Efficient Tate Pairing Computation for Supersingular Elliptic Curves over Binary Fields

Soonhak Kwon
Department of Mathematics, Sungkyunkwan University, Korea
shkwon@skku.edu

Abstract: After Miller's original algorithm for the Tate pairing computation, many improved algorithms have been suggested, to name just a few, by Galbraith et al. and Berrato et al., especially for the fields with characteristic *three*. Also Duursma and Lee found a closed formula of the Tate pairing computation for the fields with characteristic *three*. In this paper, we show that a similar argument is also possible for the finite fields with characteristic *two*. That is, we present a closed formula for the Tate pairing computation for supersingular elliptic curves defined over the binary field $\mathbb{F}_{2^m}$ of odd dimension. There are exactly three isomorphism classes of supersingular elliptic curves over $\mathbb{F}_{2^m}$ for odd m and our result is applicable to all these curves. Moreover we show that our algorithm and also the Duursma-Lee algorithm can be modified to another algorithm which does not need any inverse Frobenius operation (square root or cube root extractions) without sacrificing any of the computational merits of the original algorithm. Since the computation of the inverse Frobenius map is not at all trivial in a polynomial basis and since a polynomial basis is still a preferred choice for the Tate pairing computation in many situations, this new algorithm avoiding the inverse Frobenius operation has some advantage over the existing algorithms.

Keywords: supersingular elliptic curve, Tate pairing, divisor, automorphism, roots of unity.

1. Introduction

With increasing use of the Tate pairing in cryptographic areas, a study of efficient computation of the Tate pairing becomes the subject of active research these days. Many cryptographic schemes are based on the bilinear pairings arising from the rank two abelian group structure of the points of prescribed order of the given elliptic curve. Examples of such cryptographic protocols are, to name just a few, identity based encryption scheme by Boneh and Franklin [10], short signature scheme by Boneh et al. [11], tripartite Diffie-Hellman key agreement protocol by Joux [12], identity based authenticated key agreement protocol by Smart [25], and identity based signature schemes by Hess [7], Sakai et al. [24]. In most of these applications, the Weil pairing or Tate pairing of supersingular elliptic curves (or curves of small embedding degrees) are essential tools. Therefore efficient computation of the Weil or Tate pairings is a crucial factor for practical applications of the above mentioned cryptographic protocols. The Weil pairing for a given elliptic curve is a symmetric bilinear pairing which can be thought of two applications of the Tate pairing. Thus the Weil pairing is more slow to compute than the Tate pairing, and consequently, it is desirable to replace the Weil pairing as the Tate pairing whenever it is possible in many cryptographic schemes.

Recently many progresses have been made on the computation of the Tate pairing. Galbraith et al. [4,5] suggested a few refined techniques and ideas to speed up the computation of the Tate pairing. Eisenträger et al. [13] introduced the notion of the squared Tate pairing. Barreto et al. [1] showed that the standard algorithm of Miller [19] can be modified to so called the BKLS algorithm where division in a finite field can be omitted since the denominator becomes one after final powering. Also Duursma and Lee [3] presented a closed formula for the computation of the Tate pairing for a finite field with characteristic *three*, which significantly reduces the cost of computing and is flexible for both of software and hardware applications.

In this paper we show that an efficient closed formula can also be obtained for the computation of the Tate pairing for supersingular elliptic curves over a binary field $\mathbb{F}_{2^m}$ with odd dimension m. There are exactly three isomorphism classes of supersingular elliptic curves over $\mathbb{F}_{2^m}$ with m odd [15] and our method is applicable to all these curves, of which two are the most commonly used curves with embedding degree 4. Also we present a method of avoiding inverse Frobenius operations in our and Duursma-Lee's algorithms. When one wants to use a polynomial basis, inverse Frobenius operation is not at all trivial unlike the case of a normal basis and this inverse operation deteriorates the performance of the algorithms of Duursma-Lee and ours, which need two inverse Frobenius operations in each step of the algorithms. We propose new modified algorithms which avoid the inverse Frobenius map without affecting the computational merits of the original algorithms.

2. Elliptic curves and Miller's algorithm

Let E be an elliptic curve over a finite field $\mathbb{F}_q$ where q is a power of a prime. We may express E as the following standard Weierstrass form

$$E : Y^2 + a_1XY + a_3Y = X^3 + a_2X^2 + a_4X + a_6,$$

where the coefficients a_1, a_2, a_3, a_4, a_6 are in $\mathbb{F}_q$. Let $E(\mathbb{F}_q)$ be the set of all points $P = (x, y)$, $x, y \in \mathbb{F}_q$, on the curve with the point at infinity O (which is $(0, 1, 0)$ on the corresponding homogeneous equation of degree 3 over a projective plane). $E(\mathbb{F}_q)$ has a structure of an abelian group and the order $|E(\mathbb{F}_q)|$ is bounded by the following well known relation due to Hasse [14],

$$|E(\mathbb{F}_q)| = q + 1 - Tr(\varphi), \quad |Tr(\varphi)| \leq 2\sqrt{q}, \tag{1}$$

where $Tr(\varphi) \in \mathbb{Z}$ is the trace of the Frobenius map $\varphi : E \longrightarrow E,$ with $\varphi(x, y) = (x^q, y^q)$, and φ is a zero of the characteristic polynomial $h(X) = X^2 - Tr(\varphi)X + q$. Let l be a positive integer and let $E[l]$ (resp. $E[l](\mathbb{F}_q)$) be the set of points $P \in E(\overline{\mathbb{F}}_q)$ (resp. $P \in E(\mathbb{F}_q)$) satisfying $lP = O$, where $\overline{\mathbb{F}}_q$ is an algebraic closure of $\mathbb{F}_q$. Let k be the minimal degree of the extension satisfying $E[l] \subset E(\mathbb{F}_{q^k})$. Such k is called the embedding degree (or the security multiplier) of $E[l]$ [1,15] and is dependent on E and l. If l is prime to q, then it is well known [14,15] that $E[l] \cong \mathbb{Z}/l \oplus \mathbb{Z}/l$.

A divisor D on E is a formal (finite) sum of the points P on the curve

$$D = \sum n_p(P), \quad n_p \in \mathbb{Z}. \tag{2}$$

We call D a zero divisor if $\sum n_p = 0$. A principal divisor is a divisor of the form $(f) = \sum n_p(P)$, where f is a rational function on E and P is a point of E with n_P the order of multiplicity of f at P, i.e. $n_P > 0$ if f has a zero at P and $n_P < 0$ if f has a pole at P. We say two divisors D and D' are equivalent if $D - D'$ is a principal divisor. It is well known [14,15] that a principal divisor (f) is a zero divisor, and a divisor $D = \sum n_p(P)$ is a principal divisor if

D is a zero divisor and $\sum n_p P = O$ in the abelian group $E(\overline{\mathbb{F}}_q)$. More precisely, there is an isomorphism [15]

$$Div_0/Div_{prin} \longrightarrow E, \quad \text{with} \quad D = \sum n_p(P) \longmapsto \sum n_p P, \tag{3}$$

where the summation in the right side is the addition of points on the elliptic curve E and Div_0 (resp. Div_{prin}) is a free abelian group generated by the zero divisors (resp. principal divisors). Now suppose that $P \in E[l]$. Then the divisor $l(P) - l(O)$ is a principal divisor so that there is a rational function f_P such that $(f_P) = l(P) - l(O)$. For any rational function f and any divisor $D = \sum n_p(P)$ having disjoint supports, one naturally define $f(D) = \prod f(P)^{n_p}$. The Tate pairing τ_l on the set $E[l]$ is defined as follows.

Definition 1. *Let $P \in E[l](\mathbb{F}_q)$ and $Q \in E[l](\mathbb{F}_{q^k})$. The Tate pairing is a map*

$$\tau_l : E[l](\mathbb{F}_q) \times E[l](\mathbb{F}_{q^k}) \longrightarrow \{\zeta_l\}, \quad \text{with} \quad \tau_l(P, Q) = f_P(D_Q)^{\frac{q^k-1}{l}},$$

where f_P is a rational function satisfying $(f_P) = l(P) - l(O)$ and D_Q is a zero divisor equivalent to $(Q) - (O)$ such that D_Q and (f_P) have disjoint supports. Also $\{\zeta_l\}$ is the group of l-th roots of unity in $\mathbb{F}_{q^k}^{\times}$.

It is well known that τ_l is a non-degenerate bilinear pairing. That is, for any $P \neq O \in \mathbb{E}[l](\mathbb{F}_q)$, there exists a point $Q \in E[l](\mathbb{F}_{q^k})$ such that $\tau_l(P, Q) \neq 1$. Also we have $\tau_l(P_1 + P_2, Q) = \tau_l(P_1, Q)\tau_l(P_2, Q)$ and $\tau_l(P, Q_1 + Q_2) = \tau_l(P, Q_1)\tau_l(P, Q_2)$. Non-degeneracy is not a trivial result and a proof can be found in [6,17]. It is also easy to verify $\tau_{ld}(P, Q) = \tau_l(P, Q)$ for $P, Q \in E[l]$ and $d > 0$ with ld dividing $|E(\mathbb{F}_q)|$.

An effective algorithm for finding a rational function f_P satisfying $(f_P) = l(P) - l(O)$ with $P \in E[l]$ is found by Miller [15,19]. Let us briefly explain the idea of Miller. For any zero divisor D and D', the isomorphism in (3) implies that there exist points P and P' such that

$$D = (P) - (O) + (f), \quad D' = (P') - (O) + (f'),$$

for some rational functions f and f'. Then one easily checks that

$$D + D' = (P + P') - (O) + (ff'\frac{\ell_{P,P'}}{\ell_{P+P'}}), \tag{4}$$

where $\ell_{P,P'}$ is an equation of a line intersecting P and P', and ℓ_P is an equation of a vertical line intersecting P and $-P$. This can be verified using the relation

$$\begin{aligned}(\frac{\ell_{P,P'}}{\ell_{P+P'}}) &= (\ell_{P,P'}) - (\ell_{P+P'}) \\ &= (P) + (P') + (-P - P') - 3(O) - \{(P + P') + (-P - P') - 2(O)\} \\ &= (P) + (P') - (P + P') - (O).\end{aligned} \tag{5}$$

Thus the right side of (4) is

$$\begin{aligned}(P + P') - (O) + (ff'\frac{\ell_{P,P'}}{\ell_{P+P'}}) &= (P + P') - (O) + (ff') + (P) + (P') - (P + P') - (O) \\ &= (P) + (P') - 2(O) + (ff') = D + D'.\end{aligned}$$

An elliptic curve E over $\mathbb{F}_q$ is called supersingular if $Tr(\varphi) \equiv 0 \pmod{p}$ where φ is the Frobenius map and p is the characteristic of $\mathbb{F}_q$. If an elliptic curve E over $\mathbb{F}_q$ is supersingular, then it is well known [15] that for any l dividing $|E(\mathbb{F}_q)|$, the embedding degree k is bounded by 6. More precisely, we have $E[l] \subset E(\mathbb{F}_{q^k})$ with $k = 2, 3, 4, 6$. It is also well known [15] that the embedding degree $k = 6$ is attained when the characteristic of $\mathbb{F}_q$ is *three* and the embedding degree $k = 4$ is attained when the characteristic of $\mathbb{F}_q$ is *two*.

3. BKLS algorithm and the algorithm of Duursma and Lee

Barreto, Kim, Lynn, and Scott [1] showed that, for some supersingular curves with embedding degree $k = 2, 4, 6$, one can speed up the computation of the Tate pairing by observing that the denominators ℓ_Q appearing in the Miller's algorithm can be omitted using the idea of the distortion map ϕ introduced in [18], where ϕ is a suitably chosen nontrivial automorphism of the given supersingular elliptic curve. That is, since the line $X-\alpha$ intersecting $Q = (\alpha, \beta) \in \mathbb{F}_q$ and $-Q$ has only X-coordinate and since this X-coordinate has the value in $\mathbb{F}_{q^{k/2}}$ after applying ϕ to Q, it becomes one after taking the final power by $\frac{q^k-1}{l}$ because $l|q^{k/2}+1$ and $q^k - 1 = (q^{k/2} - 1)(q^{k/2} + 1)$. Therefore omitting ℓ_Q does not affect the final pairing value and this greatly simplifies the Miller's algorithm since the costly operation of division is not necessary. By the similar reasoning, Barreto et al. [1] also showed that it is not necessary to evaluate the Tate pairing at O, the point at infinity, since the image of O is already in the field $\mathbb{F}_q$ before taking the final power by $\frac{q^k-1}{l}$. To summarize, the BKLS algorithm can be explained as follows.

Theorem 2. (Barreto et al. [1]) *Let E be a supersingular elliptic curve over $\mathbb{F}_q$ with embedding degree $k = 2, 4, 6$ and suppose that there is a suitable distortion map ϕ for E. Let l be a positive integer dividing $|E(\mathbb{F}_q)|$ with $gcd(l, q) = 1$ and let $\{\zeta_l\}$ be the group of l-th roots of unity in $\mathbb{F}_{q^k}^{\times}$. Then the modified Tate pairing*

$$\tau_l : E[l](\mathbb{F}_q) \times E[l](\mathbb{F}_q) \longrightarrow \{\zeta_l\}, \quad \text{with} \quad \tau_l(P, Q) = f_P(\phi(Q))^{\frac{q^k-1}{l}},$$

is a non-degenerate bilinear pairing, where f_P is a rational function with denominator one, i.e. a polynomial, satisfying $(f_P) = l(P) - l(O)$.

The crucial difference between the above pairing with a distortion map ϕ and the conventional Tate pairing is that this new pairing is symmetric as long as $E[l](\mathbb{F}_q)$ is a cyclic group, while the original Tate pairing is not. The reason is that, in this new pairing, both of the points P and Q are in the same cyclic group $E[l](\mathbb{F}_q)$ generated by a point R of order l. Thus there are integers a and b satisfying $P = aR$ and $Q = bR$ so that we have

$$\tau_l(P, Q) = \tau_l(aR, bR) = \tau_l(R, R)^{ab} = \tau_l(bR, aR) = \tau_l(Q, P). \tag{6}$$

Efficient computation of the Tate pairing is closely related with efficient computation of the scalar multiplication lP of a given point P since one has to find a rational function f_P satisfying $(f_P) = l(P) - l(O)$. Usually a binary representation of l is used for the field $\mathbb{F}_{2^m}$ or the field $\mathbb{F}_p$ with p a prime. A (balanced) ternary representation of l is an optimal choice for $\mathbb{F}_{3^m}$ and both of the algorithms of BKLS [1] and Duursma-Lee [3] made careful studies for this case.

For a field with characteristic *three*, $\mathbb{F}_q$ with $q = 3^m$, Duursma and Lee [3] noticed that one can obtain a faster Tate pairing computation if one use $q^3 + 1 = 3^{3m} + 1$ instead of using

l dividing q^3+1, since the ternary expansion of q^3+1 is trivial. That is, if one write g_Q as a rational function satisfying

$$3(Q)-3(O)=(3Q)-(O)+(g_Q),$$

then, by repeated applications of the above equation, one has

$$3^{3m}(P)-3^{3m}(O)=(3^{3m}P)-(O)+(g_P^{3^{3m-1}}g_{3P}^{3^{3m-2}}\cdots g_{3^{3m-2}P}^{3}g_{3^{3m-1}P}).$$

It is shown [3] that the rational function

$$f=\prod_{i=1}^{3m}g_{3^{i-1}P}^{3^{3m-i}}=g_P^{3^{3m-1}}g_{3P}^{3^{3m-2}}\cdots g_{3^{3m-2}P}^{3}g_{3^{3m-1}P} \tag{7}$$

can be used for a computation of the Tate pairing as

$$\tau_l(P,Q)=f(\phi(Q))^{3^{3m}-1}. \tag{8}$$

Duursma and Lee showed that the value $f(\phi(Q))=\prod_{i=1}^{3m}\{g_{3^{i-1}P}(\phi(Q))\}^{3^{3m-i}}$ has certain cyclic property with regard to the polynomials $g_{3^{i-1}P}^{3^{3m-i}}$ so that they found a nice closed formula [3] for f as a product of m (not $3m$) polynomials.

4. Tate pairing computation for binary fields with closed formulas

4.1. Supersingular elliptic curves over binary fields

For cryptographic purposes, it is natural to think of elliptic curves defined over $\mathbb{F}_{2^m}$ with m odd or more strongly a prime. There are exactly three isomorphism classes of supersingular elliptic curves over $\mathbb{F}_{2^m}$ when m is odd [15]. Namely they are

$$Y^2+Y=X^3,\ \ Y^2+Y=X^3+X,\ \ Y^2+Y=X^3+X+1. \tag{9}$$

Among them, the curves

$$E_b:Y^2+Y=X^3+X+b,\quad b=0,1 \tag{10}$$

have the embedding degree (or security multiplier) $k=4$ while the curve $Y^2+Y=X^3$ has $k=2$. Thus we are mainly interested in the curves E_b though our method is also applicable to the curve $Y^2+Y=X^3$. The Frobenius map $\varphi:E_b\longrightarrow E_b$ with $\varphi(x,y)=(x^2,y^2)$ is a root of the characteristic polynomial

$$h(X)=X^2\pm 2X+2=(X-\varphi)(X-\bar{\varphi}).$$

We also have the order $|E_b(\mathbb{F}_{2^m})|$ of the group of rational points $E_b(\mathbb{F}_{2^m})$ as

$$|E_b(\mathbb{F}_{2^m})|=2^m+1-Tr(\varphi^m),$$

where $Tr(\varphi^m)=\varphi^m+\bar{\varphi}^m$ and $\varphi^m(x,y)=(x^{2^m},y^{2^m})$. Letting $c_j=Tr(\varphi^j)$, one can find the values of c_j using the following second order liner recurrence relations (or Lucas type sequences) arising from the characteristic polynomial $h(X)$,

$$c_j=2(\mp c_{j-1}-c_{j-2}),\quad j\geq 0, \tag{11}$$

with $c_0 = 2$ and $c_1 = \mp 2$. From the above relations, it is straightforward to see [15] that $E_b(\mathbb{F}_{2^m})$ is a cyclic group of order

$$\begin{aligned} |E_b(\mathbb{F}_{2^m})| &= 2^m + 1 + (-1)^b \sqrt{2 \cdot 2^m}, \quad \text{if} \quad m \equiv 1, 7 \pmod 8 \\ &= 2^m + 1 - (-1)^b \sqrt{2 \cdot 2^m}, \quad \text{if} \quad m \equiv 3, 5 \pmod 8. \end{aligned} \tag{12}$$

4.2. Closed formula of the Tate pairing for $\mathbb{F}_{2^m}$

As in the characteristic three case of Duursma and Lee [3], we want to derive a closed formula for the Tate pairing computation using the simple equality for our binary case,

$$2^{2m} + 1 = (2^m + 1 + 2^{\frac{m+1}{2}})(2^m + 1 - 2^{\frac{m+1}{2}}).$$

Let $P = (\alpha, \beta)$ be a point on the curve $E_b : Y^2 + Y = X^3 + X + b,\ b = 0, 1$. Then one has $-P = (\alpha, \beta + 1)$ and $2P = (\alpha^4 + 1, \alpha^4 + \beta^4)$. Thus we get

$$2^2P = (\alpha^{2^4}, \beta^{2^4} + 1) = -\varphi^4(P),\ 2^3P = (\alpha^{2^6} + 1, \alpha^{2^6} + \beta^{2^6} + 1),\ 2^4P = (\alpha^{2^8}, \beta^{2^8}),$$

where $\varphi^4 + 4 = 0$, i.e. $h(X) = X^2 \pm 2X + 2$ divides $X^4 + 4$. Using this cyclic property, one finds easily

$$\begin{aligned} 2^{i-1}P &= (\alpha^{2^{2i-2}} + i - 1, \beta^{2^{2i-2}} + (i-1)\alpha^{2^{2i-2}} + \epsilon_i) \\ &= (\alpha^{(2i-2)} + i - 1, \beta^{(2i-2)} + (i-1)\alpha^{(2i-2)} + \epsilon_i), \end{aligned} \tag{13}$$

where $\alpha^{(j)}$ (resp. $\beta^{(j)}$) is defined as $\alpha^{(j)} = \alpha^{2^j}$ (resp. $\beta^{(j)} = \beta^{2^j}$) and ϵ_i is defined as

$$\epsilon_i = 0 \quad \text{if} \quad i \equiv 1, 2 \pmod 4 \quad \text{and} \quad \epsilon_i = 1 \quad \text{if} \quad i \equiv 3, 4 \pmod 4. \tag{14}$$

For an effective Tate pairing computation, the following distortion map (nontrivial automorphism) is chosen [1] for E_b,

$$\phi : E_b \longrightarrow E_b, \quad \text{with} \quad \phi(x, y) = (x + s^2, y + sx + t), \tag{15}$$

where $s^2 + s + 1 = 0$ and $t^2 + t + s = 0$. That is,

$$\mathbb{F}_2(s) = \mathbb{F}_{2^2}, \quad \mathbb{F}_2(t) = \mathbb{F}_{2^4}, \quad s = t^5, \quad t^4 + t + 1 = 0, \tag{16}$$

and t is a generator of the cyclic group $\mathbb{F}_{2^4}^\times$ of order 15. Therefore if P is a point of order l in $\mathbb{F}_{2^m}$ with m odd, then $\phi(P) \in E(\mathbb{F}_{2^{4m}})$ but $\phi(P) \notin E(\mathbb{F}_{2^{2m}})$, and the two points P and $\phi(P)$ generate all points of order l as a $\mathbb{Z}/l$ module.

For any point Q on the curve E_b, let us write g_Q as a rational function satisfying

$$2(Q) - 2(O) = (2Q) - (O) + (g_Q).$$

By the Miller's formula in (4), we have $g_Q = \ell_{Q,Q}/\ell_{2Q}$ and the denominator ℓ_{2Q} can be omitted by the result of Barreto et al. [1]. Now for a given point $P \in E_b(\mathbb{F}_{2^m})$, one repeatedly has

$$\begin{aligned} 2(P) - 2(O) &= (2P) - (O) + (g_P), \\ 2^2(P) - 2^2(O) &= 2\{(2P) - (O)\} + (g_P^2) = (2^2P) - (O) + (g_P^2 g_{2P}), \\ 2^3(P) - 2^3(O) &= 2\{(2^2P) - (O)\} + (g_P^{2^2} g_{2P}^2) = (2^3P) - (O) + (g_P^{2^2} g_{2P}^2 g_{2^2P}), \\ &\cdots \\ 2^{2m}(P) - 2^{2m}(O) &= (2^{2m}P) - (O) + (g_P^{2^{2m-1}} g_{2P}^{2^{2m-2}} \cdots g_{2^{2m-2}P}^2 g_{2^{2m-1}P}). \end{aligned}$$

Letting

$$f_P = \prod_{i=1}^{2m} g_{2^{i-1}P}^{2^{2m-i}} = g_P^{2^{2m-1}} g_{2P}^{2^{2m-2}} \cdots g_{2^{2m-2}P}^{2} g_{2^{2m-1}P}, \tag{17}$$

we have

$$2^{2m}(P) - 2^{2m}(O) = (2^{2m}P) - (O) + (f_P) \quad \text{and} \quad (P) - (O) = (P) - (O) + (1).$$

Thus the equation (4) of the Miller's formula again says

$$(2^{2m}+1)\{(P) - (O)\} = (f_P \ell_P), \tag{18}$$

because $2^{2m}P = -P$. Note that the line ℓ_P can also be omitted in the actual computation by the BKLS algorithm. Therefore after adjusting the irrelevant factors, we can say that

$$(f_P) = (2^{2m}+1)\{(P) - (O)\} = \tfrac{2^{2m}+1}{l} \cdot \{l(P) - l(O)\} = \tfrac{2^{2m}+1}{l}(f'_P), \tag{19}$$

where f'_P is a rational function satisfying $l(P) - l(O) = (f'_P)$. Thus we have the Tate pairing

$$\tau_l(P,Q) = f'_P(\phi(Q))^{\frac{2^{4m}-1}{l}} = f'_P(\phi(Q))^{\frac{2^{2m}+1}{l}(2^{2m}-1)} = f_P(\phi(Q))^{2^{2m}-1}. \tag{20}$$

From the equation (17), the rational function f_P is just a product of the functions of the form $g_{2^{i-1}P}$ and, in view of the BKLS algorithm, the rational function $g_{2^{i-1}P}$ can be regarded as the tangent line at the point $2^{i-1}P$. Thus all we have to do is to find an explicit expression of $f_P = \prod_{i=1}^{2m} g_{2^{i-1}P}^{2^{2m-i}}$.

Lemma 3. *Let $P = (\alpha, \beta), Q = (x, y)$ be points in $E_b(\mathbb{F}_{2^m})$. Then one has the value of $\{g_{2^{i-1}P}(\phi(Q))\}^{2^{2m-i}} = \{g_{2^{i-1}P}(x + s^2, y + sx + t)\}^{2^{2m-i}}$ as*

$$\{g_{2^{i-1}P}(\phi(Q))\}^{2^{2m-i}} = \alpha^{(i-1)}x^{(-i)} + \beta^{(i-1)} + y^{(-i)} + s(\alpha^{(i-1)} + x^{(-i)}) + t + b,$$

where $g_R(X,Y) = \ell_{R,R}$ is an equation of the tangent line at R.

Proof. The tangent line at $P = (\alpha, \beta)$ on the curve $E_b : Y^2 + Y = X^3 + X + b$ is $Y = (\alpha^2+1)X + \beta^2 + b$. Thus we have $2(P) - 2(O) = (2P) - (O) + (\frac{g_P}{\ell_{2P}})$ where

$$g_P(x,y) = (\alpha^2+1)x + \beta^2 + b - y, \tag{21}$$

and ℓ_{2P} is the vertical line intersecting $2P$ and $-2P$. Since ℓ_{2P} can be removed in view of the BKLS algorithm [1], we are mainly interested in the computations of the lines $g_{2^{i-1}P}$. Using the equation (13), one has

$$g_{2^{i-1}P}(x,y) = (\alpha^{(2i-1)} + i)x + \beta^{(2i-1)} + (i-1)\alpha^{(2i-1)} + \epsilon_i + b - y.$$

Therefore, by applying the distortion map ϕ to the point $Q = (x, y)$, we get

$$g_{2^{i-1}P}(x+s^2, y+sx+t) = (\alpha^{(2i-1)}+i)(x+s^2) + \beta^{(2i-1)} + (i-1)\alpha^{(2i-1)} + \epsilon_i + b - (y+sx+t). \tag{22}$$

Taking 2^{2m-i}-th power of both sides of the above equality,

$$\begin{aligned}\{g_{2^{i-1}P}(\phi(Q))\}^{2^{2m-i}}&\\ &= (\alpha^{(i-1)}+i)(x^{(2m-i)}+s^{(2m-i+1)})+\beta^{(i-1)}+(i-1)\alpha^{(i-1)}+\epsilon_i+b\\ &\quad -(y^{(2m-i)}+s^{(2m-i)}x^{(2m-i)}+t^{(2m-i)})\\ &= \alpha^{(i-1)}x^{(2m-i)}+\{i-s^{(2m-i)}\}x^{(2m-i)}+\{s^{(2m-i+1)}+i-1\}\alpha^{(i-1)}\\ &\quad +\beta^{(i-1)}+b-y^{(2m-i)}+\{is^{(2m-i+1)}+\epsilon_i-t^{(2m-i)}\}.\end{aligned} \tag{23}$$

From $s^2+s+1=0$, we have $s^{(2)}=s^4=s, s^{(3)}=s^2=s+1, s^{(4)}=s, \cdots$. That is,

$$s^{(j)}=s+1 \quad \text{if } j=odd \quad \text{and} \quad s^{(j)}=s \quad \text{if } j=even. \tag{24}$$

The coefficients $i-s^{(2m-i)}$ (resp. $i-1+s^{(2m-i+1)}$) of $x^{(2m-i)}$ (resp. $\alpha^{(i-1)}$) in the equation (23) have a unique value equal to s independent of the choices of i because i and $2m-i$ always have the same parity. For example, when i is odd, $i-s^{(2m-i)}=1+s+1=s$ and also when i is even, $i-s^{(2m-i)}=0+s=s$. That is, for any i, we get

$$i-s^{(2m-i)}=s=i-1+s^{(2m-i+1)}. \tag{25}$$

From $t^2=t+s$, we have $t^{(2)}=t^{2^2}=t+s+s^2=t+1, t^{(3)}=t^{2^3}=t+s+1, t^{(4)}=t+s+s^2+1=t, t^{(5)}=t^2=t+s, \cdots$. Therefore, for any $j\geq 0$, we have

$$t^{(4j)}=t, \quad t^{(4j+1)}=t+s, \quad t^{(4j+2)}=t+1, \quad t^{(4j+3)}=t+s+1. \tag{26}$$

Now using the equations (14),(24),(26), it is trivial to show that the last term of the equation (23) has the value

$$is^{(2m-i+1)}+\epsilon_i-t^{(2m-i)}=t \tag{27}$$

independent of the choices of i. This can be proved as follows. Since the extension degree m is odd, we have $m\equiv 1 \pmod 4$ or $m\equiv 3 \pmod 4$. In any case, we get $2m\equiv 2 \pmod 4$ and letting $2m=4j+2$ for some j,

$$is^{(2m-i+1)}+\epsilon_i-t^{(2m-i)}=is^{(4j+3-i)}+\epsilon_i-t^{(4j+2-i)}. \tag{28}$$

By taking $i \pmod 4$ and noticing that our field has characteristic *two*, we easily get the equation (27). Since x, y, α, β are all in $\mathbb{F}_{2^m}$, the values $x^{(j)}, y^{(j)}, \alpha^{(j)}, \beta^{(j)}$ are determined up to the residue classes of $j \pmod m$ and $x^{(j)}$ with $j\in\mathbb{Z}$ (resp. $y^{(j)}, \alpha^{(j)}, \beta^{(j)}$) is understood as $x^{(j)}=x^{2^{j'}}$ where j', $0\leq j'\leq m-1$, is a unique integer satisfying $j'\equiv j \pmod m$. Therefore we get

$$\begin{aligned}\{g_{2^{i-1}P}(\phi(Q))\}^{2^{2m-i}} &= \alpha^{(i-1)}x^{(-i)}+sx^{(-i)}+s\alpha^{(i-1)}+\beta^{(i-1)}+y^{(-i)}+t+b\\ &= \alpha^{(i-1)}x^{(-i)}+\beta^{(i-1)}+y^{(-i)}+s(\alpha^{(i-1)}+x^{(-i)})+t+b.\end{aligned}$$

□

Theorem 4. *One has the Tate pairing* $\tau_l(P,Q)=f_P(\phi(Q))^{2^{2m}-1}$ *where*

$$f_P(\phi(Q))=\prod_{i=1}^{m}\{\alpha^{(i)}x^{(-i+1)}+\beta^{(i)}+y^{(-i+1)}+s^2(\alpha^{(i)}+x^{(-i+1)})+t^2+b\}.$$

Proof. By the equation (17) and (20), we have $f_P(\phi(Q)) = \prod_{i=1}^{2m}\{g_{2^{i-1}P}(\phi(Q))\}^{2^{2m-i}}$ and since all $x^{(j)}, y^{(j)}, \alpha^{(j)}, \beta^{(j)}$ are determined up to the residue classes of $j \pmod{m}$,

$$\begin{aligned} f_P(\phi(Q)) &= \prod_{i=1}^{m}\{g_{2^{i-1}P}(\phi(Q))\}^{2^{2m-i}} \prod_{i=m+1}^{2m}\{g_{2^{i-1}P}(\phi(Q))\}^{2^{2m-i}} \\ &= \prod_{i=1}^{m}\{g_{2^{i-1}P}(\phi(Q))\}^{2^{2m-i}} \prod_{i=1}^{m}\{g_{2^{i-1}P}(\phi(Q))\}^{2^{2m-i}} \\ &= \prod_{i=1}^{m}\{\alpha^{(i-1)}x^{(-i)} + \beta^{(i-1)} + y^{(-i)} + s(\alpha^{(i-1)} + x^{(-i)}) + t + b\}^2 \\ &= \prod_{i=1}^{m}\{\alpha^{(i)}x^{(-i+1)} + \beta^{(i)} + y^{(-i+1)} + s^2(\alpha^{(i)} + x^{(-i+1)}) + t^2 + b\}. \end{aligned}$$

□

5. Efficient field arithmetic for the computation of $f_P(\phi(Q))$

The computation of $f_P(\phi(Q))$ involves multiplications in $\mathbb{F}_{2^{4m}}$. A natural way to do this is to use a basis for $\mathbb{F}_{2^{4m}}$ over $\mathbb{F}_{2^m}$ and transforms a multiplication in $\mathbb{F}_{2^{4m}}$ into several multiplications in $\mathbb{F}_{2^m}$. Since the extension degree is 4, we may use an optimal normal basis of type I but we will stick to the polynomial basis $\{1, t, t^2, t^3\}$ for $\mathbb{F}_{2^{4m}}$ with the minimal polynomial of t as $X^4 + X + 1$ over $\mathbb{F}_{2^m}$. Using $s^2 = t^2 + t + 1$, we may express the element $\alpha^{(i)}x^{(-i+1)} + \beta^{(i)} + y^{(-i+1)} + s^2(\alpha^{(i)} + x^{(-i+1)}) + t^2 + b$ as

$$\alpha^{(i)}x^{(-i+1)} + \beta^{(i)} + y^{(-i+1)} + s^2(\alpha^{(i)} + x^{(-i+1)}) + t^2 + b = w + zt + (z+1)t^2,$$

where

$$z = \alpha^{(i)} + x^{(-i+1)}, \quad w = z + \alpha^{(i)}x^{(-i+1)} + \beta^{(i)} + y^{(-i+1)} + b. \tag{29}$$

Letting $C = c_0 + c_1t + c_2t^2 + c_3t^3$, $c_i \in \mathbb{F}_{2^m}$, be the partial product in the computation of $f_P(\phi(Q))$, we have

$$\begin{aligned} C \cdot (w + zt + (z+1)t^2) &= (c_0 + c_1t + c_2t^2 + c_3t^3)(w + zt + (z+1)t^2) \\ &= c_0' + c_1't + c_2't^2 + c_3't^3, \end{aligned} \tag{30}$$

where

$$\begin{aligned} c_0' &= c_0w + (c_2 + c_3)(z+1) + c_3 \\ c_1' &= c_0w + (c_1 + c_2 + c_3)w + (c_0 + c_2 + c_3)(w + z + 1) + c_3(z+1) + c_0 + c_3 \\ c_2' &= c_0w + (c_1 + c_2 + c_3)w + (c_0 + c_2 + c_3)(w + z + 1) + (c_1 + c_2)(w + z + 1) + c_1 \\ c_3' &= (c_1 + c_2 + c_3)w + (c_1 + c_2)(w + z + 1) + c_2. \end{aligned} \tag{31}$$

Therefore one needs only 6 multiplications for the computation of $C \cdot (w + zt + (z+1)t^2)$.

Table 1. An algorithm for computing $f_P(\phi(Q))$

Input: $P = (\alpha, \beta), Q = (x, y)$
Output: $C = f_P(\phi(Q))$

```
$C \leftarrow 1$
for $(i = 1$ to $m$ ; $i++)$
$\alpha \leftarrow \alpha^2, \quad \beta \leftarrow \beta^2$
$z \leftarrow \alpha + x, \quad w \leftarrow z + \alpha x + \beta + y + b$
$C \leftarrow C \cdot (w + zt + (z+1)t^2)$
$x \leftarrow x^{2^{m-1}}, \quad y \leftarrow y^{2^{m-1}}$
end for
```

Assuming that we are using a normal basis for $\mathbb{F}_{2^m}$ over $\mathbb{F}_2$, the Frobenius maps in Table 1 contribute a negligible cost. Moreover the map $x \leftarrow x^{2^{m-1}}$ is just a left cyclic shifting by one position of the vector x with respect to a normal basis while $x \leftarrow x^2$ is a right cyclic shifting by one position. All these Frobenius maps are especially useful if one wants an efficient hardware implementation. If we ignore the costs of Frobenius maps and $\mathbb{F}_{2^m}$-additions, we find that exactly 7 $\mathbb{F}_{2^m}$-multiplications are needed in each round of the for-loop, where the computation of w needs one multiplication in $\mathbb{F}_{2^m}$ and the computation of C needs 6 multiplications in $\mathbb{F}_{2^m}$ by the equation (31). Compare our result with the similar result in $\mathbb{F}_{3^m}$ case of Duursma and Lee where each step of the algorithm in [3] requires 17 $\mathbb{F}_{3^m}$-multiplications with trace computation technique [2] and can be reduced to 14 $\mathbb{F}_{3^m}$-multiplications [8] with loop unfolding technique.

It should be mentioned that one can also use a normal basis for $\mathbb{F}_{2^{4m}}$ over $\mathbb{F}_{2^m}$ instead of using $\{1, t, t^2, t^3\}$ with $t^4 + t + 1 = 0$. Letting $t^3 = \gamma$, one has $\gamma^5 = 1$ and the minimal polynomial of γ over $\mathbb{F}_{2^m}$ is $X^4 + X^3 + X^2 + X + 1$. Therefore we have a normal basis $\{\gamma, \gamma^2, \gamma^{2^2}, \gamma^{2^3}\} = \{\gamma, \gamma^2, \gamma^3, \gamma^4\}$ of type I over $\mathbb{F}_{2^m}$. Using the relation $t = \frac{1}{t^3+1} = \frac{1}{\gamma+1} = \gamma^3 + \gamma$, one may reformulate the equations (30) and (31) with respect to the basis $\{\gamma, \gamma^2, \gamma^3, \gamma^4\}$. In this case, the number of necessary additions in $\mathbb{F}_{2^m}$ slightly increases, however the expressions of the coefficients of C in (31) have more regular patterns which are particularly useful for a hardware implementation.

Computing the final powering by $2^{2m} - 1$ is a formidable task. However in some situations like a signature verification, one only needs to determine whether $\tau_l(P, Q) = \tau_l(P', Q')$ without having to know the exact value of $\tau_l(P, Q) = f_P(\phi(Q))^{2^{2m}-1}$. In this case, it suffices to check whether $f_P(\phi(Q))^{2^{2m}} f_{P'}(\phi(Q')) = f_P(\phi(Q)) f_{P'}(\phi(Q'))^{2^{2m}}$ and the cost of this operation is much cheaper than the cost of the exponentiation by $2^{2m} - 1$. Replacing 2^{2m} by 3^{3m}, the same technique is also applicable to the Duursma-Lee algorithm [3].

6. Algorithms without inverse Frobenius operations for polynomial basis arithmetic in $\mathbb{F}_{2^m}$ and $\mathbb{F}_{3^m}$

Many computational evidence [8,20] imply that a more efficient field arithmetic can be obtained for low characteristic finite fields by using a polynomial basis than a normal basis, especially for software purposes. Though a Gaussian normal basis of low complexity [26] is a good choice for a fast arithmetic, such basis does not appear quite frequently when compared with a polynomial basis of low hamming weight (like trinomial or pentanomial). In the case of the Tate pairing computation, the same phenomenon that a polynomial basis wins over a normal basis has been observed by Granger, Page, and Stam [8]. Granger et al. [8] showed that, even though a cube root operation (inverse Frobenius operation for characteristic *three*) in a polynomial basis is quite costly, an algorithm for the Tate pairing computation with a polynomial basis outperforms a method with a normal basis since one needs many operations of multiplication while only two cube root operations are needed in each step of the Duursma-Lee algorithm [3,8] and since the cost of a multiplication with a normal basis is quite expensive than that of a polynomial basis in general situations. With a small amount of precomputation, Granger et al. [8] showed that a cube root operation in $\mathbb{F}_{3^m}$ has roughly the same cost as two multiplications in $\mathbb{F}_{3^m}$. The same method in [8] can be applied to our characteristic *two* case so that we can show that the cost of one square root operation is roughly equal to the cost of one multiplication with a precomputation. It should be mentioned that a general case without a precomputation is not so simple and one needs at least $O(m^2 \log_2 m)$ additions in $\mathbb{F}_2$ to find a square root in $\mathbb{F}_{2^m}$ as is observed by Barreto et al. [2], though the complexity can be reduced to $O(m^2)$ additions in $\mathbb{F}_2$ if we use a low weight polynomial like a trinomial or a pentanomial.

6.1. Avoiding square root extraction

In this section, we briefly remark that a close examination of the algorithm in Table 1 reveals that one may derive a new algorithm for the Tate pairing computation which does not need any inverse Frobenius operation (like square root or cube root extractions). Our method is also applicable to the characteristic *three* case of Duursma an Lee [3] and will be explained later. Let us first study the binary case here. From Theorem 4, we know that

$$f_P(\phi(Q)) = \prod_{i=1}^{m} \{\alpha^{(i)} x^{(-i+1)} + \beta^{(i)} + y^{(-i+1)} + s^2(\alpha^{(i)} + x^{(-i+1)}) + t^2 + b\}. \tag{32}$$

We define A_i as the conjugates of the terms in the product of the above formula by

$$A_i^{(m-i)} = A_i^{2^{m-i}} = \alpha^{(i)} x^{(-i+1)} + \beta^{(i)} + y^{(-i+1)} + s^2(\alpha^{(i)} + x^{(-i+1)}) + t^2 + b$$

so that

$$f_P(\phi(Q)) = \prod_{i=1}^{m} A_i^{2^{m-i}} = A_1^{2^{m-1}} A_2^{2^{m-2}} \cdots A_m = (\cdots(((A_1)^2 A_2)^2 A_3)^2 \cdots)^2 A_m. \tag{33}$$

Since A_i is in $\mathbb{F}_{2^{4m}}$, we get $A_i^{(4m)} = A_i$. Therefore, using the fact $\alpha, \beta, x, y \in \mathbb{F}_{2^m}$, we have

$$\begin{aligned} A_i = (A_i^{(m-i)})^{2^{3m+i}} &= \alpha^{(2i)} x^{(1)} + \beta^{(2i)} + y^{(1)} + s^{(3m+1+i)}(\alpha^{(2i)} + x^{(1)}) + t^{(3m+1+i)} + b \\ &= \alpha^{(2i)} x^2 + \beta^{(2i)} + y^2 + s^{(i)}(\alpha^{(2i)} + x^2) + t^{(m-1+i)} + b, \end{aligned} \tag{34}$$

because $s^{(j)}$ is determined up to $j \pmod 2$ with $3m+1 \equiv 0 \pmod 2$ and $t^{(j)}$ is determined up to $j \pmod 4$ with $3m+1 \equiv m-1 \pmod 4$ as is clear from the equations (24) and (26). Using the cyclic property of $t^{(j)}$ in the equation (26), it is not difficult to see that, for all indices $1 \le i \le m$, A_i can be written as $A_i = A_i(t) = w + zt + (z+1)t^2$ for some z and w in $\mathbb{F}_{2^m}$. Thus, similarly as in the equations (30) and (31), one needs 6 $\mathbb{F}_{2^m}$-multiplications for computing $C \cdot A_i(t)$ with respect to the basis $\{1, t, t^2, t^3\}$ for any $C \in \mathbb{F}_{2^{4m}}$. We now have the following algorithm for computing $f_P(\phi(Q))$ which avoids inverse Frobenius operations.

Table 2. An algorithm for computing $f_P(\phi(Q))$ avoiding inverse Frobenius operation

```
Input: $P = (\alpha, \beta), Q = (x, y)$
Output: $C = f_P(\phi(Q))$
$C \leftarrow 1$
$u \leftarrow x^2, \quad v \leftarrow x^2, \quad y \leftarrow y^2$
for ($i = 1$ to $m$ ; $i++$)
$\alpha \leftarrow \alpha^4, \quad \beta \leftarrow \beta^4$
$A(t) \leftarrow \alpha(v+1) + u + \beta + y + b + \frac{m-1}{2} + (\alpha + v)t + (\alpha + v + 1)t^2$
$C \leftarrow C^2 \cdot A(t)$
$u \leftarrow u + v + 1, \quad v \leftarrow v + 1$
end for
```

Note that the coefficients of $A_i(t)$ depend on the values of $s^{(i)}$ and $t^{(m-1+i)}$ and they are recursively computed by the relation (24) and (26). We have the initial values $s^{(1)} = s^2 = t^2 + t + 1$ and $t^{(m)} = t^2 + \frac{m-1}{2}$ and thus we get

$$
\begin{aligned}
A_1(t) &= \alpha x + \beta + y + (t^2 + t + 1)(\alpha + x) + t^2 + \tfrac{m-1}{2} + b \\
&= \alpha(x+1) + x + \beta + y + b + \tfrac{m-1}{2} + (\alpha + x)t + (\alpha + x + 1)t^2, \\
A_2(t) &= \alpha x + \beta + y + (t^2 + t)(\alpha + x) + t + 1 + \tfrac{m-1}{2} + b \\
&= \alpha x + 1 + \beta + y + b + \tfrac{m-1}{2} + (\alpha + x + 1)t + (\alpha + x)t^2, \\
A_3(t) &= \alpha(x+1) + (x+1) + \beta + y + b + \tfrac{m-1}{2} + (\alpha + x)t + (\alpha + x + 1)t^2, \\
A_4(t) &= \alpha x + \beta + y + b + \tfrac{m-1}{2} + (\alpha + x + 1)t + (\alpha + x)t^2.
\end{aligned}
$$

Using the intermediate values u, v with the relations $u \leftarrow u+v+1, \ v \leftarrow v+1$, the pair covers all the possible values $(u, v) = (x, x), (1, x+1), (x+1, x), (0, x+1)$ and the algorithm in Table 2 is justified. In each step of the above algorithm, one needs 7 $\mathbb{F}_{2^m}$-multiplications which is same to the algorithm in Table 1. Since the operation $C \leftarrow C^2$ needs 4 squaring operations in $\mathbb{F}_{2^m}$ and since the operations $\alpha \leftarrow \alpha^4$, $\beta \leftarrow \beta^4$ need the same 4 squaring operations, the total number of necessary squaring is 8 in this new algorithm. On the other hand, the algorithm in Table 1 needs 2 squaring and 2 square root operations. Therefore our new algorithm in Table 2 is a more optimal choice if one is interested in the implementation with a polynomial basis since this new algorithm uses 6 Frobenius operations instead of using 2 inverse Frobenius operations.

6.2. Avoiding cube root extraction from the algorithm of Duursma and Lee

Duursma and Lee [3] found a close formula for the following supersingular elliptic curves defined over $\mathbb{F}_{3^m}$ with m prime to 6,

$$E_b : Y^2 = X^3 - X + b, \quad b = \pm 1. \tag{35}$$

For the above mentioned curves, the following distortion map (nontrivial automorphism) is used,

$$\phi : E_b \longrightarrow E_b, \quad \text{with} \quad \phi(x, y) = (\rho - x, \sigma y), \tag{36}$$

where $\sigma^2 + 1 = 0$ and $\rho^3 - \rho - b = 0$. That is, $\mathbb{F}_3(\sigma) = \mathbb{F}_{3^2}$ and $\mathbb{F}_3(\rho) = \mathbb{F}_{3^3}$. A closed formula of Duursma and Lee says that, for $P = (\alpha, \beta)$ and $Q = (x, y)$ in $E[l](\mathbb{F}_{3^m})$, the Tate pairing can be written as $\tau_l(P,Q) = f_P(\phi(Q))^{3^{3m}-1}$ where

$$f_P(\phi(Q)) = \prod_{i=1}^{m} \{-\sigma\beta^{(i)}y^{(-i+1)} - (\alpha^{(i)} + x^{(-i+1)} - \rho + b)^2\}, \tag{37}$$

and f_P is a rational function satisfying $(f_P) = (3^{3m}+1)\{(P) - (O)\}$. Now define the intermediate values μ and λ as

$$\mu = \alpha^{(i)} + x^{(-i+1)} + b \in \mathbb{F}_{3^m} \quad \text{and} \quad \lambda = -\sigma\beta^{(i)}y^{(-i+1)} - \mu^2 \in \mathbb{F}_{3^{2m}}.$$

Then the formula (37) to compute $f_P(\phi(Q))$ is realized by the following algorithm [2,3,8].

Table 3. Duursma-Lee algorithm for computing $f_P(\phi(Q))$

```
Input: $P = (\alpha, \beta), Q = (x, y)$
Output: $C = f_P(\phi(Q))$
$C \leftarrow 1$
for ($i = 1$ to $m$ ; $i++$)
$\alpha \leftarrow \alpha^3, \quad \beta \leftarrow \beta^3$
$\mu = \alpha + x + b, \quad \lambda = -\sigma\beta y - \mu^2$
$C \leftarrow C \cdot (\lambda - \mu\rho - \rho^2)$
$x \leftarrow x^{1/3}, \quad y \leftarrow y^{1/3}$
end for
```

One needs 2 cube root operations in each step of the above algorithm. However it is not so difficult, by using the same technique of the previous section, to show that one can have a new algorithm where 2 cube root operations (inverse Frobenius) are replaced by 8 cube operations (Frobenius) without affecting the number of multiplications in $\mathbb{F}_{3^m}$, which are quite useful in polynomial basis arithmetic. Let us define $A_i \in \mathbb{F}_{3^{6m}}$ as the conjugates of the terms in the product formula (37) by

$$A_i^{(m-i)} = A_i^{3^{m-i}} = -\sigma\beta^{(i)}y^{(-i+1)} - (\alpha^{(i)} + x^{(-i+1)} - \rho + b)^2 \tag{38}$$

so that

$$f_P(\phi(Q)) = \prod_{i=1}^{m} A_i^{3^{m-i}} = A_1^{3^{m-1}} A_2^{3^{m-2}} \cdots A_m = (\cdots(((A_1)^3 A_2)^3 A_3)^3 \cdots)^3 A_m. \tag{39}$$

Since A_i is in $\mathbb{F}_{3^{6m}}$, we get $A_i^{(6m)} = A_i$. From the equation (38), using the fact $\alpha, \beta, x, y \in \mathbb{F}_{3^m}$, we have

$$\begin{aligned} A_i = (A_i^{(m-i)})^{3^{5m+i}} &= -\sigma^{(5m+i)}\beta^{(2i)}y^{(1)} - (\alpha^{(2i)} + x^{(1)} - \rho^{(5m+i)} + b)^2 \\ &= (-1)^{i+1}\sigma\beta^{(2i)}y^{(1)} - (\alpha^{(2i)} + x^{(1)} - \rho + (m+1-i)b)^2, \end{aligned} \tag{40}$$

because the relations $\sigma^2 + 1 = 0$, $\rho^3 - \rho - b = 0$ imply

$$\sigma^{(j)} = (-1)^j\sigma \quad \text{and} \quad \rho^{(j)} = \rho + jb. \tag{41}$$

Letting $\mu = \alpha^{(2i)} + x^{(1)} + (m+1-i)b \in \mathbb{F}_{3^m}$ and $\lambda = (-1)^{i+1}\sigma\beta^{(2i)}y^{(1)} - \mu^2 \in \mathbb{F}_{3^{2m}}$ from the equation (40), one finds that

$$A_i = \lambda - \mu\rho - \rho^2. \tag{42}$$

Therefore the modified algorithm is given as follows.

Table 4. A modified Duursma-Lee algorithm without cube root operations

```
Input: $P = (\alpha, \beta), Q = (x, y)$
Output: $C = f_P(\phi(Q))$
$C \leftarrow 1$
$x \leftarrow x^3, \quad y \leftarrow y^3, \quad d \leftarrow mb$
for ($i = 1$ to $m$ ; $i++$)
$\alpha \leftarrow \alpha^9, \quad \beta \leftarrow \beta^9$
$\mu = \alpha + x + d, \quad \lambda = \sigma\beta y - \mu^2$
$C \leftarrow C^3 \cdot (\lambda - \mu\rho - \rho^2)$
$y \leftarrow -y, \quad d \leftarrow d - b$
end for
```

In each step of the above algorithm, the number of necessary multiplications in $\mathbb{F}_{3^m}$ is same to that of the algorithm in Table 3. Since the cube operation $C \leftarrow C^3$ with respect to the basis $\{1, \rho, \rho^2\}$ over $\mathbb{F}_{3^{2m}}$ costs 6 cube operations in $\mathbb{F}_{3^m}$ and since the operations $\alpha \leftarrow \alpha^9$, $\beta \leftarrow \beta^9$ cost 4 cube operations in $\mathbb{F}_{3^m}$, the total number of necessary Frobenius operations in each step of the above algorithm is 10. Note that the Duursma-Lee algorithm in Table 3 needs 2 Frobenius operations plus 2 inverse Frobenius operations. Therefore our modified algorithm uses 8 Frobenius operations instead of using 2 inverse Frobenius operations. In a polynomial basis, it is safe to believe [8] that the cost of 4 cube operations is cheaper than the cost of one cube root operation.

It should be mentioned that our technique of avoiding inverse Frobenius operations can also be applied to the refined algorithm of Granger et al. [8], where the for-loop in Table 3 is unrolled so that it has $\frac{m-1}{2}$ steps and a multiplication of two $\lambda - \mu\rho - \rho^2$ is executed before being multiplied to the partial product C. The only thing we have to do is to redefine A_i in the equation (39) appropriately so that the multiplication $A_i A_{i+1}$ is performed before being multiplied to C.

7. Tate pairing computation for $Y^2 + Y = X^3$

Among the three isomorphism classes of supersingular elliptic curves over a binary field $\mathbb{F}_{2^m}$ with $m = odd$, $E_b : Y^2 + Y = X^3 + X + b$, $b = 0, 1$ and $E : Y^2 + Y = X^3$, the curve $Y^2 + Y = X^3$ has the embedding degree $k = 2$. Though the curve $Y^2 + Y = X^3$ is not so interesting in terms of the bandwidth, i.e. the imbedding degree $k = 2$, we will discuss a method of efficient Tate pairing computation with a closed formula. Note that a similar formula (like the cases of characteristic *two* and *three*) is not available for a prime field $\mathbb{F}_p$ with $p \neq 2, 3$ and one has the same embedding degree $k = 2$ for this prime field case. Although the curve $E : Y^2 + Y = X^3$ is not discussed by Barreto et al. in the BKLS algorithm [1], we will show that a similar technique about the irrelevant denominators is also applicable for this curve. It seems that this technique is applicable to quite a many class of elliptic curves with nontrivial automorphisms over low characteristic finite fields.

Let $P = (\alpha, \beta)$ be a point on the curve $E : Y^2 + Y = X^3$. Then one has $-P = (\alpha, \beta + 1)$ and $2P = (\alpha^4, \beta^4 + 1) = -\varphi^2(P)$. Thus we get

$$2^2 P = (\alpha^{2^4}, \beta^{2^4}) = \varphi^4(P),$$

where $\varphi^4 - 4 = 0$, i.e. $h(X) = X^2 + 2$ divides $X^4 - 4$. Using this property, it is easy to show inductively

$$2^{i-1}P = (\alpha^{2^{2i-2}}, \beta^{2^{2i-2}} + i - 1) = (\alpha^{(2i-2)}, \beta^{(2i-2)} + i - 1). \tag{43}$$

For an effective Tate pairing computation, we will use the following distortion map (nontrivial automorphism) for E,

$$\phi : E \longrightarrow E, \quad \text{with} \quad \phi(x, y) = (x + 1, y + x + t), \tag{44}$$

where $t \in \mathbb{F}_{2^2}$ with $t^2 + t + 1 = 0$. It is clear that the proposed map ϕ is an automorphism since the following equality can be easily checked,

$$(y + x + t)^2 + (y + x + t) = (x + 1)^3. \tag{45}$$

Lemma 5. *With the above distortion map, the line $X - u$ intersecting $R = (u, v)$ and $-R = (u, v + 1)$ with $R \in E(\mathbb{F}_{2^m})$ can be omitted without altering the pairing value.*

Proof. The line $X - u$ evaluated at the point $\phi(Q)$ with $Q = (x, y) \in E(\mathbb{F}_{2^m})$ is $x + 1 - u$. By applying the final powering by $\frac{2^{2m}-1}{l} = (2^m - 1)\frac{(2^m+1)}{l}$, one has $(x+1-u)^{\frac{2^{2m}-1}{l}} = 1$ because $x, u \in \mathbb{F}_{2^m}$ and l divides $|E(\mathbb{F}_{2^m})| = 2^m + 1$.

Lemma 6. *Let $P = (\alpha, \beta), Q = (x, y)$ be points in $E(\mathbb{F}_{2^m})$ with $E : Y^2 + Y = X^3$. Then one has the value of $\{g_{2^{i-1}P}(\phi(Q))\}^{2^{m-i}} = \{g_{2^{i-1}P}(x + 1, y + x + t)\}^{2^{m-i}}$ as*

$$\{g_{2^{i-1}P}(x + 1, y + x + t)\}^{2^{m-i}} = \alpha^{(i-1)}x^{(-i)} + (\alpha + \beta)^{(i-1)} + (x + y)^{(-i)} + t.$$

Proof. The tangent line at $P = (\alpha, \beta)$ on the curve $E : Y^2 + Y = X^3$ is $Y = \alpha^2 X + \beta^2$. Thus we have $g_P(x, y) = \alpha^2 x + \beta^2 - y$ and using the equation (43), we get

$$g_{2^{i-1}P}(x + 1, y + x + t) = \alpha^{(2i-1)}(x + 1) + \beta^{(2i-1)} + i - 1 - (y + x + t). \tag{46}$$

Taking 2^{m-i}-th power of the both sides of the above equality,

$$\{g_{2^{i-1}P}(x+1, y+x+t)\}^{2^{m-i}} = \alpha^{(i-1)}(x^{(-i)}+1) + \beta^{(i-1)} + i - 1 - (y^{(-i)} + x^{(-i)} + t^{(m-i)}). \tag{47}$$

Since $t^{(1)} = t^2 = t + 1$, one has $t^{(2)} = t^4 = t, t^{(3)} = t + 1, t^{(4)} = t, \cdots$. That is,

$$t^{(j)} = t + j, \tag{48}$$

for any j because we are in the field with characteristic *two*. Thus we have $t^{(m-i)} = t + m - i$ in the equation (47) and therefore

$$\begin{aligned}\{g_{2^{i-1}P}(\phi(Q))\}^{2^{m-i}} &= \alpha^{(i-1)}(x^{(-i)} + 1) + \beta^{(i-1)} + i - 1 - (y^{(-i)} + x^{(-i)} + t + m - i) \\ &= \alpha^{(i-1)}x^{(-i)} + (\alpha + \beta)^{(i-1)} + (x + y)^{(-i)} + t.\end{aligned}$$

□

Theorem 7. *One has the Tate pairing* $\tau_l(P, Q) = f_P(\phi(Q))^{2^m-1}$ *where*

$$f_P(\phi(Q)) = \prod_{i=1}^{m}\{g_{2^{i-1}P}(\phi(Q))\}^{2^{m-i}} = \prod_{i=1}^{m}\{\alpha^{(i-1)}x^{(-i)} + (\alpha+\beta)^{(i-1)} + (x+y)^{(-i)} + t\},$$

and f_P *is a rational function satisfying* $(2^m + 1)\{(P) - (O)\}$.

One may derive the same algorithms as in Table 1 and 2 for this case also but we omit them here since the method is pretty straightforward.

8. Conclusions

In this paper we showed that an efficient closed formula can be derived for the Tate pairing computation for supersingular elliptic curves over a binary field $\mathbb{F}_{2^m}$ of odd dimension. There are exactly three isomorphism classes of supersingular elliptic curves over $\mathbb{F}_{2^m}$ with m odd and our method is applicable to all these curves. Each step of our algorithm requires two inverse Frobenius operations like the characteristic three case of Duursma and Lee. To overcome the computational complexity of the inverse Frobenius operation in a polynomial basis, we modified our algorithm and the algorithm of Duursma and Lee, and presented another closed formula which does not need any inverse Frobenius operation, which is especially useful for polynomial basis arithmetic.

References

[1] P. Barreto, H. Kim, B. Lynn, and M. Scott, “Efficient algorithms for pairing based cryptosystems,” *Crypto 2002, Lecture Notes in Computer Science*, vol. 2442, pp. 354–368, 2002.

[2] M. Scott and P. Barreto, “Compressed pairings,” *Crypto 2004, Lecture Notes in Computer Science*, to appear, 2004.

[3] I. Duursma and H. Lee, “Tate pairing implementation for hyperelliptic curves $y^2 = x^p - x + d$,” *Asiacrypt 2003, Lecture Notes in Computer Science*, vol. 2894, pp. 111–123, 2003.

[4] S. Galbraith, “Supersingular curves in cryptography,” *Asiacrypt 2001, Lecture Notes in Computer Science*, vol. 2248, pp. 495–513, 2001.

[5] S. Galbraith, K. Harrison, and D. Soldera, “Implementing the Tate pairing,” *ANTS 2002, Lecture Notes in Computer Science*, vol. 2369, pp. 324–337, 2002.

[6] F. Hess, “A Note on the Tate pairing of curves over finite fields,” *Arch. Math.* vol. 82, pp. 28–32, 2004.

[7] F. Hess, “Efficient identity based signature schemes based on pairings,” *SAC 2002, Lecture Notes in Computer Science*, vol. 2595, 310-324, 2003.

[8] R. Granger, D. Page, and M. Stam, “Hardware and software normal basis arithmetic for pairing based cryptography in characteristic three,” preprint, *available at* http://citeseer.ist.psu.edu/695313.html, 2004.

[9] K. Rubin and A. Silverberg “Torus based cryptography,” *Crypto 2003, Lecture Notes in Computer Science*, vol. 2729, pp. 349–365, 2003.

[10] D. Boneh and M. Franklin, “Identity based encryption from the Weil pairing,” *Crypto 2001, Lecture Notes in Computer Science*, vol. 2139, pp. 213–229, 2001.

[11] D. Boneh, B. Lynn, and H. Shacham, “Short signatures from the Weil pairing,” *Asiacrypt 2001, Lecture Notes in Computer Science*, vol. 2248, pp. 514–532, 2002.

[12] A. Joux, “A one round protocol for tripartite Diffie-Hellman,” *ANTS 2000, Lecture Notes in Computer Science*, vol. 1838, pp. 385–394, 2000.

[13] K. Eisenträger, K. Lauter, and P.L. Montgomery, "Improved Weil and Tate pairing for elliptic and hyperelliptic curves," preprint, 2004.

[14] J.H. Silverman, *The Arithmetic of Elliptic Curves*, Springer-Verlag, 1985.

[15] A.J. Menezes, *Elliptic Curve Public Key Cryptosystems*, Kluwer Academic Publisher, 1993.

[16] A.J. Menezes, T. Okamoto, and S.A. Vanstone, "Reducing elliptic curve logarithms to logarithms in a finite field," *IEEE Trans. Information Theory*, vol. 39, pp. 1639–1646, 1993.

[17] G. Frey and H. Rück, "A remark concerning m-divisibility and the discrete logarithm in the divisor class groups of curves," *Math. Comp.*, vol. 62, pp. 865–874, 1994.

[18] E.R. Verheul, "Evidence that XTR is more secure than supersingular elliptic curve cryptosystems," *Eurocrypt 2001, Lecture Notes in Computer Science*, vol. 2045, pp. 195–210, 2001.

[19] V. Miller, "Short programs for functions on curves," *unpublished manuscript*, 1986.

[20] D. Hankerson, J.L. Hernandez, and A.J. Menezes, "Software implementation of elliptic curve cryptography over binary fields," *CHES 2000, Lecture Notes in Computer Science*, vol. 1965, pp. 1–24 , 2000.

[21] N. Koblitz, A. Menezes, and S. Vanstone, "The state of elliptic curve cryptography," *Design, Codes and Cryptography*, vol. 19, pp. 173–193, 2000.

[22] P. Gaudry, F. Hess, and N.P. Smart, "Constructive and destructive facets of Weil descent on elliptic curves," *J. of Cryptology*, vol. 15, pp. 19–46, 2002.

[23] A. Miyaji, M. Nakabayashi, and S. Takano, "New explicit conditions of elliptic curve trace for FR-reduction," *IEICE Trans. Fundamentals*, vol. E84 A, pp. 1–10, 2001.

[24] R. Sakai, K. Ohgishi, and M. Kasahara, "Cryptosystems based on pairing," *SICS 2000, Symposium on Cryptography and Information Security*, pp. 26–28, 2000.

[25] N.P. Smart, "An identity based authentication key agreement protocol based on pairing," *Electronics Letters*, vol. 38, pp. 630–632, 2002.

[26] S. Gao, J. von zur Gathen, and D. Panario, "Gauss periods and fast exponentiation in finite fields," *Latin 1995, Lecture Notes in Computer Science*, vol. 911, pp. 311–322, 1995.