Optimized Frobenius and Cyclotomic Cubing for Enhanced Pairing Computation

Leila Ben Abdelghani[1], Nadia El Mrabet[2], Loubna Ghammam[3], and Lina Mortajine[3]

[1] *Laboratory of Analysis, Probability and Fractals, Faculty of Sciences*
[2]*Laboratory of Secure System and Architecture (SSA), Ecole des Mines de Saint Etienne*
[3]*ITK Engineering GmbH*

Abstract

Efficient implementation of a pairing-based cryptosystem relies on high-performance arithmetic in finite fields $\mathbb{F}_p$ and their extensions $\mathbb{F}_{p^k}$, where k is the embedding degree. A small embedding degree is crucial because part of the arithmetic for pairing computation occurs in $\mathbb{F}_{p^k}$ and includes operations such as squaring, multiplication, and Frobenius operations. In this paper, we present a fast and efficient method for computing the Frobenius endomorphism and its complexity. Additionally, we introduce an improvement in the efficiency of cyclotomic cubing operations for several pairing-friendly elliptic curves, which are essential for the calculation of Tate pairing and its derivatives.

Keywords— Optimal Ate Pairing Frobenius maps Kronecker products Finite fields Cyclotomic cubing.

1 Introduction

Bilinear pairing is a mathematical operation that maps a pair of points on an elliptic curve to an element of a finite field. Pairings have become a cornerstone in modern cryptography due to their unique properties (bilinearity, non-degeneracy, computability) and applications. Indeed, they enable the construction of advanced cryptographic protocols that were previously difficult to achieve. For instance, they are used, among others, in identity-based encryption [9], attribute-based encryption [19], short digital signatures [11], key exchange [22], broadcast encryption [10], deep package inspection over encrypted traffic [29, 12], and cryptocurrency [30]. However, the efficiency of cryptosystems that rely on bilinear pairings is largely dependent on the speed of pairing computation. Consequently, over the past decades, researchers have focused on optimizing this process, particularly by identifying pairing-friendly curves [18, 26]. For example, the BLS (Barreto-Lynn-Scott) curves are known for their simplicity and efficient implementation, making them popular for applications like digital signatures and identity-based encryption [30, 15, 20]. Additionally, studies have explored various types of pairings, including the widely-used Tate pairing and its variants (e.g., Optimal Ate Pairing) [31]. The Tate pairing computation relies on Miller's algorithm and a final exponentiation. Therefore, these components have also been a focus of optimization efforts [14, 2, 1].

To delve into the specifics of Miller's algorithm and final exponentiation, we consider E as an ordinary elliptic curve over the finite field $\mathbb{F}_p$, and let r be a large prime number that divides the order of the group of points $E(\mathbb{F}_p)$. The embedding degree k of E with respect to r and the prime number p is defined as the smallest integer k such that r divides $p^k - 1$. Miller's algorithm is used to compute an intermediate result, a function $f_{s,Q}(P)$ where P and Q are two points on the elliptic curve E, and s an integer related to the order of the points. This intermediate value, which is an element in the finite field $\mathbb{F}_{p^k}$, is raised to the power $\frac{p^k-1}{r}$ in order to ensure the unicity of the pairing. This final exponentiation involves raising the intermediate result to a large power, leading to complex calculations in the finite field $\mathbb{F}_{p^k}$. To secure against several attacks (e.g., ExTNS [6, 24, 4], subgroup attacks [7]), the size of the two finite fields $\mathbb{F}_p$ and $\mathbb{F}_{p^k}$ should follow the recommendations in [5]. Thus, the efficiency of pairing computation relies mainly on three factors:

- The size of $\mathbb{F}_p$;
- The size of $\mathbb{F}_{p^k}$, which depends on the embedding degree k of the curve;
- The arithmetic in $\mathbb{F}_{p^k}$, involving squaring, multiplication, and Frobenius evaluation.

Several studies have already been conducted on optimizing arithmetic operations in $\mathbb{F}_{p^k}$, and the search for suitable embedding degrees k.

In [5], the authors highlight the existence of additional families of elliptic curves with embedding degrees $k = 9$ and $k = 15$ which exhibit competitive performance when compared to the well-known BN curves and BLS curves with $k = 12$. However, these elliptic curves with odd embedding degrees face a significant drawback: the lack of fast squaring, known as cyclotomic squaring, within the cyclotomic subgroup. In [28], the authors provide the first explicit formula for performing fast cubing, in cyclotomic subgroup, referred to as cyclotomic cubing. This method is especially beneficial for elliptic curves with embedding degrees that are multiples of three. The proposed formulae utilize the cyclotomic structure to reduce the computational complexity of cubing operations, which are crucial in the final exponentiation step of pairing computations. However, this method remains less efficient compared to the simple square-and-multiply technique in the final exponentiation and necessitates further optimization. In [32], the authors demonstrate that using a ternary representation of the seed u, which is an integer used in the parameterization of the prime p and the order r of the elliptic curve, can lead to better performance in the final exponentiation in some particular cases of the seed u. To evaluate the cost of the final exponentiation for elliptic curves with embedding degrees divisible by three, they leverage the cyclotomic cubing method described in [28].

Additionally, to provide an accurate complexity of the final exponentiation step, it is essential to compute the number of operations required to perform the Frobenius morphism. However, computing the Frobenius morphism in various finite fields involves numerous algebraic operations, with the exact count depending on the specific characteristics of each field. This variability complicates the determination of operation counts and, consequently, the computation of pairing complexity. Therefore, developing a method to accurately determine the number of operations required for the Frobenius morphism is considered an important and valuable area of study.

Our contribution. In this paper, we present new methods to enhance the arithmetic in $\mathbb{F}_{p^k}$, leading to an easiest way to determine the complexity of the Frobenius evaluation as well as, a reduction in the operation counts needed to perform the final exponentiation. We also apply these new results to BLS curves with specific embedding degrees and provide a comparison with the state-of-the-art. More specifically, our main contributions are as follows:

- Frobenius computation in the finite field $\mathbb{F}_{p^k}$: we introduce a new method utilizing Kronecker product of matrices to perform more efficiently the Frobenius operation and simplifies the complexity analysis.
- Squaring: during the final exponentiation computation, squaring can sometimes be substituted with cubing, more specifically when the embedded degrees k is divisible by 3. This paper aims to provide an optimized approach for cyclotomic cubing, improving the results, in terms of operations counts, obtained in [28].

- Application of fast cubing optimization to the pairing based on the BLS curves for embedding degrees $k = 15$ and $k = 27$ cuves.

Organization of the paper. In section 2, we provide an overview of the necessary background information to comprehend this work. In section 3, we introduce a novel approach based on the Kronecker product to compute the Frobenius map. In section 4, we focus on the optimization of the cubing computation. In section 5, we demonstrate the application of the method explained in the previous section to pairing-based cryptography on BLS (Barreto-Lynn-Scott) curves and give a comparison of complexity.

Notations. In this paper, we use the following notations to define the operations in the finite field $\mathbb{F}_{p^k}$: M_k, S_k, A_k, F_k and $m_{k,c}$ denote multiplication, squaring, addition, Frobenius and multiplication by an element c in $\mathbb{F}_{p^k}$, respectively. To simplify the notations, we denote by M, S, A, and m_c multiplication, squaring, addition, and multiplication by an element c in $\mathbb{F}_p$, respectively.
In addition, we refer to C_{cyc_k} as the cyclotomic cubing and to I_{cyc_k} as an inversion in the cyclotomic subgroup of the finite field $\mathbb{F}_{p^k}$.

2 Background

This work is based on various mathematical notions, which we recall in this section to provide the necessary background to comprehend our results. We focus particularly on pairing-based cryptography and related notions such as pairings, Frobenius endomorphisms, and the Kronecker product.

2.1 Pairings

Let E be an elliptic curve defined over a finite field $\mathbb{F}_p$, where p is a large prime number. Let r be a large prime divisor of $\#E(\mathbb{F}_p)$. In practice, (E, p, r) are provided using parametric families [18]. Let k be the smallest integer such that r divides $p^k - 1$. This k is called the embedding degree of E relative to r.
Let $G_1 = E(\mathbb{F}_p)[r]$ be the r-torsion subgroup of $E(\mathbb{F}_p)$. Let $G_2 = E'(\mathbb{F}_{p^{k/d}})[r] \cap \text{Ker}(\pi_p - [p])$, where E' is the twist of E (if it exists) of degree d, π_p represents the Frobenius map over E, and $[p]$ is the scalar multiplication by p over E. The subgroup of $\mathbb{F}^{\star}_{p^k}$, consisting of r-th roots of unity, is denoted by $G_3 = \mu_r$.
Let s be an integer derived for the optimal Ate pairing [31], depending on r and p. Then the (optimal) Ate pairing is given by:

$$\begin{aligned} T_r : G_1 \times G_2 &\longrightarrow G_3 \\ (P, Q) &\longmapsto f_{s,Q}(P)^{(p^k-1)/r}. \end{aligned}$$

The pairing computation is divided into two main steps [17, Chap. 3]. First, we compute $f_{s,Q}(P)$ using an iterative algorithm known as the Miller algorithm [27]. As with any algorithm based on the double-and-add method, the overall computational cost is directly influenced by both the length and the Hamming weight of the integer s.
The second step in computing the Tate pairing and its variants is the final exponentiation, which involves raising the final result of the main loop, $f_{s,Q}(P)$, to the power $\frac{p^k-1}{r}$. The final exponentiation has become a significant component of the overall computation. Utilizing the

cyclotomic polynomial, this exponentiation can be efficiently divided into two distinct parts as follows:

$$\frac{p^k-1}{r} = \frac{p^k-1}{\phi_k(p)} \times \frac{\phi_k(p)}{r}.$$

- The computation of $\dfrac{p^k-1}{\phi_k(p)}$, known as the easy part of the final exponentiation, requires some Frobenius operations (two if k is even), several multiplications, and an inversion in $\mathbb{F}_{p^k}$.
- The hard part of the final exponentiation, $\dfrac{\phi_k(p)}{r}$, is expressed in terms of the basis p using the cyclotomic polynomial. This computation encompasses numerous multiplications, squarings, cyclotomic squarings or cubings, and several Frobenius operations. The complexity of this stage is significantly higher compared to the easy part of the final exponentiation and the Miller loop [5].

In this paper, we aim to optimize operations essential for the computation of the final exponentiation, specifically Frobenius operation and cyclotomic cubing.
Let us first recall the definition of the Frobenius endomorphism that is used at different steps of the final exponentiation.

2.2 Frobenius endomorphism

Definition 2.1. Let p be a prime number and $k \in \mathbb{N}^*$.
For a field extension $\mathbb{F}_{p^k}/\mathbb{F}_p$, the p-Frobenius mapping is the $\mathbb{F}_p$-linear mapping

$$\begin{array}{rccc} \pi_p : \mathbb{F}_{p^k}/\mathbb{F}_p & \longrightarrow & \mathbb{F}_{p^k}/\mathbb{F}_p \\ x & \longmapsto & x^p. \end{array}$$

Observe that π_p is in fact an $\mathbb{F}_p$-algebra automorphism of $\mathbb{F}_{p^k}$. In particular,

$$(x+y)^p = x^p + y^p,$$

for all $x, y \in \mathbb{F}_{p^k}$.
For $i \in \mathbb{N}^*$, the ith iterate of the Frobenius map, $\pi^i := \underbrace{\pi_p \circ \ldots \circ \pi_p}_{i \text{ times}}$, is the automorphism

$$\begin{array}{rccc} \pi^i : \mathbb{F}_{p^k}/\mathbb{F}_p & \longrightarrow & \mathbb{F}_{p^k}/\mathbb{F}_p \\ x & \longmapsto & x^{p^i}. \end{array}$$

Note that we have $\pi^k = Id_{\mathbb{F}_{p^k}}$.

In this paper, the new method proposed to improve the Frobenius endomorphism calculation is based on the use of the Kronecker product of matrices. In the following, we provide some general details on the Kronecker product.

2.3 Kronecker Product

The Kronecker product is defined for two matrices of arbitrary size over any ring. However, in this work, we focus on matrices over finite fields $\mathbb{F}$.

Definition 2.2. Let $p, q, r, s \in \mathbb{N}^*$, the Kronecker product, also known as tensor product, of matrix $A \in \mathcal{M}_{p,q}(\mathbb{F})$ and matrix $B \in \mathcal{M}_{r,s}(\mathbb{F})$ is defined as

$$A \otimes B = \begin{pmatrix} a_{11}B & \dots & a_{1q}B \\ \vdots & & \vdots \\ a_{p1}B & \dots & a_{pq}B \end{pmatrix} \in \mathcal{M}_{pr,qs}(\mathbb{F}) .$$

The Kronecker product has interesting properties; many of them are stated and proven in the basic literature about matrix analysis. We provide the following properties which are needed in the sequel.

Basic Properties

1. The product of two Kronecker products yields another Kronecker product:

   $$(A \otimes B)(C \otimes D) = AC \otimes BD \quad \forall\, A \in \mathcal{M}_{p,q}, B \in \mathcal{M}_{r,s}, C \in \mathcal{M}_{q,k}, D \in \mathcal{M}_{s,l}.$$

   In particular,

   $$(A \otimes B)^s = A^s \otimes B^s \quad \forall s \in \mathbb{N},\ \forall\, A \in \mathcal{M}_p,\ B \in \mathcal{M}_r.$$

2. If A and B are diagonal matrices, then $A \otimes B$ is a diagonal matrix.

3. $I_m \otimes I_n = I_{nm}$, where I_r is the $(r \times r)$ identity matrix.

3 Efficient Frobenius operation in a tower of subfields

The Frobenius operation is an integral part of several steps in the final exponentiation process in pairing-based cryptography. Therefore, analyzing the number of operations required for the Frobenius morphism in various finite fields is essential for optimal performance. In this section, we introduce a method to simplify the complexity analysis of the Frobenius operation using the matrix representation and the Kronecker product of matrices. This approach offers a structured and efficient means of determining the operation counts required for the Frobenius morphism. We also provide examples of this method applied to different pairing-friendly fields.

3.1 Frobenius operation in a simple field extension

We first explain how the Frobenius morphism can be computed in a simple field extension using a matrix representation, and then provide the operation counts needed to perform the Frobenius operation.

Lemma 3.1. *Let p be a prime number and $k \geq 2$ an integer. Let $\beta \in \mathbb{F}_p$ be an element such that $k \mid p-1$ and such that the polynomial $X^k - \beta$ is irreducible over $\mathbb{F}_p$.*
We denote $\mathbb{F}_{p^k} = \mathbb{F}_p(\lambda)$ and $B = (1, \lambda, \lambda^2, \dots, \lambda^{k-1})$ the standard basis of $\mathbb{F}_{p^k}/\mathbb{F}_p$.
Consider the p-Frobenius mapping

$$\begin{array}{rccc} \pi_p : \mathbb{F}_{p^k}/\mathbb{F}_p & \longrightarrow & \mathbb{F}_{p^k}/\mathbb{F}_p \\ x & \longmapsto & x^p. \end{array}$$

Then,

$$Mat_B(\pi_p) = \begin{pmatrix} 1 & 0 & \dots & 0 \\ 0 & \theta & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & \theta^{(k-1)} \end{pmatrix},$$

where $\theta := \beta^{(p-1)/k} \in \mathbb{F}_p$.
Moreover, for all $1 \le i \le k$, *we have*

$$Mat_B(\pi^i) = (Mat_B(\pi_p))^i = \begin{pmatrix} 1 & 0 & \dots & 0 \\ 0 & \theta^i & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & \theta^{i(k-1)} \end{pmatrix}.$$

Proof. We note that

$$\pi_p(\lambda) = \lambda^p = \lambda\lambda^{p-1} = \lambda(\lambda^k)^{(p-1)/k} = \lambda\beta^{(p-1)/k} = \theta\lambda.$$

Consequently, $\pi_p(\lambda^j) = \theta^j\lambda^j$ for all $0 \le j \le k-1$.
In particular, for $A = \sum_{j=0}^{k-1} a_j\lambda^j \in \mathbb{F}_{p^k}$ with $a_j \in \mathbb{F}_p$, we have

$$A^p = \pi_p(A) = Mat_B(\pi_p)\begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_{k-1} \end{pmatrix} = \begin{pmatrix} 1 & 0 & \dots & 0 \\ 0 & \theta & & \vdots \\ \vdots & & \ddots & 0 \\ 0 & \dots & 0 & \theta^{(k-1)} \end{pmatrix}\begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_{k-1} \end{pmatrix} = \sum_{j=0}^{k-1} a_j\theta^j\lambda^j.$$

□

This formula is essential for determining the operation counts of a p^i-Frobenius map. Indeed, the complexity of the Frobenius operation is determined by counting the number of multiplications involving $\theta^j \neq \pm 2^l$, for $l \in \mathbb{N}$. Multiplication by $\pm 2^l$ can be performed using bitwise shifts, combined with a negation step in the case of a multiplication by -2^l. These operations are generally very fast and considered computationally inexpensive compared to other arithmetic operations. Thus, instead of performing the computation for a general $A \in \mathbb{F}_{p^k}$, it suffices to count the occurences of $\theta^j \neq \pm 2^l$ in the matrix representation of the Frobenius map. From this observation, we derive the result specified in Proposition 3.2.

Proposition 3.2. *Using the same notations as in Lemma 3.1, we have the following*

1. *Let* $s_1 := Card\{0 \le j \le k-1 \mid \theta^j = \pm 2^l;\ l \in \mathbb{N}\}$*; then the number of multiplications to compute the* p*-Frobenius map is*
$$(k - s_1)M.$$

2. *More generally, for* $1 \le i \le k-1$, *let* $s_i := Card\{0 \le j \le k-1 \mid \theta^{ji} = \pm 2^l;\ l \in \mathbb{N}\}$. *Then, for all* $1 \le i \le k-1$, *the number of multiplications to compute the* p^i*-Frobenius map is*
$$(k - s_i)M.$$

3.2 Frobenius operation in a tower field extension

Similar to the approach used for a simple field extension, in this section, a matrix representation of the Frobenius map is used to calculate the number of operations needed to compute the Frobenius morphism. Specifically, for a Frobenius map on a tower field extension, we write a representative matrix of the Frobenius map as a Kronecker product of smaller matrices, according to the decomposition of the finite field extension.

Let p be a prime number and let $k_1, k_2 \in \mathbb{N}^*$ be two integers such that $k_1, k_2 \mid p-1$.

Suppose that $\beta_1 \in \mathbb{F}_p$ and $\beta_2 \in \mathbb{F}_{p^{k_1}}$ are two elements such that we have the following tower of finite fields:

$$\mathbb{F}_{p^{k_1}} = \mathbb{F}_p(\lambda_1);\ \lambda_1^{k_1} = \beta_1 \in \mathbb{F}_p$$

$$\mathbb{F}_{p^{k_2 k_1}} = \mathbb{F}_{p^{k_1}}(\lambda_2);\ \lambda_2^{k_2} = \beta_2 \in \mathbb{F}_{p^{k_1}}.$$

Let $B_1 := (1, \lambda_1, \lambda_1^2, \ldots, \lambda_1^{k_1-1})$ and $B_2 := (1, \lambda_2, \lambda_2^2, \ldots, \lambda_2^{k_2-1})$ be the respective standard basis of the vector spaces $\mathbb{F}_{p^{k_1}}/\mathbb{F}_p$ and $\mathbb{F}_{p^{k_2 k_1}}/\mathbb{F}_{p^{k_1}}$.

Let $B := B_2 \otimes B_1 = (\lambda_2^j, \lambda_2^j \lambda_1, \ldots, \lambda_2^j \lambda_1^{k_1-1})_{0 \leq j \leq k_2-1}$. Then it is straightforward to see that B is a basis of the vector space $\mathbb{F}_{p^{k_2 k_1}}/\mathbb{F}_p$, leading to the result stated in Proposition 3.3.

Proposition 3.3. *The representative matrix of the Frobenius map*

$$\begin{array}{rcl} \pi_p : \mathbb{F}_{p^{k_2 k_1}}/\mathbb{F}_p & \longrightarrow & \mathbb{F}_{p^{k_2 k_1}}/\mathbb{F}_p \\ x & \longmapsto & x^p \end{array}$$

with respect to the basis $B := B_2 \otimes B_1$ is given by

$$\begin{aligned} Mat_B(\pi_p) &= \begin{pmatrix} 1 & & & & & & & & & & \\ & \ddots & & & & & & & & & \\ & & \theta_1^{k_1-1} & & & & \mathbf{0} & & & & \\ & & & \ddots & & & & & & & \\ & & & & \theta_2^j & & & & & & \\ & & & & & \ddots & & & & & \\ & & & & & & \theta_2^j \theta_1^{k_1-1} & & & & \\ & & & & & & & \ddots & & & \\ & & \mathbf{0} & & & & & & \theta_2^{k_2-1} & & \\ & & & & & & & & & \ddots & \\ & & & & & & & & & & \theta_2^{k_2-1}\theta_1^{k_1-1} \end{pmatrix} \\ &= \begin{pmatrix} M_{\theta_1} & & & & \\ & \ddots & & \mathbf{0} & \\ & & \theta_2^j M_{\theta_1} & & \\ & \mathbf{0} & & \ddots & \\ & & & & \theta_2^{k_2-1} M_{\theta_1} \end{pmatrix} \\ &= M_{\theta_2} \otimes M_{\theta_1} \end{aligned}$$

where $\theta_1 := \beta_1^{(p-1)/k_1} \in \mathbb{F}_p$, $\theta_2 := \beta_2^{(p-1)/k_2} \in \mathbb{F}_{p^{k_1}}$ *and*

$$M_{\theta_l} := \begin{pmatrix} 1 & 0 & \dots & 0 \\ 0 & \theta_l & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & \theta_l^{k_l-1} \end{pmatrix}; \ l = 1, 2.$$

Proof. As π_p is a $\mathbb{F}_p$-linear map, to determine the representative matrix of the Frobenius map, it suffices to compute the images of the basis elements

$$B := B_2 \otimes B_1 = (\lambda_2^j, \lambda_2^j\lambda_1, \dots, \lambda_2^j\lambda_1^{k_1-1})_{0 \leq j \leq k_2-1}$$

of the vector space $\mathbb{F}_{p^{k_2k_1}}/\mathbb{F}_p$.
We have

$$\lambda_1^p = \lambda_1\lambda_1^{p-1} = \lambda_1(\lambda_1^{k_1})^{(p-1)/k_1} = \lambda_1\beta_1^{(p-1)/k_1} = \theta_1\lambda_1,$$

and

$$\lambda_2^p = \lambda_2\lambda_2^{p-1} = \lambda_2(\lambda_2^{k_2})^{(p-1)/k_2} = \lambda_2\beta_2^{(p-1)/k_2} = \theta_2\lambda_2.$$

Thus,

$$\pi_p(\lambda_2^j\lambda_1^i) = (\lambda_2^j)^p(\lambda_1^i)^p = \theta_2^j\theta_1^i\lambda_2^j\lambda_1^i.$$

This completes the proof and provides us with the desired representative matrix of the Frobenius map π_p. □

From the representative matrix, we derive the Theorem 3.4 which provides the formula to determine the operation counts of the Frobenius map π_p.

Theorem 3.4. *Using the same notations as in Proposition 3.3, the number of operations to compute the* p*-Frobenius map is*

$$(k_1 - s_1)M + k_1(k_2 - 1)m_{k_1,\theta_2},$$

where $s_1 = Card\{0 \leq i \leq k_1 - 1 \mid \theta_1^i = \pm 2^l;\ l \in \mathbb{N}\}$. *More precisely, we can distinguish two cases:*
- *If* $\theta_2 \in \mathbb{F}_{p^{k_1}} \setminus \mathbb{F}_p$, *then* $m_{k_1,\theta_2} = k_1M$, *and the number of multiplications to compute the* p*-Frobenius map is*

$$(k_1 - s_1)M + k_1^2(k_2 - 1)M.$$

- *If* $\theta_2 \in \mathbb{F}_p$, *then* $m_{\theta_2} = M$, *and the number of multiplications to compute the* p*-Frobenius map is*

$$(k_1 - s_1)M + k_1(k_2 - 1)M.$$

Proof. To determine the operation counts of the Frobenius map π_p, we need to compute the number of multiplications required when performing the power x^p for $x \in \mathbb{F}_{p^{k_2k_1}}/\mathbb{F}_p$.
Let $x \in \mathbb{F}_{p^{k_2k_1}}/\mathbb{F}_p$, then $x = \sum_{j=0}^{k_2-1} x_j\lambda_2^j$ with $x_j \in \mathbb{F}_{p^{k_1}}$ for $0 \leq j \leq k_2 - 1$.
For all $0 \leq j \leq k_2 - 1$, $x_j = \sum_{i=0}^{k_1-1} x_{j,i}\lambda_1^i$ with $x_{j,i} \in \mathbb{F}_p$ for $0 \leq i \leq k_1 - 1$, then

$$x = \sum_{j=0}^{k_2-1}\sum_{i=0}^{k_1-1} x_{j,i}\lambda_2^j\lambda_1^i$$

with $x_{j,i} \in \mathbb{F}_p$ for all $0 \leq i \leq k_1 - 1$, $0 \leq j \leq k_2 - 1$.
By Fermat's Theorem, $x_{j,i}^p = x_{j,i}$ for all $0 \leq i \leq k_1 - 1$, $0 \leq j \leq k_2 - 1$, thus

$$\begin{aligned}\pi_p(x) = x^p &= \sum_{j=0}^{k_2-1} \sum_{i=0}^{k_1-1} x_{j,i} (\lambda_2^p)^j (\lambda_1^p)^i \\ &= \sum_{j=0}^{k_2-1} \sum_{i=0}^{k_1-1} x_{j,i} (\theta_2 \lambda_2)^j (\theta_1 \lambda_1)^i \\ &= \sum_{j=0}^{k_2-1} \sum_{i=0}^{k_1-1} x_{j,i} \theta_2^j \theta_1^i \lambda_2^j \lambda_1^i.\end{aligned}$$

This demonstrates that raising x to the power of p requires multiplications involving $\theta_2^j \theta_1^i$ in the finite field $\mathbb{F}_p$. Consequently, the number of operations needed to compute the Frobenius map π_p is determined by the number of the multiplication by $\theta_2^j \theta_1^i$ in the finite field $\mathbb{F}_p$.
More precisely, for a simple computation of this complexity, we consider the matrix representation of π_p as the Kronecker product:

$$Mat_B(\pi_p) = M_{\theta_2} \otimes M_{\theta_1}.$$

Thus,

$$\begin{aligned}Mat_B(\pi_p(x)) &= M_{\theta_2} \otimes M_{\theta_1} \begin{pmatrix} x_{0,0} \\ \vdots \\ x_{0,k_1-1} \\ \vdots \\ x_{k_2-1,0} \\ \vdots \\ x_{k_2-1,k_1-1} \end{pmatrix} \\ &= \begin{pmatrix} x_0^p = \sum_{i=0}^{k_1-1} x_{0,i} \theta_1^i \lambda_1^i \\ \theta_2 x_1^p = \theta_2 (\sum_{i=0}^{k_1-1} x_{1,i} \theta_1^i \lambda_1^i) \\ \vdots \\ \theta_2^{k_2-1} x_{k_2-1}^p = \theta_2^{k_2-1} (\sum_{i=0}^{k_1-1} x_{k_2-1,i} \theta_1^i \lambda_1^i) \end{pmatrix}\end{aligned}$$

This implies that the count of the operations of π_p is as follows: we count the number of multiplications by θ_1^i such that $\theta_1^i \neq \pm 2^l$; $l \in \mathbb{N}$. This corresponds to $(k_1 - s_1)M$. Then, we need to perform $(k_2 - 1) \times k_1$ multiplications by θ_2^j; $1 \leq j \leq k_2 - 1$. This completes the proof. □

The preceding result can be readily extended to a finite tower of finite fields, as presented in Proposition 3.5.

Proposition 3.5. *Let $k = k_1 \ldots k_s$ be a composite number such that $k_i \mid p-1$, for all $1 \leq i \leq s$, and suppose that*

$$\mathbb{F}_{p^{k_1}} = \mathbb{F}_p(\lambda_1);\ \lambda_1^{k_1} = \beta_1 \in \mathbb{F}_p;$$

$$\vdots$$

$$\mathbb{F}_{p^{k_s \dots k_1}} = \mathbb{F}_{p^{k_{s-1} \dots k_1}}(\lambda_s); \ \lambda_s^{k_s} = \beta_s \in \mathbb{F}_{p^{k_{s-1} \dots k_1}}$$

forms a finite tower of finite fields. For $1 \leq j \leq s$, let B_j denote a standard basis of $\mathbb{F}_{p^{k_j}}/\mathbb{F}_{p^{k_{j-1}}}$, and π_p the p-Frobenius mapping,

$$\begin{array}{rcl} \pi_p : \mathbb{F}_{p^{k_s \cdots k_1}}/\mathbb{F}_p & \longrightarrow & \mathbb{F}_{p^{k_s \cdots k_1}}/\mathbb{F}_p \\ x & \longmapsto & x^p. \end{array}$$

Then,

$$Mat_B(\pi_p) = M_{\theta_s} \otimes \cdots \otimes M_{\theta_1}$$

where $B = B_s \otimes \cdots \otimes B_1$, and for $1 \leq l \leq s$; $\theta_l := \beta_l^{(p-1)/k_l}$, and

$$M_{\theta_l} := \begin{pmatrix} 1 & 0 & \dots & 0 \\ 0 & \theta_l & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & \theta_l^{k_l - 1} \end{pmatrix}$$

Remark 3.6. *We note that:*

1. *In the case of a finite tower of finite fields with more than two levels, the operation count of the p-Frobenius mapping can be computed using a recursive approach.*
2. *To determine the number of operations needed to compute an iterate of the Frobenius mapping, we use the fact that $Mat_B(\pi^i) = (Mat_B(\pi_p))^i$ for all $i \in \mathbb{N}$.*

3.3 Examples of Frobenius Computations in Pairing Friendly Fields

In [25], N. Koblitz and A. Menezes emphasize the importance of using specific finite fields to optimize the efficiency of cryptographic pairings. These fields, referred to as pairing-friendly, are chosen to enhance computation within the extension and simplify the analysis of the cost of multiplications used in pairings. Specifically, the authors define a finite field $\mathbb{F}_{p^k}$ as pairing-friendly if $p \equiv 1 \mod 12$ and k is of the form $k = 2^i 3^j$, where $i, j \in \mathbb{N}$. Under these conditions, the polynomial $X^k - \beta$ is irreducible over $\mathbb{F}_p$ if β is neither a square nor a cube in $\mathbb{F}_p$. The extension can be constructed by first adjoining a cube or square root of a small β, followed by successively adjoining a cube or square root of each newly adjoined root until the tower is fully constructed.
If $j = 0$, then it is sufficient that $p \equiv 1 \mod 4$ and that β be a quadratic non-residue in $\mathbb{F}_p$. This result provides an easy method for constructing towers over pairing-friendly fields: identify an element $\beta \in \mathbb{F}_p$ that is a quadratic non-residue, and, if necessary, a cubic non-residue. Then, adjoin successive cube and square roots of β, beginning with $\mathbb{F}p$
We can choose β as a small value in $\mathbb{F}_p$. Then, the multiplications by β can be reduced to a few additions, making their cost negligible.

3.3.1 Frobenius Computation in $\mathbb{F}_{p^{12}}$.

For the finite field $\mathbb{F}_{p^{12}}$, with p a prime number such that $p \equiv 1$ [12], we have the following tower of extension, as given in [3]:

$$\mathbb{F}_{p^2} = \mathbb{F}_p[u]; \quad \text{where } u^2 = 2$$

$$\mathbb{F}_{p^6} = \mathbb{F}_{p^2}[v]; \quad \text{where } v^3 = 2^{1/2}$$

$$\mathbb{F}_{p^{12}} = \mathbb{F}_{p^6}[w]; \quad \text{where } w^2 = 2^{1/6}$$

Let $\theta_1 = (2)^{(p-1)/2}$, $\theta_2 = 2^{(p-1)/6}$ and $\theta_3 = 2^{(p-1)/12}$. We note that, as $p \equiv 1\ [12]$, we have θ_1, θ_2 and $\theta_3 \in \mathbb{F}_p$. Thus, we obtain

$$Mat_B(\pi_p) = \begin{pmatrix} 1 & 0 \\ 0 & \theta_3 \end{pmatrix} \otimes \begin{pmatrix} 1 & 0 & 0 \\ 0 & \theta_2 & 0 \\ 0 & 0 & \theta_2^2 \end{pmatrix} \otimes \begin{pmatrix} 1 & 0 \\ 0 & \theta_1 \end{pmatrix}.$$

Lemma 3.7. *In the finite field $\mathbb{F}_{p^{12}}$, and supposing that $12 \mid (p-1)$,*

1. *The computation of the p, p^3, p^5-Frobenius maps costs $11M$.*
2. *The computation of the p^2, p^4 -Frobenius maps costs $10M$.*
3. *The computation of the p^6 -Frobenius maps costs $6M$.*

Proof. The result is a consequence of Theorem 3.4 and the fact that $\theta_1^2 = \theta_1^4 = \theta_1^6 = 1$, $\theta_2^3 = \theta_1$, $\theta_2^6 = 1$. Thus, we obtain

$$Mat_B(\pi^2) = \begin{pmatrix} 1 & 0 \\ 0 & \theta_3^2 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & \theta_2^2 & 0 \\ 0 & 0 & \theta_2^4 \end{pmatrix} \otimes I_2$$

and

$$Mat_B(\pi^6) = \begin{pmatrix} 1 & 0 \\ 0 & \theta_1 \end{pmatrix} \otimes I_3 \otimes I_2.$$

□

3.3.2 Frobenius Computation in $\mathbb{F}_{p^{15}}$.

Let p a prime number such that $\{3,\ 5\} \mid p-1$. Suppose that we have a tower extension for $\mathbb{F}_{p^{15}}$ as follows:

$$\mathbb{F}_{p^5} = \mathbb{F}_p[u]; \quad \text{with } u^5 = \beta_1 = 7$$
$$\mathbb{F}_{p^{15}} = \mathbb{F}_{p^5}[v]; \quad \text{with } v^3 = \beta_2 = u \in \mathbb{F}_{p^5}.$$

Let $\theta_1 = 7^{(p-1)/5}$ and $\theta_2 = 7^{(p-1)/15}$. Then $\theta_1 \neq 1$, $\theta_1 \in \mathbb{F}_p$ and, as $15 \mid p-1$, $\theta_2 \in \mathbb{F}_p$. Moreover, $\theta_2 \neq 1$, $(\theta_2)^3 = 7^{(p-1)/5} = \theta_1 \neq 1$ and $(\theta_2)^{15} = 1$. Thus,

$$Mat_B(\pi_p) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \theta_2 & 0 \\ 0 & 0 & \theta_2^2 \end{pmatrix} \otimes \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & \theta_1 & 0 & 0 & 0 \\ 0 & 0 & \theta_1^2 & 0 & 0 \\ 0 & 0 & 0 & \theta_1^3 & 0 \\ 0 & 0 & 0 & 0 & \theta_1^4 \end{pmatrix}$$

Lemma 3.8. *In the finite field $\mathbb{F}_{p^{15}}$, and supposing that $15 \mid (p-1)$,*

1. *For $1 \leq i \leq 14$ and $i \neq 5$ or $i \neq 10$, the computation of the p^i-Frobenius maps costs $14M$.*
2. *The computation of the p^5 or p^{10}-Frobenius map costs $10M$.*

Proof. The result is a consequence of Theorem 3.4 and the fact that $\theta_1^5 = \theta_1^{10} = 1$. Thus, we obtain

$$Mat_B(\pi^5) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \theta_2^5 & 0 \\ 0 & 0 & \theta_2^{10} \end{pmatrix} \otimes I_5.$$

□

3.3.3 Frobenius Computation in $\mathbb{F}_{p^{27}}$.

For $\mathbb{F}_{p^{27}}$, we consider the following tower extensions:

$$\mathbb{F}_{p^3} = \mathbb{F}_p[u]; \quad \text{with } u^3 = \beta_1 = 7$$

$$\mathbb{F}_{p^9} = \mathbb{F}_{p^3}[v]; \quad \text{with } v^3 = \beta_2 = 7^{1/3}$$

$$\mathbb{F}_{p^{27}} = \mathbb{F}_{p^9}[w]; \quad \text{with } w^3 = \beta_3 = 7^{1/9}.$$

Then

$$Mat_B(\pi_p) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \theta_3 & 0 \\ 0 & 0 & \theta_3^2 \end{pmatrix} \otimes \begin{pmatrix} 1 & 0 & 0 \\ 0 & \theta_2 & 0 \\ 0 & 0 & \theta_2^2 \end{pmatrix} \otimes \begin{pmatrix} 1 & 0 & 0 \\ 0 & \theta_1 & 0 \\ 0 & 0 & \theta_1^2 \end{pmatrix}.$$

where $\theta_1 = 7^{(p-1)/3}$, $\theta_2 = 7^{(p-1)/9}$ and $\theta_3 = 7^{(p-1)/27}$.
Since 7 is not a cube in $\mathbb{F}_p$, $\theta_1 = 7^{(p-1)/3} \neq 1$ and $\theta_1^3 = 1$. Observe that $\theta_1 \in \mathbb{F}_p$.
As $7^{1/3} \in \mathbb{F}_{p^3}$, we have $\theta_2 = 7^{(p-1)/9} \in \mathbb{F}_{p^3} \setminus \mathbb{F}_p$ (unless $p \equiv 1 \mod 9$ for example). Moreover, $\theta_2 \neq 1$, $(\theta_2)^3 = 7^{(p-1)/3} = \theta_1 \neq 1$ and $(\theta_2)^9 = 1$.
Finally, $\theta_3 = 7^{(p-1)/27} \in \mathbb{F}_{p^9} \setminus \mathbb{F}_p$ (unless $p \equiv 1 \mod 27$ for example). Thus we have $\theta_3 \neq 1$, $(\theta_3)^3 = 7^{(p-1)/9} = \theta_2 \neq 1$ and $(\theta_3)^{27} = 1$.
The cost of the computation of the iterates of the p-Frobenius map is given in the following lemma:

Lemma 3.9. 1. *In the worst case, the computation of the p; p^2; p^4; p^5; p^7; p^8-Frobenius maps costs:*

$$2M + 6M_{\theta_2} + 18m_{9,\theta_3} = (2 + 18 + 162)M = 182M.$$

However, in the case where $27 \mid (p-1)$, which is used for constructing pairing tower extensions, the cost of the above Frobenius mappings is **26M**.

2. *In the worst case, the computation of the p^3 and p^6-Frobenius maps costs*

$$6M + 18m_{3,\theta_2} = (6 + 54)M = 60M.$$

However, in the case where $9 \mid (p-1)$, the cost of the above Frobenius mappings is ***24 M***.

3. *The computation of the p^9-Frobenius maps costs* ***18M***.

Proof. The results are a consequence of the Theorem 3.4 and the fact that:

$$Mat_B(\pi^3) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \theta_3^3 & 0 \\ 0 & 0 & \theta_3^6 \end{pmatrix} \otimes \begin{pmatrix} 1 & 0 & 0 \\ 0 & \theta_2^3 & 0 \\ 0 & 0 & \theta_2^6 \end{pmatrix} \otimes \begin{pmatrix} 1 & 0 & 0 \\ 0 & \theta_1^3 & 0 \\ 0 & 0 & \theta_1^6 \end{pmatrix}$$

$$= \begin{pmatrix} 1 & 0 & 0 \\ 0 & \theta_2 & 0 \\ 0 & 0 & \theta_2^2 \end{pmatrix} \otimes \begin{pmatrix} 1 & 0 & 0 \\ 0 & \theta_1 & 0 \\ 0 & 0 & \theta_1^2 \end{pmatrix} \otimes I_3$$

$$Mat_B(\pi^9) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \theta_3^9 & 0 \\ 0 & 0 & \theta_3^{18} \end{pmatrix} \otimes \begin{pmatrix} 1 & 0 & 0 \\ 0 & \theta_2^9 & 0 \\ 0 & 0 & \theta_2^{18} \end{pmatrix} \otimes \begin{pmatrix} 1 & 0 & 0 \\ 0 & \theta_1^9 & 0 \\ 0 & 0 & \theta_1^{19} \end{pmatrix}$$

$$= \begin{pmatrix} 1 & 0 & 0 \\ 0 & \theta_1 & 0 \\ 0 & 0 & \theta_1^2 \end{pmatrix} \otimes I_3 \otimes I_3$$

□

Remark 3.10. *In this study, we distinguish between the different exponents of p in Frobenius evaluations, achieving more efficient Frobenius computations. For instance, in $\mathbb{F}_{p^{15}}$, Frobenius evaluation can be reduced to 10 multiplications in specific cases.*

4 Cubing computation

Besides the computation of Frobenius morphism in finite fields, efficient methods for computing cubing can have a significant impact on the performance of pairing-based cryptosystems. As highlighted by Nanjo et al. [28], while elliptic curves with odd embedding degrees such as $k = 9$ and 15 can lead to efficient pairings, they face the disadvantage that cyclotomic squaring is not applicable. This limitation necessitates alternative techniques to optimize computational efficiency. One of the methods discussed in their paper is fast cubing, referred to by Nanjo et al. as cyclotomic cubing, which is applicable to elliptic curves where k is divisible by 3. In this section, we propose other methods to compute the typical cubing and cyclotomic cubing that result in better complexity.

In the following, let p be a prime number and let $k \in \mathbb{N}^*$, divisible by 3, be such that $k \mid (p-1)$, we denote $q = p^{k/3}$. The cubic extension field of $\mathbb{F}_q$, $\mathbb{F}_{q^3}$ is defined as $\mathbb{F}_q[x]/(x^3 - \xi)$, where ξ is an element of $\mathbb{F}_q^*$ that does not have a cube root in $\mathbb{F}_q$.

4.1 Typical Cubing

In [28], Nanjo et *al.* present the complexity of computing a typical cubing in $\mathbb{F}_q$. In this section, we introduce an alternative method that combines the Chung-Hasan technique for squaring and the Karatsuba algorithm for multiplication in $\mathbb{F}_q$.

Let a be an element in $\mathbb{F}_{q^3}$ represented as $a = a_0 + a_1x + a_2x^2$, with coefficients $a_0,\ a_1,\ a_2 \in \mathbb{F}_q$. Then,

$$a^3 = (a_0 + a_1x + a_2x^2)^3 = (a_0 + a_1x + a_2x^2)^2 \cdot (a_0 + a_1x + a_2x^2).$$

For the first part, which involves squaring the polynomial $(a_0 + a_1x + a_2x^2)$, we employ the Chung-Hasan method as detailed in [13]. According to this method, the computational complexity of performing a squaring in $\mathbb{F}_q$ is given by

$$2M_{k/3} + 3S_{k/3} + 8A_{k/3} + 2m_{k/3,\xi},$$

After computing the first part (i.e., the squaring), we need to perform a multiplication. For the multiplication involving the result of the squaring and the second term $(a_0 + a_1x + a_2x^2)$, we use the Karatsuba method as described in [23]. The computational complexity of this multiplication in $\mathbb{F}_q$ is given by

$$6M_{k/3} + 15A_{k/3} + 2m_{k/3,\xi}.$$

By combining the Chung-Hasan method [13] for squaring calculations, and the Karatsuba method [23] for multiplication computations, we find that the complexity of computing a typical cubing in $\mathbb{F}_q$ is given by

$$8M_{k/3} + 3S_{k/3} + 23A_{k/3} + 4m_{k/3,\xi}$$

Thus, as shown in Table 1, using the Chung-Hasan method combined with the Karatsuba method reduces the complexity for performing a typical cubing in $\mathbb{F}_q$ compared to the method explained in [28].

Method	Complexity
This work	$\mathbf{8M_{k/3} + 3S_{k/3} + 23A_{k/3} + 4m_{k/3,\xi}}$
Nanjo et *al.* [28]	$7M_{k/3} + 5S_{k/3} + 18A_{k/3} + 4m_{k/3,\xi}$

Table 1: Comparaison of operations counts for typical cubing in $\mathbb{F}_q$

4.2 Cyclotomic cubing

In their use of fast cubing for the final exponentiation calculation in the Tate pairing on BLS curves, Nanjo et *al.* [28] demonstrate that the cyclotomic cubing calculation is more efficient than typical cubing but remains less efficient than the squaring even though it is performed in the cyclotomic subgroup. However, according to [32], incorporating cyclotomic cubing improves the efficiency of final exponentiation in the Tate pairing and its derivatives, especially when ternary representation is used in the final exponentiation evaluation.

In this section, we introduce a new method for computing cyclotomic cubing, which enhances the complexity measure obtained in [28], thus improving the efficiency of the final exponentiation in the Tate pairing and its derivatives in specific cases.

Our proposal presents a new approach for decomposing the cube a^3, where a belongs to the cyclotomic subgroup of $\mathbb{F}_q$, denoted as $G_{\phi_3(q)}$. The element $a \in G_{\phi_3(q)}$ can be represented as $a_0 + a_1x + a_2x^2$, with coefficients $a_0, a_1, a_2 \in \mathbb{F}_q$.

In [28], the authors provide an explicit formula for the cube of a, as detailed in Lemma 4.1. This formula is derived from the typical cubing equation, by utilizing the specific order of the element a and the following relation:

$$a^{\phi_3(q)} = a_0^3 + (-3a_0a_1a_2 + a_1^3)\xi + a_2^3\xi^2 = 1.$$

Lemma 4.1. *Let a an element of $G_{\phi_3(q)}$. The cyclotomic cubing a^3 is calculated as follows:*

$$\begin{aligned} a^3 &= (a_0 + a_1x + a_2x^2)^3 \\ &= 1 + 9a_0a_1a_2\xi + 3\left(\underbrace{a_0^2a_1 + a_2(a_0a_2 + a_1^2)\xi}_{(1)}\right)x + 3\left(\underbrace{a_0(a_0a_2 + a_1^2) + a_1a_2^2\xi}_{(2)}\right)x^2 \end{aligned}$$

By developing the terms in the brackets (1) and (2), we can establish a relation between them as given in Lemma 4.2.

Lemma 4.2. *Let $B_1 = a_0^2a_1 + a_2(a_0a_2 + a_1^2)\xi$ and $B_2 = a_0(a_0a_2 + a_1^2) + a_1a_2^2\xi$. We obtain the following equations:*

$$\begin{cases} B_1 + B_2 &= (a_1 + a_0)(a_1 + a_2)(\xi a_2 + a_0) - (\xi + 1)a_0a_1a_2 \\ B_1 - B_2 &= (a_1 - a_0)(a_1 - a_2)(\xi a_2 - a_0) + (\xi - 1)a_0a_1a_2 \end{cases}$$

Proof.

$$\begin{aligned}
B_1 + B_2 &= a_0^2 a_1 + a_2(a_0 a_2 + a_1^2)\xi + a_0(a_0 a_2 + a_1^2) + a_1 a_2^2 \xi \\
&= a_1(a_0^2 + a_2^2 \xi) + (a_1^2 + a_0 a_2)(a_0 + a_2 \xi) \\
&= a_1\big((a_0 + a_2\xi)(a_0 + a_2) - a_0 a_2 - a_0 a_2 \xi\big) + (a_1^2 + a_0 a_2)(a_0 + a_2\xi) \\
&= a_1(a_0 + a_2\xi)(a_0 + a_2) - a_0 a_1 a_2 (1 + \xi) + (a_0 + a_2\xi)(a_1^2 + a_0 a_2) \\
&= (a_0 + a_2\xi)\,\big(a_1(a_0 + a_2) + a_1^2 + a_0 a_2\big) - (1 + \xi) a_0 a_1 a_2 \\
&= (a_0 + a_2\xi)(a_0 a_1 + a_1 a_2 + a_1^2 + a_0 a_2) - (1 + \xi) a_0 a_1 a_2 \\
&= (a_1 + a_0)(a_1 + a_2)(\xi a_2 + a_0) - (\xi + 1) a_0 a_1 a_2
\end{aligned}$$

For the calculation of $B_1 - B_2$, we apply the same approach as for $B_1 + B_2$ detailed above. □

With this presentation of $B_1 + B_2$ and $B_1 - B_2$, we can deduce that:

$$\begin{cases} B_1 &= \dfrac{1}{2}\Big((B_1 + B_2) + (B_1 - B_2)\Big) \\ B_2 &= \dfrac{1}{2}\Big((B_1 + B_2) - (B_1 - B_2)\Big) \end{cases} \tag{1}$$

To achieve optimal complexity in computing B_1 and B_2, both terms $B_1 + B_2$ and $B_1 - B_2$ should be even. In this case, multiplication by $\dfrac{1}{2}$ is essentially free, as it can be performed with a simple bitwise shift. This is only possible if ξ is chosen as an odd positive integer as demonstrated in Lemma 4.3.

Lemma 4.3. *Let ξ be an odd positive integer. Then, for all $a_0, a_1, a_2 \in \mathbb{F}_q$,*

$$(a_1 + a_0)(a_1 + a_2)(\xi a_2 + a_0) \text{ is even,}$$

and

$$(a_1 - a_0)(a_1 - a_2)(\xi a_2 - a_0) \text{ is even.}$$

Proof. ξ is an odd positive integer. Let's assume that the terms $(a_1 + a_0)$, $(a_1 + a_2)$ and $(\xi a_2 + a_0)$ are all odd. Thus, we have:

$$\begin{cases} a_1 + a_0 &= 2s + 1 \quad (1) \\ a_2 + a_1 &= 2t + 1 \quad (2) \ ; \quad s, l, t \in \mathbb{F}_q. \\ a_0 + \xi a_2 &= 2l + 1 \quad (3) \end{cases}$$

We distinguish two possible cases:

1. **First case:** a_2 is even $\Rightarrow \xi a_2$ is even $\overset{(3)}{\Rightarrow}$ a_0 is odd $\overset{(1)}{\Rightarrow}$ a_1 is even $\overset{(2)}{\Rightarrow}$ a_2 is odd which is not possible.

2. **Second case:** a_2 is odd $\Rightarrow \xi a_2$ is odd $\overset{(3)}{\Rightarrow}$ a_0 is even $\overset{(1)}{\Rightarrow}$ a_1 is odd $\overset{(2)}{\Rightarrow}$ a_2 is even which is not possible.

Therefore, we can conclude that $(a_1 + a_0)$, $(a_1 + a_2)$ and $(\xi a_2 + a_0)$ cannot all be odd simultaneously. Consequently, $(a_1 + a_0)(a_1 + a_2)(\xi a_2 + a_0)$ is even.
By the same way, we can conclude that $(a_1 - a_0)$, $(a_1 - a_2)$ and $(\xi a_2 - a_0)$ cannot all be odd simultaneously. Consequently, $(a_1 - a_0)(a_1 - a_2)(\xi a_2 - a_0)$ is even.

□

By expanding Equation 1, we obtain the following formulas.
Let ξ be an odd positive integer, then B_1 and B_2 are given by:

$$\begin{cases} B_1 &= \dfrac{1}{2}(a_1+a_0)(a_1+a_2)(\xi a_2+a_0)+\dfrac{1}{2}(a_1-a_0)(a_1-a_2)(\xi a_2-a_0)-a_0a_1a_2 \\ B_2 &= \dfrac{1}{2}(a_1+a_0)(a_1+a_2)(\xi a_2+a_0)-\dfrac{1}{2}(a_1-a_0)(a_1-a_2)(\xi a_2-a_0)-\xi a_0a_1a_2 \end{cases}$$

In particular, as proved in Lemma 4.3, the multiplication by $\dfrac{1}{2}$ is essentially free, as it can be implemented as a right bitwise shift, which is computationally efficient.
From the above formulas, we can count the operations needed to calculate B_1 and B_2, and thus, the cyclotomic cubing in $G_{\phi_3(q)}$.

Complexity measure. To determine the complexity of the cyclotomic cubing of an element $a \in G_{\phi_3(q)}$, we first need to calculate the number of operations needed to compute B_1 and B_2.

```
Calculation of $\mathbf{B_1, B_2}$
1: $t_0 = a_1 + a_0$
2: $t_1 = a_1 + a_2$
3: $t_2 = \xi a_2$
4: $t_3 = t_2 + a_0$
5: $t_4 = a_0a_1a_2$
6: $t_5 = \xi t_4$
7: $t_6 = t_0t_1t_3$
8: $t_6 = \frac{1}{2}t_6$
9: $t_0 = a_1 - a_0$
10: $t_1 = a_1 - a_2$
11: $t_3 = t_2 - a_0$
12: $t_7 = t_0t_1t_3$
13: $t_7 = \frac{1}{2}t_7$
14: $\mathbf{B1} = \mathbf{t_6} + \mathbf{t_7} - \mathbf{t_4}$
15: $\mathbf{B_2} = \mathbf{t_6} - \mathbf{t_7} - \mathbf{t_5}$
```

In total, the calculation of B_1 and B_2 takes $6M_{k/3} + 10A_{k/3} + 2m_{k/3,\xi}$.
Moreover, to compute the cube of a (see Lemma 4.1), we have:

$$1 + 9a_0a_1a_2\xi = 1 + (2^3+1)t_5 = 1 + 2^3t_5 + t_5,$$
$$3B_1 = 2B_1 + B_1 \quad \text{and} \quad 3B_2 = 2B_2 + B_2.$$

A multiplication by 2 is equivalent to a bitwise shift, and thus it is not considered in the complexity calculation. Then, the above calculations takes $3A_{k/3}$.
In conclusion, we find that the complexity of a cyclotomic cube is $6M_{k/3} + 13A_{k/3} + 2m_{k/3,\xi}$.
A comparison of the complexity measures between this work and the results presented in [28] is given in Table 2.

Method	Complexity
This work	$\mathbf{6M_{k/3} + 13A_{k/3} + 2m_{k/3,\xi}}$
Nanjo et *al.* [28]	$5M_{k/3} + 4S_{k/3} + 9A_{k/3} + 3m_{k/3,\xi}$

Table 2: Comparaison operation counts for cyclotomic cubing

From the results presented in Table 2, we can conclude that the cyclotomic cubing calculation proposed in this work is slightly more efficient than the method calculation presented in [28]. Please note that, for the remainder of this paper, we will not consider the additions in our complexity calculations.

Example 4.4. *Based on the results presented below, we can calculate the number of multiplications needed to perform a cyclotomic cubing within the cyclotomic subgroups of $\mathbb{F}_{p^{15}}$ and $\mathbb{F}_{p^{27}}$. Indeed, using the results presented in [28], and considering the costs of multiplications and squarings in $\mathbb{F}_{p^k}$ as detailed in [1], we obtain:*

- $C_{cyc_{15}} = 5M_5 + 4S_5 + 3m_{5,\xi} = 5 \times 13M + 4 \times 13S + 3m_{5,\xi} \simeq 132\,M.$
- $C_{cyc_{27}} = 5M_9 + 4S_9 + 3m_{9,\xi} = 5 \times 36M + 4 \times (18M + 9S) + 3m_{9,\xi} \simeq 315\,M$

However, using our results, we obtain the following:

- $C_{cyc_{15}} = 6M_5 + 2m_{5,\xi} = 6 \times 13M + 2m_{5,\xi} \simeq \mathbf{88\,M}.$
- $C_{cyc_{27}} = 6M_9 + 2m_{9,\xi} = 6 \times 36M + 2m_{9,\xi} \simeq \mathbf{234\,M}.$

Operation	Nanjo et al. [28]	**This work**	**Gain**
$C_{cyc_{15}}$	$132M$	**88M**	**33.3 %**
$C_{cyc_{27}}$	$315M$	**234M**	**25.7 %**

Table 3: Examples of cyclotomic cubing complexity (in terms of multiplications)

In the existing research, multiplication by ξ is often ignored. If we omit it in this analysis, we derive the results presented in 4.

Operation	Nanjo et al. [28]	**This work**	**Gain**
$C_{cyc_{15}}$	$117M$	**78M**	**33.3 %**
$C_{cyc_{27}}$	$288M$	**216M**	**25 %**

Table 4: Examples of cyclotomic cubing complexity (without multiplications by ξ)

Remark 4.5. *In the cyclotomic subgroup of $\mathbb{F}_{p^k}$, where k is odd and divisible by 3, computing the cube of an element can be achieved through two methods: the cyclotomic cubing or the square-and-multiply routine. We can utilize our previous results to compare the efficiency of these methods, as presented in Table 5. We observe that computing a cyclotomic cubing is less expensive than performing a square-and-multiply routine in the cyclotomic subgroup of $\mathbb{F}_{p^k}$.*

Operation	Complexity in $\mathbb{F}_{p^{15}}$	Complexity in $\mathbb{F}_{p^{27}}$
C_{cyc_k}	**88M**	**234M**
$S_k + M_k$	$104M + 39S$	$369M$

Table 5: Cyclotomic Cubing vs Multiplication and Squaring

5 Pairing Evaluation

In this section, we leverage the operation counts of cyclotomic cubing obtained in the previous section to enhance the efficiency of the final exponentiation step in pairing computation.
The Barreto-Lynn-Scott (BLS) curves, a class of elliptic curves, were presented and analyzed in

[8]. These curves are defined over a finite prime field $\mathbb{F}_p$ by the equation $E : y^2 = x^3 + b$ with $j(E) = 0$. In our pairing evaluation, we will focus exclusively on the BLS15 and BLS27 curves, utilizing the improved cyclotomic cubing method presented in Section 4.

The case of BLS15 elliptic curve: BLS15 is a family of parameterized elliptic curves with an embedding degree $k = 15$, defined by the following parameters [16]:

$$\begin{cases} p(u) = \dfrac{u^{12} - 2u^{11} + u^{10} + u^7 - 2u^6 + u^5 + u^2 + u + 1}{3}, \\ r(u) = u^8 - u^7 + u^5 - u^4 + u^3 - u + 1, \\ t(u) = u + 1, \end{cases}$$

The optimal Ate pairing in the context of BLS15 is given by:

$$\begin{aligned} e \colon \mathbb{G}_2 \times \mathbb{G}_1 &\longrightarrow \mathbb{G}_3 \\ (Q, P) &\longmapsto f_{u,Q}(P)^{\frac{p^{15}-1}{r}}, \end{aligned}$$

where the groups $\mathbb{G}_1$, $\mathbb{G}_2$,and $\mathbb{G}_3$ are defined as follows:

- $\mathbb{G}_1 = E(\overline{\mathbb{F}_p})[r] \cap Ker(\pi_p - 1) \subset E(\mathbb{F}_p)$,
- $\mathbb{G}_2 = E(\overline{\mathbb{F}_p})[r] \cap Ker(\pi_p - p) \subset E(\mathbb{F}_{p^{15}})$,
- $\mathbb{G}_3 = \mu_r \subset \mathbb{F}^*_{p^{15}}$.

As explained in Section 2.1, we are primarily interested in calculating the final exponentiation given by

$$\frac{p^{15} - 1}{r} = (p^5 - 1) \times \frac{p^{10} + p^5 + 1}{r}.$$

The computation of $f^{(p^5-1)}$ is referred to as the easy part of the final exponentiation. However, computing the result of the easy part raised to the power of $\frac{p^{10}+p^5+1}{r}$ is known as the hard part of the final exponentiation. We aim to evaluate the cost of the hard part of the final exponentiation using the improved results of cyclotomic cubing.
The approach of computing the final exponentiation was presented differently in [21]. In their method, the easy part of the final exponentiation involves computing the exponent $(p^5 - 1)(p^2 + p + 1)$, while the hard part requires raising this result to the power of

$$(u - 1)^2(u^2 + u + 1) + \sum_{i=0}^{7} \lambda_i(u)p^i(u) + 3$$

where:
$\lambda_7 = 1$, $\lambda_6 = u\lambda_7 - 1$, $\lambda_5 = u\lambda_6$, $\lambda_4 = u\lambda_5 + 1$,
$\lambda_3 = u\lambda_4 - 1$, $\lambda_2 = u\lambda_3 + 1$, $\lambda_1 = u\lambda_2$, $\lambda_0 = u\lambda_1 - 1$.
In the literature, only a few works have evaluated the final exponentiation of the Tate pairing and its derivatives using cyclotomic cubing. In [28], Nanjo et *al.* present the execution time of their method for computing the Optimal Ate pairing on the BLS15 curve, utilizing their optimization of cyclotomic cubing. More recently, in [32], Haddaji et *al.* detailed the computation of the final exponentiation using a ternary basis, which allows for the application of cyclotomic cubing. Haddaji et *al.* also referenced the results from [28] to highlight the computational cost of cyclotomic cubing. To compute the complexity of the final exponentiation of the optimal Ate pairing, we have combined the findings from [32], specifically the Two Consecutive Active Bits

(TCAB) method when TCAB is at the end of the seed u, with the results we present in this paper. The complexity formula is given as follows:

$$\mathbf{I_{15}} + 53\mathbf{M_{15}} + 803\mathbf{S_{15}} + 12\mathbf{C_{cyc_{15}}} + 3\mathbf{I_{cyc_{15}}} + 10\mathbf{F_{15}}$$

Using the cost of each operation provided in [1], [32], and in this paper, we find the following:

$$\begin{aligned} 229\mathbf{M} + 53 \times (78\mathbf{M}) + 803 \times (65\mathbf{M}) + 12 \times (88\mathbf{M}) + 3 \times (78\mathbf{M}) + 8 \times (14\mathbf{M}) \\ +2 \times (10\mathbf{M}) = 57\,980\mathbf{M}. \end{aligned}$$

The case of BLS27 elliptic curve BLS27 is a family of parameterized elliptic curves with an embedding degree of $k = 27$, defined by the following parameters [16]:

$$\begin{cases} r(u) = \dfrac{u^{18} + u^9 + 1}{3}, \\ p(u) = (u-1)^2 r(u) + u, \\ t(u) = u + 1. \end{cases}$$

The optimal Ate pairing in the context of BLS27 is given by:

$$\begin{aligned} e\colon \mathbb{G}_2 \times \mathbb{G}_1 &\longrightarrow \mathbb{G}_3 \\ (Q, P) &\longmapsto f_{u,Q}(P)^{\frac{p^{27}-1}{r}}, \end{aligned}$$

where

- $\mathbb{G}_1 = E(\overline{\mathbb{F}_p})[r] \cap Ker(\pi_p - 1) \subset E(\mathbb{F}_p)$,
- $\mathbb{G}_2 = E(\overline{\mathbb{F}_p})[r] \cap Ker(\pi_p - p) \subset E(\mathbb{F}_{p^{27}})$,
- $\mathbb{G}_3 = \mu_r \subset \mathbb{F}^*_{p^{27}}$.

As explained above, we are interested in calculating the final exponentiation given by

$$\frac{p^{27} - 1}{r} = (p^9 - 1)\frac{p^{18} + p^9 + 1}{r}.$$

The computation of $f^{(p^9-1)}$ is referred to as the easy part of the final exponentiation. However, computing the result of the easy part raised to the power of $\dfrac{p^{18} + p^9 + 1}{r}$ is known as the hard part of the final exponentiation. We aim to evaluate the cost of the hard part of the final exponentiation using the improved results of cyclotomic cubing.

The approach of computing the final exponentiation was presented differently in [21]. In their method, the easy part of the final exponentiation involves computing the exponent $(p^9 - 1)$, while the hard part requires raising this result to the power of:

$$(u-1)^2(u^2 + pu + p^2)(u^6 + p^3u^3 + p^6)(u^9 + p^9 + 1) + 3$$

As in the case of BLS15, only a few works have evaluated the Optimal Ate pairing using cyclotomic cubing. We found results presented in [32], where Haddaji et *al.* detailed their new method of applying cyclotomic cubing using a ternary basis. As mentioned above, Haddaji et al. referenced the results presented in [28] for the cost of cyclotomic cubing. To evaluate the complexity of the final exponentation, we have integrated the findings from [32] which utilized the TCAB method for the 192-security level, with the results presented in this paper. Consequently, the complexity formula is given as follows:

$$\mathbf{I_{27}} + 86\mathbf{M_{27}} + 399\mathbf{S_{27}} + 20\mathbf{C_{cyc_{27}}} + 19\mathbf{I_{cyc_{27}}} + 6\mathbf{F_{27}}$$

Using the cost of each operation given in [1], [32], and in this paper, we obtain the following:

$$\begin{aligned} 536\mathbf{M} + 86 \times (216\mathbf{M}) + 399 \times (153\mathbf{M}) + 20 \times (234\mathbf{M}) + 19 \times (189\mathbf{M}) \\ +6 \times (26\mathbf{M}) = 88\,586\mathbf{M}. \end{aligned}$$

In Table 6, we compare our results with the results presented in [32], and we observe a gain of 536 multiplications in $\mathbb{F}_p$ for pairings on the BLS15 curve and $1,620$ multiplications for pairings on the BLS27 curve.

Pairings in:	Haddaji et *al.* [32]	**This work**
BLS 15	58,516	$\mathbf{57,980}$
BLS 27	90,206	$\mathbf{88,586}$

Table 6: Comparaison of pairing complexity (in terms of multiplication counts) using cyclotomic cubing

In the case where multiplications by ξ are ignored from cyclotomic cubing, as has been done for other operations, we obtain the results presented in Table 7.

Pairings in:	Haddaji et *al.* [32]	**This work**
BLS 15	58,336	$\mathbf{57,860}$
BLS 27	89,666	$\mathbf{88,266}$

Table 7: Comparaison of pairing complexity using cyclotomic cubing (omitting multiplications by ξ)

6 Conclusion

In this article, we introduced two methods to optimize pairing calculations, with a particular focus on the computationally costly operations involved in the final exponentiation, namely the Frobenius and cyclotomic cubing in $\mathbb{F}_{p^k}$. Our first method leverages the Kronecker product of matrices to perform the Frobenius operation, thereby simplifying the complexity analysis over $\mathbb{F}_{p^k}$. We applied the complexity formula defined to commonly used pairing-friendly elliptic curves. Whereas for elliptic curves with even embedding degrees, cyclotomic squaring provides a more efficient approach for the final exponentiation, in the case of curves with odd embedding degrees divisible by 3, we proposed utilizing cyclotomic cubing to enhance the final exponentiation process. We illustrated this with two examples, computing cyclotomic cubing in the cyclotomic subgroups of $\mathbb{F}_{p^{15}}$ and $\mathbb{F}_{p^{27}}$. Furthermore, we assessed the computational cost of the final exponentiation for the Optimal Ate pairing on BLS15 and BLS27 curves. Our results were compared with recent studies that also employed cyclotomic cubing.

References

[1] Diego F. Aranha, Georgios Fotiadis, and Aurore Guillevic. A short-list of pairing-friendly curves resistant to the special TNFS algorithm at the 192-bit security level. *IACR Commun. Cryptol.*, 1(3):3, 2024.

[2] Diego F. Aranha, Koray Karabina, Patrick Longa, Catherine H. Gebotys, and Julio López. Faster explicit formulas for computing pairings over ordinary curves. In Kenneth G. Paterson, editor, *Advances in Cryptology - EUROCRYPT 2011 - 30th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tallinn, Estonia, May 15-19, 2011. Proceedings*, volume 6632 of *Lecture Notes in Computer Science*, pages 48–68. Springer, 2011.

[3] Ismail Assoujaa, Siham Ezzouak, and Hakima Mouanis. Tower building technique on elliptic curve with embedding degree 72. *WSEAS TRANSACTIONS ON COMPUTER RESEARCH*, 10:126–138, 12 2022.

[4] Razvan Barbulescu and Sylvain Duquesne. Updating key size estimations for pairings. *J. Cryptol.*, 32(4):1298–1336, 2019.

[5] Razvan Barbulescu, Nadia El Mrabet, and Loubna Ghammam. A taxonomy of pairings, their security, their complexity. *IACR Cryptol. ePrint Arch.*, page 485, 2019.

[6] Razvan Barbulescu and Cécile Pierrot. The multiple number field sieve for medium- and high-characteristic finite fields. *LMS J. Comput. Math.*, 17(Theory):230–246, 2014.

[7] Paulo S. L. M. Barreto, Craig Costello, Rafael Misoczki, Michael Naehrig, Geovandro C. C. F. Pereira, and Gustavo H. M. Zanon. Subgroup security in pairing-based cryptography. In Kristin E. Lauter and Francisco Rodríguez-Henríquez, editors, *Progress in Cryptology - LATINCRYPT 2015 - 4th International Conference on Cryptology and Information Security in Latin America, Guadalajara, Mexico, August 23-26, 2015, Proceedings*, volume 9230 of *Lecture Notes in Computer Science*, pages 245–265. Springer, 2015.

[8] Paulo S. L. M. Barreto, Ben Lynn, and Michael Scott. Constructing elliptic curves with prescribed embedding degrees. In *Security in Communication Networks*, 2002.

[9] Dan Boneh and Matthew K. Franklin. Identity-based encryption from the Weil pairing. In *Advances in Cryptology – CRYPTO 2001*, volume 2139 of *LNCS*, 2001.

[10] Dan Boneh, Craig Gentry, and Brent Waters. Collusion resistant broadcast encryption with short ciphertexts and private keys. In *Advances in Cryptology - CRYPTO 2005*, volume 3621 of *Lecture Notes in Computer Science*, 2005.

[11] Dan Boneh, Ben Lynn, and Hovav Shacham. Short signatures from the Weil pairing. *J. of Cryptology*, 17(4), 2004.

[12] Sébastien Canard, Aïda Diop, Nizar Kheir, Marie Paindavoine, and Mohamed Sabt. BlindIDS: Market-compliant and privacy-friendly intrusion detection system over encrypted traffic. In *Asia Conference on Computer and Communications Security*. ACM, 2017.

[13] Jaewook Chung and M. Anwar Hasan. Asymmetric squaring formulae. In *18th IEEE Symposium on Computer Arithmetic (ARITH '07)*, pages 113–122, 2007.

[14] Craig Costello, Hüseyin Hisil, Colin Boyd, Juan Manuel González Nieto, and Kenneth Koon-Ho Wong. Faster pairings on special weierstrass curves. In Hovav Shacham and Brent Waters, editors, *Pairing-Based Cryptography - Pairing 2009, Third International Conference, Palo Alto, CA, USA, August 12-14, 2009, Proceedings*, volume 5671 of *Lecture Notes in Computer Science*, pages 89–101. Springer, 2009.

[15] Craig Costello, Kristin E. Lauter, and Michael Naehrig. Attractive subfamilies of BLS curves for implementing high-security pairings. In Daniel J. Bernstein and Sanjit Chatterjee, editors, *Progress in Cryptology - INDOCRYPT 2011 - 12th International Conference on Cryptology in India, Chennai, India, December 11-14, 2011. Proceedings*, volume 7107 of *Lecture Notes in Computer Science*, pages 320–342. Springer, 2011.

[16] Pu Duan, Shi Cui, and Choong Wah Chan. Special polynomial families for generating more suitable elliptic curves for pairing-based cryptosystems. *Cryptology ePrint Archive*, 2005.

[17] N. El Mrabet and M. Joye. *Guide to Pairing-Based Cryptography.* Chapman & Hall/CRC Cryptography and Network Security Series. CRC Press, 2017.

[18] David Freeman, Michael Scott, and Edlyn Teske. A taxonomy of pairing-friendly elliptic curves. *J. Cryptology*, 23(2):224–280, 2010.

[19] Vipul Goyal, Omkant Pandey, Amit Sahai, and Brent Waters. Attribute-based encryption for fine-grained access control of encrypted data. In Ari Juels, Rebecca N. Wright, and Sabrina De Capitani di Vimercati, editors, *Proceedings of the 13th ACM Conference on Computer and Communications Security, CCS 2006, Alexandria, VA, USA, October 30 - November 3, 2006*, pages 89–98. ACM, 2006.

[20] Laurian Azebaze Guimagang, Emmanuel Fouotsa, Nadia El Mrabet, and Aminatou Pecha. Faster beta weil pairing on BLS pairing friendly curves with odd embedding degree. *Math. Comput. Sci.*, 16(2-3):13, 2022.

[21] Daiki Hayashida, Kenichiro Hayasaka, and Tadanori Teruya. Efficient final exponentiation via cyclotomic structure for pairings over families of elliptic curves. *Cryptology ePrint Archive*, 2020.

[22] Antoine Joux. A one round protocol for tripartite Diffie-Hellman. In *Algorithmic Number Theory (ANTS-IV)*, volume 1838 of *Lecture Notes in Computer Science*, 2000.

[23] Anatolij A. Karatsuba and Yu. Ofman. Multiplication of multidigit numbers on automata. *Soviet physics. Doklady*, 7:595–596, 1963.

[24] Taechan Kim and Razvan Barbulescu. Extended tower number field sieve: A new complexity for the medium prime case. In Matthew Robshaw and Jonathan Katz, editors, *Advances in Cryptology - CRYPTO 2016 - 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part I*, volume 9814 of *Lecture Notes in Computer Science*, pages 543–571. Springer, 2016.

[25] Neal Koblitz and Alfred Menezes. Pairing-based cryptography at high security levels. In Nigel P. Smart, editor, *Cryptography and Coding*, pages 13–36, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg.

[26] Duc-Phong Le, Nadia El Mrabet, Safia Haloui, and Chik How Tan. On the near prime-order MNT curves. *Appl. Algebra Eng. Commun. Comput.*, 30(2):107–125, 2019.

[27] Victor S. Miller. Short programs for functions on curves: A STOC rejection. In Andrei Z. Broder and Tami Tamir, editors, *12th International Conference on Fun with Algorithms, FUN 2024, June 4-8, 2024, Island of La Maddalena, Sardinia, Italy*, volume 291 of *LIPIcs*, pages 34:1–34:4. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2024.

[28] Yuki Nanjo, Masaaki Shirase, Takuya Kusaka, and Yasuyuki Nogami. An explicit formula of cyclotomic cubing available for pairings on elliptic curves with embedding degrees of multiple of three. In *2020 35th International Technical Conference on Circuits/Systems, Computers and Communications (ITC-CSCC)*, pages 288–292, 2020.

[29] Justine Sherry, Chang Lan, Raluca Ada Popa, and Sylvia Ratnasamy. Blindbox: Deep packet inspection over encrypted traffic. *ACM SIGCOMM Computer communication review*, 45(4), 2015.

[30] Zcash team. Zk snark in zcash. https://z.cash.

[31] Frederik Vercauteren. Optimal pairings. *IEEE Trans. Information Theory*, 56(1):455–461, 2010.

[32] Nadia El Mrabet Walid Haddaji, Loubna Ghammam and Leila Ben Abdelghani. Optimizing final exponentiation for pairing-friendly elliptic curves with odd embedding degrees divisible by 3. *IACR*, 2025.