We are IntechOpen, the world's leading publisher of Open Access books Built by scientists, for scientists

4,800
Open access books available

122,000
International authors and editors

135M
Downloads

154
Countries delivered to

Our authors are among the

TOP 1%
most cited scientists

12.2%
Contributors from top 500 universities

WEB OF SCIENCE™

Selection of our books indexed in the Book Citation Index in Web of Science™ Core Collection (BKCI)

Interested in publishing with us? Contact book.department@intechopen.com

Numbers displayed above are based on latest data collected.
For more information visit www.intechopen.com

Chapter 3

Efficient Computation for Pairing Based Cryptography: A State of the Art

Nadia El Mrabet

Additional information is available at the end of the chapter

http://dx.doi.org/10.5772/56295

1. Introduction

Cryptographic protocols are divided in two main classes, symmetric systems where keys are secret and asymmetric approaches with public keys. The security of this second category is based on algebraic problems known to be difficult to solve. Historically, in 1976, Diffie-Hellman described a protocol [26] which was one of the first crypto-systems based on the discrete logarithm problem. Later, the introduction of the elliptic curve in cryptography was promoted by V. Miller [55] and N. Koblitz [47] and a large spectrum of crypto-systems appeared. Pairings are bilinear maps which allow to transform an approach on abelian curves, such as elliptic ones, to a problem on finite fields. A first use of such maps concerns cryptanalysis and was proposed in1993 by Menezes Okamoto and Vanstone [53] and in 1994 by G. Frey and H.G. Rück [36] they linked pairings to the discrete logarithmic problem on curves.

In 2000, A. Joux [45] had proposed a tripartite Diffie-Hellmann keys exchange using pairing. That was the beginning of a blossoming literature on the subject. In 2003, D. Boneh and M. Franklin broke a challenge given by Shamir[65] in 1984, creating an identity-based encryption scheme [19] based on pairings. The construction of the pairings is based on the algorithm proposed in 1986 by Victor Miller [54, 56]. A consequence of the rich literature on this subject [62] was the creation of a conference devoted to pairing based cryptography, Pairings [60].

With the birth of this new domain of investigation in cryptography, the problem of implementing these protocols occurs. This point is very relevant to the interest of pairings, the costs and the performances of the implementation make a cryptosystem available. Some good studies on pairings implementation are given by P. Barreto et al [13, 15], we can also refer to some books [29, 37]. We detail later what is a pairing, but at a high level: a pairing is a bilinear map between two groups $\mathbb{G}_1, \mathbb{G}_2$ into a third group $\mathbb{G}_3$ all abelian groups and of the same order.

$$e : \mathbb{G}_1 \times \mathbb{G}_2 \longrightarrow \mathbb{G}_3$$

The bilinearity is the property that

$$e(a \cdot A, b \cdot B) = e(A,B)^{a \cdot b}.$$

For efficient realization $\mathbb{G}_1$ and $\mathbb{G}_2$ are subgroups of an elliptic curve and $\mathbb{G}_3$ is a subgroup of a finite field. The size of the group is fixed by security considerations and lays on the fact that the discrete logarithm problem is hard to solve over $\mathbb{G}_1, \mathbb{G}_2$ and $\mathbb{G}_3$. The pairings are mainly computed with the Miller's algorithm. As a pairing evaluation can be enclosed in a smart card, the question of an efficient implementation is very important.

Several publications are dealing with the efficiency of implementation of pairings. Each of them focus on one aspect of the implementation. We want here to bring together each possible optimizations. The outline of the chapter is the following. First in Section 2 we present the necessary background for a pairing implementation. We present the two first pairings the Weil and the Tate pairings, as well as the optimizations of these, the Eta pairing, the Ate pairing, the twisted Ate pairing, which leads to the notion of optimal pairing and pairing lattices. We also give a first analysis of the arithmetic of pairings. In Section 4, we present the mathematical optimizations of pairings. The use of twisted elliptic curves which leads to the denominator elimination, the improvement of a squaring using cyclotomic subgroups. In Section 5, we present the arithmetical optimizations of a pairing implementation. We describe the different options for an efficient multiplication in Section 5.2, 5.3, 5.3.1 and 5.4. We describe as well how an original representation of a finite field can improve a pairing computation in Section 5.5. In Section 5.6, we describe how the choice of the model of elliptic curve and of its coordinates has a consequence on the implementation. Finally, we conclude in Section 6.

2. Background and notation

Let E be an elliptic curve over a finite field $\mathbb{F}_q$, with P_∞ denoting the identity element of the associated group of rational points $E(\mathbb{F}_p)$. For a positive integer $r|\#E(\mathbb{F}_p)$ coprime to p, let $\mathbb{F}_{p^k}$ be the smallest extension field of $\mathbb{F}_p$ which contains the r-th roots of unity in $\overline{\mathbb{F}}_p$; the extension degree k is called the security multiplier or embedding degree. Let $E(\mathbb{F}_p)[r]$ (respectively $E(\mathbb{F}_{p^k})[r]$) denote the subgroup of $E(\mathbb{F}_p)$ (respectively $E(\mathbb{F}_{p^k})$) of all points of order dividing r. The two groups $\mathbb{G}_1$ and $\mathbb{G}_2$ will be subgroups of elliptic curve groups and $\mathbb{G}_3$ is a subgroup of the multiplicative group of a finite field.

2.1. The Weil, Tate and Ate pairings

2.1.1. The Miller algorithm

The Miller algorithm is the most important step for the Weil, Tate and Ate pairings computation. It is constructed like a double and add scheme using the construction of $[r]P$. Miller's algorithm is based on the notion of divisors. We only give here the essential elements for the pairing computation.

The Miller algorithm constructs the rational function $f_{r,P}$ associated to the point P, where P is a generator of $\mathbb{G}_1 \subset E(\mathbb{F}_p)$; and at the same time, it evaluates $f_{r,P}(Q)$ for a point $Q \in \mathbb{G}_2 \subset E(\mathbb{F}_{p^k})$.

Algorithm 1: Miller(P,Q,l)

Data: $l = (l_n \dots l_0)$(radix 2 representation), $P \in \mathbb{G}_1(\subset E(\mathbb{F}_p))$ and $Q \in \mathbb{G}_2(\subset E(\mathbb{F}_{p^k}))$;
Result: $F_P(Q) \in \mathbb{G}_3(\subset \mathbb{F}^*_{p^k})$;

```
1 : T ← P ;
2 : f1 ← 1 ;
3 : f2 ← 1 ;
for i = n − 1 to 0 do
    4 : T ← [2]T, 5 : f1 ⟵ f1^2 × h1(Q), h1(x) is the equation of the tangent at the point T;
    if l_i = 1 then
        6 : T ← T + P ;
        7 : f1 ⟵ f1 × h2(Q), h2(x) is the equation of the line (PT);
    end
end
return f1
```

2.1.2. The pairings

Definition 2.1. The Weil pairing, denoted e_W, is defined by:

$$\begin{aligned} e_W : \mathbb{G}_1 \times \mathbb{G}_2 &\to \mathbb{G}_3, \\ (P,Q) &\to (-1)^r \tfrac{f_{r,P}(Q)}{f_{r,Q}(P)}. \end{aligned}$$

Definition 2.2. The Tate pairing, denoted e_{Tate}, is defined by:

$$\begin{aligned} \mathbb{G}_1 \times \mathbb{G}_2 &\mapsto \mathbb{G}_3 \\ (P,Q) &\mapsto e_{Tate}(P,Q) = f_{r,P}(Q). \end{aligned}$$

Here, the function $f_{r,P}$ is normalized, i.e. $(u_0^r f_{r,P})(P_\infty) = 1$ for some $\mathbb{F}_p$-rational uniformizer at P_∞. This pairing is only defined up to a representative of $(F_{p^k})^r$. In order to obtain a unique value we raise it to the power $\frac{p^k-1}{r}$, obtaining an r-th root of unity that we call the reduced Tate pairing

$$\hat{e}_{Tate}(P,Q) = f_{r,P}(Q)^{\frac{p^k-1}{r}}.$$

Let π_p be the Frobenius map over the elliptic curve: $\pi_p : E \to E : (x,y) \to (x^p, y^p)$. We denote the Frobenius trace by t. Let $T = t-1$, $\mathbb{G}_1 := E[r] \cap \mathrm{Ker}(\pi_p - [1])$ and $\mathbb{G}_2 := E[r] \cap \mathrm{Ker}(\pi_p - [q])$

Theorem 2.3. *For $P \in \mathbb{G}_1$ and $p \in \mathbb{G}_2$ the following properties hold [43]:*

- $f_{T,Q}(P)$ *is a bilinear pairing called the Ate pairing.*
- *Let $N = gcd(T^k - 1, p^k - 1)$ and $T^k - 1 = NL$, then $e_{Tate}(Q,P)^L = f_{T,Q}(P)^{c(p^k-1)/N}$, where $c = \sum_{i=0}^{k-1} T^{k-1-i} p^i \equiv kp^{k-1} mod(r)$*
- *for r not dividing L, the Ate pairing is non degenerated.*

We therefore obtain the reduced Ate pairing $f_{T,Q}(P)^{(p^k-1)/r}$ which is a power of the Tate pairing. As the trace t is in average of size $\sqrt{p}$, for $r \sim p$, the loop length of Miller's algorithm when computing the Ate pairing is obviously going to be two times shorter than the loop length for the Tate pairing.

2.2. The Duursma-Lee pairing

Duursma and Lee use a family of hyperelliptic curves including supersingular curves over finite fields of characteristic three and adapt it to pairing.

For $\mathbb{F}_p$ with $p = 3^m$ and $k = 6$, suitable curves are defined by an equation of the form

$$E : y^2 = x^3 - x + b,$$

with $b = \pm 1 \in \mathbb{F}_3$. If $\mathbb{F}_{p^3} = \mathbb{F}_p[\rho]/(\rho^3 - \rho - b)$, and $\mathbb{F}_{p^6} = \mathbb{F}_{p^3}[\sigma]/(\sigma^2 + 1)$ then the distortion map $\phi : E(\mathbb{F}_p) \to E(\mathbb{F}_{p^6})$ is defined by $\phi(x,y) = (\rho - x, \sigma y)$. Then, setting $\mathbb{G}_1 = \mathbb{G}_2 = E(\mathbb{F}_{3^m})$ and $\mathbb{G}_3 = \mathbb{F}_{p^6}$, Algorithm 2 computes an admissible, symmetric pairing.

Algorithm 2: The Duursma-Lee pairing algorithm.

Input : $P = (x_P, y_P) \in \mathbb{G}_1$ and $Q = (x_Q, y_Q) \in \mathbb{G}_2$.
Output: $e(P,Q) \in \mathbb{G}_3$.

$f \leftarrow 1$;
for $i = 1$ **upto** m **do**
  $x_P \leftarrow x_P^3$, $y_P \leftarrow y_P^3$;
  $\mu \leftarrow x_P + x_Q + b$;
  $\lambda \leftarrow -y_P y_Q \sigma - \mu^2$;
  $g \leftarrow \lambda - \mu\rho - \rho^2$;
  $f \leftarrow f \cdot g$;
  $x_Q \leftarrow x_Q^{1/3}$, $y_Q \leftarrow y_Q^{1/3}$;
end
return f^{p^3-1};

2.3. The η and η_G pairings

Barreto et al. [12] introduce the η pairing by generalising the Duursma-Lee pairing to allow use of supersingular curves over finite fields of any small characteristic; Kwon [49] independently used the same approach and in both cases characteristic two is of specific interest. The η pairing has already a simple final powering, but work done by Galbraith et al. [38] (see [59, Section 5.4]) demonstrates that it can be eliminated entirely; the crucial step is the lack of normal denominator elimination, which is enabled by evaluation of additional line functions. Interestingly, the analysis of this approach demonstrates no negative security implication in terms of pairing inversion and so on. We follow Whelan and Scott [71] by terming this approach to the η_G pairing.

For $\mathbb{F}_p$ with $p = 2^m$ and $k = 4$, suitable curves are defined by an equation of the form

$$E : y^2 + y = x^3 + x + b$$

Algorithm 3: The η pairing algorithm.

Input : $P = (x_P, y_P) \in \mathbb{G}_1$ and $Q = (x_Q, y_Q) \in \mathbb{G}_2$.
Output: $e(P,Q) \in \mathbb{G}_3$.

$f \leftarrow 1$;
for $i = 1$ **upto** m **do**
  $x_P \leftarrow x_P^2$, $y_P \leftarrow y_P^2$;
  $\mu \leftarrow x_P + x_Q$;
  $\lambda \leftarrow \mu + x_P x_Q + y_P + y_Q + b$;
  $g \leftarrow \lambda + \mu t + (\mu + 1)t^2$;
  $f \leftarrow f \cdot g$;
  $x_Q \leftarrow x_Q^{1/2}$, $y_Q \leftarrow y_Q^{1/2}$;
end
return f^{p^2-1};

Algorithm 4: The η_G pairing algorithm.

Input : $P = (x_P, y_P) \in \mathbb{G}_1$ and $Q = (x_Q, y_Q) \in \mathbb{G}_2$.
Output: $e(P,Q) \in \mathbb{G}_3$.

with $b \in \mathbb{F}_2$. If $\mathbb{F}_{p^2} = \mathbb{F}_p[s]/(s^2+s+1)$ and $\mathbb{F}_{p^4} = \mathbb{F}_{p^2}[t]/(t^2+t+s)$ then the distortion map $\phi : E(\mathbb{F}_p) \to E(\mathbb{F}_{p^4})$ is defined by $\phi(x,y) = (x+s^2, y+sx+t)$. Note that $s = t^5$ and that t satisfies $t^4 = t+1$, so we can also represent $\mathbb{F}_{p^4}$ as $\mathbb{F}_p[t]/(t^4+t+1)$. Then, by setting $\mathbb{G}_1 = \mathbb{G}_2 = E(\mathbb{F}_p)$ and $\mathbb{G}_3 = \mathbb{F}_{p^4}$, Algorithm 4 computes an admissible, symmetric pairing.

Historically, the Weil and Tate pairing was developed by mathematicians without any consideration for cryptography. As efficient implementation of pairings become an interesting question for cryptographers, they searched for improving these two pairings. The Ate and twisted Ate pairing were improvement of the Tate pairing, throught mathematical properties [43]. The notion of Optimal pairing [70] and pairing lattices [42] are the latest properties of pairing. The number of iterations is reduced to the minimum in [70]. In [42], F. Hess proves that every pairing are in relation, because the different pairings are in fact element of a lattice in which each pairing is a power of another pairing. To present the following Sections, we work over the Tate pairing, since as any optimizations of the Tate pairing can be easily adapted to others pairings.

2.4. Analysis of the arithmetic

In order to present the different existing options for the optimizations of a pairing computation, we will focus on the Miller's algorithm. Among the several algorithms which exist to compute a pairing, the most efficient implementations are obtained with the Miller's algorithm.

Let $P = (X_P, Y_P)$ be a point in affine coordinates of the set $E(\mathbb{F}_p)[r]$ (or in Jacobian coordinates with $Z_P = 1$). We consider the point p of order r in $E(\mathbb{F}_{p^k})$, also given in affine coordinates (x_Q, y_Q). Let $\mathbb{G}_1 =< P >$ be the subgroup of order r of $E(\mathbb{F}_p)$ generated by the point P and $\mathbb{G}_2 =< Q >$ the subgroup of order r of $E(\mathbb{F}_{p^k})$. We want to compute a pairing between $\mathbb{G}_1$ and $\mathbb{G}_2$, under the condition $\mathbb{G}_1 \neq \mathbb{G}_2$. The group $\mathbb{G}_3$ is a subgroup of order r of $\mathbb{F}_{p^k}^{\star}$.

Let $T = (X_T, Y_T, Z_T)$ be a point of $E(\mathbb{F}_{p^k})$ in Jacobian coordinates. The main advantage of Jacobian coordinates is that there is no inversion during the arithmetical operation over the elliptic curve.

The Miller's algorithm is given in Algorithm 5.

Algorithm 5: Miller(P, Q, r)

Données: $r = (r_n \dots r_0)$(binary representation), $P \in \mathbb{G}_1(\subset E(\mathbb{F}_p))$ and $Q \in \mathbb{G}_2(\subset E(\mathbb{F}_{p^k}))$;
Résultat: $f_{r,P}(Q) \in \mathbb{G}_3(\subset \mathbb{F}^\star_{p^k})$;

1. $T \leftarrow P$;
2. $f_1 \leftarrow 1$;
3. $f_2 \leftarrow 1$;

for $i = n-1$ **to** 0 **do**
  4. $T \leftarrow [2]T$;
  5. $f_1 \longleftarrow {f_1}^2 \times l_1(Q)$, l_1 is the tangent at point T of E. ;
  6. $f_2 \longleftarrow {f_2}^2 \times v_1(Q)$, v_1 is the vertical line at point $[2]T$. ;
  ($Div(\frac{l_1}{v_1}) = 2(T) - ([2]T) - P_\infty$);
  if $n_i = 1$ **then**
    7. $T \leftarrow T + P$;
    8. $f_1 \longleftarrow f_1 \times l_2(Q)$, l_2 is the line (PT) ;
    9. $f_2 \longleftarrow f_2 \times v_2(Q)$, v_2 is the vertical line at $P + T$;
    ($Div(\frac{l_2}{v_2}) = (T) + D_P - ((T) \oplus D_P) - P_\infty$);
  end
  return $\frac{f_1}{f_2}$
end

The functions $l_1(Q)$, $l_2(Q)$, $v_1(Q)$ and $v_2(Q)$ occurring in Miller's algorithm have their images in $\mathbb{F}^\star_{p^k}$. The parameters f_1 and f_2 are elements of $\mathbb{F}^\star_{p^k}$.

The order r of the subgroups is chosen with a very sparse binary decomposition. In this case, the addition step in Miller's algorithm is not often executed, whereas the doubling step is computed for every iteration of the Miller's algorithm. As a consequence, we consider that the complexity of Miller's algorithm is approximately given by the doubling step. So we will only consider the computation of l_1 and v_1 in the complexity evaluation of Miller's algorithm.

In a general case, we consider that the equation of the elliptic curve is given into the Weierstrass form $E : Y^2 = X^3 + aXZ^4 + bZ^6$, with a and b elements of $\mathbb{F}_p$. In order to be very general, we consider a and b ordinary. Indeed, it is possible to consider that $a = -3$ [20] and the value of b is also a vector of optimizations, but we do not take in consideration these options. We denote $P = (X_P, Y_P)$, $T = (X_T, Y_T, Z_T)$ is the current point in the Miller's algorithm and $2T = (X_{2T}, Y_{2T}, Z_{2T})$ the doubling of T.

The formulas of the doubling in Jacobian coordinates are the following [25]

$$C = 2Y_T^2, \quad D = Z_T^2, \quad A = 4X_T Y_T^2 = 2X_T C, \quad B = (3X_T^2 + aZ_T^4) \tag{1}$$

$$X_{2T} = B^2 - 2A, \quad Y_{2T} = B(A - X_{2T}) - 2C^2, \quad Z_{2T} = 2Y_T Z_T. \tag{2}$$

In this case, the expressions of l_1 and v_1, for $Q = (x_Q, y_Q) \in E(\mathbb{F}_{p^k})$ are given by

$$l_1(x_Q, y_Q) = Z_P^2(Z_{2T} D y_Q - B(D x_Q - X_T) - 2Y_T) \tag{3}$$

$$v_1(x_Q, y_Q) = Z_{2T}^2 Z_P x_Q + 4Y_P^2(X_P D + X_T Z_P^2) - Z_P^2 B^2. \tag{4}$$

We could remark that some intermediary results of the previous formulas may be reused, for instance Y_T^2, Z_T^2, $4X_T Y_T^2$, $(3X_T^2 + aZ_T^4)$. This precomputation reduce the cost of the doubling step, considering the number of operations over the finite field $\mathbb{F}_p$.

Let A_{p^e} (respectively Sub_{p^e}, Sq_{p^e} and M_{p^e}) denote an addition (respectively a subtraction, a squaring and a multiplication) in the finite field $\mathbb{F}_{p^e}$, for e a natural integer. Let also M_a be the cost of a multiplication by a. The Table 1 gives the cost of each operation occurring in the computation of the doubling step. Each cost is given in number of operations over the finite fields. We optimize the computation as possible without any trick different from the one which are following. We consider that a multiplication by 2 is nothing more than a shift in binary representation and thus may be neglected. As a consequence, a multiplication by 3 can be seen as a multiplication by 2 plus an addition and then a multiplication by 3 is equivalent to an addition.

Doubling of a point over E	$4A_p + 3Sub_p + M_a + 4S_p + 4M_p$
Evaluation of l_1	$2Sub_p + Sub_{p^k} S_P + (3+3k)M_P$
Evaluation of v_1	$2A_p + Sub_p + 3S_P + (5+k)M_P$
Step 1 in Algorithm 5	$6A_p + 4Sub_p + Sub_{p^k} + 8S_p + (12+4k)M_p + 2S_{p^k} + 2M_{p^k}$

Table 1. Cost of the doubling step in Miller's algorithm

We will present in Section 4 the optimizations related with mathematics and in Section the optimization in pairings related with the arithmetic of finite fields, in Section 4 the optimizations related with mathematics, in Section 5 the optimizations related with algorithmical breakout.

3. Pairing based cryptography

The first use of pairing in cryptography was destructive: in [53] the Weil pairing was used to shift the discrete logarithm problem from an elliptic curve to a finite field. As the discrete logarithm problem is more easily solved over a finite field than over an elliptic curve, the MOV attack consists in transfering a hard problem over a structure where the same problem is easier. The MOV attack is named after its authors Menezes Okamoto and Vanstome. Later on the pairing was used to improve existing protocols as tri-partite Diffie Hellman key exchange [45] and to construct original protocol like identity based encryption [19, 21].

The aim of identity based encryption is that a person λ, even if λ does not know anything about cryptography, is able to receive and more importantly to read an encrypted message with almost no help.

The public key of λ is its identity, its private key would be send to λ by a trusted authority T. This trusted authority will have all the private keys related with the identity based protocol.

The general scheme of identity based encryption is the following.

The public data are an elliptic curve E over a finite field $\mathbb{F}_p$, a pairing $\hat{e}$ and a hash function H, this hash function associates a point of $E(\mathbb{F}_p)$ to an identity: $H : \{Identity\} \to E(\mathbb{F}_p)$. We consider that two person Alice and Bob want to exchange a common secret for use it as a key in a secure communication.

With the public data, Alice can compute $Q_B = H(Bob)$ the public key of Bob and Bob can compute $Q_A = H(Alice)$ the public key of Alice.

Alice and Bob request the trusted authority to receive their secret key. The secret key is a point of $E(\mathbb{F}_p)$.

The trusted authority chooses s, as its secret key, then it generates $P_A = [s]Q_A$ the secret key of Alice and $P_B = [s]Q_B$ the secret key of Bob.

Then, Alice (respectively Bob) can compute $\hat{e}(P_A, Q_B)$ (resp. $\hat{e}(Q_A, P_B)$, by bilinearity, Alice and Bob have calculated the same key: $\hat{e}(Q_A, Q_B)^{[s]}$. Indeed:

$$\hat{e}([s]H(A), H(B)) = \hat{e}(H(A), [s]H(B)) = \hat{e}(H(A), H(B))^{[s]}.$$

4. Mathematical optimizations

We recall here the mathematical optimizations of pairings. As a pairing is defined over an elliptic curve which is an abelian variety, the first optimization for a pairing computation comes from the mathematical background of pairings. We will use the twist of an elliptic curve, the pairing friendly elliptic curve will follow. We will consider the cyclotomic subgroup of a finite field and then how the final exponentiation in a pairing computation can be improve.

4.1. The twist of an elliptic curve

The twisted elliptic curve of E is another elliptic curve isomorphic to E. Using twisted elliptic curves (when it is possible) in pairing based cryptography is a way to avoid the denominator evaluation in Miller's algorithm. The execution of Miller's algorithm involves computation over $E(\mathbb{F}_{p^k})$, considering a twist of degree d of $E(\mathbb{F}_{p^k})$ allows some computations to be executed in $\tilde{E}(\mathbb{F}_{p^{k/d}})$, where $\tilde{E}(\mathbb{F}_{p^{k/d}})$ is the twisted elliptic curve of $E(\mathbb{F}_{p^k})$ [64].

Definition 4.1. Let E and E' be two elliptic curves, the elliptic curve E' is **a twisted elliptic curve of** E if there exists an isomorphisme Φ defined over $\overline{\mathbb{F}_p}$ mapping each point of E' to a point of E.

There is a limited number of twisted elliptic curves of E. The number of twisted curves depends on the finite field on which the elliptic curve E is defined. The Theorem 4.2 from [64] gives the classification of the possible twists.

Theorem 4.2. *Let E be an elliptic curve of equation $y^2 = x^3 + ax + b$ defined over $\mathbb{F}_{p^k}$. Following the value of k, the possible degrees d of twists are 2, 3, 4 and 6. Let E' be a twist of E, the morphism between E and E' is one of the following.*

- $d = 2$, $E' : Dy^2 = x^3 + ax + b$ defined over $\mathbb{F}_{p^{k/2}}$, where $D \in \mathbb{F}_{p^{k/2}}$ is not a quadratic residue, i.e. such that the polynomial $X^2 - D$ has no solution over $\mathbb{F}_{p^{k/2}}$. The morphism Φ_d is defined by

$$\begin{aligned} \Phi_d : E' &\to E \\ \Phi_d(x,y) &\to (x, yD^{1/2}). \end{aligned}$$

- $d = 4$. The elliptic curve E has a twist of degree 4 if and only if $b = 0$. The equation of E' is then $y^2 = x^3 + \frac{a}{D}x$, where D is not a residue of degree 4, i.e. D is not solution in $\mathbb{F}_{p^{k/4}}$ of a polynomial $X^4 - D$. The morphism is then

$$\begin{aligned} \Phi_d : E' &\to E \\ \Phi_d(x,y) &\to (xD^{1/2}, yD^{3/4}). \end{aligned}$$

- $d = 3$ (resp. 6), the curve E has a twist of degree 3 or 6 if and only if $a = 0$. The equation of E' is then $y^2 = x^3 + \frac{b}{D}$, where D is not a residue of degree 3 (resp. 6), i.e. D is not solution of a polynomial $X^3 - D$ (resp. $X^6 - D$). The morphism is then

$$\begin{aligned} \Phi_d : E' &\to E \\ \Phi_d(x,y) &\to (xD^{1/3}, yD^{1/2}). \end{aligned}$$

Considering the definition above, an elliptic curve can admit a twist of degree 2, 3, 4 or 6. We will only consider here the twisted elliptic curve for an even degree. In order to simplify the notations, we will consider a twist of degree 2. The same method can be applied for twists of degree 4 and 6. The case of twist of degree 3 is a little different, but can also be considered, we refer to [31] for more details. Using a twisted elliptic curve of $E(\mathbb{F}_{p^k})$ allows to make some computation of the Miller's algorithm in a subfield of $\mathbb{F}_{p^k}$, instead of $\mathbb{F}_{p^k}$ and thus allows to simplify the computation. Using a twisted elliptic curve is the solution to avoid the denominators in the Miller's algorithm (i.e. the update of the function f_2). We will denote $\widetilde{E}(\mathbb{F}_{p^{k/2}})$ the twisted curve of $E(\mathbb{F}_{p^k})$, for an even k. We could remark that the twisted elliptic curve of E is an elliptic curve define over an extension of degree half of the initial extension ($\mathbb{F}_{p^k}$) [11]. Let $\nu \in \mathbb{F}_{p^{k/2}}$ a non square element in $\mathbb{F}_{p^{k/2}}$, then $\sqrt{\nu}$ is an element of $\mathbb{F}_{p^k} \setminus \mathbb{F}_{p^{k/2}}$. We can define $\widetilde{E}$ the twisted elliptic curve of $E(\mathbb{F}_{p^k})$ of equation $\nu y^2 = x^3 - 3x + b$. The morphism mapping $\widetilde{E}(\mathbb{F}_{p^{k/2}})$ to $E(\mathbb{F}_{p^k})$ is Ψ_2 define by

$$\begin{aligned} \Psi_2 : \widetilde{E}(\mathbb{F}_{p^{k/2}}) &\to E(\mathbb{F}_{p^k}) \\ (x,y) &\to (x, y\sqrt{\nu}). \end{aligned}$$

The probability that the point $Q = (x, y\sqrt{\nu})$ image of $Q' = (x,y) \in \widetilde{E}$ by Ψ_2 belongs to the subgroup generate by $P \in E(\mathbb{F}_p)$ is negligeable [11]. This assures us that the pairing is non degenerated between

$P \in E(\mathbb{F}_p)$ and $Q = \Psi_2(Q')$. As a consequence, we can consider that the coordinates of the point Q are element of $\mathbb{F}_{p^{k/2}}$ plus a multiplication by $\sqrt{v}$.

We give the formulae for Miller's algorithm with the use of a twisted elliptic curve. Let A, B, C, D, E and F be the intermediate values in the doubling and addition of a point over E (in Jacobian coordinates). These values are dependant only on the point $P = (X_P; Y_P; Z_P)$ and multiples of P: $T = (X_T; Y_T; Z_T)$; $2T = (X_{2T}; Y_{2T}; Z_{2T})$ and $T + P = (X_3; Y_3; Z_3)$. The equations of functions l_1, l_2, v_1 and v_2 are

$$\begin{aligned} l_1(x_Q, y_Q\sqrt{v}) &= Z_P^2(Z_{2T}Dy_Q\sqrt{v} - B(Dx_Q - X_T) - 2Y_T), \\ v_1(x_Q, y_Q\sqrt{v}) &= Z_{2T}^2 Z_P x_Q + 4Y^2(X_P D + X_T Z_P^2) - 9Z_P^2(X_T^2 - Z_T^4)^2, \\ l_2(x_Q, y_Q\sqrt{v}) &= Z_{T+P}^2(Z_T^3 E y_Q\sqrt{v} - Z_T F(Z_T^2 x_Q) - Y_T E), \\ v_2(x_Q, y_Q\sqrt{v}) &= Z_T^3 E(Z_3^3 x_Q + E(A+B) - Z_T^2 Z_P^2 F). \end{aligned} \tag{5}$$

The multiplications and additions in these formulae are made in $\mathbb{F}_p$ and $\mathbb{F}_{p^{k/2}}$. For $x_Q \in \mathbb{F}_{p^{k/2}}$, if we consider carefully the equations of v_1 and v_2, we can remark that the results $v_1(x_Q, y_Q\sqrt{v})$ and $v_2(x_Q, y_Q\sqrt{v})$ are elements of $\mathbb{F}_{p^{k/2}}$. Indeed, the y-coordinate of Q does not appear in the denominator v_1 and consequently $\sqrt{v}$ either. This simple remark allows the elimination of the denominators during the Tate pairing computation.

Property 4.3. *During the evaluation of Miller's algorithm for the Tate pairing, the evaluation of f_2 and thus the computations of v_1 and v_2 can be omited [11].*

Indeed, when using a twist, the equation shows that $v_1(Q)$, $v_2(Q) \in \mathbb{F}_{p^{k/2}}$ and then $f_2 \in \mathbb{F}_{p^{k/2}}$. By definition of the embedding degree k of the elliptic curve, $\frac{p^k-1}{r}$ is a multiple of $p^{k/2} - 1$ and $f_2^{\frac{p^k-1}{r}} = 1$ by the following proposition.

Property 4.4. *Let r be a prime divisor of $\#E(\mathbb{F}_p)$ and E be an elliptic curve of embedding degree k relatively to r. Then $\frac{p^k-1}{r}$ is a multiple of $p^{k/2} - 1$.*

Proof. The demonstration is a straight forward consequence of the construction of k as the smallest integer such that r divides $p^k - 1$. So for an even k, $p^k - 1 = (p^{k/2} - 1)(p^{k/2} + 1)$ and r a prime integer divides $p^k - 1$. Using the Gauss theorem, r divides $(p^{k/2} - 1)$ or $(p^{k/2} + 1)$. If r divides $(p^{k/2} - 1)$, then the definition of k would be wrong, thus the only possibility is that r divides $(p^{k/2} + 1)$. □

For all $\xi \in \mathbb{F}_{p^{k/2}}$, we know that $\xi^{p^{k/2}-1} \equiv 1$ (from the Little Fermat's theorem). Consequently the final exponentiation of the Tate pairing kills every factor of the result belonging to a proper subfield of $\mathbb{F}_{p^k}$. The Miller's computation can be simplified by forgetting v_1 and v_2. But with the same remark, we can also simplify the function l_1 and l_2 into

$$\begin{aligned} l_1(x_Q, y_Q\sqrt{v}) &= Z_{2T}Dy_Q\sqrt{v} - B(Dx_Q - X_T) - 2Y_T, \\ l_2(x_Q, y_Q\sqrt{v}) &= Z_T^3 E y_Q\sqrt{v} - Z_T F(Z_T^2 x_Q) - Y_T E. \end{aligned} \tag{6}$$

This method can be applied for every pairing with a final exponentiation. In the case of the Weil pairing, we can also apply it by raising the result of Weil pairing at the power $p^{k/2} - 1$. The cost of this exponentiation will be study in Section 4.4.

In order to illustrate the simplification of the computation with the use of a pairing, we compare two computations of the doubling step in Miller's algorithm. The Miller Lite execution is the computation of the Miller's algorithm for the Tate pairing ($Miller(P,Q)$). The Miller full execution is the computation of $Miller(Q,P)$. The Table 2 compare the cost of the doubling step in Miller Lite and Miller Full with and without the use of twisted elliptic curve.

Miller	Without twist	With twist
Lite	$8S_p + (12+4k)M_p + 2S_{p^k} + 2M_{p^k}$	$4S_p + (7+k)M_p + S_{p^k} + M_{p^k}$
Full	$3kM_p + 10S_{p^k} + 14M_{p^k}$	$kM_p + 5S_{p^k} + 7M_{p^k}$

Table 2. Cost of Miller Lite and Miller Full

4.2. Pairing friendly fields and elliptic curves

The computation of pairings implies computations over extension fields of the form $\mathbb{F}_{p^k}$. If the embedding degree k is smooth, than the arithmetic in $\mathbb{F}_{p^k}$ can be computed step by step. A complete an extensive nice definition of smooth number is given in [50], we recall here an intuitive naive definition.

Definition 4.5. A ***smooth integer*** is an integer such that its prime factor are composed only by small primes.

***Example* 4.6.** An integer of the form $2^i 3^j$ is smooth.

We illustrate how a smooth integer k allows a construction of $\mathbb{F}_{p^k}$ with a tower field.

***Example* 4.7.** Let l be a prime number and m an integer such that $k = lm$. The extension $\mathbb{F}_{p^k}$ of $\mathbb{F}_p$ can be constructed like an extension of degree l of $\mathbb{F}_{p^m}$. We suppose that we have already constructed the extension $\mathbb{F}_{p^m}$. Let $P(X)$ be an irreducible polynomial of degree l in $\mathbb{F}_{p^m}[X]$. Then $\mathbb{F}_{p^{lm}} = \mathbb{F}_{(p^m)^l}$ is constructed with the quotient

$$\mathbb{F}_{p^{lm}} = \mathbb{F}_{p^m}[X] / (P(X)).$$

We use the tower field construction in order to optimize the multiplication over $\mathbb{F}_{p^k}$. We will see in Section 5 that for extensions of degree 2 and 3, we can use the Karatsuba and Toom Cook multiplications. The tower field construction reduce the number of elementary operations over $\mathbb{F}_p$ to compute a multiplication in $\mathbb{F}_{p^k}$ [35].

A.Menezes and N.Koblitz [48] proposed the definition of pairing friendly elliptic curves. There are elliptic curves suitable for pairing computation. Pairing friendly fields are defined with k smooth.

Definition 4.8. A **pairing friendly field** $\mathbb{F}_{p^k}$ is an extension of a finite field $\mathbb{F}_p$ with the following property

- the characteristic p is such that $p \equiv 1 \ mod(12)$,
- the embedding degree k is such that $k = 2^i 3^j$.

Pairing friendly field are such that the polynomial reduction over the extension $\mathbb{F}_{p^k}$ is very easy to compute [50, Theorem 3.75].

Theorem 4.9. *Let $\beta \in \mathbb{F}_p$ be a neither a square nor a cube in $\mathbb{F}_p$ and $\mathbb{F}_{p^k}$ a pairing friendly field with $k = 2^i 3^j$. Then the polynomial $X^k - \beta$ is irreducible in $\mathbb{F}_p$.*

Using the definition and the above property, we construct the extension $\mathbb{F}_{p^k} = \mathbb{F}_p[X]/(X^k - \beta)$ using several extensions of degree 2 and 3. The construction is done step by step with square or cubic root of β and the results.

***Example* 4.10.** Example of possible tower field for $k = 2^2 3^1$:

$$\begin{aligned} \mathbb{F}_p \xrightarrow{2} L &= \mathbb{F}_p[T]/(T^2 - \beta), \\ K \xrightarrow{3} M &= L[U]/(U^3 - T), \\ L \xrightarrow{2} N &= M[V]/(V^2 - U). \end{aligned}$$

The representation of fields L, M and N are as follow

$$\begin{aligned} L &= \{l_0 + l_1\beta, \text{ with } l_0, l_1 \in \mathbb{F}_p\}, \\ M &= \{m_0 + m_1 T + m_2 T^2, \text{ with } m_0, m_1, m_2 \in L\}, \\ N &= \{n_0 + n_1 U, \text{ with } n_0, n_1 \in M\}. \end{aligned}$$

The arithmetic in $\mathbb{F}_{p^k}$ can be composed in each floor of the tower field construction. As k is a product of power of 2 and 3, the Karatsuba and Toom Cook methods are the more suitable for improving the multiplication in $\mathbb{F}_{p^k}$. We consider that a multiplication in $\mathbb{F}_{p^k}$ with $k = 2^i 3^j$ involves $3^i 5^j$ multiplications in $\mathbb{F}_p$, which is denoted $M_{p^k} = 3^i 5^j M_p$.

4.3. Cyclotomic subgroup and squaring

A. Lenstra and M. Stam introduce in [52] an efficient method for squaring. They use the structure of a cyclotomic subgroup. They construct an extension of degree 6 with a polynomial different from $X^6 - \beta$. The cyclotomic subgroup $\mathbb{G}_{\phi_k(p)}$ is the subgroup of order $\phi_k(p)$ of $\mathbb{F}^\star_{p^k}$, where $\phi_k(p)$ is the kth cyclotomic polynomial evaluated at p. The cyclotomic polynomials are constructed such that there roots are the primitive roots of unity.

The multiplication developed by Lenstra and Stam is interesting for computing squares in degree 6 extension of $\mathbb{F}_p$ (or a degree multiple of 6). It could be interesting to generalize it for other degree extension. They construct the degree 6 extension using the cyclotomic polynomial $\phi_k(X) = X^{k/3} - X^{k/6} + 1$. This method can be used for every degree extension multiple of 6.

Let $\alpha \in \mathbb{G}_{\phi_k(p)}$, $\alpha = \sum_{i=0}^{k-1} a_i \gamma^i$, where for all i, $a_i \in \mathbb{F}_p$ and $\mathscr{B} = (1, \gamma, \gamma^2, \ldots, \gamma^{k-1})$ is a basis of $\mathbb{F}_{p^k}$.

We are seeking for the general expression of an element in $\mathbb{G}_{\phi_k(p)}$. We consider that α is a polynomial in several variables in $\mathbb{F}_p$ (the a_is), with coefficients power of γ in $\mathbb{F}_{p^k}$.

As α belong to the cyclotomic subgroup $\mathbb{G}_{\phi_k(p)}$, the order of α divides the cardinal of $\mathbb{G}_{\phi_k(p)}$ which is $\phi_k(p)$. So, we have that $\alpha^{p^{k/3}-p^{k/6}+1} = 1$ in $\mathbb{G}_{\phi_k(p)}$. This equality can be written $\alpha^{p^{k/3}+1} = \alpha^{p^{k/6}}$.

In order to find the decomposition of $\alpha \times \alpha^{p^{k/3}} - \alpha^{p^{k/6}}$, we can then formally compute $\alpha^{p^{k/3}}$ and $\alpha^{p^{k/6}}$

$$\alpha \times \alpha^{p^{k/3}} - \alpha^{p^{k/6}} = \sum_{i=0}^{k-1} v_i \gamma^i.$$

Where

$$\begin{aligned}
v_0 &= a_1^2 - a_0a_2 - a_4 - a_4^2 + a_3a_5,\\
v_1 &= -a_0 + a_1a_2 + a_3 - 2a_0a_3 + a_3^2 - a_2a_4 - a_1a_5,\\
v_2 &= -a_0a_1 + a_3a_4 - a_5 - 2a_2a_5 + a_5^2,\\
v_3 &= -a_1 - a_2a_3 + 2a_1a_4 - a_4^2 - a_0a_5 + a_3a_5,\\
v_4 &= a_0^2 + a_1a_2 + a_3 - 2a_0a_3 - a_4a_5,\\
v_5 &= -a_2 + a_2^2 - a_1a_3 - a_0a_4 + a_3a_4 - 2a_2a_5.
\end{aligned}$$

As $\alpha \in \mathbb{G}_{\phi_k(p)}$, we have that $\sum_{i=0}^{k-1} v_i\gamma^i = 0$. With this equation, we construct a system in the α_i, the resolution of this system will give us the general form of an element in $\mathbb{G}_{\phi_k(p)}$.

The subgroup $\mathbb{G}_{\phi_k(p)}$ is the set of elements α such that $\forall i,\ v_i = 0$, which gives $\alpha^2 = \alpha^2 + \mathscr{B}.\Gamma.^t v$, with $\mathscr{B} = (1, \gamma, \gamma^2, ..., \gamma^{k-1})$ and with Γ a chosen matrix. As v is zero in $\mathbb{F}_p$, we can reduce the cost of a square with this method.

Denoting $\alpha^2 = \sum_{i=1}^{k} s_i\gamma^i$, we have the equality

$$\sum_{i=1}^{k} s_i\gamma^i = (\sum_{i=1}^{k} a_i\gamma^i)^2 + \mathscr{B}.\Gamma^t.v.$$

We can formally develop the right expression and for a well chosen matrix Γ, the formulae for a square in $\mathbb{F}_{p^k}$ would be simplified. For instance, for $k = 6$[52] :

$$\alpha^2 = \mathscr{B}.\begin{pmatrix} 2a_1 + 3a_4(a_4 - 2a_1) \\ 2a_0 + 3(a_0 + a_3)(a_0 - a_3) \\ -2a_5 + 3a_5(a_5 - 2a_2) \\ 2(a_2 - a_4) + 3a_1(a_1 - 2a_4) \\ 2(a_0 - a_3) + 3a_3(2a_0 - a_3) \\ -2a_2 + 3a_2(a_2 - 2a_5) \end{pmatrix}. \tag{7}$$

Granger, Page and Smart apply this method to construct the Table 3 [41].

Degree extension k	**cost of a square in** $\mathbb{F}_{p^k}$
6	$4,5M_p$
12	$18M_p + 12S_p$
24	$84M_p + 24S_p$

Table 3. Complexity of a square in $\mathbb{F}_{p^k}$

In the particular case where $k = 6$ and $p \equiv 2 \pmod 9$, the cost of a square with the Lenstra and Stam method is less than $0,75M_{p^k}$, which is usually the ratio of a square compare to a multiplication.

***Example* 4.11.** In $\mathbb{F}_{p^6}$, a square with Lenstra and Stam method cost $6 \times 0,75M_p \approx 4,5M_p$. With the classical ratio, a square in $\mathbb{F}_{p^6}$ costs $15 \times 0,75M_p \approx 10M_p$.

4.4. The finale exponentiation

The Tate pairing (and also the Ate, optimal Ate) is composed of two steps, first the Miller's execution and then a final exponentiation. This exponentiation is a very expensive operation as it takes place in $\mathbb{F}_{p^k}$ and the exponent $\frac{p^k-1}{r}$ is a large integer. In order to simplify this exponentiation it is split in two parts [48] using the fact that:

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

where $\phi_k(p)$ is the evaluation in p of the k-th cyclotomic polynomial.

The first part of the exponentiation uses the twisted elliptic curve and it is equivalent to computing the Frobenius map of elements in $\mathbb{F}_{p^k}$. The second part is a reduced exponentiation in $\mathbb{F}_{p^k}$ which is performed with classical method for exponentiation.

4.4.1. First part of the exponentiation

We consider here the exponentiation to the power $\frac{p^k-1}{\phi_k(p)}$. We can first remark that if $k = 2^i 3^j$, then $\phi_k(p) = p^{k/3} - p^{k/6} + 1$ and $\frac{p^k-1}{\phi_k(p)} = (p^{k/2} - 1)(p^{k/6} + 1)$. Using a twist, the result of Miller's algorithm is something like $(X + Y\sqrt{v})$ avec $X, Y \in \mathbb{F}_{p^{k/2}}$.

The computation of $(X + Y\sqrt{v})^{p^{k/2}-1}$ can be decomposed in

$$(X + Y\sqrt{v})^{p^{k/2}} \times (X + Y\sqrt{v})^{-1}.$$

As $(X+Y\sqrt{v})^{-1} = (X+Y\sqrt{v})^{p^{k/2}}$, we have that

$$(X+Y\sqrt{v})^{p^{k/2}-1} = (X+Y\sqrt{v})^{2p^{k/2}}.$$

Raising an element of $\mathbb{F}_{p^k}$ to a power $p^{k/2}$ is a Frobenius operation, which mainly consists in shifts. The total cost of the exponentiation to the power $(p^{k/2}-1)$ is a square in $\mathbb{F}_{p^k}$ and a Frobenius application. Let $(X'+Y'\sqrt{v})$ be the result of $(X+Y\sqrt{v})^{p^{k/2}-1}$.

We then have to compute $(X'+Y'\sqrt{v})^{p^{k/6}+1}$ which is another application of the Frobenius.

Let γ be a root of $X^k-\beta$ in $\mathbb{F}_{p^k}$. An element a of $\mathbb{F}_{p^k}$ can be decomposed in $a = \sum_{i=0}^{k-1} a_i\gamma^i$, with $a_i \in \mathbb{F}_p$.

The property of a finite field gives $a^p = \sum_{i=0}^{k-1} a_i\gamma^{ip}$ and recursively

$$a^{p^j} = \sum_{i=0}^{k-1} a_i\gamma^{ip^j}.$$

For i and j two integers let q_{ij} and r_{ij} be the quotient and the remainder of the Euclidien division of ip^j by k, we know that

$$\gamma^{ip^j} = \beta^{q_{ij} \bmod(p)}\gamma^{r_{ij}}.$$

The computation of $(X'+Y'\sqrt{v})^{p^{k/6}+1}$ can be decomposed in

$$(X'+Y'\sqrt{v})^{p^{k/6}+1} = (X'^{p^{k/6}} + Y'^{p^{k/6}}\sqrt{v}^{(p^{k/6})}) \times (X'+Y'\sqrt{v})$$

For example, if we describe what happened for the variable X' raised to the power $p^{k/6}$, we obtain the following step

$$\begin{cases} X' = \sum_{i=0}^{k/2-1} x_i\gamma^i, \\ X'^{p^{k/6}} = \sum_{i=0}^{k/2-1} x_i\gamma^{ip^{(k/6)}}, \\ X'^{p^{k/6}} = \sum_{i=0}^{k/2-1} (x_i\beta^{q_{i(k/6)}} \mod (p))\gamma^{r_{i(k/6)}}. \end{cases}$$

We have to compute the $\frac{k}{2}$ products $(x_i\beta^{q_{i(k/6)} \bmod(p)})$, with x_i and $\beta^{q_{i(k/6)} \bmod(p)}$ in $\mathbb{F}_p$. The total complexity of the first part of the exponentiation is $2kM_p + S_{p^k} + M_{p^k}$ plus shifts and multiplications by β.

Second part of the exponentiation

The second part of the exponentiation is the hard part. We use classical method of exponentiation like the Lucas sequences [16] or sliding windows [40]. In [67], more tricky method are developed.

The Lucas sequence method induces a cost of a square and a multiplication in the intermediate field $\mathbb{F}_{p^{k/2}}$ for each bit of the exponent. The sliding window method has the advantage that the squares are computed in the cyclotomic subgroup and consequently we can use the method described in Section 4.3. The complexity of the two methods is linearly related to the number of the bits in the binary decomposition of the exponent, we recall here the complexity of the methods and refer to for instance the book [25] for more details.

Let b_r be the number of bits of r, the prime number dividing the cardinal of E. Let b_{p^k} be the number of bits of p^k. The respective size of b_r, b_{p^k}, r and p^k are fixed by the security level we want to reach. We give them in the Table 4. The number of positive integers smaller than k and prime with k is $\varphi(k)$, the Euler totent function evaluated at k. The number $\varphi(k)$ is also the number of primitive k-roots of unity, then it is the degree of the polynomial $\phi_k(p)$. The exponent of the second part of the exponentiation is $(\frac{\varphi(k)}{k} b_{p^k} - b_r)$ bits.

The number of squares and multiplications involved for the computation of the exponentiation depends on $\frac{\varphi(k)}{k} b_{p^k} - b_r = (\tau_k \gamma - 1) b_r$, where

$$\gamma = \frac{b_{p^k}}{b_r},$$

$$\tau_k = \frac{\varphi(k)}{k} = \begin{cases} 1/2 \text{ si } k = 2^i,\ i \geqslant 1 \\ 1/3 \text{ si } k = 2^i 3^j,\ i,\ j \geqslant 1 \end{cases} .$$

The number γ is related to the security levels given in the Table 4 and its is a good appreciation of the total complexity of the exponentiation.

Security level in bits	80	128	192	256
Minimal number of for r	160	256	384	512
Minimal number of for p^k	1 024	3 072	7 680	15 360
$\gamma = \frac{b_{p^k}}{b_r}$	6,4	12	20	30

Table 4. Security level

The complexity of the Lucas sequance method is [16]

$$C_{Luc} = (M_{p^{k/2}} + S_{p^{k/2}}) \log_2 \left(\frac{\phi_k(p)}{r} \right).$$

The complexity of the sliding window method is [40]

$$C_{sw} = \left(\frac{\log_2(e)}{\log_2(p)} + \log_2(p) \right) S_{G_{\phi_k(p)}} + \left(\frac{\log_2(e)}{\log_2(p)} \left(2^{n-1} - 1 \right) + \frac{\log_2(e)}{n+2} - 1 \right) M_{p^k},$$

where $e = \frac{\phi_k(p)}{r}$, and n is the integer giving the size of the window in bits, generally $n = 4$.

5. Arithmetical optimisation

As the pairings computation lays on arithmetic over finite fields, a way to improve the efficiency of computation of pairings is to improve the arithmetic of finite fields and extension of finite fields.

The elliptic curve used in pairing based cryptography are constructed throught the complex multiplication method. These methods of constructions do not allow to fixe p the characteristic of the field $\mathbb{F}_p$, we can only choose the number of bits in the decomposition of p. As a consequence, the arithmetic of pairings is particular. We cannot choose p with a special structure which would provide an efficient arithmetic, like for example a sparse decomposition or a Mersenne or Pseudo Mersenne prime. A very nice overview of construction of elliptic curve for pairing based cryptography is available in the work of Freeman, Scott and Teske [33].

We then begin this section with the presentation of efficient multiplications in finite fields and extensions of finite fields. We recall the different methods for a multiplication and we will provide a comparison of efficiency of these multiplications in Section 5.2, 5.3, 5.4. In Section 5.5, we will consider the representation of elements in a finite field. Indeed, in Section 5.1 we describe the classical representation of a finite field, this classical representation is used for the description of the multiplications. But it is possible, to have original representations of finite field, which can offer opportunities for improvement in pairing based cryptography. In Section 5.6 we will consider how the choice of coordinates can be a way for improving the efficiency of computation of pairings and on the equation of the elliptic curve.

5.1. Setting

We consider in this Section the cost of operations over $\mathbb{F}_{p^k}$ in number of operations over $\mathbb{F}_p$. We give the notations for the rest of the chapter. Let $\mathbb{F}_p$ be a finite field field of prime characteristic p, with p of thousands digits. Let $\mathbb{F}_{p^k}$ be the extension of degree k of $\mathbb{F}_p$. The extension $\mathbb{F}_{p^k}$ is defined through an irreducible polynomial $P(X)$ of degree k. Let A and B be two elements of $\mathbb{F}_{p^k}$. The elements of $\mathbb{F}_{p^k}$ are described in the basis $\mathscr{B} = (1, \gamma, \gamma^2, \ldots, \gamma^{k-1})$, for γ a roots of $P(X)$ in $\mathbb{F}_{p^k}$. An element of $\mathbb{F}_{p^k}$ is a polynomial in γ with coefficients in $\mathbb{F}_p$:

$$\mathbb{F}_{p^k} = \{\sum_{i=0}^{k-1} a_i\gamma^i, a_i \in \mathbb{F}_p\}.$$

A is represented by $\sum_{i=1}^{k-1} a_i\gamma^i$ and B by $B = \sum_{i=0}^{k-1} b_i\gamma^i$. The product of A and B can be done in two steps. The first one is the the product of the polynomials, to obtain the polynomial $C(X) = A(X) \times B(X)$ of degree $(2k-2)$. The second step is the polynomial reduction modulo $P(X)$. The cost of this reduction depends on the form of $P(X)$. The more $P(X)$ is sparse, the more the reduction is efficient. As a consequence, $P(X)$ should be as possible chosen of the form $X^k - \beta$, with $\beta \in \mathbb{F}_p$ [50]. In this case, the polynomial reduction is reduced to multiplications by β and $(k-1)$ additions:

$$C(X) = C_0(X) + C_1(X)X^k \equiv C_0(X) + \beta C_1(X) \bmod (P(X)).$$

with, $C_0(X), C_1(X)$ of degree $(k-1)$.

The following theorem [50, Theorem 3.75] gives us a natural construction of the extension $\mathbb{F}_{p^k}$ using a sparse representation.

Theorem 5.1. *Let k be an integer and $\mathbb{F}_{p^k}$ an extension of degree k of $\mathbb{F}_p$, for p a prime number. There exists β an element of $\mathbb{F}_p$ which is not a k-th roots in $\mathbb{F}_p$ and such that the polynomial $X^k - \beta$ is irreducible over $\mathbb{F}_p$.*

Thus, we can consider that the complexity of a product in $\mathbb{F}_{p^k}$ is highly dependent on the complexity of the product of two polynomials, neglecting the complexity of the modular reduction. We introduce above the possible multiplications of polynomials.

5.2. The school book method

As the name gives the hint, the school book multiplication is the one we learned at school. The school book method of two polynomials is the following

$$A(\gamma) \times B(\gamma) = \sum_{i=0}^{2k-1} \left(\sum_{j=0}^{i} (a_j b_{i-j}) \right) \gamma^i.$$

This simple method is very expensive, indeed its complexity is quadratic in the degree of the polynomials. The cost of this method is k^2 multiplications in $\mathbb{F}_p$ plus $k(2k-1)$ addition, thus the complexity is $k(2k-1)A_p + k^2 M_p$.

The interpolation method are an alternative to the school book method, there are efficient for k greater than a fixed value. This value depends on the method.

5.3. Interpolation method

Let $A(X) = a_0 + a_1 X + \ldots + a_{k-1} X^{k-1}$ and $B(X) = b_0 + b_1 X + \ldots + b_{k-1} X^{k-1}$ be the polynomials obtained by substitution (γ becomes X). The result $C(X)$ of $A(X) \times B(X)$ is a polynomial of degree $(2k-1)$. It is known that a polynomial of degree m is determined by its image in $(m+1)$ distinct values.

Theorem 5.2. *Let $P(X)$ be a polynomial of degree m, then $P(X)$ is determined by the image of $(m+1)$ distinct values.*

The multiplications by the interpolation method use in this theorem. The methodology is to find $(2k-1)$ images of the polynomial $C(X)$ and then to reconstruct $C(X)$ by interpolation. All multiplications by interpolation follow this scheme

1. Find $(2k-1)$ distinct values in $\mathbb{F}_p$
   denoted by $\alpha_0, \alpha_1, \ldots, \alpha_{2k-2}$.
2. Evaluate the polynomials $A(X)$ and $B(X)$ in these values
   keep in memory $A(\alpha_0), \ldots, A(\alpha_{2k-2}), B(\alpha_0), \ldots, B(\alpha_0)$.

3. Compute the evaluation of C in these $(2k-1)$ values,
   $C(\alpha_i) = A(\alpha_i)B(\alpha_i)$.
4. Use these evaluations of $C(X)$ to reconstruct by interpolation the polynomial $C(X)$.

The complexity of a multiplication by interpolation depends

1. on the evaluation of the $A(\alpha_i)$, $B(\alpha_i)$,
2. on the multiplications in $\mathbb{F}_p$ $C(\alpha_i) = A(\alpha_i) \times B(\alpha_i)$,
3. and on the reconstruction of the polynomial expression of $C(X)$.

If we compare the interpolation method with the school book method, we substitute some multiplications in $\mathbb{F}_p$ by multiplications by constants in $\mathbb{F}_p$. The constants are determined by the choice of the α_i values. The drawback is that the multiplication by interpolation need more additions, but as an addition in $\mathbb{F}_p$ is less expensive than a multiplication, for some degree k interpolation methods are more efficient than the school book method.

Let M_a the cost of a multiplication by the constant a in $\mathbb{F}_p$. The evaluations in $(\alpha_i)_{\{i=0\ldots(2k-1)\}}$ cost

$$2(2k-1)(k-1)\left(A_p + CM_p\right),$$

when executed using the Horner scheme:

$$A(\alpha_i) = a_0 + \alpha_i\left(a_1 + \alpha_i(a_2 + \alpha_i[\ldots])\right).$$

The computation of the $C(\alpha_i) = A(\alpha_i) \times B(\alpha_i)$ involves $(2k-1)$ multiplications in $\mathbb{F}_p$, which costs $(2k-1)M_p$.

Two classical method of interpolation exist, the Lagrange and the Newton interpolation methods.

5.3.1. Lagrange's interpolation method

We suppose that we have obtained the evaluation of the polynomial $A(X)$ and $B(X)$ in $2k-1$, denoted $(\alpha_0, \alpha_1, \ldots, \alpha_{2k-2})$. We then have the image of $C(X) = A(X) \times B(X)$ in these $2k-1$ points. The reconstruction of the coefficients of $C(X)$ using the Lagrange interpolation is done through the formula:

$$C(X) = \sum_{i=0}^{2k-2}\left(C(\alpha_i) \times \frac{\prod_{j=0, j\neq i}^{2k-2}(X-\alpha_j)}{\prod_{j=0, j\neq i}^{2k-2}(\alpha_i-\alpha_j)}\right). \tag{8}$$

The complexity of Lagrange's interpolation is

$$(2k-1)M_p + (2k-1)(4k-3)CM_p + 2(2k-1)(3k-2)A_p. \tag{9}$$

5.3.2. Newton's interpolation

As in the Lagrange's interpolation, we dispose of the $C(\alpha_i)$s and we want to find the coefficients of $C(X)$. The Newton's interpolation needs the construction of intermediates values.

The first step is the computation of the values c'_i

$$\begin{cases} c'_0 = C(\alpha_0) \\ c'_1 = (C(\alpha_1) - c'_0)\frac{1}{(\alpha_1-\alpha_0)} \\ c'_2 = \left((C(\alpha_2) - c'_0)\frac{1}{(\alpha_2-\alpha_0)} - c'_1\right)\frac{1}{(\alpha_2-\alpha_1)} \\ \vdots = \vdots \\ c'_{2k-2} = \left((C(\alpha_{2k-2}) - c'_0)\frac{1}{(\alpha_{2k-2}-\alpha_0)} - c'_1\right)\frac{1}{(\alpha_{2k-2}-\alpha_1)} - \dots \end{cases}$$

With the c'_is, the expression of $C(X)$ is

$$\begin{aligned} C(X) = {} & c'_0 + c'_1(X-\alpha_0) + c'_2(X-\alpha_0)(X-\alpha_1) \\ & + \dots + c'_{2k-2}(X-\alpha_0)(X-\alpha_1)\dots(X-\alpha_{2k-2}). \end{aligned}$$

The reconstruction of the coefficients of $C(X)$ can be done using the Horner's scheme

$$\begin{aligned} C(X) = {} & c'_0 + (X-\alpha_0)[c'_1 + (X-\alpha_1)(c'_2 + (X-\alpha_2)(\dots \\ & \dots + (X-\alpha_{2k})[c'_{2k-1} + (X-\alpha_{2k-1})c'_{2k-2}]))]. \end{aligned}$$

The efficiency of the multiplication by interpolation depends on the choice of the α_is. The Newton's interpolation involves divisions be the differences of the α_is, these elements can be precomputed once for all as the α_is are fixed. Furthermore, the divisions by $(\alpha_i - \alpha_j)^{-1}$ can be transformed in multiplication by constants, as we work in a finite field.

The complexity of Newton's interpolation is the sum of the complexity of the computation of the $C(\alpha_i)$, the c'_i and the reconstruction of the coefficients of $C(X)$.

The complexity of Newton's interpolation is

$$4(2k^2 - 3k + 1)A_p + 4(2k^2 - 3k + 1)CM_p + (2k-1)M_p.$$

5.3.3. Comparison between the two methods

The two methods involves the same number of multiplications in the base field $\mathbb{F}_p$: $(2k-1)$, for polynomials of degree $(k-1)$.

The Lagrange's interpolation is very important when computations can be parallelised. Indeed, the computation of the $C(\alpha_i) \times \dfrac{\prod_{j \neq i}(X - \alpha_j)}{\prod_{j \neq i}(\alpha_i - \alpha_j)}$ are independent. The Newton's interpolation involves less additions and multiplications by constants than the Lagrange's one, but we cannot parallelise the computation. The c'_i must be computed one after another.

Operation \ Method	Lagrange	Newton
A_p	$12k^2 - 14k + 4$	$8k^2 - 12k + 4$
CM_p	$8k^2 - 10k + 3$	$8k^2 - 12k + 4$
M_p	$(2k - 1)$	$(2k - 1)$

Table 5. Complexity in number of operation over the base field

The Lagrange's interpolation should be privileged when computations can be parallelised and Newton when the size of the device is limited, typically for smart cards.

5.4. Karatsuba and Toom Cook methods

5.4.1. Karatsuba's method

The Karatsuba multiplication is a straightforward application of the Newton's method, for polynomials of degree 1. The result of the multiplication is a polynomial of degree 2, then we need $2 + 1 = 3$ points of interpolation. These values are $\{0, 1, \infty\}$. The Karatsuba multiplication provide the product of two polynomials of degree 1 in 3 multiplications in the base field, instead of 4 using the school book method. The multiplication by constants in the Newton multiplication are free, because of the choice of the interpolation values. Let $A(X) = A_0 + A_1 X$ and $B(X) = B_0 + B_1 X$ be two polynomials of degree 1 and $C(X) = A(X) \times B(X)$.

We evaluate the polynomial $C(X)$ in the point $\{0, 1, \infty\}$ using equations 10.

$$\begin{aligned} C(0) &= (A_1 X + A_0)(B_1 X + B_0) \bmod (X), \\ &= A_0 \times B_0, \\ C(1) &= (A_1 X + A_0)(B_1 X + B_0) \bmod (X - 1), \\ &= (A_0 + A_1) \times (B_0 + B_1), \\ C(\infty) &= (A_1 X + A_0)(B_1 X + B_0) \bmod (X - \infty), \\ &= A_1 \times B_1 \times X^2 \bmod (X - \infty). \end{aligned} \tag{10}$$

The evaluation of polynomial $C(X)$ in the 3 values involves $2A_p + 3M_p$ operations in the base field $\mathbb{F}_p$.

Then, we use the formulas in the Newton interpolation to reconstruct the polynomial $C(X)$.

$$\begin{cases} c'_0 &= C(0), \\ &= A_0B_0, \\ c'_1 &= (C(1)-c'_0)\frac{1}{(1-0)}, \\ &= (A_0+A_1)(B_0+B_1)-A_0B_0, \\ c'_2 &= \left((C(\infty)-c'_0)\frac{1}{(\infty-0)}-c'_1\right)\frac{1}{(\infty-1)}, \\ &= \left((A_1B_1X^2-A_0B_0)\frac{1}{(X-0)}-((A_0+A_1)(B_0+B_1)-A_0B_0)\right)\frac{1}{(X-1)} \bmod (X-\infty), \\ &= \frac{A_1B_1X^2}{X^2}-\frac{A_0B_0}{X^2}-\frac{((A_0+A_1)(B_0+B_1)-A_0B_0)}{X} \bmod (X-\infty), \\ &= A_1B_1. \end{cases}$$

$$\begin{aligned} C(X) &= c'_0+c'_1X+c'_2X(X-1), \\ &= A_0B_0+((A_0+A_1)(B_0+B_1)-A_0B_0)X+A_1B_1X(X-1), \\ &= A_0B_0+((A_0+A_1)(B_0+B_1)-A_0B_0-A_1B_1)X+A_1B_1X^2. \end{aligned}$$

We can resume the computation of the polynomial $C(X)$ using Karatsuba's multiplication by the following equation

$$\begin{cases} c_0 &= A_0\times B_0, \\ c_1 &= (A_0+A_1)\times(B_0+B_1), \\ c_2 &= A_1\times B_1, \\ C(X) &= c_0+(c_1-c_0-c_2)X+c_2X^2. \end{cases} \tag{11}$$

For polynomials of degree 1, the complexity of Karatsuba's multiplication is $3M_p+4A_p$.

The Karatsuba's multiplication can be recursively applied for polynomials of degree greater than 1. Let $A(X)=A_0+A_1X+\ldots A_mX^m$, we can split $A(X)$ in two parts of degree smaller or equal to $\lfloor\frac{m}{2}\rfloor$:

$$\begin{aligned} A(X) &= A_0+A_1X+\ldots A_{\lfloor\frac{m}{2}\rfloor-1}+X^{\lfloor\frac{m}{2}\rfloor}\left(A_{\lfloor\frac{m}{2}\rfloor}+A_{\lfloor\frac{m}{2}\rfloor+1}X+\ldots A_mX^{\lfloor\frac{m}{2}\rfloor}\right), \\ &= \widetilde{A_0}+Y\widetilde{A_1}, \text{ where we denote } Y=X^{\lfloor\frac{m}{2}\rfloor}. \end{aligned}$$

Then, we apply the Karatsuba's multiplication to the two parts. Each of the three multiplications can also be done using the Karatsuba's multiplication. The recursive application of Karatsuba's multiplication is the most efficient method for the computation of polynomials of degree a power of 2. The asymptotic complexity of Karatsuba's multiplication is $O(m^{\log_2(3)})$ multiplications and $O(m)$ additions, with m being the degree of the polynomials we want to multiply.

5.4.2. Toom Cook 3 multiplication

Exactly like the Karatsuba's multiplication, Toom Cook 3 multiplication is an application of Newton's interpolation. The Toom Cook 3 method provide the product of polynomials of degree 2 with 5 multiplications of coefficients, instead of 9 using the school book method multiplication. The values for the interpolation are $\{0,1,-1,2,\infty\}$. Unlike the Karatsuba's method, there are few multiplications and divisions by constants that we cannot avoid.

Let $A(X) = A_0 + A_1X + A_2X^2$ and $B(X) = B_0 + B_1X + B_2X^2$ be polynomials of degree 2 and $C(X) = A(X) \times B(X)$ obtained using the Toom Cook method. The evaluation part of Toom Cook 3 multiplication involves 10 additions of A_i and B_i, for $i = 0,1,2$. The evaluation of $A(X)$ needs 5 additions.

$$\begin{cases} A(0) &= A_0, \\ Sp_1 &= A_0 + A_2, \\ A(1) &= Sp_1 + A_1, \\ A(-1) &= Sp_1 - A_1, \\ A(2) &= A_0 + 2A_1 + 4A_2, \\ A(\infty) &= A_2X^2 \bmod (X - \infty). \end{cases}$$

We begin with the evaluation of $C(X)$ in the α_i pour $i = 0,1,2,3,4$.

$$\begin{cases} C(0) &= A(0) \times B(0) = A_0B_0, \\ C(1) &= A(1) \times B(1), \\ C(-1) &= A(-1) \times B(-1), \\ C(2) &= A(2) \times B(2), \\ C(\infty) &= A(\infty) \times B(\infty) = A_2B_2X^4 \bmod (X - \infty). \end{cases}$$

We apply the Newton's method to find the coefficients c'_i

$$\begin{cases} c'_0 &= C(0), \\ c'_1 &= C(1) - c'_0, \\ c'_2 &= \frac{1}{2}\left(C(-1) - c'_0 + c'_1\right), \\ c'_3 &= \frac{1}{6}C(2) - \frac{1}{6}c'_0 - \frac{1}{3}c'_1 - \frac{1}{3}c'_2, \\ c'_4 &= A_2B_2. \end{cases}$$

The reconstruction of $C(X)$ is then

$$\begin{aligned} C(X) = c'_0 + c'_1X \; + \; & c'_2X(X-1) + c'_3X(X-1)(X+1) \\ & c'_4X(X-1)(X+1)(-2). \end{aligned}$$

This step can be resume by the formula

$$\begin{aligned} C(X) = c'_0 \; &+ \; (c'_1 - c'_2 - c'_3 - 2c'_4)X + (c'_2 - c'_4)X^2 \\ &+ \; (c'_3 - 2c'_4)X^3 + c'_4 X^4. \end{aligned}$$

Which gives

$$\begin{cases} C_0 = c'_0, \\ C_1 = c'_1 - c'_2 - c'_3 - 2c'_4, \\ C_2 = c'_2 - c'_4, \\ C_3 = c'_3 - 2c'_4, \\ C_4 = c'_4, \\ C(X) = C_0 + C_1 X + C_2 X^2 + C_3 X^3 + C_4 X^4. \end{cases}$$

For polynomials of degree 2, the complexity of Toom Cook 3 is $5M_p + 11CM_p + 11A_p$. As for Karatsuba's method, the Toom Cook 3 method can be recursively applied. The asymptotic complexity of Toom Cook 3 multiplication is $O(m^{\log_3(5)})$ multiplications and $O(m)$ additions, where m is the degree of the polynomials we want to multiply.

5.4.3. Extensions to other extensions

The Toom Cook 3 method can be extended to Toom Cook 5, this multiplication is suited for polynomials of degree 3. Few works deal with the multiplication of polynomials of degree greater than 3. For polynomials of degree 4, we can use the Karatsuba's method. As a consequence, in pairing based cryptography, field with extension degree of the form $2^i 3^j$ are called pairing friendly because we can use tower fields and for each stage of the tower we use the Karatsuba or Toom Cook 3 multiplication. However in pairing based cryptography (and in cryptography in general) there are some cases where it is more interesting to use fields with degree extensions different from 2 and 3. We can cite the problem of compression (i.e. representing elements in a finite field subgroup with fewer bits than classical algorithms) for extension fields in terms of *algebraic tori* $T_n(\mathbb{F}_q)$ [63] or applications based on $T_{30}(\mathbb{F}_q)$, such as El Gamal encryption, El Gamal signatures and voting schemes in [69].

Let $\mathbb{F}_p$ be a finite field of characteristic greater than 5. For instance for polynomials of degree 5, we can begin with Karatsuba's method and then use Karatsuba and Toom Cook 3 for each part. This construction gives an efficient multiplication for polynomials of degree 5, but not the most efficient. For degree 5 extensions, Montgomery [58] has proposed a Karatsuba-like formula for 5-terms polynomials performed using 13 base field multiplications. This work was improved by El Mrabet et all in [30] using Newton's interpolation.

We recall here Montgomery's method for an extension of degree 5. Let $A = a_0 + a_1 X + a_2 X^2 + a_3 X^3 + a_4 X^4$ and $B = b_0 + b_1 X + b_2 X^2 + b_3 X^3 + b_4 X^4$ in $\mathbb{F}_{p^5}$ with coefficients over $\mathbb{F}_p$. Montgomery constructs the polynomial $C(X) = A(X) \cdot B(X)$ using the following formula $C = (a_0 + a_1 X + a_2 X^2 + a_3 X^3 + a_4 X^4)(b_0 + b_1 X + b_2 X^2 + b_3 X^3 + b_4 X^4)$

$$
\begin{aligned}
= & (a_0+a_1+a_2+a_3+a_4)(b_0+b_1+b_2+b_3+b_4)(X^5-X^4+X^3) \\
& +(a_0-a_2-a_3-a_4)(b_0-b_2-b_3-b_4)(X^6-2X^5+2X^4-X^3) \\
& +(a_0+a_1+a_2-a_4)(b_0+b_1+b_2-b_4)(-X^5+2X^4-2X^3+X^2) \\
& +(a_0+a_1-a_3-a_4)(b_0+b_1-b_3-b_4)(X^5-2X^4+X^3) \\
& +(a_0-a_2-a_3)(b_0-b_2-b_3)(-X^6+2X^5-X^4) \\
& +(a_1+a_2-a_4)(b_1+b_2-b_4)(-X^4+2X^3-X^2) \\
& +(a_3+a_4)(b_3+b_4)(X^7-X^6+X^4-X^3) \\
& +(a_0+a_1)(b_0+b_1)(-X^5+X^4-X^2+X) \\
& +(a_0-a_4)(b_0-b_4)(-X^6+3X^5-4X^4+3X^3-X^2) \\
& +a_4b_4(X^8-X^7+X^6-2X^5+3X^4-3X^3+X^2) \\
& +a_3b_3(-X^7+2X^6-2X^5+X^4) \\
& +a_1b_1(X^4-2X^3+2X^2-X) \\
& +a_0b_0(X^6-3X^5+3X^4-2X^3+X^2-X+1).
\end{aligned}
$$

The cost of these computations is $13M_q + 22A_q$. Note that in order to recover the final expression of the polynomial of degree 8, we have to re-organize the 13 lines to find its coefficients. We denote the products on each of the 13 lines by u_i, $0 \le i \le 12$ (i.e. $u_{12} = (a_0+a_1+a_2+a_3+a_4)(b_0+b_1+b_2+b_3+b_4)$, $u_{11} = (a_0-a_2-a_3-a_4)(b_0-b_2-b_3-b_4)$ etc.) By re-arranging the formula in function of the degree of X, we obtain the following expression for C

$$
\begin{aligned}
C = & u_3X^8 \\
& +(-u_2-u_3+u_6)X^7 \\
& +(u_0+2u_2+u_3-u_4-u_6-u_8+u_{11})X^6 \\
& +(-3u_0-2u_2-2u_3+3u_4-u_5+2u_8+u_9-u_{10}-2u_{11}+u_{12})X^5 \\
& +(3u_0+u_1+u_2+3u_3-4u_4+u_5+u_6-u_7-u_8-2u_9+2u_{10}+2u_{11}-u_{12})X^4 \\
& +(-2u_0-2u_1-3u_3+3u_4-u_6+2u_7+u_9-2u_{10}-u_{11}+u_{12})X^3 \\
& +(u_0+2u_1+u_3-u_4-u_5-u_7+u_{10})X^2 \\
& +(-u_0-u_1+u_5)X \\
& +u_0.
\end{aligned}
$$

Considering this expression, hidden additions must be taken in account. Once every simplification is done, the total complexity of Montgomery's method is $13M_p + 62A_p$.

In [30], the Newton's interpolation gives a better result for the multiplication of 5-terms polynomials. The interpolation values are $\alpha_0 = 0$, $\alpha_1 = 1$, $\alpha_2 = -1$, $\alpha_3 = 2$, $\alpha_4 = -2$, $\alpha_5 = 4$, $\alpha_6 = -4$, $\alpha_7 = 3$, $\alpha_8 = \infty$. With these values, the evaluations of A and B are only composed of shifts and additions. Details are provide in [30], the evaluations of $A(X)$ and $B(X)$ have a total complexity of $48A_p$. The evaluation of $C(X)$ in the α_is costs $9M_p$. The computation of the c_i's is not straightforward. Indeed, there are few divisions by 3, 5 and 7 that appear in the formula Section 5.3.2. To avoid the computation of a division which is an expensive operation over a finite field, using a trick on the binary decomposition of integers, they perform very efficiently the divisions. The complexity for these divisions is smaller than

$2A_p$. The global complexity for the computation of the c_i's is then $64A_p$. Finally, the reconstruction of the polynomial $C(X)$ using the Horner's scheme has a complexity of $28A_p$. And the total complexity of the 5-terms polynomials is $9M_q + 137A_q$.

The comparison with Montgomery's result is not evident, but implementations in [30] shows that the results are more efficient than the Montgomery's one.

In the two articles, the authors give also results for 6-terms and 7-terms polynomials.

The fact that we can compute efficiently the multiplication for extensions greater than 2 and 3 gives the opportunity to consider pairing computation over elliptic curve with an embedding degree k different from $2^i 3^j$ and can improve the implementation of pairings. But this work is still to be made.

5.5. Original representation of finite fields

In the previous section we consider efficient multiplications for a classical representation of finite fields and extension of finite fields. But they are many ways to represent a finite field. In [22], the authors use an original representation of finite field to provide a very efficient implementation of a pairing. This original representation is the Residue Number System (RNS) representation and it was developed in [7, 8]. The RNS representation relays on the Chinese remainder theorem. Let $\mathscr{B} = \{m_1, \ldots, m_n\}$ be a set of co-prime natural integers, $M = \prod_{i=1}^{n} m_i$ and $0 \leq X < M$. There exists a unique representation $X_{\mathscr{B}}$ of X in the basis $\mathscr{B}$, $X_{\mathscr{B}} = \{X \mod m_1, \ldots X \mod m_n\} = \{x_1, x_2, \ldots, x_n\}$. Given $X_{\mathscr{B}}$, we can reconstruct X using the Chinese Remainder theorem:

$$X = \left(\sum_{i=1}^{n} (x_i \times b_i^{-1} \mod m_i) \times b_i \right) \mod M, \text{ where } b_i = \frac{M}{m_i}.$$

The RNS representation is obviously very interesting for parallel computations. An efficient multiplication in RNS representation is described in [7, 8]. This multiplication is based on the Montgomery modular multiplication. In [22], the authors present two very efficient implementation of a pairing algorithm on an FPGA, in RNS representation. They implement the optimal Ate pairing at several security levels over Altera and Xilinx FPGA. They compare there result with previous work and obtaint very nice results.

5.6. The arithmetic of Pairings

The complexity of a computation of a pairing depends on the finite field and the arithmetic underlying, but also of the model and the equation of the elliptic curve and the choice of the coordinates. Usually, an elliptic curve is represented using the short Weierstrass equation which is on the form $E : y^2 = x^3 + ax + b$, with a and b elements of the finite field $\mathbb{F}_p$. In [20], Brier and Joye show that the value a can be chosen to be -3. This value contributes to improve the computation of pairings. But, even on a short Weierstrass equation, several cases exist, we can have $b = 0$, $a = 0$ with b a square or not just an integer. For each option, the coordinates have also an influence on the efficiency of the computation of a pairing. The coordinates are usually chosen between affine, Projective and Jacobian. The affine coordinates are often put aside. Indeed, the operations over the elliptic curve in affine coordinates involves inversion over finite fields. As inversion over a finite field is an expensive operation, one try to avoid them so far as possible. To achieve this aim, the Projective or Jacobian coordinates are suitable,

as by construction, the Projective and Jacobian coordinates substitute inversions in affine coordinates into multiplications. The fact that the affine coordinates involves inversions was a drawback to their use in pairing based cryptography. In [51], the authors analyzed the use of affine coordinates for pairing based cryptography. They adapt two known techniques for speeding up field inversion to the pairing based cryptography case. They found out that for high security levels, an implementation in affine coordinates of a pairing will be much faster than an implementation in projective coordinates. The first technique to improve the inversion consists in computing inverses in extension fields by using towers of extension field and transform inverse computation to subfield computations via the norm map. Using this technique, the authors reduce drastically the ratio of the costs of inversions to multiplications in extension fields. This is very interesting for the computation of pairings over a large extension field, typically at high level security such as 256 bits. The second trick is to take advantage of the inversion-sharing, a standard trick whenever several inversions are computed at once. This method involves the lecture of the binary expansion from right to left, instead of left to right. This second method is very interesting when multi-core processors are used, indeed, it can be easily parallelized. We can find in [51] detailed performance numbers with timing for base field and extension field arithmetic. For security level more reasonable, the Projective and Jacobian coordinates are for now more suitable.

In [24], the authors resume, compare and improve several works dealing with the optimizations of pairings, considering all the possibilities for the Weierstrass equation. They give efficient computations in Jacobian and Projective coordinates. We resume there work in Table 6.

Curve Curve order Twist deg.	Doubling Addition Result of [24]	Prev Result	Doubling Addition
$y^2 = x^3 + ax$ any $d = 2,4$	$M_a + (2k/d+2)M_p + 8S_p$ $(2k/d+12)M_p + S_p$ New coord.	[2]	$M_a + (2k/d+1)M_p + 11S_p$ $(2k/d+10)M_p + 6S_p$ Jacobian
$y^2 = x^3 + c^2$ $3 \mid \sharp E$ $d = 2,6$	$(2k/d+3)M_p + 5S_p$ $M_c + (2k/d+3)M_p + 5S_p$ Projective	[23]	$(2k/d+3)M_p + 5S_p$ $M_c + (2k/d+3)M_p + 5S_p$ Projective
$y^2 = x^3 + b$ $3 \nmid \sharp E$ $d = 2,6$	$M_b + (2k/d+2)M_p + 7S_p$ $M_b + (2k/d+2)M_p + 7S_p$ Projective	[2]	$(2k/d+3)M_p + 8S_p$ $(2k/d+3)M_p + 8S_p$ Jacobian
$y^2 = x^3 + b$ any $d = 3$	$M_b + (k+6)M_p + 7S_p$ $(k+16)M_p + 3S_p$ Projective	[31]	$M_b + (2k+8)M_p + 9S_p$ not reported Projective

Table 6. Comparaison of pairings considering Weierstrass models

There exists several model of elliptic curves, for instance

- Short Weierstrass: $y^2 = x^3 + ax + b$, for a, b in $\mathbb{K}$.
- Legendre coordinates: $y^2 = x(x-1)(x-\lambda)$, for $\lambda \in \mathbb{K}$.
- Montgomery: $by^2 = x^3 + ax^2 + x$, for a, b in $\mathbb{K}$.
- Edwards coordinates: $x^2 + y^2 = c(1 + x^2y^2)$ over $\mathbb{K}$.
- Huff's coordinates: $aX(Y^2 - Z^2) = bY(X^2 - Z^2)$ for $a^2 \neq b^2 \neq 0$ over $\mathbb{K}$.

Several works study the efficiency of an implementation of pairing over some of these models of elliptic curves. The Edwards elliptic curves were recently introduced in cryptographie. In [32], Edwards demonstrates that every elliptic curve E defined over an algebraic number field is birationally equivalent over some extension of that field to a curve given by the equation:

$$x^2 + y^2 = c^2(1 + x^2y^2). \tag{12}$$

Edwards curves became interesting for elliptic curve cryptography when it was proven by Bernstein and Lange in [18] that they provide addition and doubling formulas faster than all addition formulas known at that time. The advantage of Edwards coordinates is that the addition law can be complete (i.e. the formulas for adding or doubling two points are the same) and thus the exponentiation in Edwards coordinates is naturally protected against side channel attacks. Recently, the Edwards elliptic curves were used to compute pairings [3, 44]. In [46], the authors study the Huff's model of an elliptic curve, they provide explicit formulae for fast doubling and addition and also for Tate pairing computation. Another example is the work in [72], in this work the authors consider the Selmer elliptic curves, they present formulae for doubling, addition and pairing computations. They compare there results to various elliptic curve models such as Weierstrass, Edwards, Hessian. There is many choices for the equation/model of the elliptic curve and of the coordinates, the website [17] regroups every new result on this subject. It is a very nice overview of this topic of research.

6. Conclusions

We presented the various pairings available for cryptographic use. As the pairing are aimed to be implemented in smart cards, the efficiency of a pairing implementation is a subject of several research. We presented optimizations developed for the improvement of a pairing implementation. We introduced the twisted elliptic curve which leads to the denominator elimination. We constructed the extension field $\mathbb{F}_{p^k}$ using tower fields and the method for an efficient multiplication over each step of the tower. We described efficient squaring method combine with the cyclotomic subgroup. We also highlighted the fact that the choice of the model of the elliptic curve and the choice of the coordinates is important for an efficient implementation. We saw that the representation of an element in the base field $\mathbb{F}_p$ with original definition can leads to very efficient implementation. To conclude, the optimizations of pairing are a very interesting point of research and a lot of scientists work hardly to find new optimizations. Further research can follow the presented optimizations and adapt to the case of pairings over hyperelliptic curves, or find any other point of optimizations in the implementation.

Author details

Nadia El Mrabet

LIASD, University Paris 8, Saint Denis, France

References

[1] Ahmadi O., Hankerson D., Menezes A.: Software Implementation of Arithmetic in $\mathbb{F}_{3^m}$, WAIFI Conference, Madrid Spain 2007.

[2] C. Arene, T. Lange, M. Naehrig and C. Ritzenthaler Faster pairing computation Cryptology ePrint Archive, Report 2009/155, 2009

[3] C. Arène and T. Lange and M. Naehrig and C. Ritzenhaler, Faster Pairing Computation of the Tate pairing, Cryptology ePrint Archive, Report 2009/155, http://eprint.iacr.org/2009/155, 2009

[4] J.C.Bajard and N.El Mrabet: Pairing in cryptography: an arithmetic point de view, In *Advanced Signal Processing Algorithms, Architectures and Implementations XVI*, part of SPIE, August 2007.

[5] Bajard J.C., Imbert L., Negre Ch.: Arithmetic Operations in Finite Fields of Medium Prime Characteristic Using the Lagrange Representation, IEEE Transactions on Computers, September 2006 (Vol. 55, No. 9) p p. 1167-1177

[6] Bajard, J.C., Meloni, N., Plantard, T.: Efficient RNS bases for Cryptography IMACS'05, Applied Mathematics and Simulation, (2005)

[7] J-C. Bajard, L-S. Didier and P. Kornerup, Modular Multiplication and Base Extensions in Residue Number Systems, 15th IEEE Symposium on Computer Arithmetic (Arith-15 2001), p. 59–65, 2001.

[8] J-C. Bajard, L-S. Didier and P. Kornerup, An RNS Montgomery Modular Multiplication Algorithm, IEEE Trans. Computers, vol. 47, p. 766–776, 1998.

[9] Barreto, P.: The Pairing-Based Crypto Lounge
http://paginas.terra.com.br/informatica/paulobarreto/pblounge.html

[10] P.D. Barrett. Implementing the Rivest Shamir and Adleman Public Key Encryption Algorithm on a Standard Digital Signal Processor. In *Advances in Cryptology (CRYPTO)*, LNCS 263, 311–323, 1986.

[11] Barreto P., Lynn B., Scott M.: On the Selection of Pairing-Friendly Groups Selected Aeras in Cryptography SAC 2003, LNCS 30006, 2004,17-25

[12] P.S.L.M. Barreto, S.D. Galbraith, C. O'hEigeartaigh and M. Scott. Efficient Pairing Computation on Supersingular Abelian Varieties, In *Designs, Codes and Cryptography*, 42 (3), 239–271, 2007.

[13] Barreto P., Kim H., Lynn B., Scott M.: Efficient algorithms for pairing-based cryptosystems, Advances in Cryptology CRYPTO 2002, Lecture Notes in Computer Science, 2442 (2002), 354-368.

[14] Barreto P., Naehrig M.: Pairing-friendly elliptic curves of prime order, Selected Areas in Cryptography (SAC 2005), LNCS 3897 (2006), 319-331.

[15] Barreto P., Lynn B., Scott M.: Efficient implementation of pairing-based cryptosystems, Journal of Cryptology, 17 (2004), 321-334

[16] Barreto P., Scott M.: Compressed pairings, Advances in Cryptology | Crypto 2004, LNCS 3152, 140-156, http://eprint.iacr.org/2004/032.

[17] D. J. Bernstein and T. Lange Explicit-formulas database. http://www. hyperelliptic.org/EFD.

[18] D. J. Bernstein and T. Lange, Faster additions and doubling on elliptic curves, In Advances in cryptology - ASIACRYPT 2007, LNCS, vol. 4833, p. 29–50,2007

[19] Boneh D., Franklin M.: Identity-based encryption from the Weil pairing. SIAM Journal of Computing, 32, 586?615, 2003

[20] Brier E., Joye M.: Point multiplication on elliptic curves through isogenies, AAECC 2003, LNCS., vol. 2643, 2003, 43–50.

[21] Blake F., Seroussi G., Smart N. (editors): Advances in Elliptic Curve Cryptography, Series: London Mathematical Society Lecture Note Series (No. 317),Cambridge University Press,2005

[22] R. C. C. Cheung, S. Duquesne, J. Fan and, N. Guillermin, I. Verbauwhede and G. Xiaoxu Yao, FPGA Implementation of Pairings Using Residue Number System and Lazy Reduction, Cryptographic Hardware and Embedded Systems - CHES 2011 LNCS vol. 6917, p. 421–441, 2011.

[23] C. Costello, H. Hisil, C. Boyd, J-M. Gonz?alez Nieto and K. Koon-Ho Wong. Faster pairings on special Weierstrass curves Pairing2009, LNCS, vol. 5671, p. 89–101, 2009

[24] C. Costello, T. Lange and M. Naehrig. Faster pairing computations on curves with high-degree twists. Progress in Cryptology ? PKC 2010 (13th International Conference on Practice and Theory in Public Key Cryptography LNCS, vol. 6056, p. 224–242, 2010.

[25] Cohen, H., Frey, G. (editors): Handbook of elliptic and hyperelliptic curve cryptography. Discrete Math. Appl., Chapman & Hall/CRC (2006)

[26] Diffie W., Hellman M.: directions in cryptography, IEEE Transactions on Information Theory, 22 (1976), 644-654.

[27] I. Duursma and H. Lee. Tate Pairing Implementation for Hyperelliptic Curves $y^2 = x^p - x + d$. In *Advances in Cryptology (ASIACRYPT)*, Springer-Verlag LNCS 2894, 111–123, 2003.

[28] Duquesne S., Frey G. Background on Pairings, Chapter 6 of Cohen, H., Frey, G.: Handbook of elliptic and hyperelliptic curve cryptography. Discrete Math. Appl., Chapman & Hall/CRC (2006)

[29] Duquesne S., Frey G. Implementation of Pairings, Chapter 16 of Cohen, H., Frey, G.: Handbook of elliptic and hyperelliptic curve cryptography. Discrete Math. Appl., Chapman & Hall/CRC (2006)

[30] N. El Mrabet, A. Guillevic and S. Ionica, Efficient Multiplication in Finite Field Extensions of Degree 5. Progress in Cryptology-Africacrypt 2011, Springer-Verlag LNCS 6737, p. 188–205, 2011.

[31] N. El Mrabet, N. Guillermin and S. Ionica. A study of pairing computation for elliptic curves with embedding degree 15 Cryptology ePrint Archive, Report 2009/370, 2009.

[32] H. Edwards , A normal Form for Elliptic Curve Bulletin of the American Mathematical Society, vol. 44, nř 3, p. 393–422, July 2007

[33] D. Freeman, M. Scott and E. Teske, A Taxonomy of Pairing-Friendly Elliptic Curves, Journal of Cryptology, vol. 23, nř 2, p. 224–280, 2010

[34] Frey G., Müller M., Rück H.G.: The Tate Pairing and the Discrete Logarithm Applied to Elliptic Curve Cryptosystems, IEEE Transactions Inf. Theory, 45, 1717-1719;1999

[35] Fleischmann P., Paar C., Soria-Rodriguez P.: Fast Arithmetic for Public-Key Algorithms in Galois Fields with Composite Exponents, IEEE Transactions on Computers, vol. 48, no. 10, pp. 1025-1034, October, 1999.

[36] Frey G., Rück H.G.: A Remark Concerning m-divisibility Constructions and the Discrete Logarithmic problem in the Divisor Class Group of Curves, in Math. comp., 62, 865-874, 1994.

[37] Galbraith S.: K.G.: Pairings, Chapter IX, Advances in Elliptic Curve Cryptography, F. Blake and G. Seroussi and N. Smart editors, Series: London Mathematical Society Lecture Note Series (No. 317),Cambridge University Press,2005

[38] S.D. Galbraith, C. O'hEigeartaigh and C. Sheedy. Simplified Pairing Computation and Security Implications, In *Journal of Mathematical Cryptology*, 1 (3), 267–282, 2007.

[39] S.D. Galbraith, F. Hess and F. Vercauteren. Aspects of Pairing Inversion In *IEEE Transactions on Information Theory*, 54 (12), 5719–5728, 2008.

[40] Granger R., Page D., Stam M.: Hardware and Software Normal Basis Arithmetic for Pairing-Based Cryptography in Characteristic Three. IEEE Transactions on Computers,volume 54(7): 852–860, July 2005

[41] R. Granger and M. Scott, Faster Squaring in the Cyclotomic Subgroup of Sixth Degree Extensions, Practice and Theory in Public Key Cryptography 2010, LNCS vol. 6056, p. 209–223, 2010

[42] F. Hess, Pairing Lattices, Pairing 2008, LNCS vol. 5209, p. 18–38, 2008

[43] F. Hess, N. Smart and F. Vercauteren The Eta Pairing Revisited, IEEE Transactions on Information Theory, vol. 52, p. 4595–4602, 2006

[44] S. Ionica and A. Joux , Another Approach to Pairing Computation in Edwards Coordinates, INDOCRYPT '08, LNCS,vol. 5365, p. 400–413,2008

[45] Joux A.: one round protocol for tripartite Diffie-Hellman, Algorithmic Number Theory: Fourth International Symposium, Lecture Notes in Computer Science, 1838 (2000), 385-393. Full version: Journal of Cryptology, 17 (2004), 263-276

[46] M. Joye, M. Tibouchi and D. Vergnaud Huff's Model for Elliptic Curve Algorithmic Number Theory (ANTS-IX), LNCS vol. 6197, p. 234–250, 2012

[47] Koblitz N.: Elliptic curve cryptosystems, Mathematics of Computation, Vol. 48, 1987, 203-209.

[48] N. Koblitz and A. Menezes, Pairing-Based Cryptography at High Security Levels, Cryptography and Coding 2005, LNCS vol. 3796, p. 13–36, 2005

[49] S. Kwon. Efficient Tate Pairing Computation for Supersingular Elliptic Curves over Binary Fields. In *Cryptology ePrint Archive*, Report 2004/303, 2004.

[50] R. Lidl and H. Niederreiter Finite Fields Cambridge University Press, 1994, 0-521-39231-4

[51] K. Lauter, P. L. Montgomery and M. Naehrig An Analysis of Affine Coordinates for Pairing Computation, Pairing 2010, LNCS, vol. 6487, p. 1–20, 2010

[52] Lenstra A., Stam M.: Efficient Subgroup Exponentiation in Quadratic and Sixth Degree Extensions, Cryptographic Hardware and Embedded Systems, CHES 2002, LNCS 2523 pp. 318-332.

[53] MenezesA., Okamoto T. and Vanstone S.A.: Reducing Elliptic Curve Logarithms to Logarithms in a Finite Field, IEEE Trans. Inf . Theory 39, numéro 5, pages 1639-1646, 1993.

[54] Miller V.: The Weil pairing and its efficient calculation, J. Cryptology, 17 (2004), 235-261.

[55] Miller V.: Use of Elliptic Curves in Cryptography Advances in Cryptology-Crypto 85 pages 417-426 Vol. 218 of LNCS 1986.

[56] Miller V.: Short Programs for Functions on Curves, IBM, Thomas J. Watson Research Center, 1986 http://crypto.stanford.edu/miller/miller.pdf

[57] P.L. Montgomery. Modular Multiplication Without Trial Division. *Mathematics of Computation*, 44, 519–521, 1985.

[58] P. L. Montgomery Five, Six and Seven-terms Karatsuba-Like Formulae, IEEE Transactions on Computers 2005, vol. 54, p. 362–369.

[59] C. O'hEigeartaigh. Pairing Computation on Hyperelliptic Curves of Genus 2. PhD Thesis, Dublin City University, 2006.

[60] Pairing 2005 in Dublin Ireland, http://pic.computing.dcu.ie/
Pairing 2007 in Tokyo Japan, http://www.pairing-conference.org/
Pairing 2008 Egham UK,
Pairing 2009 Palo Alto Ca, USA
Pairing 2010 Yamanaka Hot Spring, Ishikawa, Japan
Pairing 2012 Darmstadt, Germany http://2012.pairing-conference.org/

[61] PARI/GP, version `2.1.7`, Bordeaux, 2005, http://pari.math.u-bordeaux.fr/

[62] Paterson K.G.: Cryptography from Pairings, Chapter X, Advances in Elliptic Curve Cryptography, F. Blake and G. Seroussi and N. Smart editors, Series: London Mathematical Society Lecture Note Series (No. 317),Cambridge University Press,2005

[63] K. Rubin and A. Silverberg, Torus-Based Cryptography, Advances in Cryptology CRYPTO 2003, Springer-Verlag, LNCS 2729, p. 349–365, 2003.

[64] J.H. Silverman The arithmetic of elliptic curves, Graduate Texts in Mathematics, Springer Verlag, vol. 106, 1992

[65] Shamir A.: Identity BasedCryptosystems and Signature Schemes, Advances in Cryptology Crypto '84, LNCS, Vol. 196, pp 47-53, 1984

[66] M. Scott. Computing the Tate Pairing. In *Topics in Cryptology (CT-RSA)*, Springer-Verlag LNCS 3376, 293–304, 2005.

[67] M. Scott, N. Benger, M. Charlemagne, L. J. Dominguez Perez and E. J. Kachisa, On the Final Exponentiation for Calculating Pairings on Ordinary Elliptic Curves, Pairing 2009, LNCS vol. 5671, p. 78–88,2009

[68] M. Scott, N. Costigan and W. Abdulwahab. Implementing Cryptographic Pairings on Smartcards. In *Cryptographic Hardware and Embedded Systems (CHES)*, Springer-Verlag LNCS 4249, 134–147, 2006.

[69] M. Van Dijk and R. Granger and D. Page and K. Rubin and A. Silverberg and M. Stam and D. Woodruff Practical cryptography in high dimensional tori, Progress in Cryptography Eurocrypt'2005

[70] F. Vercauteren, Optimal pairings, IEEE Trans. Inf. Theor., vol. 56, nř1,p. 455–461, Jan 2010

[71] C. Whelan and M. Scott. The Importance of the Final Exponentiation in Pairings When Considering Fault Attacks. In *Pairing-Based Cryptography*, Springer-Verlag LNCS 4575, 225–246, 2007.

[72] L. Zhang, K. Wang, H. Wang and D. Ye, Another Elliptic Curve Model for Faster Pairing Computation, ISPEC 2011, LNCS vol. 6672, p. 432–446, 2011.

IntechOpen