Fast computation of Tate pairing on general divisors of genus 3 hyperelliptic curves

* Eunjeong Lee[a], Hyang-Sook Lee[b] and Yoonjin Lee[c]

[a] *School of Computational Sciences, Korea Institute for Advanced Study, 130-722, Seoul, Korea*
[b] *Department of Mathematics, Ewha Womans University, 120-750, Seoul, Korea*
[c] *Department of Mathematics, Simon Fraser University, V5A 1S6, British Columbia, Canada*

Abstract

For the Tate pairing implementation over hyperelliptic curves, there is a development by Duursma-Lee and Barreto et al., and those computations are focused on *degenerate* divisors. As divisors are not degenerate form in general, it is necessary to find algorithms on *general* divisors for the Tate pairing computation. In this paper, we present two efficient methods for computing the Tate pairing over divisor class groups of the hyperelliptic curves $y^2 = x^p - x + d,\ d = \pm 1$ of genus 3. First, we provide the *point-wise* method, which is a generalization of the previous developments by Duursma-Lee and Barreto et al. In the second method, we use a *Resultant* for the Tate pairing computation. According to our theoretical analysis of the complexity, the *Resultant* method is approximately three times faster than the point-wise method, and our implementation result shows that the *Resultant* method is much faster than the point-wise method. These two methods are completely general in a sense that they work for general divisors, and they provide very explicit algorithms.

keywords: Tate pairing; hyperelliptic curves; divisors; eta pairing; resultant; pairing-based cryptosystem

Introduction

In recent years the Tate pairing and the Weil pairing have been getting a lot of attentions for designing various protocols in cryptosystem [4, 5, 15, 14, 23, 27, 28]. It is therefore important to develop the efficient implementation of those pairings for the practical applications in our real world. It is known that the computation of the Weil pairing is almost the same as computing the Tate pairing twice [11], therefore the Tate pairing implementation have been actively developed. The recent papers by Barreto et al. [1] and Galbraith et al. [12] provided the fast computation of the Tate pairing over the supersingular elliptic curves $y^2 = x^3 - x \pm 1$ in characteristic three. In 2003, Duursma and Lee [9] generalized their results to the hyperelliptic curves $y^2 = x^p - x \pm 1,\ p = 3 \pmod 4$ in characteristic p. In particular, they provided a closed formula for the efficient computation of the Tate pairing. After then, Barreto et al. [2] proposed a general technique for the efficient computation of the Tate pairing on supersingular abelian varieties using *Eta pairing* approach [2, 21]. They also described efficient pairing algorithms on $\mathbb{F}_q$-rational points for elliptic and hyperelliptic curves over $\mathbb{F}_q$.

[*] (e-mail) ejlee@kias.re.kr, hsl@ewha.ac.kr, yoonjinl@sfu.ca The authors[a,b] were supported by KOSEF, grant number R01-2005-000-10713-0. This work was also done while the last author[c] was visiting KIAS in Seoul during the summers of 2004 and 2005. The author expresses their gratitude to the institute.

In fact, generally divisor operations over hyperelliptic curves are more complicated than point operations over elliptic curves. Therefore, it has been pointed out that *Elliptic Curve Cryptosystem* (ECC) is more efficient than *Hyperelliptic Curve Cryptosystem* (HCC) [27]. The Tate pairing computation uses the Miller algorithm, and the Miller algorithm relies on divisor operations. Thus one expects that the Tate pairing computation over hyperelliptic curves may not be as efficient as that over elliptic curves. However, in some special cases, it was shown that HCC can be made more efficient than ECC by giving the explicit formula for divisor operations [6, 20, 24]. For the higher genus, preserving the same security level, we can decrease the size of the defining field. In fact, some examples given in [6, 20] show that for the efficiency of cryptosystems, the size of the defining field is more important than the complexity of group operation formula. For the Tate pairing, Barreto et al. [2] presented implementation results over elliptic curves and hyperelliptic curves of genus 2, where both are defined over $\mathbb{F}_{2^n}$. The running time for the Tate pairing over hyperelliptic curves was faster than that of elliptic curves. For the Tate pairing computation it is therefore certainly worthy to work over some special types of hyperelliptic curves.

Recent developments [2, 9] on the Tate pairing implementation on hyperelliptic curves over a finite field $\mathbb{F}_q$ have focused on the case of *degenerate divisors* as mentioned before. However, in the pairing-based cryptography, the efficient Tate pairing implementation over *general divisors* is significantly important. For instance, in the Boneh-Franklin identity-based encryption scheme, the private keys are general divisors, and therefore the decryption process requires computing a pairing of general divisors. For the case of genus 2, the result in [6] presents both divisor-wise and point-wise approach, and it turns out that the divisor-wise approach is more efficient than the point-wise approach. For the case of genus ≥ 3, when divisors are *general*, there has been no Tate pairing computation method developed so far.

In this paper, we develop the general method of computing the Tate pairing for the genus 3 case. We present two feasible methods by point-wise approach and Resultant approach for computing the Tate pairing over divisor class groups of the hyperelliptic curves $H_d : y^2 = x^p - x + d,\ d = \pm 1$ of $p = 7$. Those methods are completely general in a sense that they work for general divisors, and we give very explicit and feasible formulas over H_d. Furthermore, we investigate the efficiency and compare the complexity between two methods. In fact, efficient algorithms for computing the resultant have been developed; for instance in [25]. For our two methods, we use the *Eta pairing* for reducing the computation cost. We used SINGULAR [13] software package for symbolic computations in this paper.

The first method in Section 2 is a generalization of the point-wise method developed in ([2], [9]), and our method is an algorithm for computing the Tate pairing over *general* divisors. The second method in Section 3 is by using *Resultant*. In Section 4, we compare the complexity between two methods, and the result shows that the Tate pairing computation by using Resultant is approximately three times faster than the point-wise approach. In Section 5 we provide experimental results using NTL [26] software package. According to our implementation result, we conclude that the Tate pairing computation by using Resultant is much faster than the point-wise approach. We point it out that this is the first implementation over a genus 3 curve.

1 Tate pairing on divisors

Let $\mathbb{F}_q$ be a finite field with q elements, and $H/\mathbb{F}_q$ be a hyperelliptic curve over $\mathbb{F}_q$. We denote the group of degree zero divisor classes of H by J_H. Note that each divisor class can be uniquely represented by the *reduced divisor* using the *Mumford representation* [22]. Reduced divisors of the curve H can be found as discussed in ([18], [22]), and most of reduced divisors in J_H with genus 3 are written as $D = [U_D, V_D] = [x^3 + u_{D,2}x^2 + u_{D,1}x + u_{D,0},\ v_{D,2}x^2 + v_{D,1}x + v_{D,0}]$.

We recall the definition of the Tate pairing [10]. Let ℓ be a positive divisor of the order of $J_H(\mathbb{F}_q)$ with $\gcd(\ell,\ q) = 1$, and k be the smallest integer such that $\ell \mid (q^k - 1)$; such k is called *the embedding degree.*

Let $J_H[\ell] = \{D \in J_H \mid \ell D = \mathcal{O}\}$. The *Tate pairing* is a map

$$\begin{aligned} \langle \cdot \rangle_\ell : J_H[\ell] \times J_H(\mathbb{F}_{q^k})/\ell J_H(\mathbb{F}_{q^k}) &\rightarrow \mathbb{F}^*_{q^k}/(\mathbb{F}^*_{q^k})^\ell \\ \langle D, E \rangle_\ell &= f_D(E'), \end{aligned}$$

where $\mathrm{div}(f_D) = \ell D$ and $E' \sim E$ with *support*$(E') \cap$ *support*$(\mathrm{div}(f_D)) = \emptyset$. We define the Tate paring value by $t(D, E) = \langle D, E \rangle_\ell^{\frac{q^k-1}{\ell}}$ so that the pairing value is defined uniquely. We write $x^{(i)}$ for x^{p^i}.

We consider a hyperelliptic curve H_d over $\mathbb{F}_q$ defined by $y^2 = x^p - x + d$, $d = \pm 1$, for $p \equiv 3 \pmod 4$, where $q = p^n$ with $\gcd(2p, n) = 1$, and we let $F/\mathbb{F}_q$ and $K/\mathbb{F}_q$ be the extensions of degree $[F : \mathbb{F}_q] = p$ and degree $[K : \mathbb{F}_q] = 2p$, respectively. Over the extension field K, the curve is the quotient of a hermitian curve, hence it is Hasse-Weil maximal. And the class group over K is annihilated by $p^{pn} + 1$; this can be also seen from the following Lemma 1.1. It shows that for $P \in H_d(K)$, $(p^{pn} + 1)((P) - (\mathcal{O}))$ is principal [8].

Lemma 1.1 ([8])**.** *Let $P = (\alpha, \beta) \in H_d$. The function*

$$h_P = \beta^p y - (\alpha^p - x + d)^{\frac{p+1}{2}}$$

has divisor $(h_P) = p(P) + (P') - (p + 1)\mathcal{O}$, where $P' = (\alpha^{(2)} + 2d, \beta^{(2)})$.

¿From Lemma 1.1, we observe that

$$p((P) - (\mathcal{O})) \equiv ([p]P) - (\mathcal{O}),$$

thus the multiplication by p over H_d has an extremely special form such as $[p] = \phi\pi^2$, where $\phi = (x + 2d, -y)$ and π is a Frobenius map of p^{th} power.

Lemma 1.2 ([8])**.**

$$|J_{H_{(+1)}}(k)||J_{H_{(-1)}}(k)| = (p^{pn} + 1)/(p^n + 1)$$

In paticular, letting $[k : \mathbb{F}_7] = n$ and $m = (n + 1)/2$, we thus have

$$\begin{aligned} |J_{H_{(+1)}}(k)| &= (1 + 7^n)^3 + (\frac{7}{n})7^m(1 + 7^n + 7^{2n}). \\ |J_{H_{(-1)}}(k)| &= (1 + 7^n)^3 - (\frac{7}{n})7^m(1 + 7^n + 7^{2n}). \end{aligned}$$

And $|J_{H_{(+1)}}(k)||J_{H_{(-1)}}(k)| = (1 + 7^{7n})/(1 + 7^n)$.

1.1 Eta pairing on H_d

We discuss the *Eta pairing* introduced in [2] which is very useful for efficient computation of the Tate pairing.

We consider a hyperelliptic curve H_d over some finite field $\mathbb{F}_{p^n}$, and let ψ be an endomorphism on the curve H_d given by

$$\psi : H_d(K) \rightarrow H_d(K), \quad \psi(x, y) = (\rho - x, \sigma y), \qquad (1)$$

where $\rho \in F$ is a root of $\rho^p - \rho + 2d = 0$, and $\sigma, \bar{\sigma} \in K$ are the roots of $\sigma^2 + 1 = 0$.

For efficient Tate pairing computation, we concern with *the twisted Tate pairing*

$$\begin{aligned} \hat{t} \ : \ J_{H_d}[\ell] \times J_{H_d}(\mathbb{F}_{p^{2pn}})/\ell J_{H_d}(\mathbb{F}_{p^{2pn}}) \ &\longrightarrow \ \mathbb{F}^*_{p^{2pn}} \\ \hat{t}(D,\ E) \ &= \ f_D(\psi(E))^{p^{pn}-1}, \end{aligned}$$

where $(f_D) = (p^{pn}+1)D$ from [9, Theorem 4].

For two divisors D and E in J_{H_d}, the *Eta pairing* is defined by

$$\eta(D,E) = \prod_{i=0}^{n-1} h_{D_i}(\psi(E))^{p^{n-1-i}}, \tag{2}$$

where $D_{i+1} + (h_{D_i}) = pD_i$ with a divisor $D_0 = D$ and some rational function h_{D_i}.

By Lemma 1.1, H_d has a property that

$$q(P) - q(\mathcal{O}) = (\gamma(P)) - (\mathcal{O}) + (g_P), \tag{3}$$

for some automorphism γ on H_d and some function g_P. Thus γ can be given by $\gamma = \phi^n\pi^{2n}$. Now we provide the crucial theorem for efficient computation of the Tate pairing on divisors proved in [21].

Theorem 1.3 ([2], [21])**.** *Let $q = p^n, p \equiv 3 \pmod 4$, $\gamma = \phi^n\pi^{2n}$ on J_{H_d} induced from Eq. (3), and ψ be an endomorphism on the curve H_d over $\mathbb{F}_q$. Assume that*

$$\phi^n\psi^{[q]} = \psi, \tag{4}$$

where $\psi^{[q]}$ denotes a map obtained by raising the coefficients of ψ by q^{th} power. Then for divisors D and E in $J_H(\mathbb{F}_q)$, we have

$$\eta(qD,E) = \eta(D,E)^q.$$

For any divisor $E = [U_E, V_E]$ in $J_{H_d}(\mathbb{F}_q)$, the endomorphism ψ in Eq.(1) on divisors are easily deduced as follows: $\psi(E) = [U_{\psi(E)},\ V_{\psi(E)}]$, where

$$\begin{aligned} U_{\psi(E)} &= x^3 - (3\rho + u_{E,2})x^2 + (3\rho^2 + 2u_{E,2}\rho + u_{E,1})x - (\rho^3 + u_{E,2}\rho^2 + u_{E,1}\rho + u_{E,0}), \\ V_{\psi(E)} &= \sigma(v_{E,2}x^2 - (2\rho v_{E,2} + v_{E,1})x + v_{E,2}\rho^2 + v_{E,1}\rho + v_{E,0}). \end{aligned} \tag{5}$$

The following lemma shows that our curve H_d satisfies the crucial condition in Eq. (4) for Theorem 1.3, and the proof is straightforward by using Eq. (5).

Lemma 1.4. *Let E be a divisor of the curve H_d, and ϕ be a map defined on the curve H_d such that $\phi(x,y) = (x+2d, -y)$. Then we have the following*

$$\phi^n\psi^{[q]}(E) = \psi(E).$$

From Theorem 1.3 and Lemma 1.4, it follows that

$$\hat{t}(D,\ E) = \eta(D,E)^{7^{6n+1}(7^{7n}-1)}. \tag{6}$$

It is therefore enough to compute $\eta(D,E)$ to obtain $\hat{t}(D,\ E)$ for any divisors $D, E \in J_{H_d}(\mathbb{F}_q)$. When all the points in *support*(D) and *support*(E) are $\mathbb{F}_q$-rational points, using Eq. (6) makes the Tate pairing computation very efficient as mentioned in [2]. In Section 2, we will extend the concept of the Eta pairing on the general divisors, that is, the supports of D and E are not necessarily $\mathbb{F}_q$-rational points. In our two methods we use the Eta pairing for reducing the computation cost.

Throughout this paper, we focus on the hyperelliptic curve $H_d : y^2 = x^p - x + d,\ d = \pm 1, p \equiv 3$ (mod 4) of genus $g = 3$, therefore we work on the case $p = 7$; this case is cryptographically useful [9].

2 Point-wise computation of the Tate pairing

This section presents a generalization of the point-wise method developed in [2] and [9], and this method can be used for any divisors, not only for $\mathbb{F}_q$-rational points. The Eta pairing is used for reducing the computation cost as well.

As mentioned in [2], for divisors D, E in J_{H_d}, the Tate pairing can be computed by

$$\hat{t}(D,E) = \prod_{i,j=1}^{3} \hat{t}(P_i, Q_j), \tag{7}$$

where D and E have the form $D = (P_1) + (P_2) + (P_3) - 3(\mathcal{O}),\ \ E = (Q_1) + (Q_2) + (Q_3) - 3(\mathcal{O})$ for points P_i and Q_j contained in $H_d(\mathbb{F}_{7^{3n}})$ with $i, j = 1, 2, 3$.

If we want to apply the Eta pairing introduced in [2] for computing $\hat{t}(P_i, Q_j)$, then it requires that P_i and Q_j belong to $H_d(\mathbb{F}_{7^n})$ for each i, j. Therefore we define a new pairing, η_3, similar to the Eta pairing for general divisors as follows:

$$\eta_3(P,Q) = \prod_{i=0}^{3n-1} h_{D_i}(\psi(Q))^{7^{3n-1-i}}. \tag{8}$$

Then we can efficiently compute the Tate pairing for general divisors by using the following theorem and Eq. (7).

Theorem 2.1. *For $P, Q \in H_d(\mathbb{F}_{7^{3n}})$,*

$$\hat{t}(P,Q) = (\eta_3(P,Q)^{2\cdot 7^{4n}} \eta(P,Q)^{7^{6n}})^{7^{7n}-1}.$$

Proof. We have $\hat{t}(P,Q) = f_D(\psi(Q))^{7^{7n}-1}$ with $(f_D) = (7^{7n})((P) - (\mathcal{O})) - (-P) + (\mathcal{O})$. We also notice that

$$\begin{aligned} f_D(\psi(Q)) &= \eta_3(P,Q)^{7^{4n}} \eta_3(7^{3n}P,Q)^{7^n} \eta(7^{6n}P,Q) \\ &= \eta_3(P,Q)^{2\cdot 7^{4n}} \eta(P,Q)^{7^{6n}}, \end{aligned}$$

where the last equality follows from the facts

$$\eta_3(7^{3n}P,Q) = \eta_3(P,Q)^{7^{3n}},\ \ \eta(7^{6n}P,Q) = \eta(P,Q)^{7^{6n}};$$

these can be directly derived from the proof of Theorem 1.3 since $P, Q \in H_d(\mathbb{F}_{7^{3n}})$. □

Based on Theorem 2.1, we have Algorithm 1 for computing the Tate pairing of P and Q.

We notice that $f_D(\psi(Q)) = (\eta_3(P,Q)^2 \eta(P,Q)^{7^{2n}})^{7^{4n}}$. Now we discuss the complexity of Algorithm 1 by counting the number of operations which are necessary for computing

$$\prod_{i,j=1}^{3} \eta_3(P_i,Q_j)^2 \eta(P_i,Q_j)^{7^{2n}}. \tag{9}$$

We denote the time for multiplications in $\mathbb{F}_{7^n}$, $\mathbb{F}_{7^{7n}}$ and $\mathbb{F}_{7^{14n}}$ by m, m' and M, respectively, and a multiplication between $\mathbb{F}_{7^n}$ and $\mathbb{F}_{7^{7n}}$ by $\tilde{m}$. For simplicity, we assume that a squaring cost is similar to a

Algorithm 1 Point-wise computation

```
INPUT $P = (\alpha,\ \beta), Q = (x,\ y) \in \mathbb{F}_{7^{3n}}$
OUTPUT $\hat{t}(P,\ Q)$
1:  $g_1 \leftarrow 1$
2:  For $i = 0$ to $n - 1$ do
3:      compute $h = \beta^7 \cdot y \cdot \sigma - (\alpha^7 + x + d - \rho)^4$
4:      set $g_1 \leftarrow g_1^7 \cdot h$
5:      set $\alpha \leftarrow \alpha^{7^2} + 2d,\ \beta \leftarrow \beta^{7^2}$
6:  $g \leftarrow g_1$
7:  For $i = n$ to $3n - 1$ do
8:      compute $h = \beta^7 \cdot y \cdot \sigma - (\alpha^7 + x + d - \rho)^4$
9:      set $g \leftarrow g^7 \cdot h$
10:     set $\alpha \leftarrow \alpha^{7^2} + 2d,\ \beta \leftarrow \beta^{7^2}$
11: Return $(g^2 g_1^{7^{2n}})^{7^{4n}(7^{7n}-1)}$, where $g = \eta_3(P,Q),\ g_1 = \eta(P,Q)$.
```

multiplication cost, and we omit the computation cost for 7^{th} powering since it is negligible comparing to the other operations.

For each point P (resp. Q) in the support of D (resp. E), the step 3 requires 2 multiplications (mult.) and two squarings (sq.) in $\mathbb{F}_{7^{3n}}$, and the step 4 needs a multiplication in $\mathbb{F}_{7^{3(14n)}}$. Since these operations are repeated twice in the steps 8 and 9, the number of operations is

$$3n\ ((2 \text{ mult. } + 2 \text{ sq.}) \text{ in } \mathbb{F}_{7^{3n}} + 1 \text{ mult. in } \mathbb{F}_{7^{3(14n)}}) + 2 \text{ mult. in } \mathbb{F}_{7^{3(14n)}}.$$

Let m_3 and M_3 be the time cost for a multiplication in $\mathbb{F}_{7^{3n}}$ and $\mathbb{F}_{7^{3(14n)}}$ respectively. For computing Eq. (9), the total complexity is

$$2\ T_{3rt} + 9\ (3n\ (4m_3 + 1M_3) + 2M_3) + 8M_3, \tag{10}$$

which is equal to

$$T_P := 2\ T_{3rt} + n\ (108m_3 + 27M_3) + 26M_3, \tag{11}$$

where T_{3rt} is the time for finding all the roots of a cubic polynomial over $\mathbb{F}_{7^{3n}}$; this is required for obtaining the supporting points of D and E.

3 Computation of the Tate pairing by using Resultant

In this section, our goal is, for given divisor inputs with the divisor representation, to find an efficient algorithm which provides us the final Tate pairing value over H_d. The resultant is a well-known tool for evaluating a function at a divisor. With this idea we apply the *Resultant* for the efficient computation of the Tate pairing, and show that this approach is much faster than the first method. As a matter of fact, there has been much development for the resultant in terms of its properties and efficient computation, for instance in [29].

According to Eq. (6), for the Tate pairing computation over divisors $D, E \in J_H(\mathbb{F}_{7^n})$, it is sufficient to compute

$$\eta(D, E) = \prod_{i=0}^{n-1} h_{D_i}(\psi(E))^{7^{n-1-i}},$$

where $D_{i+1} + (h_{D_i}) = 7D_i$ with a divisor $D_0 = D$ and some rational function h_{D_i}. In order to obtain the value of $\eta(D, E)$, in the following subsections 3.1 and 3.2, we find the explicit formulas for $D_i = [7^i]D$ and h_{D_i}, $i \geq 1$, and we also obtain the evaluation formula of rational function h_{D_i} at a divisor in a very explicit way.

3.1 7-multiplication on divisors

Let D be a reduced divisor of H_d such that

$$D = (P_1) + (P_2) + (P_3) - 3\mathcal{O} = [U_D, V_D],$$

where $P_j = (\alpha_j, \beta_j)$ for $j = 1, 2, 3$, $U_D = x^3 + u_{D,2}x^2 + u_{D,1}x + u_{D,0}$, and $V_D = v_{D,2}x^2 + v_{D,1}x + v_{D,0} \in \mathbb{F}_{7^n}[x]$. Let $D_0 = D$, $D_{i+1} + (h_{D_i}) = 7D_i$, and $D_i = [U_{D_i}, V_{D_i}]$ for each positive integer i.

The following lemma provides us with explicit formulas for U_{D_i} and V_{D_i} in terms of the coefficients of U_D and V_D for $i \geq 1$. The proof can be obtained from the knowledge of Section 5 in Appendix of [18].

Lemma 3.1. *Let* $[7]$ *be the multiplication map by 7 on the divisor class group of* $H_d/\mathbb{F}_{7^n}$. *Then we have, for* $i \geq 1$, $[7^i]D = D_i = [U_{D_i}, V_{D_i}]$ *with*

$$\begin{aligned} U_{D_i} &= x^3 + (u_{D,2}^{(2i)} + id)x^2 + (u_{D,1}^{(2i)} + 3idu_{D,2}^{(2i)} - 2i^2)x + u_{D,0}^{(2i)} - 2idu_{D,1}^{(2i)} - 3i^2u_{D,2}^{(2i)} - i^3d, \\ V_{D_i} &= (-1)^i v_{D,2}^{(2i)}x^2 + (-1)^i(3idv_{D,2}^{(2i)} + v_{D,1}^{(2i)})x + (-1)^i(-3i^2v_{D,2}^{(2i)} - 2idv_{D,1}^{(2i)} + v_{D,0}^{(2i)}). \end{aligned}$$

In the following proposition, we find the function h_D such that $(h_D) = 7D + (D')$ in an explicit way.

Proposition 3.2. *Let* $h_D(x, y)$ *be a function such that* $(h_D) = 7D + (D')$, *and* τ *be a map*

$$\tau : H_d \to \hat{H}_d, \ \ (x, y) \to (\hat{X}, Y) = (x - \xi^7 - d, y).$$

Then, for appropriate ξ, *we have*

$$\hat{h}_{\tilde{D}}(\hat{X}, Y) := h_D(x, y) \circ \tau^{-1} = \delta_1 Y^3 + s(\hat{X})Y^2 + t(\hat{X})Y + \delta_{16}(\hat{X}),$$

where $\delta_{16}(\hat{X}) = -(\hat{X}^3 + \tilde{u}_1^7\hat{X} + \tilde{u}_0^7)^4$, *and* δ_1, $s(\hat{X})$ *and* $t(\hat{X})$ *are described in Table 1.*

Proof. Let $D = (P_1) + (P_2) + (P_3) - 3(\mathcal{O})$, and $7((P_j) - (\mathcal{O})) = (h_j) - (P'_j) + (\mathcal{O})$ for $j = 1, 2, 3$. Then

$$7D = (h_1h_2h_3) - [(P'_1) + (P'_2) + (P'_3) - 3(\mathcal{O})] = (h_D) - D',$$

where $D' = (P'_1) + (P'_2) + (P'_3) - 3(\mathcal{O})$.

For simplicity, we use the change of variable

$$\tau_\xi : x \to X = x - \xi, \ y \to Y = y$$

with $\xi = -\frac{u_{D,2}}{3}$. It transforms the curve H_d to a curve $\tilde{H_d} : Y^2 = X^7 - X + (\xi^7 - \xi + d)$.

Let $P_j = (\alpha_j, \beta_j)$, $\tilde{P_j} = (\alpha_j - \xi, \beta_j)$, $\tilde{\alpha_j} := \alpha_j - \xi$ and $\tilde{\beta_j} := \beta_j$.

From the fact that $\beta_j = V_D(\alpha_j)$ and $\tilde{\beta_j} = V_{\tilde{D}}(\tilde{\alpha_j}) = V_{\tilde{D}}(\alpha_j - \xi)$, it follows that

$$\begin{aligned} U_{\tilde{D}} &= x^3 + (3\xi^2 + 2\xi u_{D,2} + u_{D,1})x + (u_{D,2}\xi^2 + u_{D,1}\xi + u_{D,0} + \xi^3) \\ &= x^3 + \tilde{u}_1 x + \tilde{u}_0, \\ V_{\tilde{D}} &= v_{D,2}X^2 + (2\xi v_{D,2} + v_{D,1})X + (v_{D,2}\xi^2 + v_{D,1}\xi + v_{D,0}) \\ &= \tilde{v}_2 X^2 + \tilde{v}_1 X + \tilde{v}_0. \end{aligned}$$

We also have

$$\tilde{D} = (\tilde{P}_1) + (\tilde{P}_2) + (\tilde{P}_3) - 3(\mathcal{O}) = [X^3 + \tilde{u}_1 X + \tilde{u}_0,\ \tilde{v}_2 X^2 + \tilde{v}_1 X + \tilde{v}_0].$$

Furthermore, $(\tilde{h_j}) = 7(\tilde{P_j}) + (\tilde{P_j}') - 8(\mathcal{O})$, where $\tilde{h_j} = h_j(X + \xi, Y)$. Letting $\theta = \xi^7 - \xi + d$, it is easy to see the following:

$$\tilde{h_j} = \tilde{\beta_j}^7 Y - (\tilde{\alpha_j}^7 - X + \theta)^4 = (\tilde{v}_2\tilde{\alpha_j}^2 + \tilde{v}_1\tilde{\alpha_j} + \tilde{v}_0)^7 Y - (\tilde{\alpha_j}^7 - X + \theta)^4.$$

Thus we obtain

$$\begin{aligned} h_{\tilde{D}}(X,Y) &= \tilde{h_1}(X,Y)\tilde{h_2}(X,Y)\tilde{h_3}(X,Y) \\ &= \prod_{j=1}^{3} \left((\tilde{v}_2\tilde{\alpha_j}^2 + \tilde{v}_1\tilde{\alpha_j} + \tilde{v}_0)^7 Y - (\tilde{\alpha_j}^7 - X + \theta)^4 \right). \end{aligned} \tag{12}$$

If we apply the Elimination method in [7] to Eq. (12) with elimination order $\{\tilde{\alpha_1}, \tilde{\alpha_2}, \tilde{\alpha_3}\} > \{\tilde{u}_1, \tilde{u}_0\}$, then we can obtain Eq. (12) as a function of $\tilde{u}_1$ and $\tilde{u}_0$. The coefficients for $\hat{h}_{\tilde{D}}(\hat{X}, Y) = h_{\tilde{D}}(X - \theta, Y)$ are described in Table 1, where the second column shows the corresponding coefficient in terms of $\tilde{v_i}'$s and $\tilde{u_i}'$s. □

3.2 Evaluation at a divisor

In this subsection we show that the notion of *Resultant* can be used for evaluating a rational function at a divisor, which is necessary to achieve our goal.

What follows is the crucial result for the resultant, and for the proof we refer to [29, Ch. VI].

Theorem 3.3. *Let F be a field, $A, B \in F[x]$, $\alpha_1, \alpha_2, \cdots, \alpha_m \in \bar{F}$ (= algebraic closure of F) be all the roots of A, deg $A = m$, deg $B = n$, and a be the leading coefficient of A. Then we have*

$$res(A, B) = a^n \prod_{i=1}^{m} B(\alpha_i).$$

With the same notations as in Theorem 3.3, furthermore, we have

$$res(A, B) = (-1)^{mn} res(B, A). \tag{13}$$

Table 1: h_D formula for $7D$

Input	$D = [U_D, V_D] \in J_{H_d}(k)$	Cost
Output	$\hat{h}_{\tilde{D}}(\hat{X}, Y) = \delta_1 Y^3 + s(\hat{X})Y^2 + t(\hat{X})Y + \delta_{16}(\hat{X})$ $s(\hat{X}) = \delta_2 \hat{X}^4 + \delta_3 \hat{X}^3 + \delta_4 \hat{X}^2 + \delta_5 \hat{X} + \delta_6$ $t(\hat{X}) = \delta_7 \hat{X}^8 + \delta_8 \hat{X}^7 + \delta_9 \hat{X}^6 + \delta_{10} \hat{X}^5 + \delta_{11} \hat{X}^4 + \delta_{12} \hat{X}^3 + \delta_{13} \hat{X}^2 + \delta_{14} \hat{X} + \delta_{15}$	
δ_1	$(\tilde{v}_2\tilde{u}_0(\tilde{v}_2(\tilde{v}_2\tilde{u}_0) + 3\tilde{v}_1(\tilde{v}_0 + 2\tilde{v}_2\tilde{u}_1)) + \tilde{v}_1^2(\tilde{v}_0\tilde{u}_1 - \tilde{v}_1\tilde{u}_0) + \tilde{v}_0(\tilde{v}_2\tilde{u}_1 - \tilde{v}_0)^2)^7$	$8m + 2s$
δ_2	$(4(2\tilde{v}_2\tilde{u}_1 + \tilde{v}_0)(-\tilde{v}_2\tilde{u}_1 + \tilde{v}_0) - \tilde{v}_1(3\tilde{v}_2\tilde{u}_0 + \tilde{v}_1\tilde{u}_1))^7$	$3m$
δ_3	$(-2\tilde{v}_2\tilde{u}_0(2\tilde{v}_2\tilde{u}_1 + \tilde{v}_0) + \tilde{v}_1(2\tilde{v}_2\tilde{u}_0 + \tilde{v}_0\tilde{u}_1))^7$	$2m$
δ_4	$(3\tilde{v}_2^2\tilde{u}_0^2 + \tilde{v}_1\tilde{u}_0(-2\tilde{v}_2\tilde{u}_1 + 3\tilde{v}_0) + \tilde{v}_0\tilde{u}_1(2\tilde{v}_2\tilde{u}_1 - 2\tilde{v}_0))^7$	$2m + 1s$
δ_5	$(\tilde{v}_2\tilde{v}_0(2\tilde{v}_1\tilde{v}_0 - 3\tilde{u}_1\tilde{u}_0) - \tilde{v}_1\tilde{u}_1(\tilde{v}_0\tilde{u}_1 - \tilde{v}_1\tilde{u}_0) + 2\tilde{v}_0^2\tilde{u}_0)^7$	$3m$
δ_6	$(2\tilde{u}_0(\tilde{v}_2\tilde{u}_0)(\tilde{v}_2\tilde{u}_1 - 2\tilde{v}_0) - (\tilde{v}_0\tilde{u}_1 + \tilde{v}_1\tilde{u}_0)(4\tilde{v}_1\tilde{u}_0 + 2\tilde{u}_1(\tilde{v}_0 - \tilde{v}_2\tilde{u}_1)))^7$	$4m$
δ_7	$(-2\tilde{v}_2\tilde{u}_1 + 3\tilde{v}_0)^7$	$0m$
δ_8	$(2\tilde{v}_2\tilde{u}_0 - \tilde{v}_1\tilde{u}_1)^7$	$0m$
δ_9	$(2\tilde{u}_1(\tilde{v}_0 - \tilde{v}_2\tilde{u}_1) + 2(2\tilde{v}_0\tilde{u}_1 - \tilde{v}_1\tilde{u}_0))^7$	$0m$
δ_{10}	$(\tilde{u}_1(2\tilde{v}_2\tilde{u}_0 + \tilde{v}_1\tilde{u}_1) + \tilde{v}_0\tilde{u}_0)^7$	$1m$
δ_{11}	$(\tilde{u}_1^2(-2\tilde{v}_2\tilde{u}_1 + \tilde{v}_0) + \tilde{u}_0(\tilde{v}_2\tilde{u}_0 + 2\tilde{v}_1\tilde{u}_1))^7$	$2m + 1s$
δ_{12}	$(\tilde{u}_0(3\tilde{u}_1(2\tilde{v}_2\tilde{u}_1 + \tilde{v}_0) - \tilde{v}_1\tilde{u}_0))^7$	$1m$
δ_{13}	$(\tilde{u}_0^2(-\tilde{v}_2\tilde{u}_1 + 2\tilde{v}_0) + 3\tilde{u}_1^2(\tilde{v}_1\tilde{u}_0 - \tilde{v}_0\tilde{u}_1))^7$	$2m + 1s$
δ_{14}	$(2\tilde{u}_0^2(\tilde{v}_2\tilde{u}_0 + 2\tilde{v}_1\tilde{u}_1) + 3(\tilde{u}_0\tilde{v}_0)\tilde{u}_1^2)^7$	$2m$
δ_{15}	$(\tilde{u}_1(\tilde{u}_0^2(-\tilde{v}_2\tilde{u}_1 + 2\tilde{v}_0) + 3\tilde{u}_1^2(\tilde{v}_1\tilde{u}_0 - \tilde{v}_0\tilde{u}_1)) + \tilde{u}_0^2(2\tilde{u}_1(\tilde{v}_2\tilde{u}_1 - \tilde{v}_0) + 4(\tilde{v}_0\tilde{u}_1 + \tilde{v}_1\tilde{u}_0))^7$	$2m$
Total cost		$32m + 5s$
	Notation: m denotes a multiplication in $\mathbb{F}_{7^n}$, and s a squaring in $\mathbb{F}_{7^n}$.	

In addition, efficient reduction method for computing the resultant is also introduced in [29, Ch. VI]. When $m \geq n$, by Euclidean division algorithm, there exists $Q(x), R(x) \in F(x)$ such that $A(x) = Q(x)B(x) + R(x)$ with deg $R < n$. Then

$$res(A, B) = (-1)^{mn} res(B, R). \tag{14}$$

Now we are ready to apply the resultant to our Tate pairing computation.

Lemma 3.4. *For h_D given in Table 1 and $E = [U_E, V_E]$, we let $H_{D,E}(x) = h_D(x, V_E(x))$. Then we have*

$$h_D(E) = res(U_E, H_{D,E}).$$

Proof. In fact, $h_D(E) = H_{D,E}(x_1)H_{D,E}(x_2)H_{D,E}(x_3)$ with x_i's the roots of $U_E(x)$. The assertion thus follows immediately from Theorem 3.3. □

Table 2 indicates the nonzero coefficients of $H_D(x)$, and the complexity is counted by using *Karastuba's technique* [16].

Now by using the reduction method in Eq. (14), we can compute $res(U_E, H_{D,E})$ as follows:

Lemma 3.5. *Let $H_{D,E}(x) = Q(x)U_E(x) + R(x)$ with deg $R \leq 2$. Then we have*

$$h_D(E) = res(U_E, R).$$

Proof. We observe that the degree of $H_{D,E}$ is 12 and the degree of U_E is 3. Thus by using Eq. (13), we have $res(U_E, H_{D,E}) = res(H_{D,E}, U_E)$. From Eq. (14) it follows that $res(H_{D,E}, U_E) = res(U_E, R)$, so we get $res(U_E, H_{D,E}) = res(U_E, R)$. □

Table 2: $H_{D,E}$ formula complexity counting

i	ith coefficient of $H_{\tilde{D}}$	Cost
12	-1	0
10	$\delta_7 v_{E,2} + 3u_1^7$	$1m$
9	$\delta_7 v_{E,1} + \delta_8 v_{E,2} + 3u_0^7$	$2\tilde{m}$
8	$\delta_2 v_{E,2}^2 + \delta_7 v_{E,0} + \delta_8 v_{E,1} + \delta_9 v_{E,2} + u_1^{14}$	$1m + 1s + 2\tilde{m}$
7	$2\delta_2 v_{E,1} v_{E,2} + \delta_3 v_{E,2}^2 + \delta_8 v_{E,0} + \delta_9 v_{E,1} + \delta_{10} v_{E,2} + 2u_0^7 u_1^7$	$2m + 3\tilde{m}$
6	$\delta_1 v_{E,2}^3 + 2\delta_2 v_{E,0} v_{E,2} + \delta_2 v_{E,1}^2 + 2\delta_3 v_{E,1} v_{E,2} + \delta_4 v_{E,2}^2 + \delta_9 v_{E,0}$ $+\delta_{10} v_{E,1} + \delta_{11} v_{E,2} + u_0^{14} + 3u_1^{21}$	$3m + 4\tilde{m} + 1s$
5	$3\delta_1 v_{E,1} v_{E,2}^2 + 2\delta_2 v_{E,0} v_{E,1} + 2\delta_3 v_{E,0} v_{E,2} + \delta_3 v_{E,1}^2 + 2\delta_4 v_{E,1} v_{E,2}$ $+\delta_5 v_{E,2}^2 + \delta_{10} v_{E,0} + \delta_{11} v_{E,1} + \delta_{12} v_{E,2} + 2u_0^7 u_1^{14}$	$7\tilde{m} + 1m$
4	$3\delta_1 v_{E,0} v_{E,2}^2 + 3\delta_1 v_{E,1}^2 v_{E,2} + \delta_2 v_{E,0}^2 + 2\delta_3 v_{E,0} v_{E,1} + 2\delta_4 v_{E,0} v_{E,2} + \delta_4 v_{E,1}^2$ $+2\delta_5 v_{E,1} v_{E,2} + \delta_6 v_{E,2}^2 + \delta_{11} v_{E,0} + \delta_{12} v_{E,1} + \delta_{13} v_{E,2} + 2u_0^{14} u_1^7 - u_1^{28}$	$5\tilde{m} + 2m$
3	$-\delta_1 v_{E,0} v_{E,1} v_{E,2} + \delta_1 v_{E,1}^3 + \delta_3 v_{E,0}^2 + 2\delta_4 v_{E,0} v_{E,1} + 2\delta_5 v_{E,0} v_{E,2} + \delta_5 v_{E,1}^2$ $+2\delta_6 v_{E,1} v_{E,2} + \delta_{12} v_{E,0} + \delta_{13} v_{E,1} + \delta_{14} v_{E,2} + 3u_0^{21} + 3u_0^7 u_1^{21}$	$1m' + 5\tilde{m} + 1m$
2	$(3\delta_1 v_{E,0}^2 v_{E,2} + 3\delta_1 v_{E,0} v_{E,1}^2 + \delta_4 v_{E,0}^2 + 2\delta_5 v_{E,0} v_{E,1} + 2\delta_6 v_{E,0} v_{E,2} + \delta_6 v_{E,1}^2$ $+\delta_{13} v_{E,0} + \delta_{14} v_{E,1} + \delta_{15} v_{E,2} + u_0^{14} u_1^{14}$	$1m' + 2\tilde{m} + 1m + 1s$
1	$3\delta_1 v_{E,0}^2 v_{E,1} + \delta_5 v_{E,0}^2 + 2\delta_6 v_{E,0} v_{E,1} + \delta_{14} v_{E,0} + \delta_{15} v_{E,1} + 3u_0^{21} u_1^7$	$1m + 1m' + 1\tilde{m}$
0	$\delta_1 v_{E,0}^3 + \delta_6 v_{E,0}^2 + \delta_{15} v_{E,0} - u_0^{28}$	$1m' + 1s$
	Total cost	$4s + 13m + 31\tilde{m} + 4m'$

3.3 Algorithm for the Tate pairing computation by using Resultant

In this subsection, we describe an algorithm for computing the Tate pairing on divisors, and we also compute its complexity. ¿From Lemma 3.1, Proposition 3.2, and Lemma 3.4, the Tate pairing given in Eq. (6) can be computed by using Algorithm 2.

Since $v_{\hat{E},j} = \sigma \cdot (\text{some element in } \mathbb{F}_{7^{7n}}),\ j = 0, 1, 2,$ we note that $H_{\tilde{D},\hat{E}}$ in the step 7 of Algorithm 2 can be written as

$$H_{\tilde{D},\hat{E}} = -x^{12} + \sum_{i=0}^{10} (d_i\sigma + e_i)x^i, \ \ d_i, e_i \in \mathbb{F}_{7^{7n}} \text{ for } 0 \le i \le 10.$$

To find $H_{\tilde{D},\hat{E}} \pmod{U_{\hat{E}}}$ in the step 7, we use the following recursive relations:

$$\begin{aligned} x^i &\equiv a_i x^2 + b_i x + c_i, \quad 3 \le i \le 12 \\ a_3 &= -u_{\hat{E},2},\ b_3 = -u_{\hat{E},1},\ c_3 = -u_{\hat{E},0} \in \mathbb{F}_{7^{7n}} \\ a_i &= a_{i-1}a_3 + b_{i-1},\ b_i = a_{i-1}b_3 + c_{i-1},\ c_i = a_{i-1}c_3. \end{aligned}$$

Then R can be computed by

$$\begin{aligned} R &= H_{\tilde{D},\hat{E}} \pmod{U_{\hat{E}}} \\ &= (a_{12} + \sum_{i=3}^{10} a_i(d_i\sigma + e_i) + d_2\sigma + e_2)x^2 \\ &\quad + (b_{12} + \sum_{i=3}^{10} b_i(d_i\sigma + e_i) + d_1\sigma + e_1)x \\ &\quad + (c_{12} + \sum_{i=3}^{10} c_i(d_i\sigma + e_i) + d_0\sigma + e_0). \end{aligned} \tag{15}$$

Now we discuss the complexity of Algorithm 2 by counting the number of operations which are necessary for computing $\eta(D, E)$. As before, we denote the time for multiplications in $\mathbb{F}_{7^{14n}}, \mathbb{F}_{7^{7n}}$ and $\mathbb{F}_{7^n}$ by M, m' and m, respectively, and a multiplication between $\mathbb{F}_{7^n}$ and $\mathbb{F}_{7^{7n}}$ by $\tilde{m}$.

Algorithm 2 Tate pairing computation by using resultant

```
INPUT $D = [U_D,\ V_D],\ E = [U_E,\ V_E] \in J_{H_d}(\mathbb{F}_{7^n})$, endomorphism $\psi$
OUTPUT $\hat{t}(D,\ E)$
1: Set $\xi \leftarrow 2u_{D,2}$ and $\theta \leftarrow \xi^7 - \xi + d$.
2: Compute $u_1 = 3\xi^2 + 2\xi u_{D,2} + u_{D,1}$ and $u_0 = \xi^3 + u_{D,2}\xi^2 + u_{D,1}\xi + u_{D,0}$.
3: Compute $\delta_j$ for $j = 1, ..., 15$ using Table 1
4: $g \leftarrow 1$,
5: for $i = 0$ to $n - 1$ do
6:     compute $\hat{E} = \tau_{\xi+\theta}(\psi(E))$
7:     compute $H_{\tilde{D},\hat{E}} = \hat{h}_{\tilde{D}}(x, V_{\hat{E}})$ using $u_0,\ u_1$ and $\delta_j$'s (Table 2), and
       compute $R = H_{\tilde{D},\hat{E}} \pmod{U_{\hat{E}}}$.
8:     compute $h_{D_i}(\psi(E)) = res(U_{\hat{E}}, R)$.
9:     $g \leftarrow g^7 \cdot h_{D_i}(\psi(E))$
10:    set $u_0 \leftarrow u_0^{7^2},\ u_1 \leftarrow u_1^{7^2}$
11:    set $\xi \leftarrow \xi^{7^2},\ \theta \leftarrow \theta^{7^2},\ \delta_j \leftarrow \delta_j^{7^2}$ if $j = 2, 3, 4, 5, 6$, and $\delta_j \leftarrow (-1)^i \delta_j^{7^2}$ otherwise.
12: Return $g^{7^{6n+1}(7^{7n}-1)},\ g = \eta(D, E)$.
```

Noting that, in Step 6, $u_{\hat{E},j}, j = 0, 1, 2$ and $v_{\hat{E},0}, v_{\hat{E},1}$ belong to $\mathbb{F}_{7^{7n}}$, and from Eq. (5) we have $v_{\hat{E},2} = \sigma \cdot$ (some element in $\mathbb{F}_{7^n}$). The computation cost of $H_{\tilde{D},\hat{E}} = \hat{h}_{\tilde{D}}(x, V_{\hat{E}})$ in Step 7 is counted in Table 2. For computing R, given in Eq. (15), we need $24m'$ for computing $x^n \pmod{U_E}$ and $60m'$ for computing R.

The total complexity of this algorithm is therefore

$$40m + n(22m + 31\tilde{m} + 88m' + T_{res} + 1M), \tag{16}$$

where T_{res} is the computation cost for the resultant $res(U_{\hat{E}}, R)$ of $U_{\hat{E}}$ and R in $\mathbb{F}_{7^{14n}}$. In detail, we need $3m$ in the step 2, and $37m$ in the step 3 from Table 1. For each loop, we need $5m$ in the step 6, $17m + 31\tilde{m} + 4m'$ in the step 7 from Table 2, and $1M$ in the step 9.

For an asymptotically efficient computation of the resultant of a pair of polynomials, we can use the reduction method in Eq. (14) and Lemma 3.5 repeatedly. Schwartz [25] presented a very efficient algorithm for calculating the resultant by adapting the *fast polynomial GCD algorithm* by Moenck, and this is $O(N \log^2 N)$ algorithm for the resultant calculation, where N is the sum of degrees of two polynomials.

However we calculate the resultant by computing the determinant of two polynomials with degree 2 and 3 in Mumford representation. Then we have $T_{res} = 17M + 6S$, where M (resp. S) is a multiplication (resp. squaring) in $\mathbb{F}_{7^{14n}}$. Explicitly, let $U_{\hat{E}}(x) = x^3 + u_2x^2 + u_1x + u_0$ and $R(x) = r_2x^2 + r_1x + r_0$, then $res(U_{\hat{E}}, R)$ is as follows:

$$r_2^2(u_0(-2r_0u_2-r_1u_1)+r_2u_0^2+r_0u_1^2)+r_0^2(r_2(u_2^2-2u_1)-r_1u_2+r_0)+r_1^2(u_0(r_2u_2-r_1)-u_1r_0)+r_0r_1r_2(3u_0-u_1u_2).$$

4 Complexity comparison

In this section we compare the complexity of our two methods given in Section 2 and 3.

When an extension degree is of the form $k = 2^i 3^j$, the computation cost for a multiplication in $\mathbb{F}_{q^k}$ is theoretically $3^i 5^j$ times of the cost for a multiplication $\mathbb{F}_q$ ([17], [19]). From this observation, we assume that

$$1 \text{ mult. in } \mathbb{F}_{7^{3n}}(m_3) \approx 5m, \;\; 1 \text{ mult. in } \mathbb{F}_{7^{3(14n)}}(M_3) \approx 5M, \;\; 1 \text{ mult. in } \mathbb{F}_{7^{14n}}(M) \approx 3m' \tag{17}$$

and we also let $\tilde{m} \approx 7m$.

With the above assumptions, the point-wise computation cost in Eq. (11) is

$$\begin{aligned} T_P &:= 2 \cdot T_{3rt} + 27n \cdot (20m + 15m') + 270m' + 120m' \\ &= 2 \cdot T_{3rt} + (540n)m + (405n + 390)m', \end{aligned}$$

where T_{3rt} is the time for finding all the roots of a cubic polynomial over $\mathbb{F}_{7^{3n}}$. By *Berlekamp-Rabin algorithm* [3], we have $T_{3rt} = O(3^2 \log 3 \log 7^{3n}) \cdot 5m \approx 27n \cdot 4 \cdot 5m$.

Counting the cost for T_{3rt}, we finally have

$$T_P \approx (1620n)m + (405n + 390)m'. \tag{18}$$

On the other hand, the total time for the Resultant method in Eq. (16) is

$$T_R = 40m + n(239m + 91m' + T_{res}),$$

where T_{res} is the time for computing the resultant of two polynomials over $\mathbb{F}_{7^{14n}}$.

As shown in Section 3.3, we have $T_{res} = 17M + 6S$, where M (resp. S) is a multiplication (resp. squaring) in $\mathbb{F}_{7^{14n}}$.

Thus, the computation cost of our Resultant approach is approximately

$$\begin{aligned} T_R &= 40m + n(239m + 91m' + 17M + 6S) \\ &\approx 40m + n(239m + 91m' + 23M) \\ &= (239n + 40)m + (160n)m'. \end{aligned} \tag{19}$$

To compare the complexity of two methods, we summarize T_P, T_R and the ratio T_P/T_R in Table 3, where the examples are chosen for cryptographical meaningful values [19].

Table 3: Complexity comparison: examples

Security (bits)	80	128	192
bitlength of 7^{14n}	1140	3072	8192
n	29	79	211
Point-wise (T_P)	$46980m + 12135m'$	$127980m + 32385m'$	$341820m + 85845m'$
Resultant (T_R)	$6971m + 4640m'$	$18921m + 12640m'$	$50469m + 33760m'$
$\frac{T_P}{T_R}$	$2.74 \leq \frac{T_P}{T_R} \leq 3.34$	$2.69 \leq \frac{T_P}{T_R} \leq 3.30$	$2.67 \leq \frac{T_P}{T_R} \leq 3.29$

According to [17], the ratio of m' and m is $7 \leq \frac{m'}{m} \leq 49$, and the last row of Table 3 shows the range of the ratio of $\frac{T_P}{T_R}$ for each fixed value of n. Therefore, we can conclude that the Tate pairing computation by using Resultant is approximately three times faster than the point-wise computation.

5 Experimental results

We have proposed two methods by Resultant and Point-wise approach for computing the Tate pairing over the genus 3 hyperelliptic curve $H_d : y^2 = x^7 - x + d, d = \pm 1$ over $\mathbb{F}_q$. In Section 4, we compared the complexity of two proposed methods, and it turned out that the Resultant approach is approximately three times faster than the Point-wise approach. In this section, we provide experimental results for the Tate pairing computation over the genus 3 hyperelliptic curve and compare the running time. Basically our goal in this experimentation is that we verify our theoretical complexity comparison in Section 4 by actual implementation using one of the standard packages such as NTL. We make implementations when both input divisors D and E are general divisors, and we use the NTL software package. We provide implementation for genus 3 hyperelliptic curves for the first time.

We first need to find a prime n for each security level s such that $2^s \approx 7^{3n}$, and also find a large prime ℓ dividing $|J_{H_d}(\mathbb{F}_{7^n})|$ such that $\ell \approx 2^s$. The formula for $|J_{H_d}(\mathbb{F}_{7^n})|$ is given in Lemma 1.2. By searching for good candidates for ℓ and n from $n = 29$ through $n = 79$, we find the following:

When $n = 29$, for $H_{(-1)}$ curve,
$\ell =$ 295427580543981044508742175251656510425218717654351011099430750210650097.

When $n = 43$, for $H_{(-1)}$ curve,
$\ell =$ 537186185691863880188217039863742753517055763668500175524814523901957588878744075332862878883563864467.

When $n = 47$, for $H_{(-1)}$ curve,
$\ell =$ 13749772461004425111203179773331321128112017469863375270022695103034065149004498912831678964830780873139729982133.

When $n = 73$, for $H_{(+1)}$ curve,
$\ell =$ 1055339806451465619904681860606549517661466267122231937236741631980131588994036218419755332318469900781285578602047978955194093497651290723475309620425880333576516676980042149532583647.

Table 4 shows the amounts of time to perform the field multiplications in $\mathbb{F}_{7^n}$, $\mathbb{F}_{7^{3n}}$ and $\mathbb{F}_{7^{7n}}$ using NTL. The table was computed by taking average time of 5000 multiplications of random elements in each field.

Table 4: Multiplication timings (in milliseconds)

n	29	43	47	73
$\mathbb{F}_{7^n}$ (m)	0.2	0.4468	0.5312	1.2688
$\mathbb{F}_{7^{3n}}(m_3)$	1.8468	3.3156	3.8376	8.5688
$\mathbb{F}_{7^{7n}}$ (m')	5.9938	12.9062	13.2562	21.1594
m_3/m	9.234	7.42077	7.2244	6.75347
m'/m	29.969	28.8859	24.9552	16.6767

In Section 2, the complexity for the Tate pairing computation by point-wise method is given by

$$T_P := 2\ T_{3rt} + n\ (108m_3 + 27M_3) + 26M_3. \tag{20}$$

Assuming that the field operation m_3 in $\mathbb{F}_{7^{3n}}$ and the field operation m in $\mathbb{F}_{7^n}$ has the ratio $m_3/m = 5$, we obtain Table 3. However, in NTL the actual ratio is approximately 7 or 9 as shown in Table 5. According to the actual ratio of the field operations in NTL, Table 3 is adjusted to Table 5.

Table 5: Complexity comparison: examples in NTL

bitlength of 7^{14n}	1140	1690	1847	2869
n	29	43	47	73
Point-wise (T_P)	$59508m + 21843m'$	$78948m + 24927m'$	$86292m + 27195m'$	$134028m + 41937m'$
Resultant (T_R)	$6971m + 4640m'$	$10317m + 6880m'$	$11273m + 7520m'$	$17487m + 11680m'$
$\frac{T_P}{T_R}$	4.89	3.82	3.85	3.92

Table 6 shows the experiment results of the Tate pairing for selected examples. We performed fifty calculations with random samples for each method and took the average time. The experiments ran on a machine with 2.8Ghz Opteron and 4GB of RAM, and we used Microsoft Visual C++ 6.0 with speed optimizations on.

Table 6: Experiment results (in seconds)

bit-length of ℓ	237	338	373	608
bit-length of 7^{14n}	1140	1690	1847	2869
n	29	43	47	73
Point-wise method(T_P)	125.639	357.626	453.749	1638.18
Resultant method(T_R)	14.8825	45.7058	56.0878	173.698
$\frac{T_P}{T_R}$	8.44209	7.82452	8.08998	9.43119

There is room for further optimization and using NTL might not be the best choice. However, our implementation is sufficient enough to conclude that the Resultant method is much faster than the point-wise method.

References

[1] P. S. L. M. Barreto, H. Y. Kim, B. Lynn, M. Scott, Efficient Algorithms for Pairing-Based Cryptosystems. Advances in Cryptology – Crypto 2002, Lecture Notes in Computer Science, Vol. 2442, Springer-Verlag, (2002) 354–368.

[2] P. S. L. M. Barreto, S. Galbraith, C. hEigeartaigh and M. Scott, Efficient Pairing Computation on Supersingular Abelian Varieties, Cryptology eprint Atchives, Available at http://eprint.iacr.org, 2004, No. 2004/375.

[3] E.R Berlekamp, Algebraic Coding Theory, McGraw-Hill, New York, (1968).

[4] D. Boneh and M. Franklin, Identity-based encryption from the Weil pairing. Advances in Cryptology, Crypto 2001, Lecture Notes in Computer Science, Vol. 2139, Springer-Verlag, (2001) 213-229.

[5] J. C. Cha, J. H. Cheon, An Identity-Based Signature from Gap Diffie-Hellman Groups. Proceedings of PKC, Lecture Notes in Computer Science, Vol. 2567, (2003) 18-30.

[6] Y. Choie and E. Lee, Implementation of Tate pairing on hyperelliptic curves of genus 2, Information security and cryptology—ICISC 2003, 97–111, Lecture Notes in Comput. Sci., 2971, Springer, Berlin, (2004).

[7] D. Cox, J. Little and D. O'Shea, Ideals, varieties, and algorithms : an introduction to computational algebraic geometry and commutative algebra : with 91 illustrations, New York : Springer, (1997).

[8] I. Duursma, Class numbers for hyperelliptic curves. In: Arithmetic, Geometry and Coding Theory. eds. Pellikaan, Perret, Vladuts, pp. 45-52, publ. deGruyter, Berlin, (1996).

[9] I. Duursma and H. Lee, Tate pairing implementation for hyperelliptic curves $y^2 = x^p - x + d$, Advances in cryptology-AsiaCrypt 2003, LNCS 2894, (2003) 111–123.

[10] G. Frey, H.-G. Rück, A remark concerning m-divisibility and the discrete logarithm in the divisor class group of curves. Math. Comp. 62, no. 206, (1994) 865–874.

[11] S.D. Galbraith, Supersingular curves in cryptography. Asiacrypt 2001, Springer, Lecture Notes in Computer Science, Vol. 2248, (2001) 495–513.

[12] S. D. Galbraith, K. Harrison, D. Soldera, Implementing the Tate pairing. Algorithmic Number Theory Symposium, ANTS-V, Lecture Notes in Computer Science, Vol. 2369, Springer-Verlag, (2002) 324–337.

[13] G.-M. Greuel, G. Pfister, and H. Schönemann. Singular 3.0. A Computer Algebra System for PolynomialComputations. Centre for Computer Algebra, University ofKaiserslautern (2005). `http://www.singular.uni-kl.de`.

[14] F. Hess, Exponent group signature schemes and efficient identity based signature schems based on pairing, Proceedings of the Workshop Selected Areas in Cryptology, SAC, Aug. (2002).

[15] A. Joux, A one round protocol for tripartite Diffie-Hellman. Proceedings of Algorithmic Number Theory Symposium, Lecture Notes in Computer Science, Vol. 1838, Springer-Verlag, (2000) 385-394.

[16] A. Karatsuba and Y. Ofman, Multiplication of Multidigit Numbers on Automata. Sov. Phys.-Dokl. (Engl. transl.), **7**, No. 7, (1963) 595-596.

[17] Knuth, The Art of Computer Programming, Vol. II, Addison Wesley, (2004).

[18] N. Koblitz, Algebraic Aspects of Cryptography, Springer-Verlag, (1998).

[19] N. Koblitz and A. Menezes, Pairing-based cryptography at high security levels. Proceedings of the Tenth IMA International Conference on Cryptography and Coding, Lecture Notes in Computer Science, 3796, (2005) 3-36.

[20] T. Lange, Formulae for arithmetic on genus 2 hyperelliptic curves. Appl. Algebra Engrg. Comm. Comput. 15, no. 5, (2005) 295–328.

[21] E. Lee and Y. Lee, Tate pairing computation on the divisors of hyperelliptic curves for cryptosystems. Cryptology eprint Atchives, Available at http://eprint.iacr.org, 2005, No. 2005/166

[22] D. Mumford, Tata Lectures on Theta II, Birkhauser, (1984).

[23] K.G. Paterson, ID-based signature from pairings on elliptic curves, Electronics Letters. Vol. 38 (18), (2002) 1025-1026.

[24] J. Pelzl, T. Wollinger and C. Paar, Low cost security: explicit formulae for genus-4 hyperelliptic curves. Selected areas in cryptography, Lecture Notes in Comput. Sci., 3006, Springer, Berlin, (2004) 1–16.

[25] J.T. Schwartz, Fast probabilistic algorithms for verification of polynomial identities, J. Assoc. Comput. Mach. 27, no. 4, (1980) 701-717.

[26] V. Shoup, A library for doing number theory, Software, 2001; see http://www.shoup.net/ntl/.

[27] N. Smart, On the Performance of Hyperelliptic Cryptosystems, Advances in Cryptology: Proceedings of Eurocrypt'99, LNCS 1592, Springer-Verlag, (1999) 165-175.

[28] N.P. Smart, An identity based authentication key agreement protocol based on pairing, Electronics Letters, Vol 38, pp 630-632, (2002).

[29] C. K. Yap, Fundamental Problems in Algorithmic Algebra, Oxford University Press, (2000).