_Q is an isomorphism from $\mathcal{L}(D)$ into F_Q and we have $\mathcal{U}^{A}_{D,Q,\mathcal{P}} = Ev_{\mathcal{P}}|_{\mathcal{L}(D)}\circ Ev_Q^{-1}$.

Moreover, as the cardinality of the set $\mathcal{P}$ is equal to the dimension of the Riemann-Roch space $\mathcal{L}(2D)$, $Ev_{\mathcal{P}}$ is an isomorphism from $\mathcal{L}(2D)$ into $\mathbb{F}_q^{2n+g-1}$ equipped with the canonical basis $\mathcal{B}^c$.

Thus,

$$\mathcal{U}^{R}_{D,Q,\mathcal{P}} := E_Q\circ Ev_{\mathcal{P}}^{-1}|_{ImEv_{\mathcal{P}}} = E_Q|_{\mathcal{L}(2D)}\circ Ev_{\mathcal{P}}^{-1}.$$

Then, the matrix of $\mathcal{U}^{A}_{D,Q,\mathcal{P}}$ (resp. $\mathcal{U}^{R}_{D,Q,\mathcal{P}}$) is invariant under the action of σ in $GL_{\mathbb{F}_q}(n)$ (resp. in $GL_{\mathbb{F}_q}(2n+g-1)$) on the basis $\mathcal{B}_D$ (resp. $\mathcal{B}_{2D}$) since the set (E,F,H) is equal to $(F_Q,\mathcal{L}(D),\mathbb{F}_q^N)$ (resp. $(\mathbb{F}_q^{2n+g-1},\mathcal{L}(2D),F_Q)$) for $h\circ f := Ev_{\mathcal{P}}|_{\mathcal{L}(D)}\circ Ev_Q^{-1}$ (resp. $E_Q|_{\mathcal{L}(2D)}\circ Ev_{\mathcal{P}}^{-1}$). □

Remark 4.1.1. *Note that for any permutation π of the set $\mathcal{P}$, we ask whether or not $\mathcal{U}^{F,n}_{\sigma(D),Q,\pi(\mathcal{P})}$ is different from $\mathcal{U}^{F,n}_{D,Q,\mathcal{P}}$, where $\sigma(D)$ denotes the action of σ on the basis $\mathcal{B}_{2D}$ of $\mathcal{L}(2D)$ in $\mathcal{U}^{F,n}_{D,Q,\mathcal{P}}$, with a fixed basis $\mathcal{B}_Q$ of the residue class field of the place Q? In fact, the action of π corresponds to a permutation of the canonical basis $\mathcal{B}_c^n$ of $\mathbb{F}_q^{2n+g-1}$. It corresponds to a permutation of the rows of the matrix T_{2D}. In this case, $N_{zero}(T_{2D})$ is obviously constant under the action of π but nothing enables us to claim that $N_{zero}(C.T_{2D}^{-1})$ is constant.*

Proposition 4.1.2. *Let $\mathcal{U}^{F,n}_{D,Q,\mathcal{P}}$ be a CCMA of type (3.1) in a finite field $\mathbb{F}_{q^n}$, satisfying the assumptions of Proposition 4.1.1. The optimal scalar complexity $\mu^{s,o}(\mathcal{U}^{F,n}_{D,Q,\mathcal{P}})$ of $\mathcal{U}^{F,n}_{D,Q,\mathcal{P}}$ is reached for the set $\{\mathcal{B}_{D,max},\mathcal{B}_Q\}$ such that $\mathcal{B}_{D,max}$ is the basis of $\mathcal{L}(D)$ satisfying*

$$N_{zero}(T_{D,max}) = \max_{\sigma\in GL_{\mathbb{F}_q}(n)} N_{zero}(T_{\sigma(D)}),$$

where $\sigma(D)$ denotes the action of σ on the basis $\mathcal{B}_D$ of $\mathcal{L}(D)$ in $\mathcal{U}^{F,n}_{D,Q,\mathcal{P}}$, $T_{D,max}$ the matrix of the restriction of the evaluation map $Ev_{\mathcal{P}}$ on the Riemann-Roch vector space $\mathcal{L}(D)$ equipped

with the bases $\mathcal{B}_{D,max}$ *and* $\mathcal{B}_Q = Ev_Q(\mathcal{B}_{D,max})$. *In particular,*

$$\mu^{s,o}(\mathcal{U}^{F,n}_{D,Q,\mathcal{P}}) = \min_{\sigma \in GL_{\mathbb{F}_q}(n)} \{\mu^s(\mathcal{U}^{F,n}_{\sigma(D),Q,\mathcal{P}} \mid \sigma(\mathcal{B}_D) \text{ is the basis of } \mathcal{L}(D) \text{ and } \mathcal{B}_Q = Ev_Q(\mathcal{B}_D)\}$$

$$= 3n(2n+g-1) - (2N_{zero}(T_{D,max}) + N_{zero}(T^{-1}_{2D,n})),$$

where matrices C *and* T_{2D} *are defined with respect to the basis* $\mathcal{B}_Q = Ev_Q(\mathcal{B}_{D,max})$, *and* $\mathcal{B}_{2D} = \mathcal{B}_{D,max} \cup \mathcal{B}^c_D$ *with* $\mathcal{B}^c_D$ *a basis of the kernel of* $E_Q|_{\mathcal{L}(2D)}$ *and* $T^{-1}_{2D,n}$ *denotes the matrix constituted of the first* n *rows of the matrix* T^{-1}_{2D}.

Proof. It follows directly from Proposition 4.1.1 and formulae (4.1) and (4.2). Note that since the quantity $N_{zero}(CT^{-1}_{2D})$ is constant for any basis $\mathcal{B}_{2D} = \mathcal{B}_{D,max} \cup B^c_D$ of $\mathcal{L}(2D)$ (B^c_D is changed by action σ on $\mathcal{B}^c_D$) , we can take the matrix $CT^{-1}_{2D} = T^{-1}_{2D,n}$ if $\mathcal{B}^c_D$ is a basis of the kernel of $E_Q|_{\mathcal{L}(2D)}$. □

A new setup algorithm can be obtained directly from the strategy developed in Section 4.1.2. More precisely, the following setup corresponds to the optimization described by Proposition 4.1.2.

Algorithm 13: Setup algorithm for a global optimization of N_s

Input: $F/\mathbb{F}_q$, $Q, D, \mathcal{P} = \{P_1, \ldots, P_{2n+g-1}\}$.
Output: $\mathcal{B}_{2D} = \mathcal{B}_D \cup \mathcal{B}^c_D$, T_{2D} and $T^{-1}_{2D,n}$.

(i) Check that the function field $F/\mathbb{F}_q$, the place Q, the divisors D such that Conditions (i) and (ii) in Theorem 3.2.1 are satisfied.

(ii) Construct a basis $\mathcal{B}^c_D := (f_{n+1}, ..., f_{2n+g-1})$ of the complementary space $Ker E_Q|_{\mathcal{L}(2D)}$ of $\mathcal{L}(D)$ in $\mathcal{L}(2D)$.

(iii) Go through the set (or subset) of bases $\mathcal{B}_D$ of $\mathcal{L}(D)$ to compute T_{2D} and $T^{-1}_{2D,n}$ in the basis $\mathcal{B}_{2D} = \mathcal{B}_D \cup \mathcal{B}^c_D$.

(iv) Choose a basis $\mathcal{B}_D := (f_1, \ldots, f_n)$ such that N_z be the largest.

(v) Set $\mathcal{B}_Q := Ev_Q(\mathcal{B}_D)$.

Our proposed optimization strategy on the scalar complexity of CCMA associated to above setup algorithm was published in [BBD19].

With the goal to maximize the parameter $N_z = 2N_{zero}(T_D) + N_{zero}(T^{-1}_{2D,n})$ of CCMA, we pay much attention to $N_{zero}(T_D)$ which is related to the non-zero coefficients in the n columns of the matrix T_D. We note that each column in this matrix corresponds to the a column of evaluations of the basis vectors in $\mathcal{B}_D$ on the points $P_1, ..., P_N$. Moreover, in the construction of CCMA, D is chosen as an effective divisor, so $\mathcal{L}(D)$ is the vector subspace of $\mathcal{L}(2D)$ of dimension n. This inspired us to think of the following notation in algebraic code theory.

We consider an algebraic geometry code (or AG code) which was introduced by V.D. Goppa in [Gop81].

Definition 4.1.2. *Let*
$F/\mathbb{F}_q$ *be an algebraic function field of genus* g,
$P_1, ..., P_N$ *are pairwise distinct places of* $F/\mathbb{F}_q$ *of degree one,*

$G = P_1 + \cdots + P_N$,
D are divisors of $F/\mathbb{F}_q$ such that $suppP \cap suppD = \emptyset$.
The AG code $C_{\mathcal{L}}(G, D)$ associated with the divisors G and D is defined as

$$C_{\mathcal{L}}(G, D) := \{(f(P_1), ..., f(P_N)) | f \in \mathcal{L}(D)\} \subseteq \mathbb{F}_q^N.$$

Then $C_{\mathcal{L}}(G, D)$ is an $[N, k, d]$ code with parameters: dimension $k = \dim \mathcal{L}(D) - \dim \mathcal{L}(D - G)$ and minimum distance d of the lower bound $(N - \deg D)$ (see [Sti93, Theorem 2.2.2]).

If $\{f_1, ..., f_n\}$ is a basis of $\mathcal{L}(D)$, the matrix

$$M := \begin{pmatrix} f_1(P_1) & \cdots & f_1(P_N) \\ f_2(P_1) & \cdots & f_2(P_N) \\ \vdots & \vdots & \vdots \\ f_n(P_1) & \cdots & f_n(P_N) \end{pmatrix}$$

is a generator matrix for $C_{\mathcal{L}}(G, D)$.

We note that in our construction of CCMA, D is an effective divisor. We have the property that $\mathcal{L}(D) \subseteq \mathcal{L}(2D)$. Then, the image of $\mathcal{L}(D)$ under $Ev_{\mathcal{P}}$ which is a subspace of $\mathbb{F}_q^N$ of dimension n is considered as an algebraic geometry code $C_{\mathcal{L}}(G, D) = [N, n, d]$ where $G = P_1 + \cdots + P_N$.

We observe that $T_D \equiv M^t$. We have

$$N_{zero}(T_D) = n \cdot N - N_{nz}(T_D), \tag{4.6}$$

where $N_{nz}(T_D)$ denotes the number of non-zero entries of T_D. By the definition of minimum distance d of an AG code, we see that

$$N_{nz}(T_D) \geq n \cdot d. \tag{4.7}$$

The factor n in the formula (4.6) as well as (4.7) is the number of columns of T_D.

Since $d \geq N - \deg D$, we have

$$N_{nz}(T_D) \geq n(N - \deg D).$$

Thus,

$$N_{zero}(T_D) \leq n \cdot \deg D. \tag{4.8}$$

If $N = 2n + g - 1$, as in Remark 3.2.2 we take the divisor D as a place of degree $n + g - 1$, then *the upper bound of $N_{zero}(T_D)$ is $n(n + g - 1)$*. We obtain the following theorem.

Theorem 4.1.1. *Let $\mathcal{U}^{F,n}_{D,Q,\mathcal{P}}$ be a Chudnovsky-Chudnovsky multiplication algorithm in a finite field $\mathbb{F}_{q^n}$, satisfying the assumptions of Proposition 4.1.2. Then*

$$\mu^s(\mathcal{U}^{F,n}_{D,Q,\mathcal{P}}) > n(2n - 3g + 3).$$

Proof. Since $N_{zero}(T_D) \leq n(n + g - 1)$, $N_{zero}(T^{-1}_{2D,n}) < n(2n + g - 1)$ and the formula (4.3): $N_s = 3n(2n + g - 1) - N_z$, where $N_z = 2N_{zero}(T_D) + N_{zero}(T^{-1}_{2D,n})$, we can imply the strict lower-bound of N_s. □

Using this upper-bound value of $N_{zero}(T_D)$, we can give the following algorithm (Algorithm 14) to effectively seek a good basis of $\mathcal{L}(2D)$. In particular, among the

best bases of the Riemann-Roch space $\mathcal{L}(D)$ such that $N_{zero}(T_D)$ obtains its maximal value), we look for the ones that give the best value for $N_{zero}(T_{2D,n}^{-1})$.

Algorithm 14: Setup algorithm for an efficient optimization

Input: $F/\mathbb{F}_q$, $Q, D, \mathcal{P} = \{P_1, \ldots, P_{2n+g-1}\}$.
Output: $\mathcal{B}_{2D} = \mathcal{B}_D \cup \mathcal{B}_D^c$, T_D and $T_{2D,n}^{-1}$.

(i) Check that the function field $F/\mathbb{F}_q$, the place Q, the divisor D such that Conditions (i) and (ii) in Theorem 3.2.1 are satisfied.

(ii) Construct a basis $\mathcal{B}_D^c := (f_{n+1}, ..., f_{2n+g-1})$ of the complementary space $KerE_Q|_{\mathcal{L}(2D)}$ of $\mathcal{L}(D)$ in $\mathcal{L}(2D)$.

(iii) Go through the set (or subset) $\mathcal{S}$ of bases $\mathcal{B}_D$ of $\mathcal{L}(D)$, set $mB_D := \{\mathcal{B}_D \in \mathcal{S} \mid N_{zero}(T_D) = \boldsymbol{n(n+g-1)}\}$.

(iv) Search in mB_D a basis $n\mathcal{B}_D := (f_1, \ldots, f_n)$ such that $N_{zero}(T_{2D,n}^{-1})$ (with respect to $\mathcal{B}_{2D} := n\mathcal{B}_D \cup \mathcal{B}_D^c$) be the largest.

(v) Set $\mathcal{B}_Q := Ev_Q(n\mathcal{B}_D)$.

We realize that the computational efficiency of either Algorithm 13 or Algorithm 14 mainly relies on computations in Step (iii) and Step (iv). In Step (iii) of Algorithm 13, two matrices T_{2D} and T_{2D}^{-1} of size $(2n+g-1) \times (2n+g-1)$ are computed with respect to the basis $\mathcal{B}_{2D}$ of $\mathcal{L}(2D)$ in order to maximize the global parameter $N_z = 2N_{zero}(T_D) + N_{zero}(T_{2D,n}^{-1})$.

In Step (iii) of Algorithm 14, we just need to compute the matrix T_D of size $n \times (2n+g-1)$ such that $N_{zero}(T_D) = n(n+g-1)$, and then the computation of Step (iv) is no longer too heavy because the cardinality of the set mB_D is much reduced. Therefore, from this analysis we might say that the computational efficiency of Algorithm 14 is better than that of Algorithm 13. This result is confirmed by experiments carried out in Magma.

Remark 4.1.2. *From (4.8), it raises a question whether it is possible to take an effective divisor D satisfying conditions: $suppD \cap \{Q, P_1, ..., P_{2n+g-1}\} = \emptyset$, $D - Q$ is non-special as required in the kernel-type construction of CCMA and especially $n+g-1 < \deg D < 2n+g-1$, instead of choosing the divisor D as a place of degree $n+g-1$ in the algebraic function field F over $\mathbb{F}_q$ like in Section 3.3 of Chapter 3. If we could do so, then the increased upper-bound value of $N_{zero}(T_D)$ in Step (iii) of Algorithm 14 would give us an opportunity to improve significantly the parameter N_s of CCMA.*

4.1.3 Other strategies of optimization

With the view of a complete optimization concerning the scalar complexity, we have to vary the appropriate triplet $(D, Q, \mathcal{P})$.

In the first strategy, we change the set $\mathcal{P}$ of rational places in $F/\mathbb{F}_q$ in the triplet $(D, Q, \mathcal{P})$. We see that $\mathcal{P} = \{P_1, ..., P_N\}$ is replaced by another set $P' = \{P'_1, ..., P'_N\}$ of the rational places in F where $P_i \neq P'_i$, for $i = 1, ..., N$, then $\mathcal{U}_{D,Q,\mathcal{P}}^{F,n}$ and $\mathcal{U}_{D,Q,\mathcal{P}'}^{F,n}$ might almost be different.

In the following, we just consider the order of the rational places are modified.

For any permutation π of the set $\mathcal{P}$, we can ask if $\mathcal{U}_{D,Q,\pi(\mathcal{P})}^{F,n}$ is different from $\mathcal{U}_{D,Q,\mathcal{P}}^{F,n}$. The action of π corresponds to a permutation of the canonical basis $\mathcal{B}_{\mathbb{F}_q^{2n+g-1}}^c$

of $\mathbb{F}_q^{2n+g-1}$. It corresponds to a permutation of the rows of the matrix T_{2D}. In this case, $N_{zero}(T_{2D})$ is obviously constant under the action of π. The following proposition allows us to claim that $N_{zero}(CT_{2D}^{-1})$ is always constant.

Proposition 4.1.3. *Let us consider an algorithm $\mathcal{U}_{D,Q,\mathcal{P}}^{F,n}$ such that D is an effective divisor, $D-Q$ a non-special divisor of degree $g-1$, and $|\mathcal{P}| = \dim \mathcal{L}(2D) = N$.*

Then for any π in S_N where S_N is the symmetric group on the set $\{1,2,...,N\}$, we have

$$\mu^s(\mathcal{U}_{D,Q,\mathcal{P}}^{F,n}) = \mu^s(\mathcal{U}_{D,Q,\pi(\mathcal{P})}^{F,n}).$$

In particular, the quantities $N_{zero}(T_D)$ and $N_{zero}(CT_{2D}^{-1})$ are constants under the action π.

Proof. Let $\mathcal{P} := \{P_1, P_2, ..., P_N\}$ be a set of rational places of the function field F in the algorithm. We consider the action of the permutation π in the symmetric group S_N on $\mathcal{P}$. We set $\pi(\mathcal{P}) := \{P_{\pi(1)}, P_{\pi(2)}, ..., P_{\pi(N)}\}$.

Given a basis $\mathcal{B}_{2D}$ of the Riemann-Roch space $\mathcal{L}(2D)$, we consider two evaluation maps:

$$\begin{array}{rcl} Ev_{\mathcal{P}} : \mathcal{L}(2D) & \to & \mathbb{F}_q^N \\ f & \mapsto & \big(f(P_1), ..., f(P_N)\big) \end{array} \tag{4.9}$$

and

$$\begin{array}{rcl} Ev_{\pi(P)} : \mathcal{L}(2D) & \to & \mathbb{F}_q^N \\ f & \mapsto & \big(f(P_{\pi(1)}), ..., f(P_{\pi(N)})\big) \end{array} \tag{4.10}$$

We denote by $\mathcal{B}_{Id}^c = (e_1, ..., e_N)$ the canonical basis of $\mathbb{F}_q^N$ in (4.9) and by $\mathcal{B}_{\pi}^c = (e_{\pi(1)}, ..., e_{\pi(N)})$ the basis of $\mathbb{F}_q^N$ in (4.10).

We define an isomorphism $p : \mathbb{F}_q^N \to \mathbb{F}_q^N$ by $p(e_i) = e_{\pi(i)}$ for $i = 1, ..., N$. Let us denote by M the matrix permutation of p and we note that $M^{-1} = M^t$. We have

$$Ev_{\pi(P)} = p \circ Ev_{\mathcal{P}}.$$

Then

$$\mathcal{U}_{D,Q,\pi(P)}^{A} = E_{\pi(P)} \circ Ev_Q^{-1} = (p \circ E_{\mathcal{P}}) \circ Ev_Q^{-1} = p \circ \mathcal{U}_{D,Q,\mathcal{P}}^{A} \tag{4.11}$$

and

$$\mathcal{U}_{D,Q,\pi(P)}^{R} = E_Q \circ Ev_{\pi(P)}^{-1}|_{Im(Ev_{\pi(P)})} = E_Q \circ \big(p \circ Ev_{\mathcal{P}}|_{Im(Ev_{\mathcal{P}})}\big)^{-1} = \mathcal{U}_{D,Q,\mathcal{P}}^{R} \circ p^{-1}. \tag{4.12}$$

Observing the change in the positions of rows of T_D and the columns of CT_{2D}^{-1} respectively by (4.11) and (4.12), we have $N_{zero}(T_D)$ and $N_{zero}(CT_{2D}^{-1})$ are constants for any $\pi \in S_N$. By the formulae (4.1) we obtain

$$\mu^s(\mathcal{U}_{D,Q,\mathcal{P}}^{F,n}) = \mu^s(\mathcal{U}_{D,Q,\pi(\mathcal{P})}^{F,n}),$$

for any $\pi \in S_N$. □

We thus can answer for an undefined statement in Remark 4.1.1 and conclude that one cannot perform a strategy of optimization by using the permutations of rational places $P_1, ..., P_N$ in F.

Besides, we can vary the couples (D, Q) and apply the previous step: for instance, we can start by fixing the place Q and then vary the suitable divisors D. We can then search for an appropriate fixed algebraic function field of genus g, up to isomorphism, and repeat all the previous steps.

Furthermore, it is still possible to look at the trade-off between scalar complexity and bilinear complexity by increasing the genus and then re-conduct all the previous optimizations.

4.2 Optimization of scalar complexity in the elliptic case

Now, we study a specialisation of CCMA of type (3.1) in the case of the elliptic curves. In particular, we improve the scalar complexity of the algorithm which was constructed by U. Baum and M.A. Shokrollahi in [BS91]. Their construction of the algorithm is optimal in the model that only the bilinear complexity is taken into account and it is assumed that all scalar operations are free.

We recall some results of the computation in Section 3.2.1 which are necessary parameters for Algorithm 10, in particular

$$T_D = \begin{pmatrix} 0 & 0 & 0 & 1 \\ \omega^2 & 0 & 1 & 0 \\ \omega^2 & 0 & \omega & 0 \\ \omega^2 & \omega^2 & 0 & \omega^2 \\ \omega^2 & \omega^2 & \omega^2 & \omega^2 \\ \omega & \omega^2 & 0 & 1 \\ \omega & \omega^2 & \omega & 1 \\ \omega & 1 & 0 & \omega^2 \end{pmatrix} \tag{4.13}$$

and

$$CT_{2D}^{-1} = \begin{pmatrix} 1 & \omega & 1 & \omega & 1 & 1 & \omega & 0 \\ 1 & 0 & \omega^2 & \omega & 1 & \omega^2 & 1 & \omega \\ 1 & \omega & \omega & \omega^2 & 1 & \omega^2 & \omega & \omega \\ 0 & \omega & \omega^2 & \omega & 1 & \omega^2 & 0 & 0 \end{pmatrix}. \tag{4.14}$$

We have

$$N_{zero}(T_D) = 10 \text{ and } N_{zero}(CT_{2D}^{-1}) = 5.$$

Thus, the total number N_s of scalar multiplications in the algorithm constructed by Baum and Shokrollahi in [BS91] is $N_s = 71$ by the formula (4.1).

4.2.1 Implementation of optimization algorithm

In this section, we present the Magma implementations our construction of CCMA in $\mathbb{F}_{256}/\mathbb{F}_4$ that be better in term of scalar complexity, than the result of Baum-Shokrollahi.

1/ We present the implementation based on Algorithm 13. In the first evaluation map $Ev_Q : \mathcal{L}(D) \to F_Q$, for the fixed basis of the residue class field F_Q, we vary the basis of $\mathcal{L}(D)$ (without needing to vary the basis of the complementary space of $\mathcal{L}(D)$ in $\mathcal{L}(2D)$) in order to maximize the parameter

$$N_s = \left(2N_{zero}(T_D) + N_{zero}(T_{2D,n}^{-1})\right)$$

where T_D is the first n columns of T_{2D} and $T_{2D,n}^{-1}$ is the first n rows of T_{2D}^{-1}.

The matrix T_{2D} of the second evaluation map in CCMA:

$$\begin{array}{rccc} Ev_{\mathcal{P}} : & \mathcal{L}(2D) & \to & \mathbb{F}_q^{2n+g-1} \\ & f & \mapsto & (f(P_i))_{i=1}^{2n+g-1} \end{array}$$

is computed with respect of the canonical basis of $\mathbb{F}_q^{2n+g-1}$ and the basis $\mathcal{B}_{\mathcal{L}(2D)}$ of $\mathcal{L}(2D)$ determined by

$$\mathcal{B}_{\mathcal{L}(2D)} = \mathcal{B}_{\mathcal{L}(D)} \cup \mathcal{B}_K$$

where K is complementary space of $\mathcal{L}(D)$ in $\mathcal{L}(2D)$.

In our construction, we choose K as the kernel space of $E_Q|_{\mathcal{L}(2D)}$ with $E_Q : \mathcal{O}_Q \to \mathcal{O}_Q/\langle Q\rangle = F_Q$ be the canonical projection from the evaluation ring $\mathcal{O}_Q$ of the place Q onto its residue class field F_Q.

Let us give the following pre-computing function in Magma program.

The function `MatrixSecondEval` is as follows.

Function `MatrixSecondEval`(B, P, m, N)
Input: $B = \{f_1, ..., f_m\}$, $\mathcal{P} = \{P_1, ..., P_N\}$.
Output: T_{2D}.

```
ST := [ ];
for j := 1 to N do
    for i := 1 to m do
        ST := Append(ST, Evaluate(B[i,1], P[j]));
    end for;
end for;
T := Matrix(n, m, ST);
```

return T.

Let S be an ordered set of nonsingular matrices of size $n \times n$ over $\mathbb{F}_q$. The cardinality $l(S)$ of this set is determined by the formula $l(S) := |GL(n, \mathbb{F}_q)| = \prod_{i=1}^{n}(q^n - q^{i-1})$ (here $l(S) = 2961100800$ for $n = q = 4$).

We propose a function in Magma program as follows.

Function `NumScaMult`(F, LP, TM, E, M, BK).
Input: - $F/\mathbb{F}_q$.

- $LP = \{P_1, ..., P_N\}$.
- TM is the transformation matrix in S.
- E is the matrix of Ev_Q.
- M is the matrix of the basis of $\mathcal{L}(D)$.
- B_K is the matrix of the basis of K.

Output: The number N_s of scalar multiplications in CCMA.

1. $nBasisLD := Matrix(F, TM) \cdot E^{-1} \cdot M$;
2. $nML2D := Matrix(2n + g - 1, 1, [nBasisLD[i, 1] \ : \ i \text{ in } [1..n]] \text{ cat } [BK[i, 1] \ : \ \text{i in } [1..n + g - 1]])$;
3. $nT :=$ `MatrixSecondEval`$(nML2D, LP, 2n + g - 1, 2n + g - 1)$;
4. $nCounterT :=$ `CountZeros`$(nT, 2n + g - 1, n)$;
5. $nCounterTI :=$ `CountZeros`$(nT^{-1}, n, 2n + g - 1)$;
6. $N_s := 6n^2 - (2 \times nCounterT + nCounterTI)$;

return N_s.

In this function `NumScaMult`, we have just recalled the function `MatrixSecondEval` as mentioned above and used the function `CountZeros(-)` to count the number of zeros appearing in some matrix.

We now present the algorithm to find an optimized basis of L(2D) for CCMA.

Algorithm 15: Finding the optimized basis of $\mathcal{L}(2D)$ in CCMA

Input: $F/\mathbb{F}_q, Q, D, \mathcal{P} = \{P_1, ..., P_{2n+g-1}\}$.
Output: An optimized basis of space $\mathcal{L}(2D)$.

Compute MLD the column matrix of the basis vectors of $\mathcal{L}(D)$;
Compute B_K the column matrix of the basis vectors of complementary space of $\mathcal{L}(D)$ in $\mathcal{L}(2D)$;
Compute $BELD$ the matrix of the evaluation map Ev_Q ;
$G \leftarrow GL(n, \mathbb{F}_q)$;
$S \leftarrow Classes(G)$ ▷ S `is the set of conjugacy classes of` G;
for k *in* $[1\,..\,|S|]$ **do**
  $rep \leftarrow S[k][3]$;
  $C \leftarrow Class(G, rep)$ ▷ C `is` k`-th class of the representation element` rep `in` S;
  while $i < |C|$ **do**
    $N_s i \leftarrow NumScaMult(F, \mathcal{P}, C[i], BELD, MLD, B_K)$;
    $N_s i1 \leftarrow NumScaMult(F, \mathcal{P}, C[i+1], BELD, MLD, B_K)$;
    Compare $N_s i$ and $N_s i1$ to obtain $minNs$;
    $i \leftarrow i + 2$;
Compute $\mathcal{B}_{opt}LD$ be the basis of $\mathcal{L}(D)$ with respect to $minNS$;
$\mathcal{B}_{opt}L2D \leftarrow \mathcal{B}_{opt}LD \cup B_K$;
return $\mathcal{B}_{opt}L2D$

We can refer to the detail of Magma implementation program of Algorithm 15 in Section A.2 of Appendix.

2/ For another implementation to find the optimization basis of $\mathcal{L}(2D)$ in CCMA, based on the construction in Algorithm 14, we can propose the following approach whose computational efficiency is better than the previous one. However, we will see that this approach does not cover all cases of the optimization.

Indeed, in the first step we will use the approach of transforming all bases of the Riemann-Roch vector space $\mathcal{L}(D)$ such that the *number of zeros of* T_D is $n(n+g-1)$. In other words, $N_{zero}(T_D)$ reaches the upper bound. In the second step, we find a basis among the new after-transformed bases of $\mathcal{L}(D)$ obtained in the previous step satisfying the condition that $N_{zero}(T^{-1}_{2D,n})$ is maximal. We see that the computation complexity of the optimization processing mainly depends on the first step.

Remark 4.2.1. *At the second step, the process of searching the best basis* $\mathcal{B}_{D,max}$ *in the set* $mB_D := \{\mathcal{B}_D \mid N_{zero}(T_D) = n(n+g-1)\}$ *actually makes sense if there exist at least two different bases* $\mathcal{B}^{(1)}_D$ *and* $\mathcal{B}^{(2)}_D$ *in* mB_D *such that* $N_{zero}(T^{-1}_{2D,n})^{(1)}$ *corresponding to* $\mathcal{B}^{(1)}_D$ *is different from* $N_{zero}(T^{-1}_{2D,n})^{(2)}$ *corresponding to* $\mathcal{B}^{(2)}_D$.

In fact, by the practical computation, there are two different bases $B^{(1)}_D, B^{(2)}_D$ *in set* mB_D, *for example:*

- $B_D^{(1)} \cup \mathcal{B}_D^c = B_{2D}^{(1)} := \{f_1, ..., f_8\}$, *where*

$$
\begin{aligned}
f_1 &= \frac{\omega x^2 + \omega^2 x}{x^2 + x + \omega}, \quad f_2 = \frac{\omega x^2 + \omega x}{x^2 + x + \omega}, \\
f_3 &= \frac{\omega^2 y + \omega x + 1}{x^2 + x + \omega}, \quad f_4 = \frac{\omega^2 y + \omega^2 x + \omega}{x^2 + x + \omega}, \\
f_5 &= \frac{(\omega x^2 + \omega x)y + \omega^2 x^4 + \omega x^3 + x^2 + x + \omega}{x^4 + x^2 + \omega^2}, \\
f_6 &= \frac{(\omega^2 x^2)y + \omega x^4 + \omega x^3 + x^2 + \omega x}{x^4 + x^2 + \omega^2}, \\
f_7 &= \frac{(x^2 + \omega^2 x)y + \omega x^4 + \omega x^2}{x^4 + x^2 + \omega^2}, \\
f_8 &= \frac{(\omega x + \omega)y + \omega x^4}{x^4 + x^2 + \omega^2}
\end{aligned}
$$

- $B_D^{(2)} \cup \mathcal{B}_D^c = B_{2D}^{(2)} := \{g_1, ..., g_8\}$, *where*

$$
\begin{aligned}
g_1 &= \frac{\omega x^2 + \omega x}{x^2 + x + \omega}, \quad g_2 = \frac{\omega^2 y + x + \omega}{x^2 + x + \omega}, \\
g_3 &= \frac{\omega x^2 + x + \omega^2}{x^2 + x + \omega}, \quad g_4 = \frac{\omega x^2 + \omega^2 x}{x^2 + x + \omega}, \\
g_5 &= \frac{(\omega x^2 + \omega x)y + \omega^2 x^4 + \omega x^3 + x^2 + x + \omega}{x^4 + x^2 + \omega^2}, \\
g_6 &= \frac{(\omega^2 x^2)y + \omega x^4 + \omega x^3 + x^2 + \omega x}{x^4 + x^2 + \omega^2}, \\
g_7 &= \frac{(x^2 + \omega^2 x)y + \omega x^4 + \omega x^2}{x^4 + x^2 + \omega^2}, \\
g_8 &= \frac{(\omega x + \omega)y + \omega x^4}{x^4 + x^2 + \omega^2}.
\end{aligned}
$$

The matrices $T_D^{(1)}$ *and* $T_D^{(2)}$ *corresponding to* $B_D^{(1)}$ *and* $B_D^{(2)}$ *respectively have the same value* $N_{zero}(T_D) = n(n + g - 1) = 16$.

We have:

$$
(T_{2D,n}^{-1})^{(1)} = \begin{pmatrix}
\omega^2 & 1 & \omega & \omega^2 & \omega & \omega & \omega & \omega \\
\omega^2 & 0 & 0 & \omega & 1 & 0 & \omega & \omega \\
\omega^2 & 1 & 0 & 0 & \omega^2 & 1 & 1 & \omega \\
1 & \omega & \omega^2 & 1 & 1 & 0 & \omega & 1
\end{pmatrix}
$$

has $N_{zero}(T_{2D,n}^{-1})^{(1)} = 6$,

$$
(T_{2D,n}^{-1})^{(2)} = \begin{pmatrix}
\omega & \omega & 1 & 0 & 0 & 1 & 0 & \omega \\
\omega & \omega^2 & \omega^2 & 1 & \omega & 1 & \omega^2 & \omega^2 \\
\omega^2 & 1 & 0 & 0 & \omega^2 & 1 & 1 & \omega \\
1 & \omega & \omega^2 & 1 & 0 & \omega & 1 & 0
\end{pmatrix}
$$

has $N_{zero}(T_{2D,n}^{-1})^{(2)} = 7$.

Now we present the following algorithm to optimize the basis $\mathcal{B}_{2D}$ of $\mathcal{L}(2D)$.

Algorithm 16: Finding the optimized basis of $\mathcal{L}(2D)$ with maximal $N_z(T_D)$ in CCMA

Input: $F/\mathbb{F}_q, Q, D, \mathcal{P} = \{P_1, ..., P_{2n+g-1}\}$.
Output: An optimized basis of space $\mathcal{L}(2D)$.

```
Compute MLD the column matrix of basis vectors of L(D);
Compute B_K the column matrix of basis vectors of complementary space of
  L(D) in L(2D);
Compute BELD the matrix of the evaluation map Ev_Q ;
G ← GL(n, F_q);
S ← Classes(G);
for k in [1 .. |S|] do
    rep ← S[k][3];
    C ← Class(G, rep);
    for i in [1 .. |C|] do
        nBasisLD ← Matrix(F, C[i]) · BELD^{-1} · MLD;
        nT_D ← MatrixSecondEval(nBasisLD, P, n, 2n + g − 1);
        N_zT_D ← CountZeros(nT_D, 2n + g − 1, n);
        if N_zT_D = n(n+g-1) do
            mB_D ← Append(mB_D, nBasisLD)
    while j < |mB_D| do
        N_zTIj ← NumZeroInv(F, P, mB_D[j], B_K) ;
        N_zTIj1 ← NumZeroInv(F, P, mB_D[j + 1], B_K) ;
        Compare N_zTIj and N_zTIj1 to obtain maxN_zTI ;
        j ← j + 2
Compute B_opt LD the basis of L(D) with respect to maxN_zTI;
B_opt L2D ← B_opt LD ∪ B_K;
return B_opt L2D.
```

where `NumZeroInv` is the function to calculate the number of zeros of $T^{-1}_{2D,n}$. This pre-computed function is given in Magma program as follows.

Function `NumZeroInv`(F, LP, B_D, B_K).
Input:

- $F/\mathbb{F}_q$.
- $LP = \{P_1, ..., P_{2n+g-1}\}$.
- B_D is the basis of $\mathcal{L}(D)$.
- B_K is the basis of the complementary space K.

Output: $N_{zero}(T^{-1}_{2D,n})$.

1. B_{2D} := Matrix$(2n + g - 1, 1, [B_D[i, 1]$: i in $[1..n]]$ cat $[B_K[i, 1]$: i in $[1..n + g - 1]])$;
2. nT_{2D} := `MatrixSecondEval`$(B_{2D}, LP, 2n + g - 1, 2n + g - 1)$;
3. $Nz_nRowsTI$:= `CountZeros`$((nT_{2D})^{-1}, n, 2n + g - 1)$;

return $Nz_n RowsTI$.

4.2.2 New design of the Baum-Shokrollahi construction

The new construction of CCMA for the multiplication in $\mathbb{F}_{256}/\mathbb{F}_4$ using the strategy given in Section 4.1.2 and Section 4.2.1 gives us an optimized basis $\mathcal{B}_{2D}^{opt} = (f_1, f_2, ..., f_8)$ of $\mathcal{L}(2D)$ and the matrices T_{2D} of the second evaluation map $Ev_{\mathcal{P}}$, where $\mathcal{P} := \{P_\infty, P_1, P_2, P_7, P_8, P_3, P_4, P_5\}$ used in CCMA:

$$
\begin{aligned}
f_1 &= \frac{y + \omega x + \omega^2}{x^2 + x + \omega}, \\
f_2 &= \frac{y + \omega^2 x + \omega}{x^2 + x + \omega}, \\
f_3 &= \frac{\omega x^2 + \omega^2 x}{x^2 + x + \omega}, \\
f_4 &= \frac{\omega y}{x^2 + x + \omega}, \\
f_5 &= \frac{(\omega x^2 + \omega x)y + \omega^2 x^4 + \omega x^3 + x^2 + x + \omega}{x^4 + x^2 + \omega^2}, \\
f_6 &= \frac{\omega^2 x^2 y + \omega x^4 + \omega x^3 + x^2 + \omega x}{x^4 + x^2 + \omega^2}, \\
f_7 &= \frac{(x^2 + \omega^2 x)y + \omega x^4 + \omega x^2}{x^4 + x^2 + \omega^2}, \\
f_8 &= \frac{(\omega x + \omega)y + \omega x^4}{x^4 + x^2 + \omega^2}.
\end{aligned}
$$

$$
T_{2D} = \begin{pmatrix}
0 & 0 & \omega & 0 & \omega^2 & \omega & \omega & \omega \\
\omega^2 & 0 & 0 & \omega & \omega^2 & 0 & 0 & 1 \\
0 & \omega^2 & 0 & \omega^2 & \omega^2 & 0 & 0 & \omega \\
\omega^2 & \omega^2 & \omega^2 & 0 & 1 & 1 & 0 & \omega^2 \\
0 & 0 & \omega^2 & 1 & 1 & 0 & \omega^2 & \omega^2 \\
0 & 1 & 0 & 0 & 0 & \omega^2 & 1 & \omega \\
\omega & \omega^2 & 0 & \omega^2 & 1 & \omega & 0 & 1 \\
\omega^2 & 0 & \omega & 0 & \omega & 0 & 1 & \omega^2
\end{pmatrix}
$$

and

$$
T_{2D,4}^{-1} = \begin{pmatrix}
1 & 0 & 0 & \omega^2 & \omega & 0 & \omega^2 & \omega^2 \\
1 & 1 & \omega^2 & 0 & 0 & \omega^2 & 0 & 1 \\
0 & 1 & \omega & 1 & \omega^2 & \omega & 0 & 0 \\
\omega^2 & \omega & \omega^2 & 0 & \omega & 0 & \omega^2 & 0
\end{pmatrix}.
$$

Therefore, $N_{zero}(T_D) = 16$ and $N_{zero}(T_{2D,4}^{-1}) = 12$. By the formula (4.3), we obtain $N_s = 52$, a gain of 27% over Baum and Shokrollahi's method.

Example 4.2.1. *An example of the multiplication in $\mathbb{F}_{4^4}/\mathbb{F}_4$ whose basis is determined by:*

$$\mathcal{B}_Q := (Ev_Q(f_1), ..., Ev_Q(f_8)) = (b^{179}, b^{29}, b^{19}, b^{211})$$

where b is primitive root of the irreducible polyomial $Q(x) = x^4 + x^3 + \omega x^2 + \omega x + \omega$ in $\mathbb{F}_4[x]$.

In this basis, let us consider two elements in $\mathbb{F}_{4^4}$ *over* $\mathbb{F}_4$*:*

$$x = (1, \omega^2, 0, \omega)_{\mathcal{B}_Q} = 1 \cdot b^{179} + \omega^2 \cdot b^{29} + 0 \cdot b^{19} + \omega \cdot b^{211}$$

and

$$y = (1, \omega, \omega^2, 0)_{\mathcal{B}_Q} = 1 \cdot b^{179} + \omega \cdot b^{29} + \omega^2 \cdot b^{19} + 0 \cdot b^{211}.$$

We now have the concrete computations at each step of Algorithm 12,
-Step 1:

$$\bar{x} := T_{2D} \cdot \begin{pmatrix} 1 \\ \omega^2 \\ 0 \\ \omega \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \omega^2 \\ 1 \\ \omega \\ \omega^2 \\ 1 \\ \omega^2 \end{pmatrix} \text{ and } \bar{y} := T_{2D} \cdot \begin{pmatrix} 1 \\ \omega \\ \omega^2 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ \omega^2 \\ 1 \\ 0 \\ \omega \\ \omega \\ \omega^2 \\ \omega \end{pmatrix}.$$

- Step 2:

$$u := \bar{x} \odot \bar{y} = \begin{pmatrix} 0 \\ 0 \\ \omega^2 \\ 0 \\ \omega^2 \\ 1 \\ \omega^2 \\ 1 \end{pmatrix}.$$

- Step 3, 4:

$$z := T_{2D,4}^{-1} \cdot u = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 1 \end{pmatrix}.$$

Hence, we obtain the product of x and y:

$$z = (0, 0, 1, 1)_{\mathcal{B}_Q} = 0 \cdot b^{179} + 0 \cdot b^{29} + 1 \cdot b^{19} + 1 \cdot b^{211}.$$

In Section A.3 of the Appendix, we present an implementation in Magma program in order to test a particular example of the multiplication of two elements in $\mathbb{F}_{4^4}/\mathbb{F}_4$ using the optimized CCMA as mentioned above.

At present, we have submitted our results related to the computational approach in the scalar complexity optimization strategy (Algorithm 14) based on criteria on the bound of $N_{zero}(T_D)$, and the proposition confirmed that it is unnecessary to use the permutation of evaluation points in CCMA for our strategy (Proposition 4.1.3). In this submission, we improve the results published in [BBD19].

4.2.3 A comparison of different methods

In this subsection, we are interested in the bilinear, the scalar and the total complexity of the multiplication algorithms in $\mathbb{F}_{4^4}$. We summarize the complexity of the methods using *Chudnovsky algorithm* and the method of well-known *non-Chudnovsky*

algorithms such as Karatsuba, Toom-Cook, Schönhage-Strassen which were investigated in Section 1.2 of Chapter 1.

CCMA-based methods. For the constructions of using elliptic CCMA in $\mathbb{F}_{4^4}/\mathbb{F}_4$, we have known that their optimal bilinear complexity is

$$m_q^b(n) = \mu_{q,sym}^b = 2n = 8.$$

For the construction of Baum and Shokrollahi in Section 3.2.1, the *number of additions* is $(2 \times 14) + 23 = 51$. For the kernel-type construction in Section 3.3.3, the *number of additions* is $(2 \times 16) + 19 = 51$.

The number of scalar multiplications of the construction of Baum and Shokrollahi and of the kernel-type construction in Section 3.3.3 calculated respectively by the formula (4.1):

$$N_s = 3n(2n + g - 1) - \left(2N_{zero}(T_D) + N_{zero}(CT_{2D}^{-1})\right)$$

and the formula (4.3):

$$N_s = 3n(2n + g - 1) - \left(2N_{zero}(T_D) + N_{zero}(T_{2D,n}^{-1})\right)$$

give us the same results $N_s = 71$. There is a random coincidence of the same values in both constructions.

If we use a normal basis $\mathcal{B}_Q^n = \{b, b^4, b^{16}, b^{64}\}$ of the residue class field F_D, where b be the primitive root of the normal polynomial $Q(x) = x^4 + x^3 + \omega x^2 + \omega x + \omega \in \mathbb{F}_4[x]$, then the number of additions and the number of scalar multiplications of the kernel-type construction of CCMA are 58 and 78, respectively. The total number of operations is 144.

For our construction, we can calculate the number of additions used in Algorithm 12. In particular, Step 1 of the algorithm needs 16 additions to compute $T_{D,max} \cdot (x_1, ..., x_n)^t$ and $T_{D,max} \cdot (y_1, ..., y_n)^t$. The remaining number of additions of the algorithm which is calculated in the matrix $T_{2D,4}^{-1} \cdot (u_1, ..., u_8)^t$ (where $u = (u_i)_{i=1}^8$ is the Hadamard product in Step 2 of the algorithm) is 16. In conclusion, we need 8 bilinear multiplications, 52 scalar multiplications and 32 additions in our construction of CCMA in $\mathbb{F}_{4^4}/\mathbb{F}_4$.

Non-CCMA methods. In order to multiply two elements $x, y \in \mathbb{F}_{4^4}/\mathbb{F}_4$ by using a non-Chudnovsky algorithm, we remind that we proceed the multiplication of their representation polynomials and then obtained the product polynomial by the modular reduction in $\mathbb{F}_4[X]/\langle Q(X)\rangle$ where $Q(X)$ is a degree 4 irreducible polynomial over $\mathbb{F}_4$.

Given $x, y \in \mathbb{F}_{4^4} \cong \mathbb{F}_4(\beta) = \mathbb{F}_4[X]/\langle Q(X)\rangle$ with β a primitive root of the irreducible polynomial $Q(X)$ in $\mathbb{F}_4[X]$, we write them in the canonical basis polynomial representation:

$$x = (a_0, a_1, a_2, a_3)_{\mathcal{B}_Q^c} \equiv f(\beta) := a_0 + a_1\beta + a_2\beta^2 + a_3\beta^3,$$

$$y = (b_0, b_1, b_2, b_3)_{\mathcal{B}_Q^c} \equiv g(\beta) := b_0 + b_1\beta + b_2\beta^2 + b_3\beta^3.$$

where $a_i, b_i \in \mathbb{F}_4$, for $i = 0, ..., 3$ and $\mathcal{B}_Q^c = \{1, \beta, \beta^2, \beta^3\}$ is a canonical basis of $\mathbb{F}_{4^4}/\mathbb{F}_4$.

We first compute $h(X) = f(X) \cdot g(X)$ by using one of algorithms such as Karatsuba, Toom-Cook, Schönhage-Strassen. In the case of field extension over $\mathbb{F}_4$ of size

$n = 4$, we prefer to choose the method of Karatsuba since in Chapter 1, it is known that for $q \geq 4$, Toom-Cook method is a generalization of Karatsuba'one and Algorithm 5, 6 of Schönhage-Strassen also re-use the Karatsuba algorithm. The Karatsuba's algorithm uses $n^{\log_2 3} = 4^{\log_2 3} = 9$ bilinear multiplications and $6n^{\log_2 3} - 8n + 2 = 24$ additions (see Sect. 1.2.1).

In the second step, we do the modular reduction of $h(X)$ in $\mathbb{F}_4[X]/\langle Q(X)\rangle$ by using one of the most known used modular reduction algorithms for example, Barrett reduction algorithm and Montgomery's one (see Sect. 1.2.2). Here we choose the Barrett's algorithm (Algorithm 7) which uses *two multiplications* of two polynomials in $\mathbb{F}_4[X]$ of degree 4 and one degree 4-polynomial subtraction.

In other words, we need $2n^{\log_2 3} = 18$ bilinear multiplications, $2(6n^{\log_2 3} - 8n + 2) + n = 52$ additions to compute the modular reduction. We conclude that in order to multiply $x, y \in \mathbb{F}_{4^4}$ the number of bilinear multiplications and additions used in two steps are $3n^{\log_2 3} = 27$ and $3(6n^{\log_2 3} - 8n + 2) + n = 76$, respectively.

Therefore, the total cost of the multiplication in $\mathbb{F}_{4^4}/\mathbb{F}_4$ by using a *non-Chudnovsky algorithm* (in our consideration, the Karatsuba's algorithm for the polynomial multiplication and Barrett's modular reduction algorithm are applied) is $21n^{\log_2 3} - 23n + 6 = 103$ operations.

Now, we give Table 4.1 to compare the complexity of different methods:

Method \ Complexity	$m_4^b(4)$	$m_4^s(4)$	$a_4(4)$	$M_4(4)$
Polynomial basis multiplication (e.g. Karatsuba)	27	–	76	103
Baum-Shokrollahi's original construction of CCMA	8	71	51	130
Kernel-type construction of CCMA (Sect. 3.3.3)				
Kernel-type construction of CCMA$^{(*)}$	8	78	58	144
Our construction	**8**	**52**	**32**	**92**

TABLE 4.1: Complexity of the different methods for the multiplication in $\mathbb{F}_{256}$ over $\mathbb{F}_4$.

In Table 4.1, we use $(*)$ to indicate the kernel-type construction of CCMA given in Algorithm 12 with respect of the choice of a fixed normal basis $\mathcal{B}_Q^n := \{b, b^4, b^{16}, b^{64}\}$ of the residue class field F_Q, where b be a primitive root of a normal polynomial $Q(x) = x^4 + x^3 + \omega x^2 + \omega x + \omega$ associated to the place Q.

We can conclude that the scalar complexity as well as the total complexity of CCMA using our proposed construction is better than other methods in case study $\mathbb{F}_{256}/\mathbb{F}_4$.

4.3 Further developments and future work

In this section, we discuss the perspectives as well as open questions that can be studied in further works.

For the problem of optimizing scalar complexity of CCMA of type (3.1), an open question is how to choose appropriate geometrical objects such as the algebraic curve $\mathcal{C}$ and place Q which are associated to the function field $F/\mathcal{C} \cong \mathbb{F}_q[x]/\langle Q\rangle$ as well as the divisor D in order that we can figure out the matrix of the evaluation map on the points of the curve such that the parameter N_s of the algorithm is minimal. In other words, instead of taking into consideration these geometrical objects

of CCMA being fixed as in the proposed optimization strategy in Section 4.1.2, we change and select good ones in order to minimize N_s of the algorithm. It is noted that this parameter is also affected by the set of rational points (of degree 1) of the curve $\mathcal{C}$ when one uses them in evaluating the basis vector functions of the Riemann-Roch space $\mathcal{L}(2D)$ in the algorithm.

We recall that we could give the upper bound of the number of zeros appearing in matrix T_D:

$$N_{zero}(T_D) \leq n \cdot \deg D \qquad (4.8)$$

based on the observation related to the algebraic geometry code. Remark 4.1.2 suggests us a new direction to minimize the scalar complexity of the kernel-type construction of CCMA. Besides, in effort of finding a good upper bound of the scalar complexity of CCMA, the question is whether one can discover a bound on $N_{zero}(T^{-1}_{2D,n})$ by thinking of a certain algebraic code. Exploring the structure, especially the *sparsity* of $T^{-1}_{2D,n}$ in the algorithm is a theoretical difficulty at the moment.

The problem of optimizing the scalar complexity of the symmetric CCMA using the places of higher degree with derivated evaluations has not yet considered. For the *asymmetric* version of the algorithm, the problem of the optimization of the scalar complexity might be more complicated.

Going back to the definition of scalar multiplication in $\mathbb{F}_{q^n}$ (see Section 1.1.1) we have considered the multiplication of an element $a \in \mathbb{F}_q$ by the unit element $c = 1 \in F_q$ as a normal scalar multiplication operation. In practice there is no cost for the multiplication of an element by a unit in the finite field. As we can see, there are still many ways to optimize the scalar complexity of CCMA.

Appendix A

Magma source code

A.1 Computational verification program of the kernel-type construction of CCMA

A.1.1 Example 3.3.3

```
//%%%%%%%%%%%%%%%%%%%% PRE-COMPUTING FUNCTIONS %%%%%%%%%%%%%%%%%%%%
//Count the number of zeros in k rows et l column of a matrix A
function CountZeros(A, k, l)
CounterZ:=0;
for i:=1 to k do
    for j:=1 to l do
        if A[i,j] eq 0 then CounterZ:=CounterZ + 1;
        end if;
    end for;
end for;
return(CounterZ);
end function;
//Create matrix of the evaluation map of basis matrix B of m vectors at n
//points in P
function MatrixSecondEval(B,P,m,n)
ST:=[];
for j:=1 to n do
    for i:=1 to m do
        ST:=Append(ST, Evaluate(B[i,1], P[j]));
    end for;
end for;
T:=Matrix(n,m, ST);
return(T);
end function;
//%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
n:=4;g:=1;q:=4;
F4<a>:= GF(4);
Kx<x>:= FunctionField(F4);
Kxy<y>:= PolynomialRing(Kx);

f:= y^2 + y - x^3 -1;
F<c> := FunctionField(f);
LP:=Places(F,1);
eLP:=SetToSequence({LP[1],LP[2],LP[3],LP[8],LP[9],LP[4],LP[5],LP[6]});
```

```
QQ := x^4 + x^3 + a*x^2 + a*x + a;
Q := Decomposition(F,Zeros(Kx!QQ)[1])[1];
K<b> := ResidueClassField(Q);
"degree of Q is ", Degree(Q);

DD:= x^2+x+a;
D:= Decomposition(F,Zeros(Kx!DD)[1])[1];
D:=1*D;
"dim L(D) is ", Dimension(D);

"D-Q is special? ",IsSpecial(D-Q);
print"Dimension of D-Q:", Dimension(D-Q);

// Construction of the Riemann space
LD, h :=RiemannRochSpace(D);
L2D, h2 :=RiemannRochSpace(2*D);
BaseLD:=[(h(v))@@h2 : v in Basis(LD)];
Base:=ExtendBasis(BaseLD,L2D);

L2D:=[];
for i in[1..2*n+g-1] do
    L2D:=Append(L2D, h2(Base[i]));
end for;

ML2D:=Matrix(2*n+g-1,1,L2D);

// Construction of E : E=Evalf(Q)
L:=[];
for i in [1..n] do
    L:=Append(L,ElementToSequence(Evaluate(L2D[i],Q)));
end for;
E:=Transpose(Matrix(L));

M:=Matrix(F,E^-1);
Ev:=Matrix(1,n,[L2D[i] : i in [1..n]]);

// Construction of L(2D) with the required properties
EL2D:=Matrix(2*n+g-1,1,[Evaluate(L2D[i],Q) : i in [1..2*n+g-1]]);
BEL2D:=Matrix(F4,2*n+g-1,n,[ElementToSequence(EL2D[i][1]) : i in [1..2*n+g-1]]);
MM:=Parent(ZeroMatrix(F,n+g-1,2*n+g-1))!Matrix(Basis(NullSpace(BEL2D)))*ML2D;

for i in [1..n] do
    L2D[i]:=Transpose(Ev*M)[i,1];
end for;

// rows of ML2D form a basis of L(2D)
ML2D:=Matrix(2*n+g-1,1,[L2D[i] : i in [1..n]] cat [MM[i,1] : i in [1..n+g-1]]);

print"Basis of L(2D) with the required properties:"; ML2D;

// Construction of T and T^-1
```

```
T:=MatrixSecondEval(ML2D,eLP, 2*n+g-1,2*n+g-1);
TI:=T^-1;
print"Matrix T used in CCMA:"; T;
print"Matrix T^-1 used in CCMA:"; TI;

//%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
//ALGORITHM FOR THE MULTIPLICATION IN $F_{4^4}/F_4$
//%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

x:=b^244; y:=b^72; //  --> Find the product of x and y?

// x,y in the standard basis
X:=Matrix(n,1,ElementToSequence(x)); //--> X=(0,0,1,a)
Y:=Matrix(n,1,ElementToSequence(y)); //--> Y=(a,a^2,0,0)

fX:=VerticalJoin(X,ZeroMatrix(F4,n+g-1,1));
fY:=VerticalJoin(Y,ZeroMatrix(F4,n+g-1,1));

// u = T(fX)@T(fY) //Hadamard product
u:= Matrix(2*n+g-1,1,[(T*fX)[i][1]*(T*fY)[i][1] : i in [1..2*n+g-1]]);
P:=VerticalJoin(HorizontalJoin(ScalarMatrix(F4,n,1),ZeroMatrix(F4,n,n+g-1)),
                                              ZeroMatrix(F4,n+g-1,2*n+g-1));

C:=Matrix(n,1,[(P*TI*u)[i][1] : i in [1..n]]);

print"The result of product X and Y is:", C;
//output C= (a,a,0,1) in the standard basis

z:=a*1+a*b+0*b^2+1*b^3;
print"Product of x=b^244 and y=b^72 in F_{4^4}/F_4 is", z;
```

A.2 Setup program for optimizing the scalar multiplication of CCMA in $\mathbb{F}_{256}/\mathbb{F}_4$

```
//%%%%%%%%%%%%%%%%%%%% PRE-COMPUTING FUNCTIONS %%%%%%%%%%%%%%%%%%%%
//Count the number of zeros in k rows et l column of a matrix A
function CountZeros(A, k, l)
CounterZ:=0;
for i:=1 to k do
    for j:=1 to l do
        if A[i,j] eq 0 then CounterZ:=CounterZ + 1;
        end if;
    end for;
end for;
return(CounterZ);
end function;
//Create matrix of the evaluation map of basis matrix B of m vectors
//at n points in P
function MatrixSecondEval(B,P,m,n)
ST:=[];
```

```
for j:=1 to n do
    for i:=1 to m do
        ST:=Append(ST, Evaluate(B[i,1], P[j]));
    end for;
end for;
T:=Matrix(n,m, ST);
return(T);
end function;
//Number of scalar multiplications in CCMA with input data = {Function
//field F, List of point LP, degree n, genus g, transformation matrix TM
//in GL(n,Fq); Matrix E of 1st Evaluation map, Matrix M of basis vector
//of L(D); Matrix BK of basis vector of complementary space of L(D) in L(2D)}
function NumScaMult(F,LP,TM,E,M,BK)
    nBasisLD:= Matrix(F,TM)*E^-1*M;
    nML2D:= Matrix(2*n+g-1,1,[nBasisLD[i,1] : i in [1..n]] cat [BK[i,1] :
                                                        i in [1..n+g-1]]);
    nT:=MatrixSecondEval(nML2D,LP, 2*n+g-1,2*n+g-1);
    nCounterT:=CountZeros(nT,2*n+g-1,n);
    nCounterTI:=CountZeros(nT^-1,n,2*n+g-1);
    Ns:=6*n^2-(2*nCounterT+nCounterTI);
    return(Ns);
end function;
//%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
n:=4;g:=1;q:=4;
F4<a>:= GF(4);

Kx<x>:= FunctionField(F4);
Kxy<y>:= PolynomialRing(Kx);

f:= y^2 + y - x^3 -1;
F<c> := FunctionField(f);

QQ := x^4 + x^3 + a*x^2 + a*x + a;
Q := Decomposition(F,Zeros(Kx!QQ)[1])[1];
"degree of Q is ", Degree(Q);

DD:= x^2+x+a;
D:= Decomposition(F,Zeros(Kx!DD)[1])[1];
D:=1*D;
//"dim L(D) is ", Dimension(D);
Degree(D);

"D-Q is special? ",IsSpecial(D-Q);
Dimension(D-Q);

// Construction of the residue class field and the degree one places
R<b> := ResidueClassField(Q);
LP:=Places(F,1);
eLP:=SetToSequence({LP[1],LP[2],LP[3],LP[8],LP[9],LP[4],LP[5],LP[6]});

// Construction of the Riemann space
```

```
LD, h :=RiemannRochSpace(D);
L2D, h2 :=RiemannRochSpace(2*D);
BaseLD:=[(h(v))@@h2 : v in Basis(LD)];
Base:=ExtendBasis(BaseLD,L2D);

L2D:=[];
for i in[1..2*n+g-1] do L2D:=Append(L2D, h2(Base[i])); end for;
ML2D:=Matrix(2*n+g-1,1,L2D);

// Construction of E : E=Evalf(Q)
L:=[];
for i in [1..n] do
    L:=Append(L,ElementToSequence(Evaluate(L2D[i],Q)));
end for;
E:=Transpose(Matrix(L));
M:=Matrix(F,E^-1);
Ev:=Matrix(1,n,[L2D[i] : i in [1..n]]);
// Construction of L(2D) with the required properties
EL2D:=Matrix(2*n+g-1,1,[Evaluate(L2D[i],Q) : i in [1..2*n+g-1]]);
BEL2D:=Matrix(F4,2*n+g-1,n,[ElementToSequence(EL2D[i][1]) : i
                                                    in [1..2*n+g-1]]);
BK:=Parent(ZeroMatrix(F,n+g-1,2*n+g-1))!Matrix(Basis(NullSpace(BEL2D)))*ML2D;

for i in [1..n] do L2D[i]:=Transpose(Ev*M)[i,1]; end for;
// rows of ML2D form a basis of L(2D)
ML2D:=Matrix(2*n+g-1,1,[L2D[i] : i in [1..n]] cat [BK[i,1] : i
                                                    in [1..n+g-1]]);
MLD:=Matrix(n,1,[L2D[i] : i in [1..n]]);

EL2D:=Matrix(2*n+g-1,1, [Evaluate(ML2D[i,1],Q) : i in [1..2*n+g-1]]);
BELD:=Matrix(F4,n,n, [ElementToSequence(EL2D[i][1]) : i in [1..n]]);

//%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
//%%%%%%%%    OPTIMIZATION PROCESSING     %%%%%%%%%%
//%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
G:=GL(4,F4);
//for k:=1 to 252 do
//---Replace k by 1 until 252 (252 is the number of conjugacy class of GL(4,F4))

rep:=Classes(G)[k][3];
C:=Class(G,rep);

Ns1:=NumScaMult(F,eLP,C[1],BELD,MLD,BK);
Ns2:=NumScaMult(F,eLP,C[2],BELD,MLD,BK);
i:=0;
if #C mod 2 eq 0 then
    if Ns1 gt Ns2 then
       minNs:= Ns2;
    else
       minNs:=Ns1;
    end if;
```

```
    i:=3;
else
    minNs:=Ns1;
    i:=2;
end if;

imin:=i;
while i lt #C do
    Nsi:=NumScaMult(F,eLP,C[i],BELD,MLD,BK);
    Nsi1:=NumScaMult(F,eLP,C[i+1],BELD,MLD,BK);
    if Nsi gt Nsi1 then
        if Nsi1 lt minNs then
            minNs:=Nsi1; imin:= i+1;
        end if;
    else
        if Nsi lt minNs then
            minNs:=Nsi; imin:= i;
        end if;
    end if;
    i:=i+2;
end while;
//end for;
print"Number of scalar multiplications in CCMA:", minNs;
print"Position of matrix in conjugacy class C of GL(4,F4) used to transform
the basis of L(D):", imin;

mBasisLD:= Matrix(F,C[imin])*BELD^-1*MLD;
mML2D:= Matrix(2*n+g-1,1,[mBasisLD[i,1] : i in [1..n]] cat [BK[i,1] :
                                                    i in [1..n+g-1]]);
print"The good basis of L(2D) used in CCMA:", mML2D;
```

A.3 Testing program for an example of the improved multiplication in $\mathbb{F}_{256}/\mathbb{F}_4$

```
// %%%%%%%%%%%%%%%%%%%%%%%%%%%
// ALGORITHM FOR THE MULTIPLICATION
// %%%%%%%%%%%%%%%%%%%%%%%%%%%
//SETUP THE INPUTS
n:=4;g:=1;q:=4;
F4<a>:= GF(4);
G:=GL(4,F4);
Kx<x>:= FunctionField(F4);
Kxy<y>:= PolynomialRing(Kx);
f:=y^2 + y - x^3 -1;
F<c> := FunctionField(f);

// Construction of the place Q and divisor D
QQ := x^4 + x^3 + a*x^2 + a*x + a;
Q := Decomposition(F,Zeros(Kx!QQ)[1])[1];
"degree of Q is ", Degree(Q);
```

```
DD:= x^2 + x + a;
D:= Decomposition(F,Zeros(Kx!DD)[1])[1];
D:=1*D;
"D-Q is special? ",IsSpecial(D-Q);
"dim L(D) is ", Dimension(D);

// Construction of the residue class field and the degree one places
R<b> := ResidueClassField(Q);
LP:=Places(F,1);
eLP:=SetToSequence({LP[1],LP[2],LP[3],LP[8],LP[9],LP[4],LP[5],LP[6]});

//Optimized basis of L(2D) given by Algorithm 16
f1:= (c+a*x + a^2)/(x^2 + x + a);
f2:= (c+ a^2*x + a)/(x^2 + x + a);
f3:= (a*x^2 + a^2*x)/(x^2 + x + a);
f4:= a*c/(x^2 + x + a);
f5:= ((a*x^2 + a*x)*c + a^2*x^4 +a*x^3 + x^2 + x + a)/(x^4 + x^2 + a^2);
f6:= (a^2*x^2*c + a*x^4 + a*x^3 + x^2 + a*x)/(x^4 + x^2 + a^2);
f7:= ((x^2 + a^2*x)*c + a*x^4 + a*x^2)/(x^4 + x^2 + a^2);
f8:= ((a*x+a)*c + a*x^4)/(x^4 + x^2 + a^2);

BasisL2D:=Matrix(2*n+g-1,1,[f1,f2,f3,f4,f5,f6,f7,f8]);
ST:=[];
for j:=1 to 2*n+g-1 do
    for i:=1 to 2*n+g-1 do
        ST:=Append(ST, Evaluate(BasisL2D[i,1], eLP[j]));
    end for;
end for;
T:=Matrix(2*n+g-1, ST);
TI:=T^-1;

P:=VerticalJoin(HorizontalJoin(ScalarMatrix(F4,n,1),
            ZeroMatrix(F4,n,n+g-1)),ZeroMatrix(F4,n+g-1,2*n+g-1));

//INPUT X & Y
X:=Matrix(F4,4,1,[1,a^2,0,a]); // example
Y:=Matrix(F4,4,1,[1,a,a^2,0]); // example
fx:=VerticalJoin(X,ZeroMatrix(F4,n+g-1,1));
fy:=VerticalJoin(Y,ZeroMatrix(F4,n+g-1,1));

// u = T(fx)@T(fy) //Hadamard product
u:= Matrix(2*n+g-1,1,[(T*fx)[i][1]*(T*fy)[i][1] : i in [1..2*n+g-1]]);

// OUTPUT Z=X*Y
Z:=Matrix(n,1,[(P*TI*u)[i][1] : i in [1..n]]);
"Product of X and Y is", Z;

// %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
// CHECKING THE RESULT OF MULTIPLICATION
// %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
BFqn:=Matrix(n,1, [Evaluate(BasisL2D[i],Q) : i in [1..n]]);
"The basis of F_256/F_4 is", BFqn;

XY:=(1*b^179 + a^2*b^29 + 0*b^19 + a*b^211)*(1*b^179 +
                                        a*b^29 + a^2*b^19 + 0*b^211);
Z:=0*b^179 + 0*b^29 + 1*b^19 + 1*b^211;
"Is XY equals Z ?", XY eq Z; //b^11
```

Bibliography

[Arn06] Nicolas Arnaud. "Évaluations dérivés, multiplication dans les corps finis et codes correcteurs". PhD thesis. Institut de Mathématiques de Luminy: Université de la Méditerranée, 2006.

[Ati+17] Kevin Atighehchi, Stéphane Ballet, Alexis Bonnecaze, and Robert Rolland. "Arithmetic in finite fields based on the Chudnovsky-Chudnovsky multiplication algorithm". In: *Mathematics of Computation* 86.308 (2017), pp. 2975–3000.

[BIP04] Jean-Claude Bajard, Laurent Imbert, and Thomas Plantard. "Modular Number Systems: Beyond the Mersenne Family". In: *Selected Areas in Cryptography*. 2004.

[Bal99] Stéphane Ballet. "Curves with Many Points and Multiplication Complexity in Any Extension of $\mathbb{F}_q$". In: *Finite Fields and Their Applications* 5 (1999), pp. 364–377.

[Bal02] Stéphane Ballet. "Quasi-optimal Algorithms for Multiplication in the Extensions of $\mathbb{F}_{16}$ of degree 13, 14, and 15". In: *Journal of Pure and Applied Algebra* 171 (2002), pp. 149–164.

[Bal+17] Stéphane Ballet, Nicolas Baudru, Alexis Bonnecaze, and Mila Tukumuli. "On the Construction of the Asymmetric Chudnovsky Multiplication Algorithm in Finite Fields Without Derivated Evaluation". In: *Comptes Rendus de l'Académie des Sciences, Série I* 355 (2017), pp. 729–733.

[Bal+] Stéphane Ballet, Nicolas Baudru, Alexis Bonnecaze, and Mila Tukumuli. "On the Effective Construction of Asymmetric Chudnovsky Multiplication Algorithms in Finite Fields Without Derivated Evaluation". In: *ArXiv e-prints* (). arXiv: 1611.02883v1 [math.AG].

[BBD19] Stéphane Ballet, Alexis Bonnecaze, and Thanh-Hung Dang. "On the Scalar Complexity of Chudnovsky^2 Multiplication Algorithm in Finite Fields". In: *Algebraic Informatics, CAI 2019, Lecture Notes in Computer Science, vol 11545*. Springer Cham, 2019, pp. 64–75. URL: https://doi.org/10.1007/978-3-030-21363-3_6.

[BL06] Stéphane Ballet and Dominique Le Brigand. "On the existence of non-special divisors of degree g and $g-1$ in algebraic function fields over $\mathbb{F}_q$". In: *Journal of Number Theory* 116 (2006), pp. 293–310.

[BP11] Stéphane Ballet and Julia Pieltant. "On the tensor rank of multiplication in any extension of $\mathbb{F}_2$". In: *J. Complexity* 27.2 (2011), pp. 230–245. DOI: 10.1016/j.jco.2011.01.008. URL: https://doi.org/10.1016/j.jco.2011.01.008.

[BR04] Stéphane Ballet and Robert Rolland. "Multiplication Algorithm in a Finite Field and Tensor Rank of the Multiplication". In: *Journal of Algebra* 272.1 (2004), pp. 173–185.

[BR05] Stéphane Ballet and Robert Rolland. "On the bilinear complexity of the multiplication in finite fields". In: *Proceedings of the Conference Arithmetic, Geometry and Coding Theory (AGCT 2003)*. Vol. 11. Société Mathématique de France, sér. Séminaires et Congrès, 2005, pp. 179–188.

[Bal+19] Stéphane Ballet et al. *On the tensor rank of multiplication in finite extensions of finite fields and related issues in algebraic geometry*. 2019. arXiv: 1906.07456 [math.AG].

[Bar87] Paul Barrett. "Implementing the Rivest Shamir and Adleman Public Key Encryption Algorithm on a Standard Digital Signal Processor". In: *Advances in Cryptology — CRYPTO' 86*. Ed. by Andrew M. Odlyzko. Berlin, Heidelberg: Springer Berlin Heidelberg, 1987, pp. 311–323. ISBN: 978-3-540-47721-1.

[BS91] Ulrich Baum and Amin Shokrollahi. "An optimal algorithm for multiplication in $\mathbb{F}_{256}/\mathbb{F}_4$". In: *Applicable Algebra in Engineering, Communication and Computing* 2.1 (1991), pp. 15–20.

[Ber09] Daniel J. Bernstein. "Batch Binary Edwards". In: *Advances in Cryptology - CRYPTO 2009, 29th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2009. Proceedings*. 2009, pp. 317–336. DOI: 10.1007/978-3-642-03356-8_19. URL: https://doi.org/10.1007/978-3-642-03356-8%5C_19.

[BCS97] Peter Bürgisser, Michael Clausen, and Amin Shokrollahi. *Algebraic Complexity Theory*. Springer, 1997.

[CK91] David G. Cantor and Erich Kaltofen. "On Fast Multiplication of Polynomials over Arbitrary Algebras". In: *Acta Inf.* 28.7 (1991), pp. 693–701. DOI: 10.1007/BF01178683. URL: https://doi.org/10.1007/BF01178683.

[Cas+12] Ignacio Cascudo, Ronald Cramer, Chaoping Xing, and An Yang. "Asymptotic bound for Multiplication complexity in the extensions of small finite fields". In: *IEEE Transactions on Information Theory* 58.7 (2012), pp. 4930–4935.

[CÖ10] Murat Cenk and Ferruh Özbudak. "On multiplication in finite fields". In: *Journal of Complexity* (2010), pp. 172–186.

[Cha06] Jean Chaumine. "On the bilinear complexity of multiplication in small finite fields". In: *Comptes Rendus de l'Académie des Sciences, Série I* 343 (2006), pp. 265–266.

[CC88] D. V. Chudnovsky and G. V. Chudnovsky. "Algebraic complexities and algebraic curves over finite fields". In: *J. Complexity* 4.4 (1988), pp. 285–316. DOI: 10.1016/0885-064X(88)90012-X. URL: https://doi.org/10.1016/0885-064X(88)90012-X.

[CT65] James Cooley and John Tukey. "An Algorithm for the Machine Calculation of Complex Fourier Series". In: *Mathematics of Computation* 19.90 (1965), pp. 297–301.

[CL09] Jean-Marc Couveignes and Reynald Lercier. "Elliptic periods for finite fields". In: *Finite Fields and Their Applications* 15.1 (2009), pp. 1–22.

[Dhe03] Jean-François Dhem. "Efficient Modular Reduction Algorithm in $\mathbb{F}_q[x]$ and Its Application to "Left to Right" Modular Multiplication in $\mathbb{F}_2[x]$". In: *Cryptographic Hardware and Embedded Systems - CHES 2003, 5th International Workshop, Cologne, Germany, September 8-10, 2003, Proceedings*. 2003, pp. 203–213. DOI: 10.1007/978-3-540-45238-6_17. URL: https://doi.org/10.1007/978-3-540-45238-6%5C_17.

[Dum+15] Jean-Guillaume Dumas, Jean-Louis Roch, Eric Tannier, and Sébastien Varrette. *Foundations of Coding - Compression, Encryption, Error Correction*. Wiley, 2015. ISBN: 978-1-118-88144-6. URL: http://eu.wiley.com/WileyCDA/WileyTitle/productCd-1118881443.html.

[EG12] Nadia El Mrabet and Nicolas Gama. "Efficient Multiplication over Extension Fields". In: *Arithmetic of Finite Fields*. Ed. by Ferruh Özbudak and Francisco Rodríguez-Henríquez. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 136–151.

[EN09] Nadia El Mrabet and Christophe Negre. "Finite Field Multiplication Combining AMNS and DFT Approach for Pairing Cryptography". In: *Information Security and Privacy*. Ed. by Colin Boyd and Juan González Nieto. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 422–436.

[Für07] Martin Fürer. "Faster integer multiplication". In: *Proceedings of the 39th Annual ACM Symposium on Theory of Computing, San Diego, California, USA, June 11-13, 2007*. 2007, pp. 57–66. DOI: 10.1145/1250790.1250800. URL: http://doi.acm.org/10.1145/1250790.1250800.

[GS13] S. B. Gashkov and I. S. Sergeev. "Complexity of computation in finite fields". In: *Journal of Mathematical Sciences* 191.5 (June 2013), pp. 661–685. ISSN: 1573-8795. DOI: 10.1007/s10958-013-1350-5. URL: https://doi.org/10.1007/s10958-013-1350-5.

[GG13] Joachim von zur Gathen and Jürgen Gerhard. *Modern Computer Algebra (3. ed.)* Cambridge University Press, 2013. ISBN: 978-1-107-03903-2.

[Gop81] V.D. Goppa. "Codes on algebraic curves". In: *Dokl. Akad. Nauk SSSR* 259 (1981), pp. 1289–1290. URL: http://www.ams.org/mathscinet-getitem?mr=0628795.

[HHL17] David Harvey, Joris van der Hoeven, and Grégoire Lecerf. "Faster Polynomial Multiplication over Finite Fields". In: *J. ACM* 63.6 (2017), 52:1–52:23. DOI: 10.1145/3005344. URL: https://doi.org/10.1145/3005344.

[Hes02] Florian Hess. "Computing Riemann-Roch Spaces in Algebraic Function Fields and Related Topics". In: *J. Symb. Comput.* 33.4 (2002), pp. 425–445. DOI: 10.1006/jsco.2001.0513. URL: https://doi.org/10.1006/jsco.2001.0513.

[KO62] A. Karatsuba and Yu. Ofman. "Multiplication of Many-Digital Numbers by Automatic Computers". In: *Proceedings of USSR Academy of Sciences* 145.7 (1962), pp. 293–294.

[Kne+08] Miroslav Knezevic, Kazuo Sakiyama, Junfeng Fan, and Ingrid Verbauwhede. "Modular Reduction in $GF(2^n)$) without Pre-computational Phase". In: *Arithmetic of Finite Fields, 2nd International Workshop, WAIFI 2008, Siena, Italy, July 6-9, 2008, Proceedings*. 2008, pp. 77–87. DOI: 10.1007/978-3-540-69499-1_7. URL: https://doi.org/10.1007/978-3-540-69499-1%5C_7.

[KA98] Cetin K. Koc and Tolga Acar. "Montgomery Multiplication in GF(2k)". In: *Designs, Codes and Cryptography* 14.1 (Apr. 1998), pp. 57–69. ISSN: 1573-7586. DOI: 10.1023/A:1008208521515. URL: https://doi.org/10.1023/A:1008208521515.

[LN97] Rudolf Lidl and Harald Niederreiter. *Finite fields*. English. 2nd ed. Previous ed.: Reading, Mass. ; London : Addison Wesley, 1983. Cambridge ; New York : Cambridge University Press, 1997. ISBN: 0521392314. URL: http://site.ebrary.com/id/10450828.

[Mon85] Peter L. Montgomery. "Modular Multiplication without Trial Division". In: *Mathematics of Computation* 44.170 (1985), pp. 519–521.

[Mul+89] Ronald C. Mullin, I. M. Onyszchuk, Scott A. Vanstone, and R. M. Wilson. "Optimal normal bases in $GF(p^n)$". In: *Discrete Applied Mathematics* 22.2 (1989), pp. 149–161. DOI: 10.1016/0166-218X(88)90090-X. URL: https://doi.org/10.1016/0166-218X(88)90090-X.

[OM86] Jimmy K. Omura and James L. Massey. "Computational method and apparatus for finite field arithmetic". Patent 4587627 (US). May 1986.

[Pie12] Julia Pieltant. "Tours de corps de fonctions algébriques et rang de tenseur de la multiplication dans les corps finis". PhD thesis. Université d'Aix-Marseille, Institut de Mathématiques de Luminy, 2012.

[Ran12] Hugues Randriambololona. "Bilinear complexity of algebras and the Chudnovsky-Chudnovsky interpolation method". In: *Journal of Complexity* 28.4 (2012), pp. 489–517.

[RH00] A. Reyhani-Masoleh and M.A. Hasan. "On Efficient Normal Basis Multiplication". In: *Progress in Cryptology —INDOCRYPT 2000*. Ed. by Bimal Roy and Eiji Okamoto. Berlin, Heidelberg: Springer Berlin Heidelberg, 2000, pp. 213–224. ISBN: 978-3-540-44495-4.

[Sch77] Arnold Schönhage. "Schnelle Multiplikation von Polynomen über Körpern der Charakteristik 2". In: *Acta Inf.* 7 (1977), pp. 395–398. DOI: 10.1007/BF00289470. URL: https://doi.org/10.1007/BF00289470.

[Ser07] I. S. Sergeev. "On constructing circuits for transforming the polynomial and normal bases of finite fields from one to the other". In: *Discrete Mathematics and Applications* 17 (Jan. 2007), pp. 361–373. DOI: 10.1515/dma.2007.031.

[Sho92] Amin Shokhrollahi. "Optimal Algorithms for Multiplication in Certain Finite Fields using Algebraic Curves". In: *SIAM Journal on Computing* 21.6 (1992), pp. 1193–1198.

[STV92] Igor Shparlinski, Michael Tsfasman, and Serguei Vlăduţ. "Curves with Many Points and Multiplication in Finite Fields". In: *Coding Theory and Algebraic Geometry*. Ed. by H. Stichtenoth and M.A. Tsfasman. Lectures Notes in Mathematics 1518. Proceedings of AGCT-3 conference, June 17-21, 1991, Luminy. Berlin: Springer-Verlag, 1992, pp. 145–169.

[Sti93] Henning Stichtenoth. *Algebraic Function Fields and Codes*. Lectures Notes in Mathematics 314. Springer-Verlag, 1993.

[Sun04] Berk Sunar. "A Generalized Method for Constructing Subquadratic Complexity $GF(2^k)$ Multipliers". In: *IEEE Trans. Computers* 53.9 (2004), pp. 1097–1105. DOI: 10.1109/TC.2004.52. URL: https://doi.org/10.1109/TC.2004.52.

[Too63] Andrei L. Toom. "The complexity of a scheme of functional elements realizing the multiplication of integers". In: *Soviet Mathematics Doklady* 3 (1963), pp. 714–716. ISSN: 0197–6788.

[Win+80] S. Winograd, Society for Industrial, Applied Mathematics, Conference Board of the Mathematical Sciences, and National Science Foundation (Estats Units d'Amèrica). *Arithmetic Complexity of Computations*. CBMS-NSF Regional Conference Series in Applied Mathematics. Society for Industrial and Applied Mathematics, 1980. ISBN: 9780898711639. URL: https://books.google.fr/books?id=GU1NQJBcWIsC.