國立清華大學

資訊系統與應用研究所

碩士學位論文

Weil/Tate Pairing 基於 Radix-2^{2w}+1 之快速演算法

An Efficient Algorithm for Weil/Tate Pairing Based on Radix 2^{2w}+1 Representation

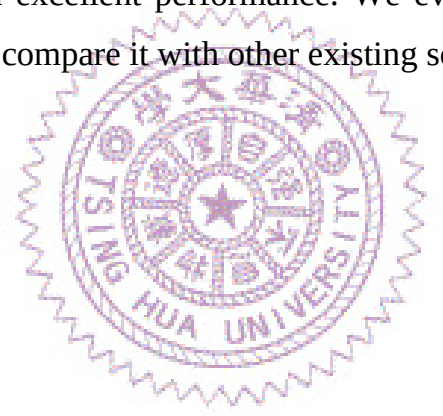
研究生　：秦茂原

指導教授：孫宏民 博士

中華民國 九十七年七月

Abstract

Weil pairing and Tate pairing act important roles in many cryptographic applications. Speeding up pairing has been an interesting and important issue. Some recent applications require pairing executed on supersingular elliptic curves with characteristics that are larger than 4 and congruent to 2 mod 3. There are very few papers working on speeding up pairing on these curves. In this thesis, we provide an efficient pairing calculation method that suits curve with characteristics 5. This method can also be applied to other characteristics with excellent performance. We evaluate this method using Markov model and compare it with other existing schemes.



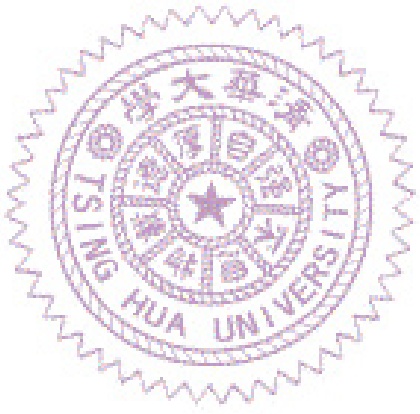
摘要

Weil pairing 和 Tate pairing 在許多密碼應用上扮演重要的角色，因此加速pairing運算成了有趣且重要的議題。近年來，有些應用所使用的pairing 需要在characteristics大於4並且mod 3同餘2的supersingular曲線上。目前卻很少有關於在這些曲線上改進效能的研究。在這篇論文中，我們提供一個有效計算pairing的方法，並適合characteristics為5的曲線。這個方法在其他characteristics時也有傑出的效果。我們使用Markov模型分析效能，並且與其他現行的方法做比較。

Acknowledge

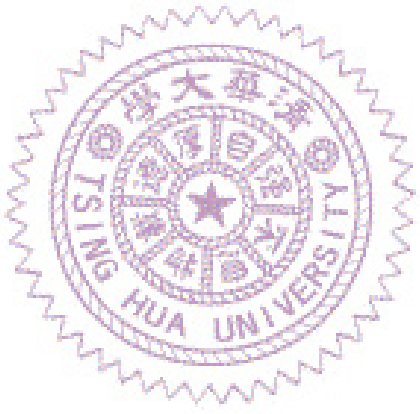
首先感謝指導教授孫宏民老師，老師在研究上嚴謹的指導，使我獲益良多。其次感謝口試委員曾文貴教授、顏嵩銘教授與洪國寶教授，在口試時對此篇論文提出精闢的見解以及建議。再來感謝博士班的王景行學長、林岳勳學長，在研究上給予我不少的建議和協助。也感謝實驗室的所有成員以及朋友們，使我的研究所生活多采多姿。特別感謝家人的栽培和支持，要感謝的人太多了，族繁不及備載，所有曾幫助過我的大家，我衷心的感謝你們。

Table of Contents



List of Figures

List of Tables



Chapter 1

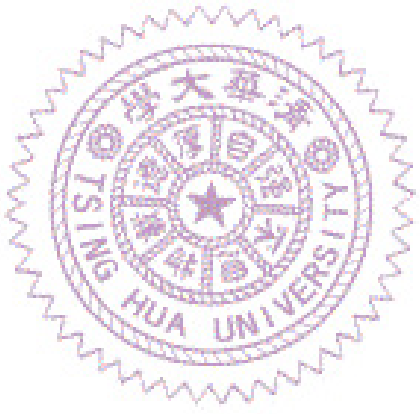
Introduction

Bilinear pairing has been widely applied in different cryptographic applications nowadays. For example, MOV attack [13] utilized Weil pairing to reduce discrete log problem in ECC to the discrete log problem in finite field. Boneh *et al.*[4] constructed the first Identity-Based Encryption (IBE) system using modified Weil pairing. Joux [9] proposed a tripartite key agreement protocol based on pairing. Recently, Boneh [5] proposed another application with bilinear pairing on supersingular curve. The scheme specifies that the characteristic p of field F_{p^m} under the curve should satisfy $p \equiv 2 \; mod \; 3, p \geq 5$. As a result, providing an efficient computation algorithm for bilinear pairing becomes a hot topic in the literature. Moreover, it is also worthy to investigate the special case when $p \equiv 2 \; mod \; 3, p \geq 5$.

The computation of pairing involves complicated calculation on divisors. To effectively calculate pairing, Miller [14] proposed a classic algorithm. This algorithm adopts the idea of double-and-add to simplify the calculation of divisors to the multiplications of lines. Since then, many different aspects improvement on Miller's algorithm have been given, Barreto *et al.*[1], and Galbraith *et al.*[7] focus particularly on the Tate pairing over some special curves. Recently, Blake *et al.*[2] pointed out that part of the line computations can be reduced using some lemmas. They presented different algorithms when the order of the pairing n in represented in radices 2, 3 and 5. Some successive works [12, 19] further improve Blake *et al.*'s work for curves work over characteristic 2 and 3. Nevertheless, none of these approaches has focused on improving the special case when $p \equiv 2 \; mod \; 3$ and $p \geq 5$.

In this thesis, we extend the above methods by considering n in another number representation. This allows us to effectively compute pairings. The core idea is that there exists a reduction equation when n is represented in radix 5 or radix M, for some $M = 2^{2w} + 1$. This reduction equation eliminates some redundant terms in lines multiplications. We examined the performance of our algorithm through analyses. Regardless some necessary precomputations, like converting a number into radix M representation, our method has a better performance than most of the previous schemes. Notice that n is a fixed constant in most of the applications, so these precomputations can be done offline. Moreover, when the characteristic p equals to 5, our scheme has additional advantages over other schemes.

The rest of the thesis is organized as follows. We reviews the preliminaries and some related work in Chapter 2 and 3. Chapter 4 describes our algorithm in details. It is followed by experiments and analyses in Chapter 5. Lastly we conclude our thesis in Chapter 6.



Chapter 2

Mathematical Background

In this chapter we present the mathematical theory required to understand the elliptic curve cryptography and the pairing-based cryptosystems. The first section introduces the basic properties of group, ring and field. In the second section, we present the elliptic curve arithmetic. Include elliptic curve, group laws and torsion. In the last section we introduce the divisor theory and present the formal definition of the Weil pairing and Tate pairing.

2.1 Finite Fields

Finite field is the fundamental element of the elliptic curve cryptography and pairing based cryptosystems. The following subsections we give the definitions and basic properties for group, ring and field.For more details on finite fields, see [8, 11].

2.1.1 Groups

Definition 2.1. *Let G be a non-empty set together with a binary operation $\circ$.We say that $(G, \circ)$ is a group, if the following axioms hold :*

1. $(Closure) : a, b \in \mathbb{G} \Rightarrow a \circ b \in G$.
2. $(Associative) : \forall a, b, c \in \mathbb{G} \Rightarrow a \circ (b \circ c) = (a \circ b)c \in \mathbb{G}$.
3. $(Identity)$: *There is an identity element $e \in \mathbb{G}$, such that $\forall a \in G$,$a \circ e = e \circ a = a$.*

4. $(Inverse) : \forall a \in \mathbb{G}, \exists a^{-1} \in \mathbb{G} \Rightarrow a \circ a^{-1} = a^{-1} \circ a = e.$

If the group satisfies the abelian property:
$(Abelian) : \forall a, b \in \mathbb{G}$, we have $a \circ b = b \circ a$. The group $\mathbb{G}$ is said to be $commutative$ or $abelian$.

We say $\mathbb{G}$ is $cyclic$ if exists an element $a \in \mathbb{G}$, such that every element $b \in \mathbb{G}$, can be written as $b = a^n = \underbrace{a \circ a \circ ... \circ a}_{n}$ for some $n \in \mathbb{Z}$. And a is called a $generator$ of $\mathbb{G}$.

2.1.2 Rings and Fields

Definition 2.2. *A Ring is a set together with two composition laws* $+$ *and* $\times$. *We say* $\mathbb{R} = (R, +, \times)$ *is a ring, if the following axioms hold:*

1. $(R, +)$ *is an abelian group with identity element* 0.
2. $\times$ *is associative:* $\forall a, b, c \in \mathbb{R}$ *, we have* $(a \times b) \times c = a \times (b \times c)$.
3. $\times$ *has Identity element* 1*:*$\forall a \in \mathbb{R}, \exists e \in R \Rightarrow a \times e = e \times a = a$.
4. $\times$ *distributes over* $+$ *:* $\forall a, b, c \in \mathbb{R}$ *, we have* $a \times (b + c) = a \times b + a \times c$ *and* $(b + c) \times a = b \times a + c \times a$.

Definition 2.3. *A* $field$ $\mathbb{F}$ *is a commutative ring* $(F, +, \times)$ *such that* 0 *does not equal* 1 *and all elements of* F *except* 0 *have a multiplicative inverse.(Note that* 0 *and* 1 *here stand for the identity elements for the* $+$ *and* $\times$ *operations respectively, which may differ from the familiar real numbers* 0 *and* 1*.)*

The number of elements in a finite group is called the order of $\mathbb{F}$. An element $g \in \mathbb{F}$ has order a if $g^a = 1$, where 1 is the multiplicative identity element of $\mathbb{F}$.

2.2 Elliptic Curves

In this section, we give an overview on the fundamentals of elliptic curve.[6, 10, 15, 18]

2.2.1 Elliptic Curves properties and group law

Let $E : y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6$ be an elliptic curve over a finite field K, a_1, a_2, a_3, a_4, a_6 are real numbers that satisfy some conditions [16]. Let P and Q be points on E and $\mathcal{O}$ be the point at infinity. The points addition rules are defined as follows:

1. $\mathcal{O}$ + P = P and P + $\mathcal{O}$ = P. The point at infinity $\mathcal{O}$ is the additive identity of the group.

2. $\mathcal{O}$ = -$\mathcal{O}$.

3. If $P = (x_1, y_1) \neq \mathcal{O}$, then $-P = (x_1, -y_1 - a_1x_1 - a_3)$.

4. P + $(-P)$ = $(-P)$ + P = $\mathcal{O}$. (Figure.2.1)

5. If $P = (x_1, 0) \neq \mathcal{O}$, then $P = -P$,$[2]P$ = $\mathcal{O}$. (Figure.2.2)

6. If $P \neq Q$ and $P, Q \neq \mathcal{O}$, then the line L= $\overline{\text{PQ}}$ intersects E in a third point $-R$. The operation are defined : $P + Q = R$. Figure.2.3 illustrates the operation.

7. If P = Q and $P, Q \neq \mathcal{O}$, then the line L is the tangent at P and intersects E in a third point $-R$. The operation are defined : $P + Q = [2]P = R$. Figure.2.4 illustrates the operation.

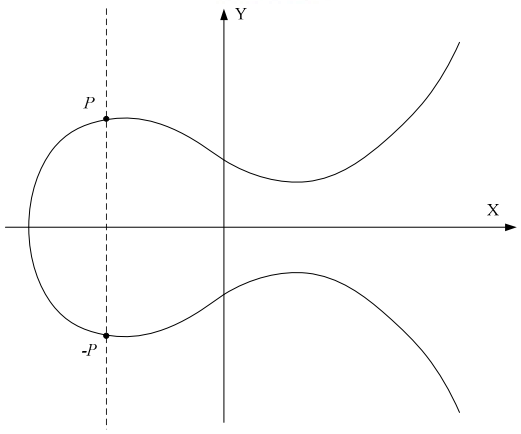


Figure 2.1: The negative of a point on a curve.

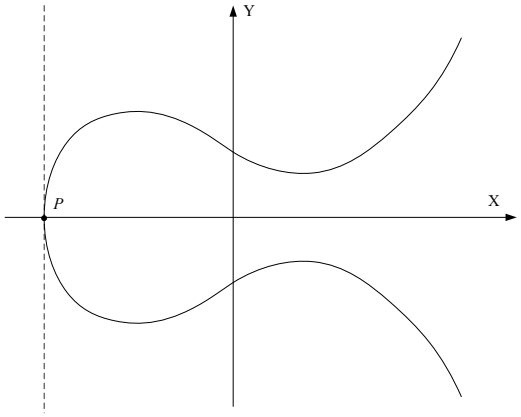


Figure 2.2: 2-torsion point: $[2]P = \mathcal{O}$.

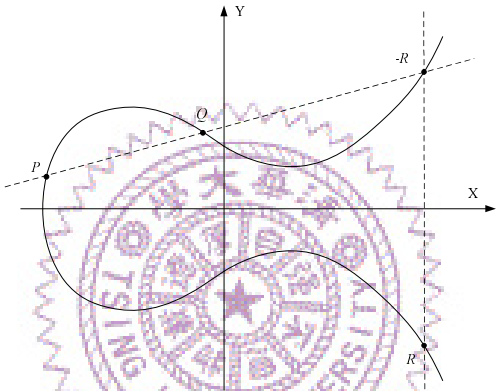


Figure 2.3: Addition of two point : $P + Q = R$.

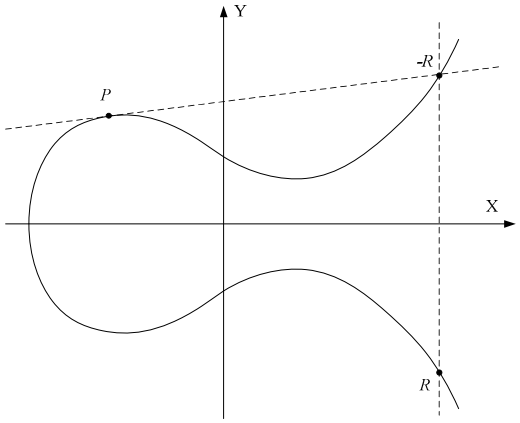


Figure 2.4: Addition of two point : $[2]P = R$.

In the field of characteristic p ($p \neq 2, 3$), two nonneqative integers, a and b, less than p and satisfying $4a^3 + 27b^2 \pmod{p} \neq 0$, a curve equation is

$$y^2 = x^3 + ax^2 + b \pmod{p}$$

$4a^3+27b^2 \pmod{p} \neq 0$, elliptic curves has two basic forms.(See Figure.2.5 and 2.6). When $4a^3 + 27b^2 \pmod{p} = 0$, its graph has distinct tangent directions or cusp, called singular curve. (See Figure.2.7 and 2.8)

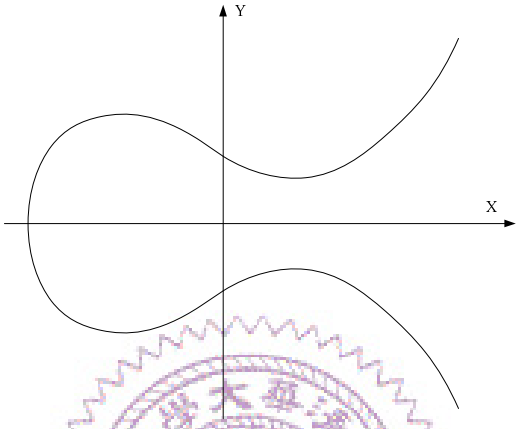


Figure 2.5: $y^2 = x^3 + 3x + 3$,
An elliptic curve with one real component

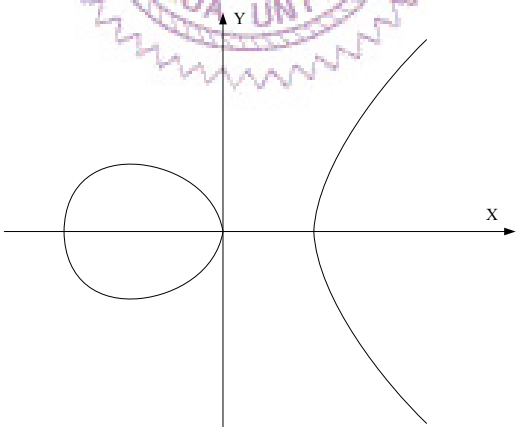


Figure 2.6: $y^2 = x^3 - 3x$,
An elliptic curve with two real components.

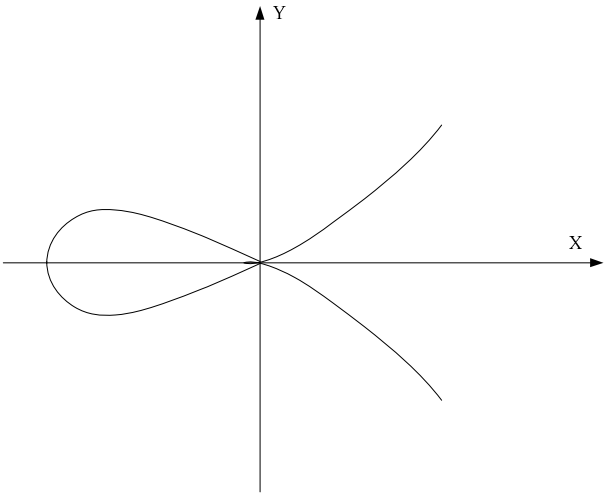


Figure 2.7: $y^2 = x^2(x+1)$,
A singular curve with distinct tangent directions.

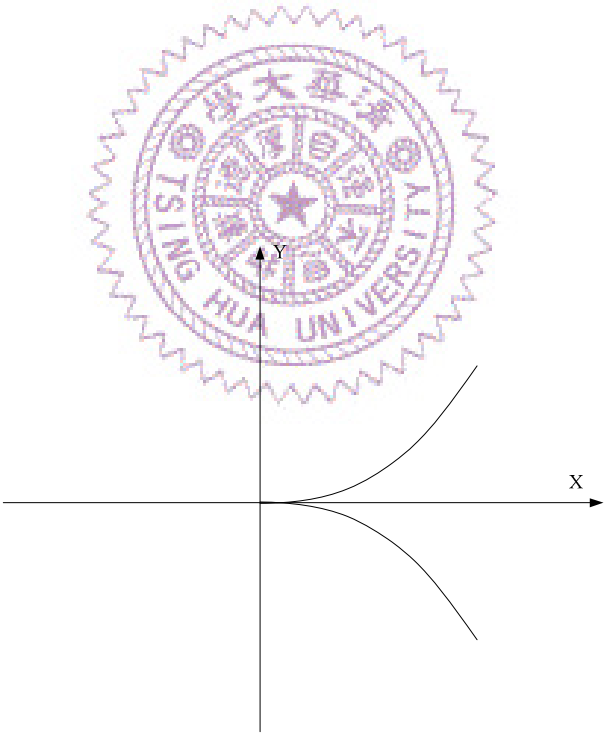


Figure 2.8: $y^2 = x^3$,
A singular curve with a cusp.

If $P, Q, R \in E$, $P = (x_1, y_1), Q = (x_2, y_2)$, then $R = P + Q = (x_3, y_3)$,where

$$x_3 = \lambda^2 - x_1 - x_2 \text{ (mod } p)$$
$$y_3 = (x_1 - x_3)\lambda - y_1 - x_2 \text{ (mod } p)$$

$$\lambda = \begin{cases} \frac{y_2 - y_1}{x_2 - x_1}, if\ P \neq Q \\ \frac{3x_1^2 + a}{2y_1}, if\ P = Q \end{cases}$$

Note that when $P \neq Q$ the variable λ is the slope of a chord line, which pass through P and Q . When $P = Q$, λ is the slope of a tangent line.

Besides, the definition of the scalar multiplication with an integer m to a point P on E is represented as : $mP = \underbrace{P + P + \cdots + P}_{m}$.

2.2.2 Order of Point and Elliptic Curves

Let K be a finite field with order q. The number of points in an elliptic curve $E(K)$, represented by $\#E(K)$, is called order of an elliptic curve over K. The order of a curve is determined by $\#E(F_q) = q + 1 \pm t$, where $|t| \leq 2\sqrt{q}$ is the Frobenius' trace [3]. For the supersingular curves, the Frobenius' trace is a multiple from the field characteristic.

The order of a point $P \in E(K)$ is the smallest positive integer r, such that $[r]P = \mathcal{O}$. The order of a point always divides the order of the curve, this is, $r|\#E(K)$

2.3 Pairings

In this section we introduce the divisor theory and present the formal definition of the Weil pairing and Tate pairing.

2.3.1 Divisors

Divisors are used in the definition of the Tate pairing. In this section, we present basic definition and properties of divisor function. A detail content can be found in Chapter 11 of [18].

Definition 2.4. *Let E be an elliptic curve defined over a field* $\mathbb{K}$. *For each point* $P \in$

$E(\overline{\mathbb{K}})$, *define a formal symbol* $[P]$.

A $divisor$ D *on* E *is a finite linear combination of such symbols with integer coefficients:*

$$D = \sum_j a_j[P_j],\ \ a_j \in Z.$$

Definition 2.5. *A divisor it therefore an element of the free abelian group generated by the symbols* $[P]$. *The group of divisor is denoted* $Div(E)$. *Define the* $degree$ *and* sum *of a divisor by*

$$deg(\sum_j a_j[P_j]) = \sum_j a_j \in Z.$$
$$sum(\sum_j a_j[P_j]) = \sum_j a_j P_j \in E(\overline{\mathbb{K}}).$$

Definition 2.6. *If* f *is a function on* E *that is not identically* 0, *define the* $divisor$ *of* f *to be*

$$div(f) = \sum_{P \in E(\overline{\mathbb{K}})} ord_P(f)[P] \in Div(E).$$

Theorem 2.1. *This is a finite sum, where* $ord_P(f)$ *is the* $order$ *of the zero or the pole at the point* P.

$$div(f) = D$$

if and only if

$$sum(D) = \infty$$

2.3.2 Weil/Tate Pairing

Assuming that an elliptic curve E over a finite field $\mathbb{F}_q$ and $q = p^r$. p is a prime and is named as the characteristic of $\mathbb{F}_q$. Let n be a positive integer that is relatively prime to p and divides the order of a elliptic curve E, such that $n|\ \#E(F_q)$. The embedding degree k is the smallest positive integer such that $n|\ q^k - 1$. Then, the field $\mathbb{F}_q^k$ contains the n-th roots of unity. We further assume points $P, Q_W \in\ E[n], Q_T \in\ E(\mathbb{F}_{q^k}), R \in E$, where $E[n]$ is called the n-torsion group such that $E[n] = \{P \in E(\overline{\mathbb{F}}_{q^k}) : nP = \mathcal{O}\}$. There existing a nonzero rational function f_P satisfies $div(f_P) = n(P) - n(\mathcal{O})$. If $div(f_P)$ and

$D = \sum_{P \in E} n_P(P)$ have disjoint supports, we name $f(D) = \prod_{P \in E} f(P)^{n_P}$. Weil pairing is a bilinear map $e_n : E[n] \times E[n] \to \mathbb{F}_{q^k}^*$ given by equation (2.1).

$$e_n(P, Q_W) = \frac{f_{Q_W}(D_P)}{f_P(D_{Q_W})} \tag{2.1}$$

Weil pairing satisfies the following properties:

1. e_n is $bilinear$ in each variable. This means that

$$e_n(P_1 + P_2, Q_W) = e_n(P_1, Q_W)e_n(+P_2, Q_W)$$

and

$$e_n(P, Q_{W1} + Q_{W2}) = e_n(P, Q_{W1})e_n(+P, Q_{W2})$$

$for\ all\ P, P_1, P_2, Q_W, Q_{W1}, Q_{W2} \in E[n]$

2. e_n is $nondegenerate$ in each variable. This means that if $e_n(P, Q_W) = 1$ for all $P \in E[n]$ then $Q_W = \mathcal{O}$ and also that if $e_n(P, Q_W) = 1$ for all $Q_W \in E[n]$ then $P = \mathcal{O}$

3. e_n is $alternating$ in each variable: $e_n(P, Q) = e_n(Q, P)^{-1}$ for all $P, Q_W \in E[n]$

Tate pairing is a bilinear map $\tau_n : E[n] \times E(\mathbb{F}_{q^k})/nE(\mathbb{F}_{q^k}) \to \mathbb{F}_{q^k}^*$ given by equation (2.2).

$$\tau_n(P, Q_T) = f_P(D_{Q_T})^{(q^k-1)/n} \tag{2.2}$$

Tate pairing satisfies the following properties:

1. τ_n is $bilinear$ in each variable. This means that

$$\tau_n(P_1 + P_2, Q) = \tau_n(P_1, Q)\tau_n(P_2, Q)$$

and

$$\tau_n(P, Q_1 + Q_2) = \tau_n(P, Q_1)\tau_n(P, Q_2)$$

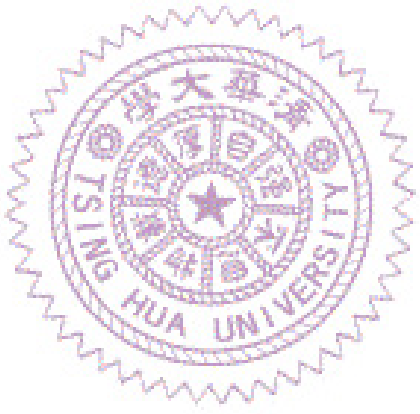
$for\ all\ P, P_1, P_2, \in E[n],\ and\ all\ Q, Q_1, Q_2 \in E(\mathbb{F}_{q^k})/nE(\mathbb{F}_{q^k})$.

So,for any integer m ,we have

$$\tau_n(mP, Q) = \tau_n(P, mQ) = \tau_n(P, Q)^m$$

2. e_n is $nondegenerate$ in each variable. This means that if $\tau_n(P, Q) = 1$ for all $Q \in E(\mathbb{F}_{q^k})/nE(\mathbb{F}_{q^k})$, then $P = \mathcal{O}$.

   Conversely, for each $P \neq \mathcal{O}, \exists Q \in E(\mathbb{F}_{q^k})/nE(\mathbb{F}_{q^k})$ so that $\tau_n(P, Q) \neq 1$



Chapter 3

Computation of Weil/Tate Pairing

3.1 Miller Algorithm

Both pairings require to calculate f such that $div(f) = n[P+R] - n[R]$. Miller [14] proposed an efficient method to evaluate this term. Let the functions $h_{P,Q}$ and h_P represent the connection line between point P and point Q and the vertical line pass through P respectively. He found out that if a function f_i is defined as $div(f_i) = i(P) - (iP) - (i-1)(\mathcal{O})$, the divisor function $div(f)$ can be calculated using the double-and-add method, as shown in equation (3.1).

$$f_{k_1+k_2} = f_{k_1} f_{k_2} \frac{h_{k_1P,k_2P}}{h_{(k_1+k_2)P}} \tag{3.1}$$

Notice that $f_0 = 1$ and $f_1 = h_{P+R}/h_{P,R}$.

A complete Miller's algorithm is described in Algorithm 1.

3.2 BMX Algorithms

In 2006, Blake *et al.*[2] spotted that the number of lines in Miller's algorithm can be reduced using some lemmas.

Lemma 1. *Denote the norm* $N_{K(x,y)/K(x)}(h) = h(x,y)\bar{h}(x,y)$. *If the line* $h(x,y) = 0$ *intersects with* E *at points* $P = (a,b), Q = (c,d)$ *and* $-(P+Q) = (\alpha,\beta)$, *then* $N_{K(x,y)/K(x)}(h) = -(x-a)(x-c)(x-\beta)$, *where* $\bar{h}(x,y)$ *is the conjugate of h with* $h(R) = \bar{h}(-R) for R \in E$.

Algorithm 1 : Miller's algorithm

INPUT:*Integer* $n = \sum_{i=0}^{t} b_i 2^i$ with $b_i \in \{0, 1\}$ and $b_t = 1$, and a point $S \in E$.
OUTPUT:$f = f_n(S)$.
$f \leftarrow f_1$; $Z \leftarrow P$;
for $j \leftarrow t-1, t-2, ..., 1, 0$ **do**
    $f \leftarrow f^2 \frac{h_{Z,Z}(S)}{h_{2Z}(S)}$; $Z \leftarrow 2Z$;
    if $b_j = 1$ **then**
        $f \leftarrow f_1 f \frac{h_{Z,P}(S)}{h_{Z+P}(S)}$; $Z \leftarrow Z + P$;
    end if
end for
return f;

Lemma 2. *Let $Q \in E[n]$,$S \neq Q, 2Q, \cdots, nQ$ and $k \in Z$ then:*

$$\frac{h_{Q,Q}(S)}{h_Q^2(S)h_{2Q}(S)} = -\frac{1}{h_{Q,Q}(-S)}. \tag{3.2}$$

$$\frac{h_{(k+1)Q,kQ}(S)}{h_{(k+1)Q}(S)h_{(2k+1)Q}(S)} = -\frac{h_{kQ}(S)}{h_{(k+1)Q,kQ}(-S)}. \tag{3.3}$$

$$\frac{h_{Q,Q}(S)h_{2Q,Q}(S)}{h_{2Q}(S)h_{3Q}(S)} = -\frac{h_{Q,Q}(S)h_Q(S)}{h_{2Q,Q}(-S)}. \tag{3.4}$$

Proof. By Lemma 1, we have

(3.2)

$$\begin{aligned}\frac{h_{Q,Q}(S)}{h_Q^2(S)h_{2Q}(S)} &= \frac{h_{Q,Q}(S)h_{Q,Q}(-S)}{h_Q^2(S)h_{2Q}(S)h_{Q,Q}(-S)} \\ &= \frac{N_{K(x,y)/K(x)}(h_{Q,Q})(S)}{(x_S - x_Q)\,(x_S - x_{2Q})\,h_{Q,Q}(-S)} \\ &= -\frac{1}{h_{Q,Q}(-S)}.\end{aligned}$$

(3.3)

$$\begin{aligned}\frac{h_{(k+1)Q,kQ}(S)}{h_{(k+1)Q}(S)h_{(2k+1)Q}(S)} &= \frac{h_{(k+1)Q,kQ}(S)h_{(k+1)Q,kQ}(-S)}{h_{(k+1)Q}(S)h_{(2k+1)Q}(S)h_{(k+1)Q,kQ}(-S)} \\ &= \frac{N_{K(x,y)/K(x)}(h_{(k+1)Q,kQ})(S)}{(x_S - x_{(k+1)Q})\,(x_S - x_{(2k+1)Q})\,h_{(k+1)Q,kQ}(-S)} \\ &= -\frac{h_{kQ}(S)}{h_{(k+1)Q,kQ}(-S)}.\end{aligned}$$

(3.4)

$$\begin{aligned}\frac{h_{Q,Q}(S)h_{2Q,Q}(S)}{h_{2Q}(S)h_{3Q}(S)} &= \frac{h_{Q,Q}(S)h_{2Q,Q}(S)h_{2Q,Q}(-S)}{h_{2Q}(S)h_{3Q}(S)h_{2Q,Q}(-S)} \\ &= \frac{h_{Q,Q}(S)N_{K(x,y)/K(x)}(h_{2Q,Q})(S)}{(x_S - x_{2Q})\,(x_S - x_{3Q})\,h_{2Q,Q}(-S)} \\ &= -\frac{h_{Q,Q}(S)(x_S - x_Q)}{h_{2Q,Q}(-S)} \\ &= -\frac{h_{Q,Q}(S)h_Q(S)}{h_{2Q,Q}(-S)}.\end{aligned}$$

Blake *et al.*employed equation (3.2), (3.3), and (3.4) to eliminate some redundant terms generated by Miller's algorithm. They have proposed three algorithms namely BMX-1, BMX-2, and BMX-3. BMX-1 considers n in radix 4. It excels at the case when n has many 0. BMX-2 treats n in binary string. It has a better performance when there are many 1. BMX-3 converts n into base 3 that suits curves over fields of characteristic 3. It takes the advantage of fast point tripling in characteristic 3 to speed up the pairing process.

3.3 LHC Algorithm

In 2007, Liu *et al.*[12] combined BMX-1 and BMX-2 to further reduce redundant terms. Consider n in radix 2, divide n into some segments of patterns: $l_0 = 0^m, l_1 = 1^m, l_2 = (01)^m, l_3 = 01^w, l_4 = 1^m0$ *and* $l_5 = 01^m0$, where $m \geq 1$ and $w \geq 2$. Design an algorithm to reduce these patterns. Reduce more lines than BMX-1 and BMX-2 in average cases. Their improved Blake *et al.*'s schemes on fields with any characteristic.

3.4 Wu's Algorithm

Later Wu *et al.*[19] stated a new lemma, equation (3.5), and improved BMX-3 by classifying a base 3 n into more different cases: $l_0 = (AB)^m, l_1 = (0A)^m, l_2 = (00)^m, l_3 = A, l_4 = 0$, where $m \geq 1$ and $A, B \in \{1, 2, 3, 4\}$. This method work efficiently in curve over fields of characteristic 3. But their performance will be degraded if it is applied to fields with other characteristics.

Algorithm 2 : BMX-1

INPUT:*Integer* $n = \sum_{i=0}^{t} b_i 4^i$ with $b_i \in \{0, 1, 2, 3\}$ and $b_t \neq 0$, and a point $S \in E$.
OUTPUT:$f = f_n(S)$.
$f \leftarrow f_1,\ Z \leftarrow P$;
if $b_t = 2$ **then** $f \leftarrow f^2 \frac{h_{P,P}(S)}{h_{2P}(S)};\ Z \leftarrow 2P$;
end if
if $b_t = 3$ **then** $f \leftarrow f^3 \frac{h_{P,P}^2(S) h_P(S)}{h_{2P,P}(-S)};\ Z \leftarrow 3P$;
end if
for $j \leftarrow t-1, t-2, ..., 1, 0$ **do**
    if $b_j = 0$ **then**
        $f \leftarrow f^4 \frac{h_{Z,Z}^2(S)}{h_{2Z,2Z}(-S)};\ Z \leftarrow 4Z$;
    end if
    if $b_j = 1$ **then**
        $f \leftarrow f_1 f^4 \frac{h_{Z,Z}^2(S) h_{4Z,P}(S)}{h_{2Z,2Z}(-S) h_{4Z+P}(S)};\ Z \leftarrow 4Z + P$;
    end if
    if $b_j = 2$ **then**
        $f \leftarrow f_1^2 f^4 \frac{h_{Z,Z}^2(S) h_{2Z,P}^2(S)}{h_{2Z}^2(S) h_{2Z+P,2Z+P}(-S)}; Z \leftarrow 4Z + 2P$;
    end if
    if $b_j = 3$ **then**
        $f \leftarrow f_1^3 f^4 \frac{h_{Z,Z}^2(S) h_{2Z,P}^2(S) h_{4Z+2P,P}(S)}{h_{2Z}^2(S) h_{2Z+P,2Z+P}(-S) h_{4Z+3P}(S)};\ Z \leftarrow 4Z + 3P$;
    end if
end for
return f;

Algorithm 3 : BMX-2

INPUT:*Integer* $n = \sum_{i=0}^{t} b_i 2^i$ with $b_i \in \{0, 1\}$ and $b_t = 1$, and a point $S \in E$.
OUTPUT:$f = f_n(S)$.
if $b_{t-1} = 0$ **then**
    $f \leftarrow f_1^2 h_{P,P}(S);\ Z \leftarrow 2P$;
else
    $f \leftarrow f_1^3 \frac{h_{P,P}(S) h_{2P,P}(S)}{h_{2P}(S)}; Z \leftarrow 3P$;
end if
for $j \leftarrow t-2, t-3, ..., 1, 0$ **do**
    if $b_j = 0$ **then**
        $f \leftarrow f^2 \frac{h_{2Z}(S)}{h_{Z,Z}(-S)};\ Z \leftarrow 2Z$;
    else
        $f \leftarrow f_1 f^2 \frac{h_{2Z,P}(S)}{h_{Z,Z}(-S)};\ Z \leftarrow 2Z + P$;
    end if
end for
return f;

Algorithm 4 : BMX-3

INPUT:*Integer* $n = \sum_{i=0}^{t} b_i 4^i$ with $b_i \in \{0, 1, 2, 3\}$ and $b_t \neq 0$, and a point $S \in E$.
OUTPUT:$f = f_n(S)$.
$f_2 \leftarrow f_1 \frac{h_{P,P}(S)}{h_{2P}(S)}$;
if $b_t = 1$ **then**
    $f \leftarrow f_1,\ Z \leftarrow P$;
end if
if $b_t = 2$ **then**
    $f \leftarrow f_2;\ Z \leftarrow 2P$;
end if
for $j \leftarrow t-1, t-2, ..., 1, 0$ **do**
    $f \leftarrow f^3 \frac{h_{Z,Z}(S) h_Z(S)}{h_{2Z,Z}(-S)};\ Z \leftarrow 3Z$;
    if $b_j = 1$ **then**
        $f \leftarrow f_1 f \frac{h_{Z,P}(S)}{h_{Z+P}(S)};\ Z \leftarrow Z + P$;
    end if
    if $b_j = 2$ **then**
        $f \leftarrow f_2 f \frac{h_{Z,2P}(S)}{h_{Z+2P}}$; $Z \leftarrow Z + 2P$;
    end if
end for
return f;

Lemma 3. *Let* $Q \in E[n]$,$S \neq Q, 2Q, \cdots, nQ$ *and* $k \in Z$ *then:*

$$\frac{h_{kQ,kQ}(S)h_{2kQ,kQ}(S)}{h_{kQ}^3(S)h_{2kQ}(S)h_{3kQ}(S)} = \frac{h_{2kQ}(S)}{h_{kQ,kQ}(-S)h_{2kQ,kQ}(-S)}. \tag{3.5}$$

Proof. By Lemma 1 and Lemma 2, we have

(3.5)

$$\begin{aligned}
& \frac{1}{h_{kQ}^3(S)} \frac{h_{kQ,kQ}(S)h_{2kQ,kQ}(S)}{h_{2kQ}(S)h_{3kQ}(S)} \\
= & \frac{1}{h_{kQ}^3(S)} \frac{h_{kQ,kQ}(S)h_{2kQ,kQ}(S)h_{kQ,kQ}(-S)h_{2kQ,kQ}(-S)}{h_{2kQ}(S)h_{3kQ}(S)h_{kQ,kQ}(-S)h_{2kQ,kQ}(-S)} \\
= & -\frac{1}{h_{kQ}(S)} \frac{h_{2kQ,kQ}(S)h_{2kQ,kQ}(-S)}{h_{3kQ}(S)h_{kQ,kQ}(-S)h_{2kQ,kQ}(-S)} \\
= & \frac{h_{2kQ}(S)}{h_{kQ,kQ}(-S)h_{2kQ,kQ}(-S)}.
\end{aligned}$$

The above methods do not optimize on supersingular curves with characteristic $p \equiv$

$2 \ mod \ 3$ and $p \geq 5$, despite the popularity of these curves in cryptographic application. Recently Wang *et al.*[17] suggested that one may apply Frobenius Endomorphism to speed up the calculation for some particular curves over fields with characteristic 5 and 7. However, they does not provide efficient terms eliminating method to enhancing general curves with characteristics 5 or 7.

Algorithm 5 : Segmentation-LHC

INPUT: $n = (b_r b_{r-1} \cdots b_0)_5$, $b_j \in \{0, 1, 2\}$, $b_r \neq 0$.
OUTPUT: $\{ B_0, \cdots, B_t \}$, $B_i \in \{l_0, l_1, \cdots, l_5\}^*$.
function TRACKER(x, y)
    while ($x \neq m$ **and** $b_x \neq y$) **do**
        $x \leftarrow x + 1$;
    end while
    if $x = m$ **then return** -1
    else return x
    end if
end function
$i \leftarrow 0$; $j \leftarrow 0$; $k \leftarrow 0$;
while $j < m - 1$ **do**
    if $(b_{j+1}, b_j) = (0, 0)$ **then**
        $k \leftarrow Tracker(j + 2, 1)$;
        if $k = -1$ **then**
            $B_i \leftarrow (l_0, m - j)$; $j \leftarrow m$;
        else
            $B_i \leftarrow (l_0, k - j - 1)$; $j \leftarrow k - 1$; $i \leftarrow i + 1$;
        end if
    end if
    if $(b_{j+1}, b_j) = (0, 1)$ **then**
        if ($i - 1 > 0$ **and** $B_{j-1} = (l_2, \alpha)$) **then**
            $B_i \leftarrow (l_2, \alpha + 1)$; $j \leftarrow j + 2$;
            if $j = m$ **then** $i \leftarrow i - 1$;
            end if
        else
            $B_i \leftarrow (l_2, 1)$; $j \leftarrow j + 2$;
            if $j \neq m$ **then** $i \leftarrow i + 1$;
            end if
        end if
    end if

```
    if (b_{j+1}, b_j) = (1, 0) then
        k ← Tracker(j + 2, 0);
        if k = −1 then
            B_i ← (l_4, m − j − 1); j ← m;
        else
            B_i ← (l_5, k − j − 1); j ← k + 1;
            if j ≠ m then i ← i + 1;
            end if
        end if
    end if
    if (b_{j+1}, b_j) = (1, 1) then
        k ← Tracker(j + 2, 0);
        if k = −1 then
            B_i ← (l_1, m − j); j ← m;
        else
            B_i ← (l_3, k − j); j ← k + 1;
            if j ≠ m then i ← i + 1;
            end if
        end if
    end if
end while
if j = m then return {B_i, B_{i−1}, · · · , B_0}
else
    if b_{m−1} = 1 then B_i ← (l_1, 1);
    else B_i ← (l_0, 1);
    end if
end if
return {B_i, B_{i−1}, · · · , B_0}.
```

Algorithm 6 : LHC Algorithm

INPUT:$n = (b_r b_{r-1} \cdots b_0)_2$ and segments $\{B_0, B_1 \cdots, B_t\}, B_j \in \{l_0, \cdots, l_5\}^*$, and a point $S \in E$.
OUTPUT:$f = f_n(S)$
$f \leftarrow f_1;\ Z \leftarrow P;$
for $j \leftarrow r, r-1 \cdots, 0,$ **do**
  if $B_j = (l_0, m_j)$ **then**
    if m is even **then**
      $m \leftarrow \frac{m}{2};$
      for $x \leftarrow 1, 2 \cdots, m,$ **do**
        $f \leftarrow f^4 \frac{h^2_{Z,Z}(S)}{h_{2Z,2Z}(-S)};$
        $Z \leftarrow 4Z;$
      end for
    else
      $m \leftarrow \frac{m-1}{2};$
      for $x \leftarrow 1, 2 \cdots, m,$ **do**
        $f \leftarrow f^4 \frac{h^2_{Z,Z}(S)}{h_{2Z,2Z}(-S)};$
        $Z \leftarrow 4Z;$
      end for
      $f \leftarrow f^2 \frac{h_{Z,Z}(S)}{h_{2Z}(S)};$
      $Z \leftarrow 2Z;$
    end if
  end if
  if $B_j = (l_1, m_j)$ **then**
    if $m = 1$ **then**
      $f \leftarrow f_1 f^2 \frac{h_{P,P}(S) h_{2P,P}(S)}{h_{2P}(S) h_{3P}(S)};$
      $Z \leftarrow 3P;$
    else if $m = 2$ **then**
      $f \leftarrow f_1^3 f^4 \frac{h^2_{P,P}(S) h^2_{2P,P}(S) h_{6P,P}(S)}{h^2_{2P}(S) h_{3P,3P}(-S) h_{7P}(S)};$
      $Z \leftarrow 7P;$
    else
      $f \leftarrow f_1 f^2 \frac{h_{P,P}(S) h_{2P,P}(S)}{h_{2P}(S)};$
      $Z \leftarrow 3P;$
      for $x \leftarrow 1, 2 \cdots, m_j - 2,$ **do**
        $f \leftarrow f_1 f^2 \frac{h_{2Z,P}(S)}{h_{Z,Z}(-S)};$
        $Z \leftarrow 2Z + P;$
      end for
      $f \leftarrow f_1 f^2 \frac{h_{2Z,P}(S)}{h_{Z,Z}(-S) h_{2Z+P}(S)};$
      $Z \leftarrow 2Z + P;$
    end if
  end if
  if $B_j = (l_2, m_j)$ **then**
    for $x \leftarrow 1, 2 \cdots, m_j,$ **do**
      $f \leftarrow f_1 f^4 \frac{h^2_{Z,Z}(S) h_{4Z,P}(S)}{h_{2Z,2Z}(-S) h_{4Z,P}(-S)};$
      $Z \leftarrow 4Z + P;$
    end for
  end if

if $B_j = (l_3, m_j)$ **then**
  $f \leftarrow f^2 h_{Z,Z}(S)$;
  $Z \leftarrow 2Z$;
  for $x \leftarrow 1, 2 \cdots, m_j - 1$, **do**
    $f \leftarrow f_1 f^2 \frac{h_{2Z,P}(S)}{h_{Z,Z}(-S)}$;
    $Z \leftarrow 2Z + P$;
  end for
  $f \leftarrow f_1 f^2 \frac{h_{2Z,P}(S)}{h_{Z,Z}(-S) h_{2Z+P}(S)}$;
  $Z \leftarrow 2Z + P$;
end if
if $B_j = (l_4, m_j)$ **then**
  if $m = 1$ **then**
    $f \leftarrow f_1^6 \frac{h_{P,P}^2(S) h_{2P,P}^2(S)}{h_{2P}^2(S) h_{3P,3P}(-S)}$;
    $Z \leftarrow 6P$;
  else
    $f \leftarrow f_1^3 \frac{h_{P,P}(S) h_{2P,P}(S)}{h_{2P}(S)}$;
    $Z \leftarrow 3P$;
    for $x \leftarrow 1, 2 \cdots, m_j - 1$, **do**
      $f \leftarrow f_1 f^2 \frac{h_{2Z,P}(S)}{h_{Z,Z}(-S)}$;
      $Z \leftarrow 2Z + P$;
    end for
    $f \leftarrow f^2 \frac{1}{h_{Z,Z}(-S)}$;
    $Z \leftarrow 2Z$;
  end if
end if
if $B_j = (l_5, m_j)$ **then**
  $f \leftarrow f^2 h_{Z,Z}(S)$;
  $Z \leftarrow 2Z$;
  while $m_j \neq 1$ **do**
    $f \leftarrow f_1 f^2 \frac{h_{2Z+P}(S)}{h_{2Z,2Z}(-S)}$;
    $Z \leftarrow 2Z + P$;
    $m_j \leftarrow m_j - 1$;
  end while
  $f \leftarrow f_1^2 f^4 \frac{h_{2Z,P}^2(S)}{h_{Z,Z}^2(-S) h_{2Z+P,2Z+P}(-S)}$;
  $Z \leftarrow 4Z + 2P$;
end if
end for
return f;

Algorithm 7 : Wu's Algorithm

INPUT:$n = (b_r b_{r-1} \cdots b_0)_5$ and segments $\{B_0, B_1 \cdots, B_t\}, B_j \in \{l_0, \cdots, l_4\}^*$
OUTPUT:$f = f_n(S)$
$f_2 \leftarrow f_1^2 \frac{h_{P,P}(S)}{h_{2P}(S)}$; $u \leftarrow r-1$;
if $b_r = 1$ **then**
    $f \leftarrow f_1$; $Z \leftarrow P$;
end if
if $b_r = 2$ **then**
    $f \leftarrow f_1^2 h_{P,P}(S)$; $Z \leftarrow 2P$;
end if
for $j \leftarrow 0, 1 \cdots, t$, **do**
    if $B_j = (l_0, m_j)$ **then**
        if ($j = 0$ **and** $b_r = 0$) **then**
            $f \leftarrow f_{b_u} f^3 \frac{h_{Z,Z}(S) h_Z(S) h_{3Z,b_u P}(S)}{h_{2Z,Z}(-S)}$;
            $Z \leftarrow 3Z + b_u P$;
            $f \leftarrow f_{b_{u-1}} f^3 \frac{h_{2Z}(S) h_{3Z,b_{u-1}P}(S)}{h_{Z,Z}(-S) h_{2Z,Z}(-S)}$;
            $Z \leftarrow 3Z + b_{u-1} P$;
            $m_j \leftarrow m_j - 1$; $u \leftarrow u - 2$;
        end if
        for $x \leftarrow 1, 2 \cdots, m_j$, **do**
            $f \leftarrow f_{b_u} f^3 \frac{h_{2Z}(S) h_{3Z,b_u P}(S)}{h_{Z,Z}(-S) h_{2Z,Z}(-S)}$;
            $Z \leftarrow 3Z + b_u P$;
            $f \leftarrow f_{b_{u-1}} f^3 \frac{h_{2Z}(S) h_{3Z,b_{u-1}P}(S)}{h_{Z,Z}(-S) h_{2Z,Z}(-S)}$;
            $Z \leftarrow 3Z + b_{u-1} P$; $u \leftarrow u - 2$;
        end for
    end if
    if $B_j = (l_1, m_j)$ **then**
        if ($j = 0$ **and** $b_r = 1$) **then**
            $f \leftarrow f^3 \frac{h_{Z,Z}(S) h_Z(S)}{h_{2Z,Z}(-S)}$;
            $Z \leftarrow 3Z$;
            $f \leftarrow f_{b_{u-1}} f^3 \frac{h_{Z,Z}(S) h_Z(S) h_{3Z,b_{u-1}P}(S)}{h_{2Z,Z}(-S)}$;
            $Z \leftarrow 3Z + b_{u-1} P$;
            $m_j \leftarrow m_j - 1$; $u \leftarrow u - 2$;
        end if
        for $x \leftarrow 1, 2 \cdots, m_j$, **do**
            $f \leftarrow f^3 \frac{h_{2Z}(S)}{h_{Z,Z}(-S) h_{2Z,Z}(-S)}$;
            $Z \leftarrow 3Z$;
            $f \leftarrow f_{b_{u-1}} f^3 \frac{h_{Z,Z}(S) h_Z(S) h_{5Z,b_{u-1}P}(S)}{h_{2Z,2Z}(-S)}$;
            $Z \leftarrow 3Z + b_{u-1} P$; $u \leftarrow u - 2$;
        end for
    end if

if $B_j = (l_2, m_j)$ **then**
  if ($j = t$) **then** $m_j \leftarrow m_j - 1$;
  end if
  if ($j = 0$ **and** $b_r = 1$) **then**
    $f \leftarrow f^3 \frac{h_{Z,Z}(S)h_Z(S)}{h_{2Z,Z}(-S)}$;
    $Z \leftarrow 3Z$;
    $f \leftarrow f^3 \frac{h_{Z,Z}(S)h_{2Z,Z}(S)}{h_{2Z}(S)}$;
    $Z \leftarrow 3Z$;
    $m_j \leftarrow m_j - 1$; $u \leftarrow u - 2$;
  end if
  for $x \leftarrow 1, 2 \cdots, m_j$, **do**
    $f \leftarrow f^3 \frac{h_{2Z}(S)}{h_{Z,Z}(-S)h_{2Z,Z}(-S)}$;
    $Z \leftarrow 3Z$;
    $f \leftarrow f^3 \frac{h_{Z,Z}(S)h_{2Z,Z}(S)}{h_{2Z}(S)}$;
    $Z \leftarrow 3Z$; $u \leftarrow u - 2$;
  end for
end if
if $B_j = (l_3, 1)$ **then**
  if ($j = 0$ **and** $b_r = 1$) **then**
    $f \leftarrow f_{b_u} f^3 \frac{h_{Z,Z}(S)h_Z(S)h_{3Z,b_uP}(S)}{h_{2Z,Z}(-S)}$;
  else
    $f \leftarrow f_{b_u} f^3 \frac{h_{2Z}(S)h_{3Z,b_uP}(S)}{h_{Z,Z}(-S)h_{2Z,Z}(-S)}$;
    $Z \leftarrow 3Z + b_uP$; $u \leftarrow u - 1$;
  end if
end if
if $B_j = (l_4, 1)$ **then**
  if ($j = 0$ **and** $b_r = 1$) **then**
    $f \leftarrow f^3 \frac{h_{Z,Z}(S)h_Z(S)}{h_{2Z,Z}(-S)}$; **return** f;
  else
    $f \leftarrow f^3 \frac{h_{2Z}(S)}{h_{Z,Z}(-S)h_{2Z,Z}(-S)}$; **return** f;
  end if
end if
end for
if $B_t = (l_2, m_t)$ **then**
  $f \leftarrow f^3 \frac{h_{2Z}(S)}{h_{Z,Z}(-S)h_{2Z,Z}(-S)}$; $Z \leftarrow 3Z$;
  $f \leftarrow f^3 \frac{h_{Z,Z}(S)h_Z(S)}{h_{2Z,Z}(-S)}$;
else
  $f \leftarrow f/h_Z(S)$;
end if
return f;

Chapter 4

Proposed Scheme

4.1 Main Idea of Scheme

Before going into our scheme, we introduce a few lemmas that will be used in our scheme. These lemmas are the core of lines reduction.

Lemma 4. *Let $Q \in E[n], S \neq Q, 2Q, \cdots, nQ$ and $k \in Z$ then:*

$$\frac{h_{4kQ,kQ}(S)}{h_{kQ}(S)\, h_{4kQ}(S)\, h_{5kQ}(S)} = -\frac{1}{h_{4kQ,kQ}(-S)}. \tag{4.1}$$

Proof. By Lemma 1, we have

(4.1)

$$\begin{aligned}
\frac{h_{4kQ,kQ}(S)}{h_{kQ}(S)\, h_{4kQ}(S)\, h_{5kQ}(S)} &= \frac{h_{4kQ,kQ}(S)h_{4kQ,kQ}(-S)}{h_{kQ}(S)\, h_{4kQ}(S)\, h_{5kQ}(S)h_{4kQ,kQ}(-S)} \\
&= \frac{N_{K(x,y)/K(x)}(h_{4kQ,kQ})(S)}{(x_S - x_{kQ})\,(x_S - x_{4kQ})\,(x_S - x_{5kQ})\, h_{4kQ,kQ}(-S)} \\
&= -\frac{1}{h_{4kQ,kQ}(-S)}.
\end{aligned}$$

Lemma 5. *Let $Q \in E[n]$,$S \neq Q, 2Q, \cdots, nQ$ and $k \in Z$ then:*

$$\begin{aligned} & \frac{1}{h_{kQ}^5(S)}(\frac{h_{kQ,kQ}(S)}{h_{2kQ}(S)})^2 \frac{h_{2kQ,2kQ}(S)}{h_{4kQ}(S)} \frac{h_{4kQ,kQ}(S)}{h_{5kQ}(S)} \\ = & \frac{h_{2kQ,2kQ}(S)}{h_{kQ,kQ}^2(-S)h_{4kQ,kQ}(-S)}. \end{aligned}$$

Proof. By Lemma 2 and Lemma 4, we have

$$\begin{aligned} & \frac{1}{h_{kQ}^5(S)}(\frac{h_{kQ,kQ}(S)}{h_{2kQ}(S)})^2 \frac{h_{2kQ,2kQ}(S)}{h_{4kQ}(S)} \frac{h_{4kQ,kQ}(S)}{h_{5kQ}(S)} \\ = & \frac{h_{kQ,kQ}^2(S)h_{2kQ,2kQ}(S)h_{4kQ,kQ}(S)}{h_{kQ}^5(S)h_{2kQ}^2(S)h_{4kQ}(S)h_{5kQ}(S)} \frac{h_{kQ,kQ}^2(-S)h_{4kQ,kQ}(-S)}{h_{kQ,kQ}^2(-S)h_{4kQ,kQ}(-S)} \\ = & \frac{-h_{2kQ,2kQ}(S)h_{4kQ,kQ}(S)h_{4kQ,kQ}(-S)}{h_{kQ}(S)h_{4kQ}(S)h_{5kQ}(S)h_{kQ,kQ}^2(-S)h_{4kQ,kQ}(-S)} \\ = & \frac{h_{2kQ,2kQ}(S)}{h_{kQ,kQ}^2(-S)h_{4kQ,kQ}(-S)}. \end{aligned}$$

Lemma 6. *(For radix $2^{2w}+1$ representation) Let $Q \in E[n]$,$S \neq Q, 2Q, \cdots, nQ$ and $k \in Z$ then:*

$$\begin{aligned} & \frac{1}{h_{kQ}^{2^{2w}+1}(S)}(\frac{h_{kQ,kQ}^{2^{2w-1}}(S)}{h_{2kQ}(S)})(\frac{h_{2kQ,2kQ}^{2^{2w-2}}(S)}{h_{2^2kQ}^{2^{2w-2}}(S)}) \cdots (\frac{h_{2^{2w-1}kQ,2^{2w-1}kQ}(S)}{h_{2^{2w}kQ}(S)} \frac{h_{2^{2w}kQ,kQ}(S)}{h_{(2^{2w}+1)kQ}(S)}) \\ = & (-1)^{w+1}(\frac{1}{h_{kQ,kQ}(-S)})^{2^{2w-1}}(\prod_{i=1}^{w-1} \frac{h_{2^{2i-1}kQ,2^{2i-1}kQ}^{2^{2w-2i}}(S)}{h_{2^{2i}kQ,2^{2i}kQ}^{2^{2w-1-2i}}(-S)}) \frac{(h_{2^{2w-1}kQ,2^{2w-1}kQ}(S))}{(h_{2^{2w}kQ,kQ}(-S))}. \end{aligned}$$

Proof. By Lemma 2 and Lemma 5, we have

$$\frac{1}{h_{kQ}^{2^{2w}+1}(S)}(\frac{h_{kQ,kQ}^{2^{2w-1}}(S)}{h_{2kQ}(S)})(\frac{h_{2kQ,2kQ}^{2^{2w-2}}(S)}{h_{2^2kQ}^{2^{2w-2}}(S)})\cdots(\frac{h_{2^{2w-1}kQ,2^{2w-1}kQ}(S)}{h_{2^{2w}kQ}(S)}\frac{h_{2^{2w}kQ,kQ}(S)}{h_{(2^{2w}+1)kQ}(S)})$$

$$=(\frac{1}{h_{kQ}^{2^{2w}+1}(S)}\frac{h_{kQ,kQ}^{2^{2w-1}}(S)}{h_{2kQ}(S)})(\frac{h_{2kQ,2kQ}^{2^{2w-2}}(S)}{h_{2^2kQ}^{2^{2w-2}}(S)}\frac{h_{2^2kQ,2^2kQ}^{2^{2w-3}}(S)}{h_{2^3kQ}^{2^{2w-3}}(S)})\cdots(\frac{h_{2^{2w-1}kQ,2^{2w-1}kQ}(S)}{h_{2^{2w}kQ}(S)}\frac{h_{2^{2w}kQ,kQ}(S)}{h_{(2^{2w}+1)kQ}(S)})$$
$$\times(\frac{h_{kQ,kQ}^{2^{2w-1}}(-S)}{h_{kQ,kQ}^{2^{2w-1}}(-S)})(\frac{h_{2^2kQ,2^2kQ}^{2^{2w-3}}(-S)}{h_{2^2kQ,2^2kQ}^{2^{2w-3}}(-S)})\cdots(\frac{h_{2^{2w}kQ,kQ}(-S)}{h_{2^{2w}kQ,kQ}(-S)})$$

$$=(\frac{1}{h_{kQ}^{2^{2w}+1}(S)}\frac{h_{kQ,kQ}^{2^{2w-1}}(S)}{h_{2kQ}(S)})(\frac{h_{2kQ,2kQ}^{2^{2w-2}}(S)}{h_{2^2kQ}^{2^{2w-2}}(S)}\frac{h_{2^2kQ,2^2kQ}^{2^{2w-3}}(S)}{h_{2^3kQ}^{2^{2w-3}}(S)})\cdots(\frac{h_{2^{2w-1}kQ,2^{2w-1}kQ}(S)}{h_{2^{2w}kQ}(S)}\frac{h_{2^{2w}kQ,kQ}(S)}{h_{(2^{2w}+1)kQ}(S)})$$
$$\times(\frac{h_{kQ,kQ}^{2^{2w-1}}(-S)}{h_{kQ,kQ}^{2^{2w-1}}(-S)})(\prod_{i=1}^{w-1}\frac{h_{2^{2i}kQ,2^{2i}kQ}^{2^{2w-1-2i}}(-S)}{h_{2^{2i}kQ,2^{2i}kQ}^{2^{2w-1-2i}}(-S)})(\frac{h_{2^{2w}kQ,kQ}(-S)}{h_{2^{2w}kQ,kQ}(-S)})$$

$$=(-1)^{w+1}(\frac{1}{h_{kQ,kQ}(-S)})^{2^{2w-1}}(\frac{h_{2kQ,2kQ}^{2^{2w-2}}(S)}{h_{2^2kQ,2^2kQ}^{2^{2w-3}}(-S)})\cdots(\frac{h_{2^{2w-1}kQ,2^{2w-1}kQ}(S)}{h_{2^{2w}kQ,kQ}(-S)})$$

$$=(-1)^{w+1}(\frac{1}{h_{kQ,kQ}(-S)})^{2^{2w-1}}(\prod_{i=1}^{w-1}\frac{h_{2^{2i-1}kQ,2^{2i-1}kQ}^{2^{2w-2i}}(S)}{h_{2^{2i}kQ,2^{2i}kQ}^{2^{2w-1-2i}}(-S)})(\frac{h_{2^{2w-1}kQ,2^{2w-1}kQ}(S)}{h_{2^{2w}kQ,kQ}(-S)}).$$

Lemma 5 is the main lemma to construct our scheme and is proven by Lemma 4. Lemma 6 is an auxiliary lemma when the scheme is extended from radix-5 representation to radix-$2^{2w}+1$ representation. We remarks that $div(f) = div(cf)$ for any nonzero constant c. Therefore the negative signs in the above lemmas can be omitted.

4.2 Rules, Representation of Lines and Segmentation Algorithm

The basic idea of our scheme is to represent n as a radix 5 integer and eliminate some reductant terms during the calculations of f_i. Our method first converts n into a radix 5 string $(b_r b_{r-1} \cdots b_0)$. Then, we divide the string into some segments of the following five types of patterns: $l_0 = (AB)^m, l_1 = (0A)^m, l_2 = (00)^m, l_3 = A, l_4 = 0$, where $m \geq 1$ and $A, B \in \{1, 2, 3, 4\}$. Notice that the pattern l_0 is the set of m-repetitions of any two non-zero base 5 symbols for any $m \in \mathbb{N}$, for instance, $(423324)_5 \in l_0$. The patterns of l_3 will only be followed by either a pattern in l_1 or l_2, since two consecutive non-zero radix 5 symbols will be classified as l_0. There is only one pattern in l_4 and will only appear at the end of the string. The segmentation method is illustrated in Algorithm 8.

Algorithm 8 : Segmentation

INPUT: $n = (b_r b_{r-1} \cdots b_0)_5$, $b_j \in \{0, 1, 2\}$, $b_r \neq 0$.
OUTPUT: $\{ B_0, \cdots, B_t \}$, $B_i \in \{l_0, \cdots, l_4\}^*$.
$j \leftarrow r-1, m \leftarrow 0, i \leftarrow 0, old\_state = -1$;
while $j > 0$ **do**
    if ($b_j \neq 0$ **and** $b_{j-1} \neq 0$) **then** $state \leftarrow 0$;
    end if
    if ($b_j \neq 0$ **and** $b_{j-1} = 0$) **then**
        if $old\_state \neq -1$ **then**
            $B_i \leftarrow (l_{old\_state}, m), i \leftarrow i + 1$;
        end if
        $B_i \leftarrow (l_3, 1), j \leftarrow j - 1, i \leftarrow i + 1$;
        $old\_state \leftarrow -1, m \leftarrow 0$;
        continue;
    end if
    if ($b_j = 0$ **and** $b_{j-1} \neq 0$) **then** $state \leftarrow 1$;
    end if
    if ($b_j = 0$ **and** $b_{j-1} = 0$) **then** $state \leftarrow 2$;
    end if
    if ($old\_state = state$ **or** $old\_state = -1$) **then**
        $m \leftarrow m + 1, j \leftarrow j - 2, old\_state \leftarrow state$;
    else
        $B_i \leftarrow (l_{old\_state}, m), i \leftarrow i + 1$;
        $j \leftarrow j - 2, m \leftarrow 1, old\_state \leftarrow state$;
    end if
    if $j = -1$ **then** $B_i \leftarrow (l_{old\_state}, m)$;
    end if
    if $j = 0$ **then**
        if $old\_state \neq -1$ **then**
            $B_i \leftarrow (l_{old\_state}, m), i \leftarrow i + 1$;
        end if
        if $b_j = 0$ **then**
            $B_i \leftarrow (l_4, 1)$;
        else
            $B_i \leftarrow (l_3, 1)$;
        end if
    end if
end while
return $\{ B_0, \cdots, B_t \}$.

Now, we convert each block into a product of lines. There are three kinds of lines, namely regular lines, singular lines, conjugate lines. Regular lines include connection lines $h_{P,Q}(S)$ and vertical lines $h_{P+Q}(S)$ for distinct points P and Q. Singular lines are the connection lines $h_{P,P}(S)$ and vertical lines $h_{2P}(S)$. Conjugate lines include connec-

tion lines $h_{P,Q}(-S)$. We adopt the notations used in [12, 19] to denote a regular line by $\bullet$, a singular line by $\circ$, and a conjugate line by $*$. Notice that regular and singular vertical lines will only appears in denominators, while regular and singular connection lines only appears in nominators.

We illustrate the basic idea of our algorithm here. Suppose n is represented as $(B_0, B_1, \ldots, B_{i-1}, B_i, \ldots, B_r)$ and the algorithm has already processed the block B_{i-1}.

Let's consider $B_i = (l_1, 0)$, that is, B_i is in the form of "AB" for some non-zero A and non-zero B. Assume the function f calculated up to B_i is given by

$$f = T\frac{1}{h_Z(S)}$$

We first handle the symbol "A" by

$$f' \leftarrow f_A f^5 \frac{h^2_{Z,Z}(S)h_{2Z,2Z}(S)h_{4Z,Z}(S)h_{5Z,AP}(S)}{h^2_{2Z}(S)h_{4Z}(S)h_{5Z}(S)h_{5Z+AP}(S)}; Z' = 5Z$$

Then

$$f' = f_A T^5 \frac{1}{h^5_Z(S)} \frac{h^2_{Z,Z}(S)h_{2Z,2Z}(S)h_{4Z,Z}(S)h_{5Z,AP}(S)}{h^2_{2Z}(S)h_{4Z}(S)h_{5Z}(S)h_{5Z+AP}(S)}$$

By Lemma 5, We can reduce f' as

$$f' = f_A T^5 \frac{h_{2Z,2Z}(S)h_{5Z,AP}(S)}{h^2_{Z,Z}(-S)h_{4Z,Z}(-S)h_{5Z+AP}(S)}$$

Let

$$T' = f_A T^5 \frac{h_{2Z,2Z}(S)h_{5Z,BP}(S)}{h^2_{Z,Z(-S)}h_{4Z,Z}(-S)}$$

Then

$$f = T'\frac{1}{h_{5Z+AP}(S)} = T'\frac{1}{h_{Z'}(S)}$$

Next, we process the symbol "B" by

$$f'' \leftarrow f_B f'^5 \frac{h^2_{Z',Z'}(S)h_{2Z',2Z'}(S)h_{4Z',Z'}(S)h_{5Z',BP}(S)}{h^2_{2Z'}(S)h_{4Z'}(S)h_{5Z'}(S)h_{5Z'+BP}(S)}$$
$$Z'' = 5Z' + BP$$

Then

$$f'' = f_B T'^5 \frac{1}{h^5_Z(S)} \frac{h^2_{Z',Z'}(S)h_{2Z',2Z'}(S)h_{4Z',Z'}(S)h_{5Z',BP}(S)}{h^2_{2Z'}(S)h_{4Z'}(S)h_{5Z'}(S)h_{5Z'+BP}(S)}$$

By Lemma 5, We have

$$f'' = f_B T'^5 \frac{h_{2Z',2Z'}(S)h_{5Z',BP}(S)}{h^2_{Z',Z'}(-S)h_{4Z',Z'}(-S)} \frac{1}{h_{5Z'+BP}(S)}$$

The fraction $\frac{1}{h_{5Z'+BP}(S)}$ will be used to reduce terms in B_{i+1} in next iteration. The above example demonstrates the case when the block B_i is in the form of "AB" which belongs to the pattern l_0. It consumes a fraction $\frac{1}{h_Z(S)}$ from previous result and contributes another fraction $\frac{1}{h_{Z''}(S)}$ to next block.Then the computation of this case can be denoted as follows:

$$l_0 = (AB)^m : \tfrac{1}{\bullet}(\tfrac{\bullet\bullet\bullet\bullet}{\bullet\bullet\bullet\bullet}\tfrac{\bullet\bullet\bullet\bullet}{\bullet\bullet\bullet\bullet})^m = (\tfrac{\circ\bullet}{**})^{(2m)}\tfrac{1}{\bullet}.$$

When $B_i = (l_1, 1)$, that is, B_i is in the form of "$0A$" for some non-zero A. Assume the function f calculated up to B_i is given by

$$f = T\frac{1}{h_Z(S)}$$

We first handle the symbol "0" by

$$f' \leftarrow f^5 \frac{h^2_{Z,Z}(S)h_{2Z,2Z}(S)h_{4Z,Z}(S)}{h^2_{2Z}(S)h_{4Z}(S)h_{5Z}(S)}; Z' = 5Z$$

Then

$$f' = T^5 \frac{1}{h^5_Z(S)} \frac{h^2_{Z,Z}(S)h_{2Z,2Z}(S)h_{4Z,Z}(S)}{h^2_{2Z}(S)h_{4Z}(S)h_{5Z}(S)}$$

By Lemma 5, We can reduce f' as

$$f' = T^5 \frac{h_{2Z,2Z}(S)}{h^2_{Z,Z}(-S)h_{4Z,Z}(-S)}$$

Next, we process the symbol "A" by

$$f'' \leftarrow f_A f'^5 \frac{h^2_{Z',Z'}(S)h_{2Z',2Z'}(S)h_{4Z',Z'}(S)h_{5Z',AP}(S)}{h^2_{2Z'}(S)h_{4Z'}(S)h_{5Z'}(S)h_{5Z'+AP}(S)}$$
$$Z'' = 5Z' + AP$$

By Lemma 2, We have

$$f'' = f_A f'^5 \frac{h^2_{Z',Z'}(S)h_{4Z',Z'}(S)h_{5Z',AP}(S)}{h_{2Z',2Z'}(-S)h_{5Z'}(S)} \frac{1}{h_{5Z'+AP}(S)}$$

The fraction $\frac{1}{h_{5Z'+AP}(S)}$ will be used to reduce terms in B_{i+1} in next iteration. The above example demonstrates the case when the block B_i is in the form of "$0A$" which belongs to the pattern l_1. It consumes a fraction $\frac{1}{h_Z(S)}$ from previous result and contributes another fraction $\frac{1}{h_{Z''}(S)}$ to next block.Then the computation of this case can be denoted as follows:

$$l_1 \;=\; (0A)^m : \tfrac{1}{\bullet}(\tfrac{\bullet\bullet\bullet\ \bullet\bullet\bullet\bullet}{\bullet\bullet\bullet\ \bullet\bullet\bullet\bullet})^m = (\tfrac{\circ}{**}\tfrac{\circ\bullet\bullet}{*\bullet})^m\ \tfrac{1}{\bullet}.$$

When $B_i = (l_1, 2)$, that is, B_i is in the form of "00". Assume the function f calculated up to B_i is given by

$$f = T\frac{1}{h_Z(S)}$$

We first handle the symbol "0" by

$$f' \leftarrow f^5 \frac{h^2_{Z,Z}(S)h_{2Z,2Z}(S)h_{4Z,Z}(S)}{h^2_{2Z}(S)h_{4Z}(S)h_{5Z}(S)}; Z' = 5Z$$

Then

$$f' = T^5 \frac{1}{h^5_Z(S)} \frac{h^2_{Z,Z}(S)h_{2Z,2Z}(S)h_{4Z,Z}(S)}{h^2_{2Z}(S)h_{4Z}(S)h_{5Z}(S)}$$

By Lemma 5, We can reduce f' as

$$f' = T^5 \frac{h_{2Z,2Z}(S)}{h^2_{Z,Z}(-S)h_{4Z,Z}(-S)}$$

Next, we process the symbol "0" by

$$f'' \leftarrow f'^5 \frac{h^2_{Z',Z'}(S)h_{2Z',2Z'}(S)h_{4Z',Z'}(S)}{h^2_{2Z'}(S)h_{4Z'}(S)h_{5Z'}(S)}; Z'' = 5Z'$$

By Lemma 2, We have

$$f'' = f'^5 \frac{h^2_{Z',Z'}(S)h_{4Z',Z'}(S)}{h_{2Z',2Z'}(-S)} \frac{1}{h_{5Z'}(S)}$$

The fraction $\frac{1}{h_{5Z'}(S)}$ will be used to reduce terms in B_{i+1} in next iteration. The above example demonstrates the case when the block B_i is in the form of "00 which belongs to the pattern l_2. It consumes a fraction $\frac{1}{h_Z(S)}$ from previous result and contributes another fraction $\frac{1}{h_{Z''}(S)}$ to next block.Then the computation of this case can be denoted as follows:

$$l_2 \;=\; (00)^m : \tfrac{1}{\bullet}(\tfrac{\bullet\bullet\bullet\ \bullet\bullet\bullet}{\bullet\bullet\bullet\ \bullet\bullet\bullet})^m = (\tfrac{\circ}{**}\tfrac{\circ\bullet}{*})^m\ \tfrac{1}{\bullet}$$

When $B_i = (l_1, 3)$, that is, B_i is in the form of "A" for some non-zero A. Assume the function f calculated up to B_i is given by

$$f = T\frac{1}{h_Z(S)}$$

We first handle the symbol "A" by

$$f' \leftarrow f_A f^5 \frac{h_{Z,Z}^2(S)h_{2Z,2Z}(S)h_{4Z,Z}(S)h_{5Z,AP}(S)}{h_{2Z}^2(S)h_{4Z}(S)h_{5Z}(S)h_{5Z+AP}(S)}; Z' = 5Z + AP$$

Then

$$f' = f_A T^5 \frac{1}{h_Z^5(S)} \frac{h_{Z,Z}^2(S)h_{2Z,2Z}(S)h_{4Z,Z}(S)h_{5Z,AP}(S)}{h_{2Z}^2(S)h_{4Z}(S)h_{5Z}(S)h_{5Z+AP}(S)}$$

By Lemma 5, We can reduce f' as

$$f' = f_A T^5 \frac{h_{2Z,2Z}(S)h_{5Z,AP}(S)}{h_{Z,Z}^2(-S)h_{4Z,Z}(-S)} \frac{1}{h_{5Z+AP}(S)}$$

The fraction $\frac{1}{h_{5Z+AP}(S)}$ will be used to reduce terms in B_{i+1} in next iteration. The above example demonstrates the case when the block B_i is in the form of "A" which belongs to the pattern l_3. It consumes a fraction $\frac{1}{h_Z(S)}$ from previous result and contributes another fraction $\frac{1}{h_{Z'}(S)}$ to next block.Then the computation of this case can be denoted as follows:

$$l_3 \ = \ A : \frac{1}{\bullet}\left(\frac{\bullet\bullet\bullet\bullet}{\bullet\bullet\bullet\bullet}\right) = \left(\frac{\circ\bullet}{**}\right)\frac{1}{\bullet}$$

When $B_i = (l_1, 4)$, that is, B_i is in the form of "0". Assume the function f calculated up to B_i is given by

$$f = T^{\frac{1}{h_Z(S)}}$$

We first handle the symbol "0" by

$$f' \leftarrow f^5 \frac{h_{Z,Z}^2(S)h_{2Z,2Z}(S)h_{4Z,Z}(S)}{h_{2Z}^2(S)h_{4Z}(S)h_{5Z}(S)}; Z' = 5Z$$

Then

$$f' = T^5 \frac{1}{h_Z^5(S)} \frac{h_{Z,Z}^2(S)h_{2Z,2Z}(S)h_{4Z,Z}(S)}{h_{2Z}^2(S)h_{4Z}(S)h_{5Z}(S)}$$

By Lemma 5, We can reduce f' as

$$f' = T^5 \frac{h_{2Z,2Z}(S)}{h_{Z,Z}^2(-S)h_{4Z,Z}(-S)}$$

The above example demonstrates the case when the block B_i is in the form of "0" which belongs to the pattern l_4. It consumes a fraction $\frac{1}{h_Z(S)}$ from previous result. The pattern l_4 will only appear in the last block. This block will only consumes a fraction from previous block without contributing any.Then the computation of this case can be denoted as follows:

Table 4.1: The terms reductions for different cases.

Case		NL
$l_0 = (AB)^m$	$\frac{1}{\bullet}(\frac{\bullet\bullet\bullet\bullet\ \bullet\bullet\bullet\bullet}{\bullet\bullet\bullet\bullet\ \bullet\bullet\bullet\bullet})^m = (\frac{\circ\bullet}{**})^{(2m)}\frac{1}{\bullet}$	$8m$
$l_1 = (0A)^m$	$\frac{1}{\bullet}(\frac{\bullet\bullet\bullet\ \bullet\bullet\bullet\bullet}{\bullet\bullet\bullet\ \bullet\bullet\bullet\bullet})^m = (\frac{\circ\ \circ\bullet\bullet}{**\ *\bullet})^m\ \frac{1}{\bullet}$	$8m$
$l_2 = (00)^m$	$\frac{1}{\bullet}(\frac{\bullet\bullet\bullet\ \bullet\bullet\bullet}{\bullet\bullet\bullet\ \bullet\bullet\bullet})^m = (\frac{\circ\ \circ\bullet}{**\ *})^m\ \frac{1}{\bullet}$	$6m$
$l_3 = A$	$\frac{1}{\bullet}(\frac{\bullet\bullet\bullet\bullet}{\bullet\bullet\bullet\bullet}) = (\frac{\circ\bullet}{**})\frac{1}{\bullet}$	4
$l_4 = 0$	$\frac{1}{\bullet}(\frac{\bullet\bullet\bullet}{\bullet\bullet\bullet}) = (\frac{\circ}{**})$	3

$$l_4 = 0 : \frac{1}{\bullet}(\frac{\bullet\bullet\bullet}{\bullet\bullet\bullet}) = (\frac{\circ}{**})$$

Therefore, by consuming the fraction contributed by the previous block, each block can eliminate terms using Lemma 5 and 2. The terms reductions for different cases are illustrated as follows:

Notice that it requires special treatments in the first block since there are no fraction prepared at the beginning of the algorithm. The detail algorithm can be found in Algorithm 9.

We give a tiny example with $n = 1591$

$$n = (\underbrace{2}_{Init.}\ \underbrace{23}_{l_0}\ \underbrace{31}_{l_0})_5$$

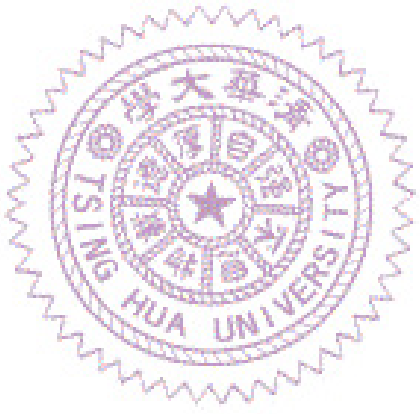
Then, we have

$$f = f_1^{1251} f_2^{125} f_3^{30} h_{1,1}^{625} \frac{h_{4,4}^{125} h_{10,2}^{125}}{\overline{h}_{2,2}^{250} \overline{h}_{8,2}^{125}} \frac{h_{24,24}^{25} h_{60,3}^{25}}{\overline{h}_{12,12}^{50} \overline{h}_{48,12}^{25}} \frac{h_{126,126}^{5} h_{315,3}^{5}}{\overline{h}_{63,63}^{10} \overline{h}_{252,63}^{5}} \frac{h_{636,636} h_{1590,1}}{\overline{h}_{318,318}^{2} \overline{h}_{1272,318} h_{1591}}$$

Notice that the computation of f_2 and f_3 requires 4 lines. Although f_4 is not used here, usually it is needed for a larger n, that requires additional 2 lines. Also, if n is taken as the order of the curve, h_{np} will be trivial and does not need to calculate. Hence one line can be omitted in general. In the above example, we have totally 20 lines to compute f.

In addition, f_1 can be chosen as 1 and therefore this term can also be omitted [12, 19]. When the same number $n = 1591$ is processed by Miller's Algorithm, it takes 31 lines.

The method can also be extended to base $2^{2w} + 1$ representation. We convert n into a radix $2^{2w} + 1$ integer and divide it into blocks of the five patterns. The terms elimination method can be referred to Lemma 6.



Algorithm 9 : Our algorithm : radix-5 representation

INPUT:$n = (b_r b_{r-1} \cdots b_0)_5$ and segments $\{B_0, B_1 \cdots, B_t\}, B_j \in \{l_0, \cdots, l_4\}^*$
OUTPUT:$f = f_n(S)$
$f_2 \leftarrow f_1^2 \frac{h_{P,P}(S)}{h_{2P}(S)}$; $f_3 \leftarrow f_2 \frac{h_{2P,P}(S)}{h_{3P}(S)}$;
$f_4 \leftarrow f_2^2 \frac{h_{2P,2P}(S)}{h_{4P}(S)}$; $u = r - 1$;
if $b_r = 1$ **then** $f \leftarrow f_1 h_P(S)$; $Z \leftarrow P$;
end if
if $b_r = 2$ **then** $f \leftarrow f_1^2 h_{P,P}(S)$; $Z \leftarrow 2P$;
end if
if $b_r = 3$ **then** $f \leftarrow f_1^3 \frac{h_{P,P}(S) h_{2P,P}(S)}{h_{2P}(S)}$; $Z \leftarrow 3P$;
end if
if $b_r = 4$ **then** $f \leftarrow f_1^4 \frac{h_{P,P}^2(S) h_{2P,2P}(S)}{h_{2P}^2(S)}$; $Z \leftarrow 4P$;
end if
for $j \leftarrow 0, 1 \cdots, t$, **do**
    if $B_j = (l_0, m_j)$ **then**
        for $x \leftarrow 1, 2 \cdots, m_j$, **do**
            $f \leftarrow f_{b_u} f^5 \frac{h_{2Z,2Z}(S) h_{5Z,b_u P}(S)}{h_{Z,Z}^2(-S) h_{4Z,Z}(-S)}$;
            $Z \leftarrow 5Z + b_u P$;
            $f \leftarrow f_{b_{u-1}} f^5 \frac{h_{2Z,2Z}(S) h_{5Z,b_{u-1} P}(S)}{h_{Z,Z}^2(-S) h_{4Z,Z}(-S)}$;
            $Z \leftarrow 5Z + b_{u-1} P$; $u \leftarrow u - 2$;
        end for
    end if
    if $B_j = (l_1, m_j)$ **then**
        for $x \leftarrow 1, 2 \cdots, m_j$, **do**
            $f \leftarrow f^5 \frac{h_{2Z,2Z}^2(S)}{h_{Z,Z}^2(-S) h_{4Z,Z}(-S)}$;
            $Z \leftarrow 5Z$;
            $f \leftarrow f_{b_{u-1}} f^5 \frac{h_{Z,Z}^2(S) h_{4Z,Z}(S) h_{5Z,b_{u-1} P}(S)}{h_{2Z,2Z}(-S) h_{5Z}(S)}$;
            $Z \leftarrow 5Z + b_{u-1} P$; $u \leftarrow u - 2$;
        end for
    end if
    if $B_j = (l_2, m_j)$ **then**
        for $x \leftarrow 1, 2 \cdots, m_j$, **do**
            $f \leftarrow f^5 \frac{h_{2Z,2Z}(S)}{h_{Z,Z}^2(-S) h_{4Z,Z}(-S)}$;
            $Z \leftarrow 5Z$;
            $f \leftarrow f^5 \frac{h_{Z,Z}^2(S) h_{4Z,Z}(S)}{h_{2Z,2Z}(-S)}$;
            $Z \leftarrow 5Z$; $u \leftarrow u - 2$;
        end for
    end if
    if $B_j = (l_3, 1)$ **then**
        $f \leftarrow f_{b_u} f^5 \frac{h_{2Z,2Z}(S) h_{5Z,b_u}(S)}{h_{Z,Z}^2(-S) h_{4Z,Z}(-S)}$;
        $Z \leftarrow 5Z + b_u P$; $u \leftarrow u - 1$;
    end if
    if $B_j = (l_4, 1)$ **then**
        $f \leftarrow f^5 \frac{h_{2Z,2Z}(S)}{h_{Z,Z}^2(-S) h_{4Z,Z}(-S)}$; **return** f;
    end if
end for
$f \leftarrow f / h_Z(S)$;
return f;

Chapter 5

Performance Evaluation

Based on different choices of the order n, the running times of pairing computations are different. The running time of our algorithm and the previous schemes [14, 2, 12, 19] depend on the length of n (represented in base 5) and the number of zeros in n. If n is selected in the form of $5^k + 1$ for some integers k, a work from Wang *et al.*[17] tells us it can be easily computed using radix-5 representation. This type of inputs in the best case for our algorithm. In contrast, the worst case for our algorithm is n is represented as a radix 5 string that does not contain any zero. Similar result can be found in the previous schemes, except that their n is represented in binary and radix 3.

In general, n can be selected as any large enough integer that divides the curve order $\#E(F_q)$ and is not divided by the characteristic p where $q = p^r$. For the same n that can be represented in base 5, base 3, and binary, let them be n_5, n_3, n_2 respectively. On one hand, n_5 is shorter than n_3 by a factor of n_3 by a factor of $\ln_3 5 \approx 1.46$ and n_2 by a factor of $\ln_2 5 \approx 2.32$. On the other hand, n_5 has a smaller portion of zeros than n_3 and n_2. The probabilities for a digit be zero in n_5, n_3, and n_2 are $1/5$, $1/3$, and $1/2$ respectively. We should consider both the length and the probability of zero when we compare the performances of different schemes.

Previous schemes were evaluated by counting the numbers of f functions required to calculate given an input k-bits long n with m non-zero digitals represented in binary or radix 3. Some precomputation costs like convertions between different representations and modules exponential of f functions were omitted. In this thesis, we compare our scheme with the three Miller's algorithms (in binary, radix 3 and radix 5 representation),

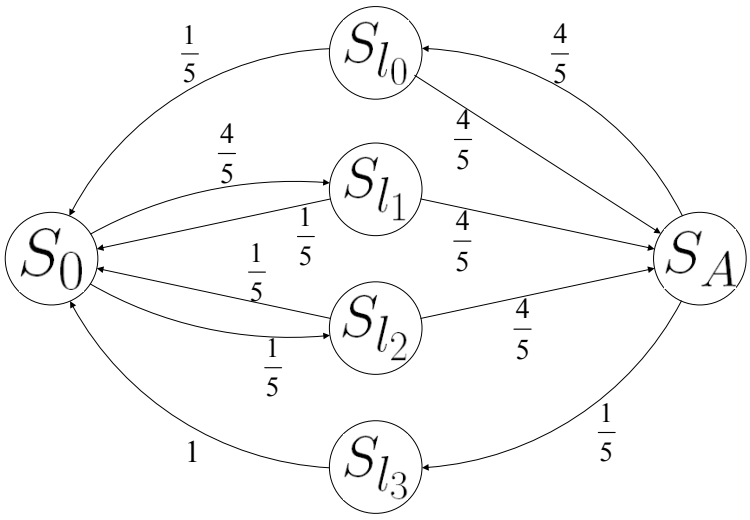


Figure 5.1: Markov chain of the state transitions of our scheme: radix-5

the three Blake *et al.*'s algorithms (BMX-1, BMX-2, BMX-3), Wu *et al.*'s algorithm and Liu *et al.*'s algorithm. We assume that the n is randomly chosen such the value of each digit (represented in radix 5, radix 3 and binary) is independent and identically-distributed random variable. We draw Markov chains for each scheme to evaluate the probability of each state. These probabilities multiplied by the cost (number of f functions) of each states are the average cost per state. This number is further normalized by factors of the average digits consumed by a state and the number of digits in each representations. The Markov chain for our algorithm shown in Fig. 5.1.

Notice that we do not include l_4 in our Markov chain since it only appears at the end of the string and is an absorbing state. By solving the Markov chain, we can obtain the probabilities of occurrences P_S of each state S.

$$\begin{aligned} P &= (P_{l_0}, P_{l_1}, P_{l_2}, P_{l_3}, P_A, P_0) \\ &= (0.276, 0.124, 0.031, 0.068, 0.345, 0.155) \end{aligned}$$

The cost consumed in each state can be viewed as the followings. Let $(C_0, C_1, C_2, C_3, C_4)$ be the cost vector of the pattern l_0, l_1, l_2, l_3, l_4. We can count the value of C in previous section.

$$(C_0, C_1, C_2, C_3, C_4) = (8, 8, 6, 4, 3)$$

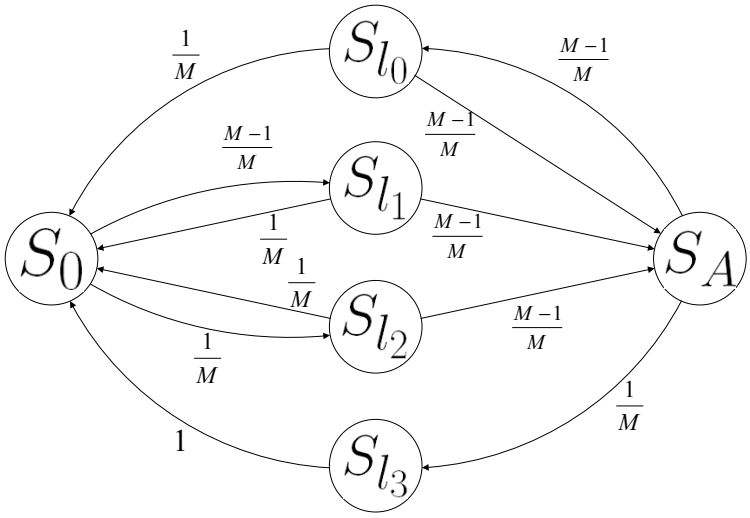


Figure 5.2: Markov chain of the state transitions of our scheme: radix-M

A block usually contains two digits, except for the blocks in patterns l_3 and l_4. We define two states S_A and S_0. Each of them represents a transitive state that takes a digit A or 0 respectively. The states $S_{l_0}, S_{l_1}, S_{l_2}, S_{l_3}$ represent the corresponding pattern handlers that process a block. Since states $S_{l_0}, S_{l_1}, S_{l_2}$ consume two digits and S_{l_3} consumes only one digit, we assign each state S with the cost C_S.

$$C = (C_{l_0}, C_{l_1}, C_{l_2}, C_{l_3}, C_A, C_0) = (4, 4, 3, 0, 4, 3.8)$$

Take an example, a block in the form of $0A$ belongs to pattern l_1. It transits through S_0 and S_{l_1} where each of them has a cost of 4. Totally it has a cost of $8 = C_1$.

We can calculate the average cost per state by

$$P \cdot C^T = 3.662 \tag{5.1}$$

Since all states consume one digit except state S_{l_3} that does not consume any state. On average, each state consumes x digits where

$$x = P \cdot (1, 1, 1, 0, 1, 1)^T = 0.931 \tag{5.2}$$

Therefore, we can normalize the average cost per state to the number of digits consumed by each state as $3.662/0.931 = 3.933$. By considering the difference in the lengths of

base 5 representation and binary representation, this value should been divided by $log_2 5$. The final cost of our algorithm for a k-bits binary input n is $1.694k$.

We evaluate other schemes using the same method. The results of these schemes are illustrated in Table 5.1. Our algorithm is the second best amongst all schemes and very closed to the best algorithm. We have simulated the our algorithm and LHC algorithm by randomly picking 1000 n for each of the following sizes: 10, 20, 30,..., 160 bits. The result of our algorithm and LHC algorithm are represented as "$\times$" and "+" in Fig 5.3 respectively. They are also plotted with the theoretical results that is calculated in Table 5.1. It is shown that our theoretical results are very close to the experimental results. Also, the different between our's and LHC's are indeed very subtle. In addition, we has the best performance in base 5 representation. If characteristic 5 curve is used, our algorithm can take the advantage of fast multiplication of $5Z$ that will be faster than Liu's scheme.

Table 5.1: Comparing the costs in different schemes.

	Radix	Cost Per Digit	Cost Per Bit
Miller_2	2	3	3
Miller_3	3	5.333	3.365
Miller_5	5	7.6	3.273
BMX-1	4	4	2
BMX-2	2	2	2
BMX-3	3	4.556	2.874
LHC	2	1.667	1.667
Wu *et al.*	3	3.667	2.313
Our	5	3.933	1.694

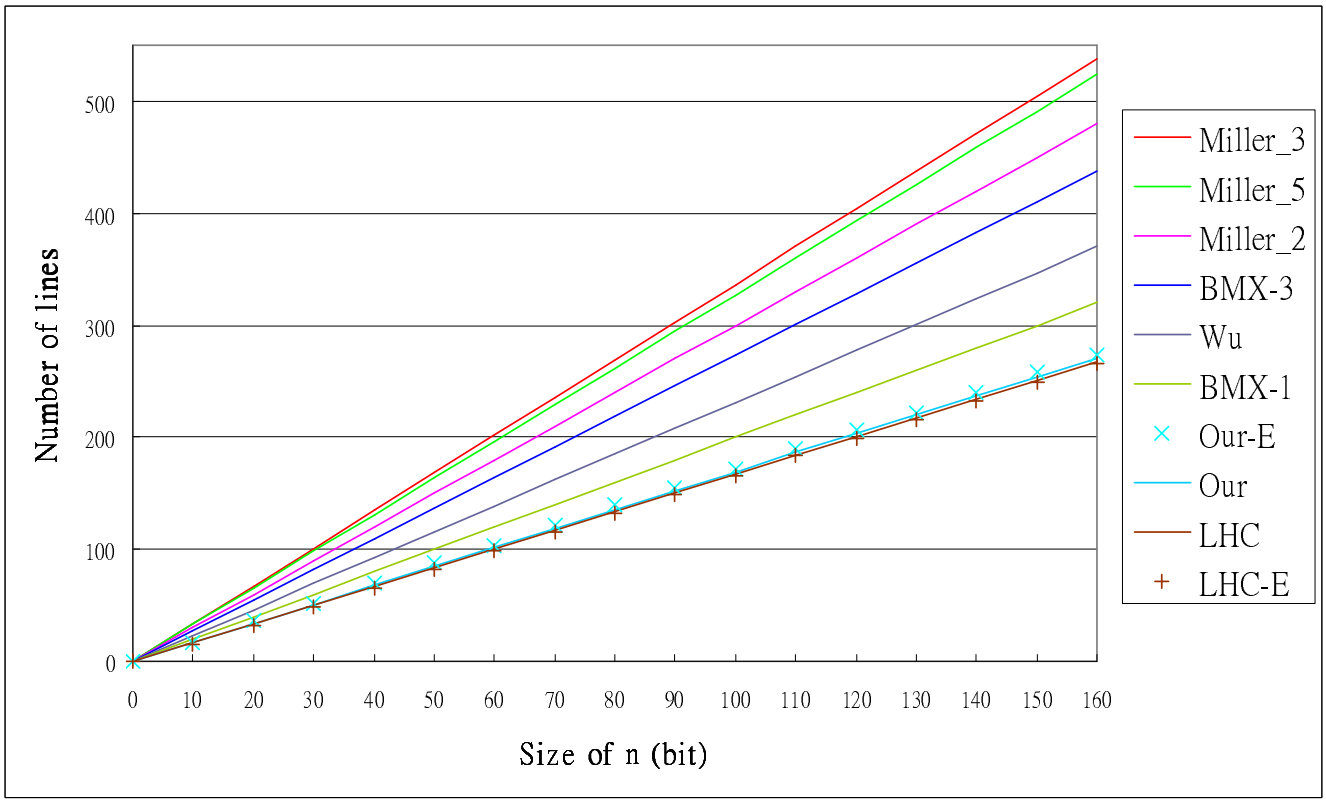


Figure 5.3: Comparing the costs in different schemes.

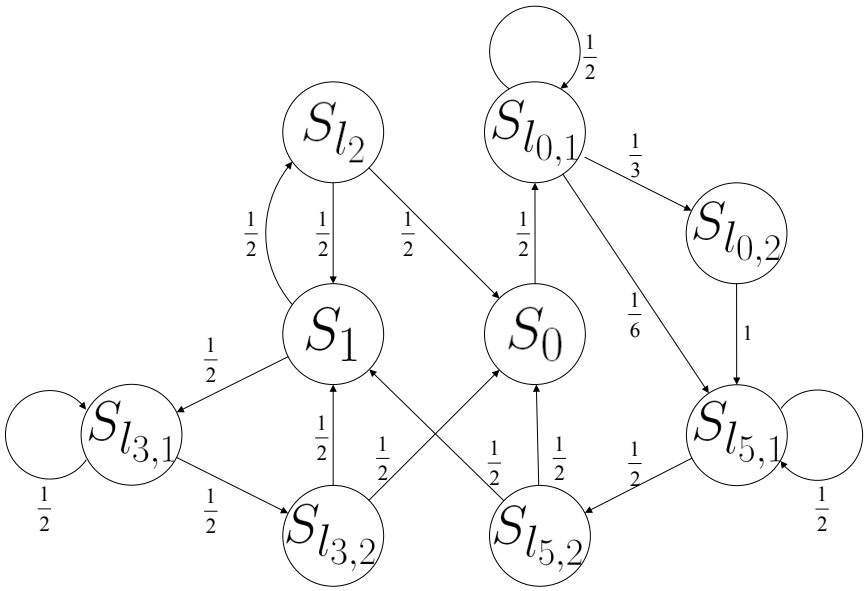


Figure 5.4: Markov chain of the state transitions of LHC algorithm

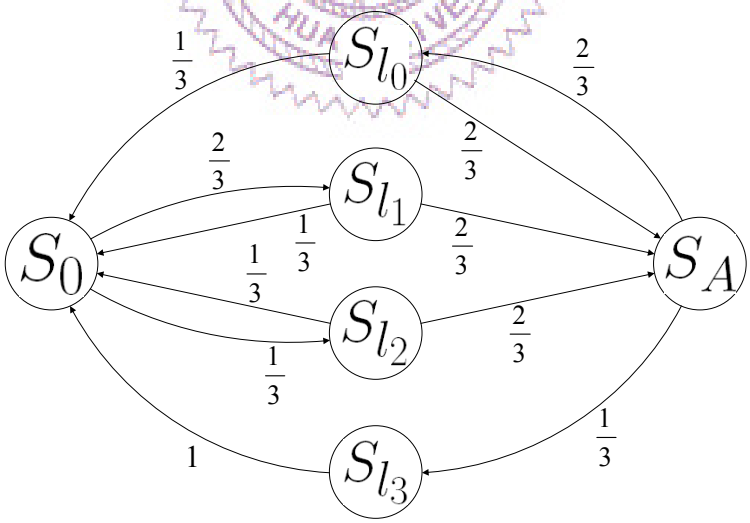


Figure 5.5: Markov chain of the state transitions of Wu's algorithm

Chapter 6

Conclusion

In this thesis we have presented an efficient algorithm to improve the computation of Weil pairing and Tate pairing by reducing the lines calculations during Miller's algorithm. Our algorithm performs excellently and is very suitable for curve with characteristic 5. Similar technique can be applied to curves with $2^{2w}+1$ characteristic using radix $2^{2w}+1$ representation.

In both theoritical analysis and experimental results show that our scheme have pretty good performances in radix 5. In the future, we will also simulate our scheme in other characteris in the form $2^{2w}+1$. We also believe there are rooms to improve our scheme in computation efficiency.

Bibliography

[1] P. Barreto, H.Y. Kim, B. Lynn, and M. Scott. Efficient algorithms for pairing-based cryptosystems. *Advances in Cryptology–Crypto*, 2442:354–368, 2002.

[2] I.F. Blake, V. Kumar Murty, and G. Xu. Refinements of Miller's algorithm for computing the Weil/Tate pairing. *Journal of Algorithms*, 58(2):134–149, 2006.

[3] I.F. Blake, G. Seroussi, and N.P. Smart. *Elliptic Curves in Cryptography*. Cambridge University Press, 1999.

[4] D. Boneh and M.K. Franklin. Identity-Based Encryption from the Weil Pairing. *Proceedings of the 21st Annual International Cryptology Conference on Advances in Cryptology*, pages 213–229, 2001.

[5] D. Boneh, E.J. Goh, and K. Nissim. Evaluating 2-DNF formulas on ciphertexts. *TCC*, 3378:325–341, 2005.

[6] H. Cohen, G. Frey, and R. Avanzi. *Handbook of Elliptic and Hyperelliptic Curve Cryptography*. CRC Press, 2006.

[7] S.D. Galbraith, K. Harrison, and D. Soldera. Implementing the Tate Pairing. *Proceedings of the 5th International Symposium on Algorithmic Number Theory*, pages 324–337, 2002.

[8] I.N. Herstein. *Topics in algebra*. Xerox College Pub Lexington, Mass, 1975.

[9] A. Joux. A One Round Protocol for Tripartite Diffie-Hellman. *Proceedings of the 4th International Symposium on Algorithmic Number Theory*, pages 385–394, 2000.

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

[11] S. Lang. *Algebra*. Springer, 2002.

[12] C.L. Liu, G. Horng, and T.Y. Chen. Further refinement of pairing computation based on Millers algorithm. *Applied Mathematics and Computation*, 189(1):395–409, 2007.

[13] AJ Menezes, T. Okamoto, and SA Vanstone. Reducing elliptic curve logarithms to logarithms in a finite field. *Information Theory, IEEE Transactions on*, 39(5):1639–1646, 1993.

[14] V. Miller. Short programs for functions on curves. *Unpublished manuscript*, 97:101–102, 1986.

[15] J.H. Silverman. *The Arithmetic of Elliptic Curves*. Springer Verlag, 1986.

[16] W. Stallings. *Cryptography and network security*. Prentice Hall Upper Saddle River, NJ.

[17] K. Wang and B. Li. Computation of Tate Pairing for Supersingular Curves over characteristic 5 and 7.

[18] L.C. Washington. *Elliptic Curves: Number Theory and Cryptography*. CRC Press, 2003.

[19] T. Wu, H.Q. Du, M. Zhang, and R.B. Wang. Improved Algorithm of the Tate Pairing in Characteristic Three. *Data, Privacy, and E-Commerce, 2007. ISDPE 2007. The First International Symposium on*, pages 453–455, 2007.