Speeding up Ate Pairing Computation in Affine Coordinates*

Duc-Phong Le and Chik How Tan

Temasek Laboratories, National University of Singapore,
5A Engineering Drive 1, #09-02, Singapore 117411.
(tslld,tsltch)@nus.edu.sg

Abstract. At Pairing 2010, Lauter et al's analysis showed that Ate pairing computation in affine coordinates may be much faster than projective coordinates at high security levels. In this paper, we further investigate techniques to speed up Ate pairing computation in affine coordinates. On the one hand, we improve Ate pairing computation over elliptic curves admitting an even twist by describing an 4-ary Miller algorithm in affine coordinates. This technique allows us to trade one multiplication in the full extension field and one field inversion for several multiplications in a smaller field. On the other hand, we investigate pairing computations over elliptic curves admitting a twist of degree 3. We propose new fast explicit formulas for Miller function that are comparable to formulas over even twisted curves. We further analyze pairing computation on cubic twisted curves by proposing efficient subfamilies of pairing-friendly elliptic curves with embedding degrees $k = 9$, and 15. These subfamilies allow us not only to obtain a very simple form of curve, but also lead to an efficient arithmetic and final exponentiation.

Keywords: Ate pairing, Pairing computation, final exponentiation, affine coordinates, cubic twisted curves, pairing-friendly elliptic curves.

1 Introduction

In recent years, the pairings have become extremely useful in public-key cryptography. Pairings used in cryptography are efficiently computable bilinear maps on torsion subgroups of points on a (hyper-)elliptic curve that map into the multiplicative group of a finite field. We call such a map a *cryptographic pairing*. Let $\mathbb{G}_1, \mathbb{G}_2$ be finite abelian groups written additively, and let $\mathbb{G}_3$ be a finite abelian group written multiplicatively. A cryptographic pairing is a map:

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

The first pairing application to cryptography was introduced in Joux' seminal paper [17] describing a one-round tripartite Diffie-Hellman key exchange protocol in 2000. Since then, the use of cryptographic protocols based on pairings has had a huge success with some notable breakthroughs such as practical Identity-based Encryption (IBE) schemes [7], and many other new cryptographic primitives.

Due to the high cost of pairing operations, the efficiency of pairing computation and the construction of pairing-friendly curves have become an active field of research. The former concerns many techniques having been exploited to dramatically improve the performance of the Miller algorithm, see [2][3][20][8][24]. The later focuses on constructing curves that are suitable for pairing-based cryptosystems. Whereas standard elliptic curve cryptography can be implemented using randomly generated elliptic curves, the elliptic curves required to implement pairing-based protocols must have a small embedding degree such that pairings can be efficiently computed in extension finite fields. Many works on constructing pairing-friendly elliptic curves have been presented in [28][9][4] and this research is collected and extended in the recent paper [13].

* In this version, we correct the computation of the final exponentiation for cubic twisted curves in [25]. We would like to thank Emmanuel Fouotsa for his comments.

Projective coordinates are usually preferred than affine coordinates for implementing pairings. That is because point addition or doubling operations in affine coordinates involve a field modular inversion that is much expensive than one field multiplication in the base field $\mathbb{F}_p$. However, recent analysis in [22] showed that over $\mathbb{F}_{p^d}$, for larger d, the inversion-to-multiplication ratio is significant reduced. Ate pairing computation in affine coordinates is thus much faster than that in projective coordinates at high security levels.

This work presents our optimizations to Miller loop using a 4-ary algorithm with direct formulas to compute quadrupling of points and a multiplication of two line functions in affine coordinates. Our techniques make a trade-off between one multiplication in the full extension field $\mathbb{F}_{p^k}$, one inversion in the subfield $\mathbb{F}_{p^e}$ for some multiplications in $\mathbb{F}_{p^e}$, where k is the embedding degree of the elliptic curve E over the finite field $\mathbb{F}_p$, $e = k/d$, and d is the degree of the twist admitted during pairing computation.

This work also focuses on pairing computations over pairing-friendly elliptic curves admitting a cubic twist. Although, such a curve doesn't provide a full denominators elimination technique, but it allows a shorter Miller loop. We first present new fast formulas in affine coordinates for doubling/addition steps of Miller's algorithm over cubic twisted curves. Then, we give a finer choice for curves of embedding degrees $k = 9, 15$. By carefully choosing parameters, we point out that the desired curve is always of form $y^2 = x^3 + 1$. Finally, we present improvements for the hard part in final exponentiation for such curves.

The rest of the paper is organized as follows: Section 2 briefly recalls some basic knowledges about Ate pairing and its computation. Section 3 presents our improvements for the curves with even twisted degree. Section 4 presents new explicit formulas to speed up pairing computation on curves with cubic twisted degree. We conclude in Section 5.

2 Background on Pairings

For p prime and $p > 3$, an elliptic curve defined over a finite field $\mathbb{F}_p$ in short Weierstrass form is the set of solutions (x, y) to the equation

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

where $a, b \in \mathbb{F}_p$ such that the discriminant $\Delta = 4a^3 + 27b^2$ is non-zero. We denote by $\mathcal{O}$ the point at infinity on E, and by $\#E(\mathbb{F}_p)$ the number of points on E defined over $\mathbb{F}_p$. We have $n = \#E(\mathbb{F}_p) = p + 1 - t$, where t is the trace of Frobenius, which satisfies $|t| \leq 2\sqrt{p}$ (Hasse's theorem). Let r be a prime number that divides the number of points n and is co-prime to the characteristic p. Let k be the embedding degree of the elliptic curve E with respect to r, i.e., k the smallest positive integer such that $r|p^k - 1$. By this setting, we can define subgroups of points of prime order r on E and a multiplicative group of order r in the extension field $\mathbb{F}^*_{p^k} = \mathbb{F}_{p^k} \backslash \{0\}$, i.e., $\mathbb{F}^*_{p^k}$ contains the group μ_r of r-roots of unity.

2.1 The Ate pairing

We denote subgroups of points of prime order r on $E(\mathbb{F}_{p^k})$ by $E[r]$. Let $m \in \mathbb{Z}$, $P \in E[r]$ and $f_{m,P}$ be a rational function on E with divisor $div(f_{m,P}) = m(P) - (mP) - (m-1)(\mathcal{O})$. Let π_p be the p-power Frobenius endomorphism on E, $\pi_p : E \to E$ given by $\pi_p(x, y) = (x^p, y^p)$. Let $T = t - 1$. We denote by $\mathbb{G}_1 = E[r] \cap Ker(\pi_p - [1]) = E(\mathbb{F}_p)[r]$, $\mathbb{G}_2 = E[r] \cap Ker(\pi_p - [p]) \subseteq E(\mathbb{F}_{p^k})[r]$. For $Q \in \mathbb{G}_2$ and $P \in \mathbb{G}_1$, the Ate pairing is defined as [16] (so with the arguments swapped in comparison to Tate pairing):

$$a_T = \mathbb{G}_2 \times \mathbb{G}_1 \to \mu_r, \qquad (Q, P) \mapsto f_{T,Q}(P)^{(p^k-1)/r}. \tag{1}$$

The length of Miller loop during Ate pairing computation is determined by the trace of Frobenius t. The Ate pairing is thus particularly suitable for pairing-friendly elliptic curves with small values of t. Usually, when implementing Tate pairing and its variants, instead of inputing the point Q on the curve $\mathbb{G}_2 \subseteq E(\mathbb{F}_{p^k})[r]$, one can take $Q' \in \mathbb{G}'_2 \subseteq E'(\mathbb{F}_{p^{k/d}})[r]$, where E' is a twist of E, and $d|k$ is the degree of the twist. Points on the twisted curve are defined over a smaller field, and are thus obviously much faster for computation.

Let $\psi : E' \to E, Q' \mapsto Q$ be an isomorphism mapping points of the twisted curve to that of the original curve. The computation of $a_T(\psi(Q'), P)$ consists of two parts: evaluation of the function $f_{T,Q}$ at P and final exponentiation ensuring a unique result of the pairing. The first part is computed using Miller's algorithm [27] that is described in Algorithm 1.

Input: $T = \sum_{i=0}^{l-1} t_i 2^i$ (radix 2), $t_i \in \{0, 1\}$, $Q' \in \mathbb{G}'_2$ not a multiple of $P \in \mathbb{G}_1$.
Output: $f_{T,\psi(Q')}(P)$ representing a class in $\mathbb{F}^*_{p^k}/(\mathbb{F}^*_{p^k})^r$
$R' \leftarrow Q'$, $f \leftarrow 1$, ;
for $i = l - 2$ **to** 0 **do**
    $f \leftarrow f^2 g_{\psi(R'),\psi(R')}(P)$, $R' \leftarrow [2]R'$;
    if $r_i = 1$ **then**
        $f \leftarrow f g_{\psi(R'),\psi(Q')}(P)$, $R' \leftarrow R' + Q'$;
end
return f

Algorithm 1: Miller's algorithm for Ate-like pairings

3 Improvements for the even twisted curves

Pairing-friendly elliptic curves with an even embedding degree k are preferred in implementing Tate pairing and its variants. That is because the denominators elimination techniques can be used (see [2][14]) on such curves. Furthermore, such curves can admit a high-degree twist, e.g., twists of degree 4 or 6 such that the points on the twisted curve E' can be represented in a smaller finite field.

Lauter *et al.* analyzed the costs of Miller's algorithm in affine coordinates over curves with even embedding degrees [22, Table 1, 2]. They pointed out that when implementing one of the optimal Ate pairings [35] in high security levels, affine coordinates could be much faster than projective coordinates. This is because the ratio of the computational costs of inversions to multiplications for point doubling/addition operations is drastically reduced in extension fields.

3.1 4-ary Miller algorithm

In this subsection, we present our optimizations of Miller loop using a 4-ary algorithm with direct formulas in affine coordinates. Usually, Miller's algorithm computes pairings using the *double-and-add* method. In [6], Blake *et al.* present the idea to compute the pairing using a 4-ary algorithm for the purpose of elimination vertical lines (i.e., denominators) in Miller's algorithm. Their algorithm can be applied on any curves (i.e., curves don't admit a twist and thus there isn't any denominators elimination technique), and has advantage if the binary expansion of the trace t has many zeros. Costello *et al.* [10] also addressed this problem by introducing a new algorithm so-called the Miller 2^n-tuple-and-add algorithm. They also presented explicit formulas in projective coordinates for cases of $n = 2, 3$.

Direct computation of $\ell_{R,R} \times \ell_{[2]R,[2]R}$: We assume that E', twisted curve of E defined in § 2, is given by an equation $E' : y^2 = x^3 + (a/\alpha^4)x + (b/\alpha^6)$ for some $\alpha \in \mathbb{F}_{p^k}$ with an isomorphism $\psi : E' \to E,\ (x, y) \mapsto$

$(\alpha^2 x, \alpha^3 y)$. Furthermore, we assume that $\mathbb{F}_{p^k} = \mathbb{F}_{p^e}(\alpha)$, and we have $\alpha^d = \omega \in \mathbb{F}_{p^e}$, where d is the degree of the twist. Each element in $\mathbb{F}_{p^k}$ is given by a polynomial of degree $d-1$ in α with coefficients in $\mathbb{F}_{p^e}$.

Let $P \in E(\mathbb{F}_p)$, $R' \in E'(\mathbb{F}_{p^e})$ and $R = \psi(R')$. Let $R_3 = [2]R = (x_{R_3}, y_{R_3})$. Let $\ell_1 = \ell_{R,R}(P)$, and $\ell_2 = \ell_{R_3,R_3}(P)$. In the following computation, we use the abscissas of the point $-R_3$ instead of that of the point R in the line function ℓ_1 passing points R and $-R_3$. We also compute $\frac{\ell_1 \cdot \ell_2}{x_P - x_{R_3}}$ instead of $\ell_1 \cdot \ell_2$. Note that, for even twisted curves, the factor $x_P - x_{R_3}$ is in the proper subfield, thus we can make this division without changing the final result of Tate pairing. Two consecutive doubling steps are performed as follows:

$$\begin{aligned}
\ell = \frac{\ell_1 \cdot \ell_2}{x_P - x_{R_3}} &= \frac{(y_P + y_{R_3} - \lambda_1(x_P - x_{R_3}))(y_P - y_{R_3} - \lambda_2(x_P - x_{R_3}))}{x_P - x_{R_3}} \\
&= \frac{y_P^2 - y_{R_3}^2}{x_P - x_{R_3}} - \lambda_1(y_P - y_{R_3}) - \lambda_2(y_P + y_{R_3}) + \lambda_1\lambda_2(x_P - x_{R_3}) \\
&= x_P^2 + x_P x_{R_3} + x_{R_3}^2 + a - \lambda_1(y_P - y_{R_3}) - \lambda_2(y_P + y_{R_3}) + \lambda_1\lambda_2(x_P - x_{R_3}),
\end{aligned}$$

where λ_1, λ_2 are slopes when computing $[2]R$ and $[4]R$. Let $R'_3 = [2]R' = (x_{R'_3}, y_{R'_3})$. The details of computations is as follows:

$$\begin{aligned}
\ell = \ell_1 \cdot \ell_2 = \ell_{\psi(R'),\psi(R')}(P) \cdot \ell_{\psi([2]R'),\psi([2]R')}(P) &= x_P^2 + \alpha^2 x_{R'_3} x_P + \alpha^4 x_{R'_3}^2 \\
&+ a - \alpha\lambda'_1(y_P - \alpha^3 y_{R'_3}) - \alpha\lambda'_2(y_P + \alpha^3 y_{R'_3}) + \alpha^2\lambda'_1\lambda'_2(x_P - \alpha^2 x_{R'_3}) \\
= x_P^2 + a - \alpha(\lambda'_1 + \lambda'_2)y_P + \alpha^2(x_{R'_3} + \lambda'_1\lambda'_2)x_P &+ \alpha^4(x_{R'_3}^2 + (\lambda'_1 + \lambda'_2)y_{R'_3} + \lambda'_1\lambda'_2 x_{R'_3}),
\end{aligned}$$

where $\lambda_1 = \alpha\lambda'_1$, and $\lambda_2 = \alpha\lambda'_2$. Since P is fixed throughout the computation, we assume that value x_P^2 is precomputed, the costs of updating ℓ are summarized in the following table. Note that, we use the same notations for field arithmetic costs as in [22]. Notations $\mathbf{I}_{p^e}$, $\mathbf{M}_{p^e}$, $\mathbf{S}_{p^e}$, $\mathbf{add}_{p^e}$, $\mathbf{sub}_{p^e}$, $\mathbf{neg}_{p^e}$ denote the costs for inversion, multiplication, squaring, addition, subtraction, and negation in the field $\mathbb{F}_{p^e}$, where $e = k/d$. The cost for a multiplication by a constant ω is denoted by $\mathbf{M}_{(\omega)}$.

Table 1. Number of operations for updating two consecutive line function values

	$\mathbf{M}_p$	$\mathbf{M}_{p^e}$	$\mathbf{S}_{p^e}$	$\mathbf{M}_{(\omega)}$	$\mathbf{add}_{p^e}$	$\mathbf{neg}_{p^e}$
$d=2$	k	3	1	2	6	1
$d=4$	$k/2$	3	1	1	5	1
$d=6$	$k/3$	3	1	-	4	1

Fast quadrupling. Let $R'_4 = [4]R' = (x_{R'_4}, y_{R'_4})$. Traditionally, R'_4 can be obtained using two repeated doublings that require 2 field inversions. In [23], Le introduced fast algorithms for quadrupling a point on elliptic curves in affine coordinates. His algorithm requires $1\mathbf{I}_{p^e} + 8\mathbf{S}_{p^e} + 8\mathbf{M}_{p^e}$, and is better than two repeated doublings whenever $\mathbf{I}_{p^e} > 4\mathbf{M}_{p^e} + 4\mathbf{S}_{p^e}$. It performs even better for curves that allow "$a = 0$" speedup (found in pairing-friendly elliptic curves admitting twists of degrees 2, 3, or 6) as $[4]R'$ in affine coordinates can be computed just using only $1\mathbf{I}_{p^e} + 5\mathbf{S}_{p^e} + 6\mathbf{M}_{p^e}$. This section presents the revised formula for point quadrupling that requires fewer additions in comparison to that in [23] for pairing computation over curves with $a = 0$. Let $d = y_{R'}^4 + 18by_{R'}^2 - 27b^2$, $I = \frac{3x_{R'}^2}{2y_{R'}d}$, and ℓ is the product of two consecutive line function values as described above. One also can precompute and cache values $s = 18b$ and $t = 27b^2$.

This quadrupling formula only requires $1\mathbf{I}_{p^e} + 6\mathbf{M}_{p^e} + 5\mathbf{S}_{p^e} + e\mathbf{M}_p + 8\mathbf{add}_{p^e} + 9\mathbf{sub}_{p^e}$. If an inversion in $\mathbb{F}_{p^e}$ is more than $2\mathbf{M}_{p^e} + 1\mathbf{S}_{p^e} + e\mathbf{M}_p + 1\mathbf{sub}_{p^e}$, then the new quadrupling formula is faster than two

$$\begin{array}{c}\lambda_1 = I \cdot d, \quad \lambda_2 = I(y_R^2 - 9b)^2/2 \\ x_{R'_3} = \lambda_1^2 - 2x_R, \quad y_{R'_3} = \lambda_1(x_R - x_{R'_3}) - y_R, \\ x_{R'_4} = \lambda_2^2 - 2x_{R'_3}, \quad y_{R'_4} = \lambda_2(x_{R'_3} - x_{R'_4}) - y_{R'_4}, \\ \ell = \ell_1 \cdot \ell_2 = \ell_{\psi(R'),\psi(R')}(P) \cdot \ell_{\psi(R'_3),\psi(R'_3)}(P),\end{array}$$

$$\begin{array}{c}A = y_{R'}^2, \quad B = A^2, \quad C = 3x_{R'}^2, \quad d = B + sA + t, \\ D = 2dy_{R'}, \quad E = D^{-1}, \quad I = C \cdot E, \quad \lambda_1 = I \cdot d, \\ x_{R'_3} = \lambda_1^2 - 2x_{R'}, \quad y_{R'_3} = \lambda_1(x_{R'} - x_{R'_3}) - y_{R'}, \quad \lambda_2 = \frac{(B - sA + 3t)I}{2}, \\ x_{R'_4} = \lambda_2^2 - 2x_{R'_3}, \quad y_{R'_4} = \lambda_2(x_{R'_3} - x_{R'_4}) - y_{R'_3}, \\ \ell = \ell_1 \cdot \ell_2 = \ell_{\psi(R'),\psi(R')}(P) \cdot \ell_{\psi(R'_3),\psi(R'_3)}(P)\end{array}$$

repeated doublings. In the case of curves with a twist of degree 4 (i.e., $y^2 = x^3 + ax$), a similar quadrupling can be performed by $1\mathbf{I}_{p^e} + 9\mathbf{M}_{p^e} + 5\mathbf{S}_{p^e} + 14\mathbf{add}_{p^e} + 10\mathbf{sub}_{p^e}$.

Table 2 summarizes and compares the costs of our technique to those from [22] in affine coordinates and [10] in projective coordinates. Again, we assume that all values that depend only on fixed parameters, are precomputed and cached, and small multiples are computed by additions.

Table 2. Operation counts for two doubling steps in the Ate-like Miller loop

d	Technique	$\mathbf{M}_p$	$\mathbf{M}_{p^k}$	$\mathbf{I}_{p^e}$	$\mathbf{M}_{p^e}$	$\mathbf{S}_{p^e}$	$\mathbf{M}_{(\cdot)}$	$\mathbf{add}_{p^e}$	$\mathbf{sub}_{p^e}$	$\mathbf{neg}_{p^e}$
2	Ours	$5k/2$	1	1	9	6	2	14	9	1
$(a = 0)$	Lauter *et al.* [22]	k	2	2	6	4	2	8	12	-
	Costello *et al.* [10]	$2k$	1	-	14	16	4	60	24	2
4	Ours	k	1	1	12	6	1	19	10	1
$(b = 0)$	Lauter *et al.* [22]	$k/2$	2	2	6	4	-	8	10	2
	Costello *et al.* [10]	k	1	-	11	20	3	55	27	2
6	Ours	$5k/6$	1	1	9	6	-	12	9	1
$(a = 0)$	Lauter *et al.* [22]	$k/3$	2	2	6	4	-	8	10	2
	Costello *et al.* [10]	$2k/3$	1	-	14	16	4	60	24	2

As showed in Table 2, the costs of two doubling steps on curves having a twist of degree $d = 2$ requires $\frac{5k}{2}\mathbf{M}_p + 1\mathbf{M}_{p^k} + 1\mathbf{I}_{p^{k/2}} + 9\mathbf{M}_{p^{k/2}} + 6\mathbf{S}_{p^{k/2}} + 2\mathbf{M}_{(\omega)} + 14\mathbf{add}_{p^{k/2}} + 9\mathbf{sub}_{p^{k/2}} + 1\mathbf{neg}_{p^{k/2}}$, while analysis in [22] require $k\mathbf{M}_p + 2\mathbf{M}_{p^k} + 2\mathbf{I}_{p^{k/2}} + 6\mathbf{M}_{p^{k/2}} + 4\mathbf{S}_{p^{k/2}} + 2\mathbf{M}_{(\omega)} + 8\mathbf{add}_{p^{k/2}} + 12\mathbf{sub}_{p^{k/2}}$. If $1\mathbf{M}_{p^k} + 1\mathbf{I}_{p^{k/2}} > 3\mathbf{M}_{p^{k/2}} + 2\mathbf{S}_{p^{k/2}} + 3\mathbf{add}_{p^{k/2}} + \mathbf{neg}_{p^{k/2}}$, then our technique is better.

4 Improvements for the cubic twisted curves

4.1 Updating Miller function

Pairing computation over cubic twisted curves with embedding degrees 9 or 15 were investigated in papers [26][12][11]. Although such curves only admit a cubic twist $d = 3$, and there exists no full denominators elimination technique, but they provide a shorter Miller loop. In [31], Scott pointed out that in the contexts of multi-pairings in conjunction with fixed arguments, these curves have more advantages than curves admitting a higher twist (i.e., 4 or 6). This section gives the first analysis about the costs of Miller's algorithm in such curves in affine coordinates.

Recall that cubic twisted curves have the form $y^2 = x^3 + b$. In [26], Lin *et al.* proposed a denominators elimination trick during Ate pairing computation on a $k = 9$ curve due to the following observation about the factor $1/v_{R+Q}(P)$:

$$\frac{1}{v_{R+Q}(P)} = \frac{1}{x_P - x_{R+Q}} = \frac{x_P^2 + x_P x_{R+Q} + x_{R+Q}^2}{(y_P - y_{R+Q})(y_P + y_{R+Q})}$$

Since $(y_P - y_{R+Q})(y_P + y_{R+Q})$ lies in a subfield when the curve admits a cubic twist, f function can be updated by multiplying by $x_P^2 + x_P x_{R+Q} + x_{R+Q}^2$ instead of dividing it by $v_{R+Q}(P)$. The updated factor is :

$$\ell'_{R,Q}(P) = (y_P - \lambda(x_P - x_{R+Q}) - y_{R+Q}) \cdot (x_P^2 + x_P x_{R+Q} + x_{R+Q}^2) \tag{2}$$

where λ is the slope of the line function passing points R and Q. This formula needs one full extension field multiplication than the full denominators elimination technique of Barreto et al [2]. The following lemma allows us to save one multiplication in the full extension field in comparison to the analysis in [26].

Lemma 1. *For elliptic curves admitting a cubic twist, the rational function $g_{R,Q}(P)$ in Miller's algorithm can be rewritten as follows:*

$$g_{R,Q}(P) = \frac{\ell_{R,Q}(P)}{v_{R+Q}(P)} = \frac{x_{R+Q}^2 + x_{R+Q}x_P + x_P^2 - \lambda(y_P + y_{R+Q})}{y_P - y_{R+Q}} \tag{3}$$

Proof. For the line function $\ell_{R,Q}(P)$, using the coordinates of the point $-(R+Q)$ instead of that of R, we have:

$$\begin{aligned}
\frac{\ell_{R,Q}(P)}{v_{R+Q}(P)} &= \frac{y_P - \lambda(x_P - x_{R+Q}) + y_{R+Q}}{x_P - x_{R+Q}} \\
&= -\lambda + \frac{y_P^2 - y_{R+Q}^2}{(x_P - x_{R+Q})(y_P - y_{R+Q})} = -\lambda + \frac{x_P^3 - x_{R+Q}^3}{(x_P - x_{R+Q})(y_P - y_{R+Q})} \\
&= \frac{-\lambda(y_P - y_{R+Q}) + x_{R+Q}^2 + x_{R+Q}x_P + x_P^2}{y_P - y_{R+Q}}
\end{aligned}$$

The factor $(y_P - y_{R+Q})$ lying in a proper subfield of $\mathbb{F}_{p^k}$ can be cancelled out. The actual updated factor is $x_{R+Q}^2 + x_{R+Q}x_P + x_P^2 - \lambda(y_P - y_{R+Q})$. The computation of this updated factor doesn't require one more multiplication in the full extension field and it is much faster than that given in [26].

Doubling step. Let the notations be described in Section 3. Let $e = k/3$, and let $\nu \in \mathbb{F}_{p^k}$ be not a cubic residue but a quadratic residue over $\mathbb{F}_{p^e}$, and $\nu^{1/2} = \omega \in \mathbb{F}_{p^e}$. Furthermore, we assume that $\mathbb{F}_{p^k} = \mathbb{F}_{p^e}(\nu^{1/6})$, i.e., each element in $\mathbb{F}_{p^k}$ can be represented by a polynomial $A + B\nu^{1/6} + C\nu^{1/3}$, where $A, B, C \in \mathbb{Z}_{p^e}$. Let the twisted curve is of the form $E' : y^2 = x^3 + b/\nu$. There exists an isomorphism $\psi : E'(\mathbb{F}_{p^e}) \to E(\mathbb{F}_{p^k})$, $(x, y) \mapsto (\nu^{1/3}x, \nu^{1/2}y)$. Let $P \in \mathbb{G}_1$, $R', Q' \in \mathbb{G}'_2$, and let $R = \psi(R')$, $Q = \psi(Q')$, where $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}'_2$ are defined as in Section 2.1.

As showed in Lemma 1, the computation of the line functions need squarings of x-coordinates. This implies that a new coordinate (x, y, z), where $z = x^2$ matches the computation. Let $R'_3 = [2]R'$. Doubling steps can be computed as follows:

$$\begin{aligned}
\ell_{R,R}(P) &= \nu^{2/3}x_{R'_3}^2 + \nu^{1/3}x_{R'_3}x_P + x_P^2 - \lambda(y_P - \nu^{1/2}y_{R'_3}) \\
&= z_P - \nu^{1/6}\lambda' y_p + \nu^{1/3}x_{R'_3}x_P + \nu^{2/3}(z_{R'_3} + \lambda' y_{R'_3}) \\
&= z_P + \nu^{1/6}(\omega(z_{R'_3} + \lambda' y_{R'_3}) - \lambda' y_p) + \nu^{1/3}x_{R'_3}x_P,
\end{aligned}$$

where $x_{R'_3} = \lambda'^2 - 2x_{R'}$, $y_{R'_3} = \lambda'(x_{R'} - x_{R'_3}) - y_{R'}$ and $z_{R'_3} = x^2_{R'_3}$. We have $\lambda' = 3x^2_{R'}/2y_{R'} = 3z_{R'}/2y_{R'}$ and $\lambda = 3x^2_R/2y_R = \nu^{1/6}\lambda'$.

The double of R' needs $\mathbf{I}_{p^e} + 2\mathbf{M}_{p^e} + 2\mathbf{S}_{p^e} + 3\mathbf{add}_{p^e} + 4\mathbf{sub}_{p^e}$, where the computation of the slope λ' need $\mathbf{I}_{p^e} + \mathbf{M}_{p^e} + 3\mathbf{add}_{p^e}$. Assume that the multiplication of elements in $\mathbb{F}_{p^e}$ with a small constant (e.g., $3z_{R'}$, $2y_{R'}$) is computed by additions. Then, we need $2e\mathbf{M}_p + \mathbf{M}_{p^e} + \mathbf{M}_{(\omega)} + \mathbf{add}_{p^e} + \mathbf{sub}_{p^e}$ to compute the line function value. In total, our new formula requires $2e\mathbf{M}_p + \mathbf{I}_{p^e} + 3\mathbf{M}_{p^e} + 2\mathbf{S}_{p^e} + \mathbf{M}_{(\omega)} + 4\mathbf{add}_{p^e} + 5\mathbf{sub}_{p^e}$ for each doubling step.

Addition step. The line function is computed similarly as in doubling steps.

$$\ell_{R,R}(P) = z_P + \nu^{1/6}(\omega(z_{R'_3} + \lambda' y_{R'_3}) - \lambda' y_p) + \nu^{1/3} x_{R'_3} x_P,$$

where $R'_3 = R' + Q'$, and $x_{R'_3} = \lambda'^2 - x_{R'} - x_{Q'}$, $y_{R'_3} = \lambda'(x_{R'} - x_{R'_3}) - y_{R'}$ and $z_{R'_3} = x^2_{R'_3}$. The slope $\lambda' = (y_{R'} - y_{Q'})/(x_{R'} - x_{Q'})$. We have $\lambda = (y_R - y_Q)/(x_R - x_Q) = \nu^{1/6}\lambda'$.

Computation of the line function in addition steps has the same cost as in the doubling steps. It needs $\mathbf{I}_{p^e} + \mathbf{M}_{p^e} + 2\mathbf{sub}_{p^e}$ for computing the slope λ' and $\mathbf{M}_{p^e} + 2\mathbf{S}_{p^e} + 4\mathbf{sub}_{p^e}$ for computing the addition of R' and Q' from the slope λ'. In total, we need $2e\mathbf{M}_p + \mathbf{I}_{p^e} + 3\mathbf{M}_{p^e} + 2\mathbf{S}_{p^e} + \mathbf{M}_{(\omega)} + \mathbf{add}_{p^e} + 7\mathbf{sub}_{p^e}$ for each addition step.

We summarize the number of operations required by the Miller loop over cubic twisted curves in Table 3. We also make a comparison on the number of operations between affine coordinates and projective coordinates taken from [11].

Table 3. Number of operations in the Ate-like Miller loop over cubic twisted curves

	coord.	$\mathbf{M}_p$	$\mathbf{I}_{p^e}$	$\mathbf{M}_{p^e}$	$\mathbf{S}_{p^e}$	$\mathbf{M}_{(\cdot)}$	$\mathbf{add}_{p^e}$	$\mathbf{sub}_{p^e}$	$\mathbf{neg}_{p^e}$
DBL	affine	$2k/3$	1	3	2	$\mathbf{M}_{(\omega)}$	4	5	-
	proj. [11]	k	-	6	7	$\mathbf{M}_{(b/\omega)}$	11	10	1
ADD	affine	$2k/3$	1	3	2	$\mathbf{M}_{(\omega)}$	1	7	-
	proj. [11]	k	-	13	3	-	6	8	3

The above analysis showed that the number of operations in doubling steps over cubic twisted curves is similar to that over even twisted curves as analyzed in [22]. Addition steps require only $1\mathbf{S}_{p^e} + 1\mathbf{M}_{(\omega)}$ more than that for even twisted curves. Table 3 also showed that the doubling steps in affine coordinates are better than that in projective coordinates [11] if:

$$\mathbf{I}_{p^e} \leq e\mathbf{M}_p + 3\mathbf{M}_{p^e} + 5\mathbf{S}_{p^e} + 7\mathbf{add}_{p^e} + 5\mathbf{sub}_{p^e} + \mathbf{neg}_{p^e}, \tag{4}$$

where $e = k/3$.

Example 1. In the case of $k = 9$, we can obtain pairing-friendly elliptic curves of form $y^2 = x^3 + b$ admitting a cubic twist [26]. During Ate pairing computation, point operations are performed over $\mathbb{F}_{p^3}$ (i.e., $e = 3$). Analysis in [21, §5.1] showed that inversion over $\mathbb{F}_{p^3}$ needs 12 multiplications and one inversion over $\mathbb{F}_p$. If the inversion-to-multiplication ratio is around 13 as benchmarks in [22] and is used in this analysis, the cost of one inversion over $\mathbb{F}_{p^3}$ is around 25 multiplications over $\mathbb{F}_p$. Obviously, this cost is much less than $3\mathbf{M}_p + 3\mathbf{M}_{p^3} + 5\mathbf{S}_{p^3} \approx 21\mathbf{M}_p + 30\mathbf{S}_p$ (from Eq. 4). Note that using Karatsuba algorithm, $\mathbf{M}_{p^3} \approx 6\mathbf{M}_p$ and $\mathbf{S}_{p^3} \approx 6\mathbf{S}_p$.

4.2 Choice of curves

In this section, we present efficient subfamilies of pairing-friendly elliptic curves with embedding degrees $k = 9, 15$ presented in [26] and [12].

The family of curves with $k = 9$ is described by the following polynomials:

$$\begin{aligned} p(x) &= ((x+1)^2 + ((x-1)^2(2x^3+1)^2)/3)/4, \\ r(x) &= (x^6 + x^3 + 1)/3, \\ n(x) &= (x-1)^2(x^6 + x^3 + 1)/3, \\ t(x) &= x + 1, \end{aligned} \tag{5}$$

where $t(x)$ is the trace of Frobenius, $p(x)$ represents the field size and $r(x)$ represents the pairing-friendly subgroup. In comparison to BN curves at 128-bit security level [4], this family supports a shorter Miller loop. But, BN curves provide a much more efficient tower extension field arithmetic.

El Mrabet *et al.* [12] introduced a family of pairing-friendly elliptic curve of embedding degree $k = 15$ and compared its performance with KSS curves [18] at 192-bit security level. Their family of curves is described as follows:

$$\begin{aligned} p(x) &= (x^{12} - 2x^{11} + x^{10} + x^7 - 2x^6 + x^5 + x^2 + x + 1)/3, \\ r(x) &= x^8 - x^7 + x^5 - x^4 + x^3 - x + 1, \\ n(x) &= (x-1)^2(x^2 + x + 1)(x^8 - x^7 + x^5 - x^4 + x^3 - x + 1)/3, \\ t(x) &= x + 1. \end{aligned} \tag{6}$$

For both families of curves, the ρ-value is equal to $4/3$ and the elliptic curves are of the form $y^2 = x^3 + b$. By using the above parameters when $x_0 \equiv 1 \pmod 3$, one is able to get all involved parameters being integers and construct a curve. The following theorem show that by choosing $x \equiv 1 \pmod 6$, we always choose the curve constant b equal to 1. That means that the multiplications with b is free.

Lemma 2. *Let $E : y^2 = x^3 + b$ be an elliptic curve defined over $\mathbb{F}_p$ where p prime and $p \equiv 1 \pmod 6$. Let $\#E(\mathbb{F}_p) = n$. If $2 \mid n$ and $3 \mid n$, then b is both a square and a cube in $\mathbb{F}_p$.*

Proof. Assume that $2 \mid n$, then the elliptic curve $E : y^2 = x^3 + b$ contains points of order 2. Let $P = (x_1, y_1) \in E$ be a point having order 2. The tangent at P meets $\mathcal{O}$ and hence $\left(\frac{dy}{dx}\right)_P = \infty$ or $y_1 = 0$. We have $y_1^2 = x_1^3 + b$, and hence $b = -x_1^3$ or b is a cube in $\mathbb{F}_p$.

When $3 \mid n$, E contains points of order 3. Assume that $P = (x_1, y_1)$ has order 3, that means $3P = \mathcal{O}$ or $2P = -P$. Let $Q = [2]P = (x_2, y_2)$. Then we have $x_2 = x_1$ which implies $\lambda^2 - 2x_1 = x_1,$ where $\lambda = 3x_1^2/2y_1$. Therefore, we obtain $9x_1(y_1^2 - b) = 12x_1y_1^2$, so that $x_1 = 0$ or $y_1^2 = -3b$.

In the former case $x_1 = 0$, it is easy to verify that the point $0, \delta$ has order 3 for some δ, and $b = \delta^2$ or b is a square in $\mathbb{F}_p$. For the later case $-3b = y_1^2$, to prove b square in $\mathbb{F}_p$ we need to show that -3 is a square in $\mathbb{F}_p$. We consider the Legendre symbol:

$$\left(\frac{-3}{p}\right) = \left(\frac{-1}{p}\right)\left(\frac{3}{p}\right) = (-1)^{\frac{p-1}{2}} \times (-1)^{\lfloor \frac{p+1}{6} \rfloor}.$$

– If $p \equiv 1 \pmod{12}$, we have

$$\left(\frac{-3}{p}\right) = 1 \times 1 = 1.$$

– If $p \equiv 7 \pmod{12}$, we have

$$\left(\frac{-3}{p}\right) = (-1) \times (-1) = 1.$$

In the other words, -3 is a square in $\mathbb{F}_p$.

Theorem 1. *By choosing $x_0 \equiv 1 \pmod 6$ for both above families of curves with embedding degrees $k = 9$ or 15, the desired curve is always of form $E(\mathbb{F}_p) : y^2 = x^3 + 1$.*

Proof. In [34, §X.5], Silverman showed that an curve defined over $\mathbb{F}_p$ with the j *invariant* $j(E) = 0$ (i.e. the curves of the form $y^2 = x^3+b$) will only have six possible curve orders. More precisely, the CM construction only ensures that the order of a curve satisfying the norm equation $3y^2 = 4p - t^2$ has one of the six forms $\{p + 1 \pm t, p + 1 \pm (t \pm 3y)/2\}$. Moreover, assume that γ be both quadratic and cubic non-residue modulo p, these possible group orders occur as the order of one of the 6 twists with $b \in \{1, \gamma, \gamma^2, \gamma^3, \gamma^4, \gamma^5\}$.

For $x_0 \equiv 1 \pmod 6$ in (5) (and (6), resp.), is is easy to see that $n_0 = (x_0 - 1)^2(x_0^6 + x_0^3 + 1)/3$ ($n_0 = (x_0 - 1)^2(x_0^2 + x_0 + 1)(x_0^8 - x_0^7 + x_0^5 - x_0^4 + x_0^3 - x_0 + 1)/3$, resp.) is congruent to 0 modulo 6, i.e., $2|n_0$ and $3|n_0$. It is also easy to verify that $p(x_0) \equiv 1 \pmod 6$. From Lemma 2, b must be both a square and a cube in $\mathbb{F}_p$, it follows that $b = 1$ is the only option.

4.3 Final exponentiation

After the main Miller loop, the Tate pairing (and its variants) must carry out the final exponentiation to ensure a unique result of the pairing. The output of the Miller loop f must be raised to be power of $(p^k - 1)/r$ to obtain a result of order r. Scott et al. [33] introduced an efficient algorithm to compute such a computation. Their algorithm splits the final exponentiation into two parts: the first part is "easy" as raising to the power of p is an almost free application of the Frobenius operator; the second so-called "hard" is to power of $\Phi_k(p)/r \in \mathbb{F}_p[x]$. The exponent of the hard part can be expanded to the base p as $a_{n-1}p^{n-1} + \cdots + a_1 p + a_0$, where $n = \varphi(k)$, the Euler-phi function. We refer readers to [33] for more details about this computation. In this section, we give an efficient version of the hard part in final exponentiation for curves of embedding degrees $k = 9, 15$.

In the case of k = 9. By setting $x = 6u + 1$, we obtain the new explicit polynomials as follows :

$$\begin{aligned}
t(u) &= 6u + 2, \\
p(u) &= 559872u^8 + 559872u^7 + 233280u^6 + 54432u^5 + 7776u^4 + 648u^3 + 36u^2 + 6u + 1, \\
r(u) &= 15552u^6 + 15552u^5 + 6480u^4 + 1512u^3 + 216u^2 + 18x + 1.
\end{aligned}$$

The cost of the final exponentiation for the Ate pairing on curves with $k = 9$ was analyzed by Lin *et al.* [26]. Let the hard part $\frac{p(u)^6+p(u)^3+1}{r(u)} = \sum_{i=0}^{5} a_i(u)p(u)^i$, where $a_i(u)$ are following explicit polynomials (see in [26, §6.1]):

$$
\begin{aligned}
a_5 &= 36u^2, \\
a_4 &= 216u^3 + 36u^2 = a_5(6u+1), \\
a_3 &= 1296u^4 + 432u^3 + 36u^2 = a_4(6u+1), \\
a_2 &= 7776u^5 + 3888u^4 + 648u^3 + 72u^2 = a_3(6u+1) + a_5, \\
a_1 &= 46656u^6 + 31104u^5 + 7776u^4 + 1080u^3 + 72u^2 = a_2(6u+1), \\
a_0 &= 279936u^7 + 233280u^6 + 77760u^5 + 14256u^4 + 1512u^3 + 72u^2 + 3 = a_1(6u+1) + 3.
\end{aligned} \tag{7}
$$

Their calculation requires $65\mathbf{M}_{p^9} + 375\mathbf{S}_{p^9} + 45\mathbf{M}_p$ for computing this hard part (see [26, Section 6.2]). The following computation allows us to save $17\mathbf{M}_{p^9} + 73\mathbf{S}_{p^9} + 45\mathbf{M}_p$.

Let $T = t - 1$, where $t = 6u + 2$ is the trace of Frobenius. Furthermore, let f be the output of Miller algorithm, and $m = f^{p^3-1}$ (i.e., easy part). We compute the hard part as follows:

$$
m^{\frac{p(u)^6+p(u)^3+1}{r(u)}} = \mu_0 \cdot \mu_1^p \cdot \mu_2^{p^2} \cdot \mu_3^{p^3} \cdot \mu_4^{p^4} \cdot \mu_5^{p^5},
$$

where μ_i can be computed as follows:

$$
\mu_5 = (m^{T-1})^{T-1},\ \mu_4 = (\mu_5)^T,\ \mu_3 = (\mu_4)^T,
$$

$$
\mu_2 = (\mu_3)^T \cdot \mu_5,\ \mu_1 = (\mu_2)^T,\ \mu_0 = (\mu_1)^T \cdot m^3.
$$

This part requires 7 exponentiations by T or $T-1$, 8 multiplications and one squaring in $\mathbb{F}_{p^9}$, and 5 p-power Frobenius operations. Let T be a number of 44 bits length and Hamming weight of T and $T-1$ is 7 and 6, resp. (as the example given in [26]). This part requires $2(43\mathbf{S}_{p^9} + 5\mathbf{M}_{p^9}) + 5(43\mathbf{S}_{p^9} + 6\mathbf{M}_{p^9}) + 8\mathbf{M}_{p^9} + 1\mathbf{S}_{p^9} = 302\mathbf{S}_{p^9} + 48\mathbf{M}_{p^9}$. We save $17\mathbf{M}_{p^9} + 73\mathbf{S}_{p^9} + 45\mathbf{M}_p$ (more than 20%) in comparison to computations in [26].

In the case of k = 9. We present here the first analysis for the hard part during Tate pairing computation over elliptic curves with embedding degree $k = 15$. Likewise, by setting $x = 6u + 1$, we obtain the new explicit polynomials as follows :

$$
\begin{aligned}
t(u) &= 6u + 2, \\
p(u) &= 725594112u^{12} + 1209323520u^{11} + 906992640u^{10} + 403107840u^9 + 117573120u^8 \\
&\quad + 23607936u^7 + 3343680u^6 + 336960u^5 + 23760u^4 + 1080u^3 + 36u^2 + 6u + 1, \\
r(u) &= 1679616u^8 + 1959552u^7 + 979776u^6 + 279936u^5 + 50544u^4 + 6048u^3 \\
&\quad + 504u^2 + 24x + 1.
\end{aligned}
$$

Once again, assume that the hard part $\frac{p(u)^{10}+p(u)^5+1}{r(u)}$ of the final exponentiation can be expanded as $\sum_{i=0}^{9} a_i(u)p(u)^i$. It is easy to verify $a_i(u)$ to be following explicit polynomials.

$$
\begin{aligned}
a_9 &= 432u^4 + 216u^3 + 36u^2, \\
a_8 &= 2592u^5 + 1728u^4 + 432u^3 + 36u^2 = a_9T, \\
a_7 &= 15552u^6 + 12960u^5 + 4320u^4 + 648u^3 + 36u^2 = a_8T, \\
a_6 &= 93312u^7 + 93312u^6 + 38880u^5 + 8208u^4 + 864u^3 + 36u^2 = a_7T, \\
a_5 &= 559872u^8 + 653184u^7 + 326592u^6 + 88128u^5 + 13392u^4 + 1080u^3 + 36u^2 = a_6T, \\
a_4 &= 3359232u^9 + 4478976u^8 + 2612736u^7 + 855360u^6 + 168480u^5 + 20304u^4 + 1512u^3 \\
&\quad + 72u^2 = a_5T + a_9, \\
a_3 &= 20155392u^{10} + 30233088u^9 + 20155392u^8 + 7744896u^7 + 1866240u^6 + 290304u^5 \\
&\quad + 29376u^4 + 1944u^3 + 72u^2 = a_4T, \\
a_2 &= 120932352u^{11} + 201553920u^{10} + 151165440u^9 + 66624768u^8 + 18942336u^7 \\
&\quad + 3608064u^6 + 466560u^5 + 41040u^4 + 2376u^3 + 72u^2 + 1 = a_3T + 1, \\
a_1 &= 120932352u^{11} + 201553920u^{10} + 147806208u^9 + 62705664u^8 + 16982784u^7 + 3063744u^6 \\
&\quad + 375840u^5 + 31536u^4 + 1728u^3 + 36u^2 + 1 = (T^2-1)a_4 + (T^2-1)a_7 + a_8 + 1, \\
a_0 &= 120932352u^{11} + 181398528u^{10} + 120932352u^9 + 47029248u^8 + 11757312u^7 + 1975104u^6 \\
&\quad + 228096u^5 + 18144u^4 + 864u^3 + 1 = (T-1)(a_3 + a_6 + a_9) + 1,
\end{aligned}
\tag{8}
$$

where $T = 6u + 1$. Similarly, we assume that f is the output of Miller algorithm, and $m = f^{p^5-1}$. The hard part can be performed as follows:

$$
m^{\frac{p(u)^{10}+p(u)^5+1}{r(u)}} = \prod_{i=0}^{9} \mu_i^{p^i},
$$

where μ_i can be computed as follows:

$$
\mu_9 = ((m^{T-1})^{(T-1)/3} \cdot m^{T-1} \cdot m)^{(T-1)^2},\ \mu_i = (\mu_{i+1})^T \text{ for } i \in \{3,5,6,7,8\},\ \mu_4 = (\mu_5)^T \cdot \mu_9,
$$

$$
\mu_2 = (\mu_3)^T \cdot m,\ \mu_1 = (\mu_4 \cdot \mu_7)^{(T^2-1)} \cdot \mu_8 \cdot m,\ \mu_0 = (\mu_3 \cdot \mu_6 \cdot \mu_9)^{(T-1)} \cdot m.
$$

This part requires 2 exponentiations by $T-1$, 7 exponentiations by T, 1 exponentiation by $(T-1)/3$, 1 exponentiation by $(T-1)^2$, and 1 exponentiation by T^2-1. It also requires 19 multiplications in $\mathbb{F}_{p^{15}}$, and 9 p-power Frobenius operations. Assume that we apply this family of curves for pairing computation at 192-bit security level. The sizes in bits of r, and T are 384 and 64, respectively. By carefully choosing parameters, we can get a value of T with low Hamming weight (*e.g.*, $H(T) = 7$). This final exponentiation will require $2(63\mathbf{S}_{p^{15}} + 5\mathbf{M}_{p^{15}}) + 7(63\mathbf{S}_{p^{15}} + 6\mathbf{M}_{p^{15}}) + (63\mathbf{S}_{p^{15}} + 5\mathbf{M}_{p^{15}}) + 2(127\mathbf{S}_{p^{15}} + 12\mathbf{M}_{p^{15}}) + 19\mathbf{M}_{p^{15}} = 100\mathbf{M}_{p^{15}} + 884\mathbf{S}_{p^{15}}$.

4.4 Discussion

At 128-bit security level, the current public-key security recommendations, Barreto-Naehrig curves [4] lead a very efficient implementation. Many results have been reported in papers [29][5][30][1]. That is because BN curves can exploit a sextic twist and there exist efficient algorithms for squarings in $\mathbb{F}_{p^{12}}$ [15][19]. The former allows us to work on points of the twisted curve whose coordinates are in $\mathbb{F}_{p^2}$ instead of $\mathbb{F}_{p^{12}}$ during Miller loop computation. The later provides an efficient speedup for the final exponentiation step.

In [26] the authors consider curves with $k = 9$ at 128-bit security level. One advantage of such a curve compared with BN curve is that it will have an Ate pairing with $2/3$ Miller loop length compared with the BN equivalent. With many optimizations in both Miller loop and the final exponentiation, BN curves are perfectly suited for implementing a single pairing. However, when we need to compute several pairings in parallel, where only one final exponentiation required to compute, curves with shorter Miller loop may offer a good choice. Our above analysis allowing to speed up pairing computation over cubic twisted curves in affine coordinates for both Miller loop and final exponentiation, are helpful for this case.

5 Conclusion

In this paper we further analyzed techniques to speed up Ate pairing computation in affine coordinates using 4-ary Miller algorithm. We focused on pairing computations over pairing-friendly elliptic curves admitting a cubic twist and presented the first and fast explicit formulas in affine coordinates for such curves. We also gave a finer choice for curves of embedding degrees $k = 9, 15$, and show that this choice leads to an efficient arithmetic and final exponentiation.

References

1. Aranha, D., Karabina, K., Longa, P., Gebotys, C., López, J.: Faster explicit formulas for computing pairings over ordinary curves. In Paterson, K., ed.: Advances in Cryptology – EUROCRYPT 2011. Volume 6632 of Lecture Notes in Computer Science. Springer Berlin / Heidelberg (2011) 48–68 10.1007/978-3-642-20465-4_5.
2. Barreto, P.S.L.M., Kim, H.Y., Lynn, B., Scott, M.: Efficient algorithms for pairing-based cryptosystems. In: CRYPTO '02: Proceedings of the 22nd Annual International Cryptology Conference on Advances in Cryptology, London, UK, Springer-Verlag (2002) 354–368
3. Barreto, P.S.L.M., Lynn, B., Scott, M.: On the selection of pairing-friendly groups. In Matsui, M., Zuccherato, R.J., eds.: Selected Areas in Cryptography. Volume 3006 of Lecture Notes in Computer Science., Springer (2003) 17–25
4. Barreto, P.S.L.M., Naehrig, M.: Pairing-friendly elliptic curves of prime order. In: Proceedings of SAC 2005, volume 3897 of LNCS, Springer-Verlag (2005) 319–331
5. Beuchat, J.L., González-Díaz, J.E., Mitsunari, S., Okamoto, E., Rodríguez-Henríquez, F., Teruya, T.: High-speed software implementation of the optimal Ate pairing over barreto-naehrig curves. In Joye, M., Miyaji, A., Otsuka, A., eds.: Pairing. Volume 6487 of Lecture Notes in Computer Science., Springer (2010) 21–39
6. Blake, I.F., Murty, V.K., Xu, G.: Refinements of Miller's algorithm for computing the Weil/Tate pairing. J. Algorithms **58**(2) (2006) 134–149
7. Boneh, D., Franklin, M.K.: Identity-Based Encryption from the Weil Pairing. In: CRYPTO '01: Proceedings of the 21st Annual International Cryptology Conference on Advances in Cryptology, Springer-Verlag (2001) 213–229
8. Boxall, J., El Mrabet, N., Laguillaumie, F., Le, D.P.: A variant of miller's formula and algorithm. In: Proceedings of the 4th international conference on Pairing-based cryptography. Pairing'10, Berlin, Heidelberg, Springer-Verlag (2010) 417–434
9. Brezing, F., Weng, A.: Elliptic curves suitable for pairing based cryptography. Des. Codes Cryptography **37** (October 2005) 133–141
10. Costello, C., Boyd, C., Nieto, J.M.G., Wong, K.K.H.: Avoiding full extension field arithmetic in pairing computations. In: AFRICACRYPT'10. (2010) 203–224
11. Costello, C., Lange, T., Naehrig, M.: Faster Pairing Computations on Curves with High-Degree Twists. In Nguyen, P., Pointcheval, D., eds.: Public Key Cryptography – PKC 2010. Volume 6056 of Lecture Notes in Computer Science. Springer Berlin / Heidelberg (2010) 224–242
12. El Mrabet, N., Guillermin, N., Ionica, S.: A study of pairing computation for elliptic curves with embedding degree 15. Cryptology ePrint Archive, Report 2009/370 (2009) http://eprint.iacr.org/.
13. Freeman, D., Scott, M., Teske, E.: A Taxonomy of Pairing-Friendly Elliptic Curves. J. Cryptol. **23** (April 2010) 224–280
14. Galbraith, S.D., Harrison, K., Soldera, D.: Implementing the tate pairing. In: ANTS-V: Proceedings of the 5th International Symposium on Algorithmic Number Theory, London, UK, Springer-Verlag (2002) 324–337
15. Granger, R., Scott, M.: Faster squaring in the cyclotomic subgroup of sixth degree extensions. In Nguyen, P.Q., Pointcheval, D., eds.: Public Key Cryptography. Volume 6056 of Lecture Notes in Computer Science., Springer (2010) 209–223

16. Hess, F., Smart, N.P., Vercauteren, F.: The eta pairing revisited. IEEE Transactions on Information Theory **52** (2006) 4595–4602
17. Joux, A.: A One Round Protocol for Tripartite Diffie-Hellman. In: ANTS-IV: Proceedings of the 4th International Symposium on Algorithmic Number Theory, Springer-Verlag (2000) 385–394
18. Kachisa, E.J., Schaefer, E.F., Scott, M.: Constructing Brezing-Weng Pairing-Friendly Elliptic Curves Using Elements in the Cyclotomic Field. In: Proceedings of the 2nd international conference on Pairing-Based Cryptography. Pairing '08, Berlin, Heidelberg, Springer-Verlag (2008) 126–135
19. Karabina, K.: Squaring in cyclotomic subgroups. Cryptology ePrint Archive, Report 2010/542 (2010) http://eprint.iacr.org/.
20. Kobayashi, T., Aoki, K., Imai, H.: Efficient algorithms for tate pairing. IEICE Trans. Fundam. Electron. Commun. Comput. Sci. **E89-A**(1) (January 2006) 134–143
21. Kobayashi, T., Morita, H., Kobayashi, K., Hoshino, F.: Fast elliptic curve algorithm combining frobenius map and table reference to adapt to higher characteristic. In: Proceedings of the 17th international conference on Theory and application of cryptographic techniques. EUROCRYPT'99, Berlin, Heidelberg, Springer-Verlag (1999) 176–189
22. Lauter, K., Montgomery, P.L., Naehrig, M.: An analysis of affine coordinates for pairing computation. In: Proceedings of the 4th international conference on Pairing-based cryptography. Pairing'10, Berlin, Heidelberg, Springer-Verlag (2010) 1–20
23. Le, D.-P.: Fast Quadrupling of a Point in Elliptic Curve Cryptography. Cryptology ePrint Archive, Report 2011/039 (2011) http://eprint.iacr.org/.
24. Le, D.-P., Liu, C.L.: Refinements of Miller's Algorithm over Weierstrass Curves Revisited. The Computer Journal **54**(10) (2011) 1582–1591
25. Le, D.-P. and Tan, C.H.: Speeding Up Ate Pairing Computation in Affine Coordinates. In: Proceedings of the 15th International Conference on Information Security and Cryptology – ICISC 2012, Berlin, Heidelberg, Springer-Verlag (2013) 262–277
26. Lin, X., Zhao, C.A., Zhang, F., Wang, Y.: Computing the Ate pairing on elliptic curves with embedding degree k = 9. IEICE Trans. Fundam. Electron. Commun. Comput. Sci. **E91-A**(9) (2008) 2387–2393
27. Miller, V.S.: The Weil Pairing, and Its Efficient Calculation. Journal of Cryptology **17**(4) (2004) 235–261
28. Miyaji, A., Nakabayashi, M., Takano, S.: New Explicit Conditions of Elliptic Curve Traces for FR-Reduction. IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences **84**(5) (2001) 1234–1243
29. Naehrig, M., Niederhagen, R., Schwabe, P.: New software speed records for cryptographic pairings. In: Proceedings of the First international conference on Progress in cryptology: cryptology and information security in Latin America. LATINCRYPT'10, Berlin, Heidelberg, Springer-Verlag (2010) 109–123
30. Pereira, G.C.C.F., Simplício, J.M.A., Naehrig, M., Barreto, P.S.L.M.: A family of implementation-friendly bn elliptic curves. J. Syst. Softw. **84** (August 2011) 1319–1326
31. Scott, M.: On the efficient implementation of pairing-based protocols. In Chen, L., ed.: IMA Int. Conf. Volume 7089 of Lecture Notes in Computer Science., Springer (2011) 296–308
32. Scott, M., Barreto, P.S.: Compressed pairings. In Franklin, M., ed.: Advances in Cryptology – CRYPTO 2004. Volume 3152 of Lecture Notes in Computer Science. Springer Berlin / Heidelberg (2004) 73–82 10.1007/978-3-540-28628-8_9.
33. Scott, M., Benger, N., Charlemagne, M., Perez, L.J.D., Kachisa, E.J.: On the final exponentiation for calculating pairings on ordinary elliptic curves. In Shacham, H., Waters, B., eds.: Pairing. Volume 5671 of Lecture Notes in Computer Science., Springer (2009) 78–88
34. Joseph H. Silverman: The Arithmetic of Elliptic Curves (2nd Edition). Springer-Verlag (May 2009)
35. Vercauteren, F.: Optimal pairings. IEEE Transactions on Information Theory **56**(1) (2010) 455–461