Efficient Pairing Computation on Supersingular Abelian Varieties

Paulo S. L. M. Barreto[1], Steven Galbraith[2], Colm Ó hÉigeartaigh[3], and Michael Scott[3]

[1] Department of Computing and Digital Systems Engineering,
Escola Politécnica, Universidade de São Paulo.
Av. Prof. Luciano Gualberto, tr. 3, n. 158.
05508-900 São Paulo (SP), Brazil.
pbarreto@larc.usp.br

[2] Mathematics Department, Royal Holloway University of London.
Egham, Surrey TW20 0EX, UK.
steven.galbraith@rhul.ac.uk

[3] School of Computing, Dublin City University.
Ballymun, Dublin 9, Ireland.
{coheigeartaigh,mike}@computing.dcu.ie

Abstract. We present a general technique for the efficient computation of pairings on supersingular Abelian varieties. This formulation, which we call the eta pairing, generalises results of Duursma and Lee for computing the Tate pairing on supersingular elliptic curves in characteristic three.
We then show how our general technique leads to a new algorithm which is about twice as fast as the Duursma-Lee method.
These ideas are then used for elliptic and hyperelliptic curves in characteristic 2 with very efficient results. In particular, the hyperelliptic case is faster than all previously known pairing algorithms.

Keywords: Tate pairing, supersingular curves, pairing-based cryptosystems, efficient algorithms.

1 Introduction

Efficient computation of pairings is essential to the large and ever growing area of pairing-based cryptosystems (see e.g. Chapter 10 of [6] or [9] for a comprehensive overview).

There has been a lot of work on efficient implementation of pairings on elliptic curves. Supersingular curves lead to more efficient implementations in terms of processing speed [3, 15, 10] and bandwidth requirements [28, 17] than the best available algorithms for ordinary curves [4]. Pairings on hyperelliptic curves have received considerably less attention than their elliptic counterparts. The best results are by Duursma and Lee [10] for a very special family of supersingular hyperelliptic curves. These results suggest that supersingular hyperelliptic curves may provide similar efficiency to elliptic curves, but these issues have not been at all clear until now.

We tackle this problem by providing criteria under which pairings on supersingular hyperelliptic curves are efficiently computable. Our method is fairly general and includes that of Duursma-Lee [10] as a particular case. We also obtain a significant improvement over previous methods, even in the characteristic three case. We illustrate the method by describing efficient pairing algorithms for supersingular genus 1 and genus 2 curves in characteristic 2.

This paper is organised as follows. Section 2 gives a brief summary on standard techniques for the efficient computation of the Tate pairing. Section 3 discusses the contributions of Duursma and Lee for certain supersingular curves, and section 4 generalises those contributions using the simpler, unified approach of *eta pairings*. Section 5 shows how the Duursma-Lee results fit into the eta pairing framework and then gives a significant improvement. Sections 6 and 7 explore the consequences of the eta pairing approach for certain elliptic curves and genus 2 curves in characteristic 2. We compare our pairings on genus 2 curves with the work of Rubin and Silverberg in section 9, and present some experimental results in section 10. Finally, we draw our conclusions in section 11.

Parts of this work were presented by one of the authors [2] at the ECC'2004 conference on September 20–22, 2004. Subsequently and independently, on November, 14 2004, a paper [21] containing some results related to those in this paper was posted on the ePrint archive.

2 The Tate pairing on supersingular curves

Let C be a smooth, projective, absolutely irreducible curve over a finite field $K = \mathbb{F}_{q^k}$. We denote the degree zero divisor class group of C over K by $\mathrm{Pic}_0^K(C)$. Let r be an integer such that $r \mid \#\mathrm{Pic}_0^K(C)$. We denote by $\mathrm{Pic}_0^K(C)[r]$ the divisor classes of order dividing r.

Let D_1 be a divisor representing a class in $\mathrm{Pic}_0^K(C)[r]$ and let D_2 be a divisor on C defined over K such that the supports of D_1 and D_2 are disjoint. Since rD_1 is principal there is a function f on C defined over K such that $(f) = rD_1$. The Tate pairing (also called the Tate-Lichtenbaum pairing) is

$$\langle D_1, D_2\rangle_r = f(D_2).$$

One can show (see Frey and Rück [14]) that the Tate pairing is a well-defined, non-degenerate, bilinear pairing

$$\mathrm{Pic}_0^K(C)[r] \times \mathrm{Pic}_0^K(C)/r\mathrm{Pic}_0^K(C) \to K^*/(K^*)^r.$$

The fact that the Tate pairing is only defined up to r-th powers is often undesirable. To obtain a unique value, one defines the *reduced* pairing

$$e(D_1, D_2) = \langle D_1, D_2\rangle_r^{(q^k-1)/r}.$$

Throughout the paper we will refer to the extra powering required to compute the reduced pairing as the *final exponentiation*.

One very important property of the reduced pairing is the following [15]. Let $N = hr$ for some h.

$$e(D_1, D_2) = \langle D_1, D_2\rangle_r^{(q^k-1)/r} = \langle D_1, D_2\rangle_N^{(q^k-1)/N}. \qquad (1)$$

2.1 Miller's algorithm in the elliptic case

We recall how the Tate pairing can be computed in polynomial time using Miller's algorithm [24]. For simplicity we restrict to the case of elliptic curves. The divisor class group of an elliptic curve is isomorphic to the curve itself, so all divisors may be assumed to have the form $D = (P) - (\infty)$.

Let E be an elliptic curve over $\mathbb{F}_q$ and let $r \mid \#E(\mathbb{F}_q)$ be a prime. Suppose the embedding degree is k (i.e., k is the smallest positive integer such that $r \mid (q^k - 1)$). Let $P \in E[r]$ and $Q \in E(\mathbb{F}_{q^k})$, where typically Q is the image of some multiple of P under a non-rational endomorphism called a distortion map. We construct an $\mathbb{F}_{q^k}$-rational divisor D equivalent to $(Q) - (\infty)$ by taking a random point $R \in E(\mathbb{F}_{q^k})$ and defining $D = (Q + R) - (R)$. We aim to compute

$$e(P, Q) = e((P) - (\infty), D).$$

For every integer n and point P there is a function $f_{n,P}$ such that

$$(f_{n,P}) = n(P) - ([n]P) - (n - 1)(\infty).$$

Miller's algorithm builds up these functions $f_{n,P}$ according to the following formula: If l and v are the lines which arise in the addition rule for adding $[n]P$ and $[m]P$ then we have

$$f_{n+m,P} = f_{n,P} f_{m,P} l/v.$$

The pairing value $\langle (P) - (\infty), D\rangle_r$ is $f_{r,P}(D)$.

Miller's algorithm is explicitly described in Algorithm 1. Note that the addition in the final iteration is simplified in that l is a vertical line and v disappears.

Miller's algorithm can be generalised to general divisor class groups. The basic algorithm is the same, but the functions are more complicated.

2.2 Improvements to Miller's algorithm

Several improved implementation techniques to compute the reduced Tate pairing on supersingular elliptic curves have been proposed [3, 15]. These include:

Exploiting properties of the field of definition: It is typical in pairing applications to pair a point defined over $\mathbb{F}_q$ with a point defined over $\mathbb{F}_{q^k}$. Hence it makes sense to represent $\mathbb{F}_{q^k}$ as an extension of $\mathbb{F}_q$ and to try to simplify the operations in $\mathbb{F}_{q^k}$ as much as possible.

The final exponentiation eliminates terms defined over subfields. Hence, terms defined over subfields can be omitted from the calculations. For example, if $k > 1$ then the point R can be chosen to be defined over a subfield, in which case all terms $l(R)$ and $v(R)$ may be ignored.

Algorithm 1 Miller's algorithm (base 2)

```
INPUT: $r$, $P$, $Q + R$, $R$, where the binary representation of $r$ is $\{r_i\}$.
OUTPUT: $\langle P, Q\rangle_r$
 1: $T \leftarrow P$
 2: $f \leftarrow 1$
 3: for $i \leftarrow \lfloor \log_2(r) \rfloor - 1$ downto 0 do
 4:     ▹ Calculate lines $l$ and $v$ in doubling $T$
 5:     $T \leftarrow [2]T$
 6:     $f \leftarrow f^2 \cdot l(Q + R)v(R)/(v(Q + R)l(R))$
 7:     if $r_i = 1$ then
 8:         ▹ Calculate lines $l$ and $v$ in adding $P$ to $T$
 9:         $T \leftarrow T + P$
10:         $f \leftarrow f \cdot l(Q + R)v(R)/((v(Q + R)l(R))$
11:     end if
12: end for
13: return $f$
```

Changing the base in Miller's algorithm: Miller's algorithm is usually presented as a loop through the binary expansion of the group order. It is sometimes more efficient to use other bases, for example to write the group order in base three when implementing pairings in characteristic three (in which case, in line 6 of Miller's algorithm above, we change f^2 to f^3 and there are now two cases in the addition step).

Replacing divisors by points: As explained above, the point R can be ignored. In fact, one can choose $R = \infty$ by [3, Theorem 1]. Hence, the reduced pairing can be computed as

$$e(P, Q) = f_{r,P}(Q)^{(q^k-1)/r},$$

where the function f_r is now evaluated on a *point* rather than on a divisor.

Exploiting the form of the distortion maps and denominator elimination: If the distortion map is chosen so that the x-coordinates always lies in a subfield, then all terms $v(Q)$ may be eliminated. As a result there are no longer any divisions in Miller's algorithm.

Note that a distortion map with this property can always be obtained by combining with a map into the trace zero subgroup (see [27] for an example of this).

Hamming weight/group order issues: Miller's algorithm to compute f_r involves a number of arithmetic operations proportional to the Hamming weight of r, and for this reason it is advantageous to choose r with low Hamming weight (with respect to the base being used) whenever possible. In many cases it is worth using a small multiple of r which has low Hamming weight and exploiting formula (1).

Speeding up the final exponentiation: The naive way to compute the final powering to $(q^k - 1)/N$ (for some multiple N of r) has cubic complexity. However, this exponent has a rather simple structure for supersingular curves when one chooses N to be the full curve order rather than a factor r thereof. By carefully exploiting that structure, one can replace the powering by a few applications of the Frobenius, some multiplications, and one inversion. Details can be found in [3, Appendix A.2].

Pairing value compression: It is possible to reduce the bandwidth requirements of pairing values by storing and manipulating traces [28], or by working on a torus [17]. These methods compress pairing values to half their usual size, or to a third thereof in the case of supersingular elliptic curves with embedding degree 6 in characteristic 3, supersingular genus 2 curves with embedding degree 12 in characteristic 2, or ordinary BN elliptic curves [5] with embedding degree 12 in large prime characteristic.

The techniques mentioned above give impressive results for pairing implementation. For the remainder of the paper we focus on further improvements. We consider only supersingular curves over $\mathbb{F}_q$ with embedding degree $k > 1$ and with suitable distortion maps ψ. We will always be computing a modified pairing

$$\hat{e}_r(P, Q) = \langle P, \psi(Q) \rangle_r$$

where P and Q are defined over $\mathbb{F}_q$, or its reduced version $\hat{e}(P, Q) = \hat{e}_r(P, Q)^{(q^k-1)/r}$.

3 The Duursma-Lee techniques

Duursma and Lee [10] gave a significant improvement to the computation of pairings on curves of the form $y^2 = x^p - x + d$ over $\mathbb{F}_{p^m}$ where $p \geq 3$ and $(m, 2p) = 1$ (these curves have embedding degree $2p$). In particular, their results apply to the case of the embedding degree 6 curve in characteristic three.

One crucial aspect of [10] is that they replace the group order r by the value $p^{mp} + 1$ which has Hamming weight 2 in base p. Also, the final exponentiation is to the power $(p^{2mp} - 1)/(p^{pm} + 1) = p^{mp} - 1$, which is simply computing a Frobenius conjugation and a division. Hence, both the final exponentiation and the main body of Miller's algorithm are simplified, at the expense of extending the main loop from m iterations to mp iterations. Duursma and Lee show the surprising fact that this loop can be shortened from mp iterations to m iterations.

A careful reading of [10] shows that it contains four independent contributions:

1. A nice choice of function for computing pD in the divisor class group;
2. The definition of a pairing on *points* (in $g > 1$). In other words, they propose the use of degenerate divisors rather than general divisors;
3. A shorter loop than would be expected for the given group order;
4. Incorporating Frobenius operations directly into the formulae (this has a huge saving, since it removes the exponentiation of f to the power p in line 6 of Miller's algorithm).

Generalising points 1, 2 and 4 is relatively straightforward. In this paper we show how to generalise the loop shortening idea to many other cases.

4 The eta pairing approach

Let C be a curve over $\mathbb{F}_q$ (where $q = p^m$) with a single point at infinity. In all the examples in this paper, C will be an elliptic or hyperelliptic curve. We will always assume that C is supersingular, with even embedding degree $k > 1$, and that there is a distortion map ψ which allows denominator elimination (i.e., if $P \in C(\mathbb{F}_q)$ then $\psi(P) \in C(\mathbb{F}_{q^k})$ has x-coordinate defined over $\mathbb{F}_{q^{k/2}}$).

Let D, D' be reduced divisors on C defined over $\mathbb{F}_q$ which represent divisor classes of order dividing N. In all examples in this paper, these divisors will be represented using the Mumford notation (see Cantor [8]) which, in the elliptic curve case, corresponds to just a single point. Let $M = (q^k - 1)/N$. We want to efficiently compute the Tate pairing (including the final exponentiation) $\langle D, \psi(D')\rangle_N^M$.

Let $n \in \mathbb{N}$. We use the notation D_n for a reduced divisor equivalent to nD and $f_{n,D}$ for a function whose divisor is $nD - D_n - m(\infty)$ for some $m \in \mathbb{N}$. In the elliptic case we have $D = (P) - (\infty)$ and so $D_n = (nP) - (\infty)$ and $f_{n,D}$ is the Miller function introduced in subsection 2.1 . If $n \in \mathbb{Z}$ with $n < 0$ then $nD = (-n)(-D)$. We therefore write D_n for a divisor equivalent to $(-n)(-D)$ and write $f_{n,D}$ for a function with divisor $(-n)(-D) - (D_n) - m(\infty)$ for some m. The Tate pairing is defined to be $\langle D, D'\rangle_N = f_{N,D}(D')$.

An important observation is that, for many supersingular curves, multiplication by p has an extremely special form. This has already been exploited by many authors. In this paper we will be concerned with cases where multiplication by some power of p can be represented by an automorphism on the curve (which we will call γ).

Definition 1. *For $T \in \mathbb{Z}$ we define the* eta pairing *to be*

$$\eta_T(D, D') = f_{T,D}(\psi(D')). \tag{2}$$

In general, this definition will not give a non-degenerate, bilinear pairing. The aim of this paper is to explain some cases where the resulting pairing is non-degenerate and bilinear. The key property of the eta pairing is that we do not necessarily demand that TD is equivalent to zero. The aim is to choose values of T which are smaller than N. This is a generalisation of the loop reduction idea of Duursma and Lee.

As we will see, the Duursma-Lee method arises from the choice $T = q$ in the above definition while our improved version uses the choice $T = q - N$. In the later part of the paper, when discussing running times, we will drop the subscript in the case $T = q$ and refer to the pairing as η. Hence, the notation η_T will generally be reserved for the improved version.

The following theorem is the main result of this paper. It relates the eta pairing to the Tate pairing for certain values of T. From this relation one immediately deduces (as long as L, a and T are coprime to N) that the eta pairing (for these values of T) is non-degenerate and bilinear.

Theorem 1. *Let C be a supersingular curve over $\mathbb{F}_q$ with distortion map ψ and embedding degree k as above. Let D be a divisor on C defined over $\mathbb{F}_q$ with order dividing $N \in \mathbb{N}$ and let $M = (q^k - 1)/N$. Suppose $T \in \mathbb{Z}$ is such that*

1. *$TD \equiv \gamma(D)$ in the divisor class group where γ is an automorphism of C which is defined over $\mathbb{F}_q$.*
2. *γ and ψ satisfy the condition*[4]
$$\gamma\psi^q(Q) = \psi(Q) \tag{3}$$
for all points $Q \in C(\mathbb{F}_q)$.
3. *$T^a + 1 = LN$ for some $a \in \mathbb{N}$ and $L \in \mathbb{Z}$.*
4. *$T = q + cN$ for some $c \in \mathbb{Z}$.*

Then
$$\left(\langle D, \psi(D')\rangle_N^M\right)^L = (\eta_T(D, D')^M)^{aT^{a-1}}.$$

4.1 Proof of Theorem 1

We split the proof into a number of lemmas.

First note that, since TD is equivalent to $\gamma(D)$ we have $D_{T^i} = \gamma^i(D)$. Write d for the degree of the finite part of D. Then $D = \sum_{j=1}^d (P_j) - d(\infty)$ and so $D_{T^i} = \sum_{j=1}^d (\gamma^i(P_j)) - d(\infty)$.

The key result is the following.

Lemma 1. *With notation as above and D any divisor such that TD is equivalent to $\gamma(D)$. Then*
$$f_{T,D}(\psi(D'))^{TM} = f_{T,TD}(\psi(D'))^M.$$

Proof. We have $(f_{T,D}) = TD - D_T - (T-1)d(\infty)$ and $(f_{T,D}^T) = T(f_{T,D})$ and $(f_{T,TD}) = TD_T - D_{T^2} - (T-1)d(\infty)$.

We now use the assumption that $TD \equiv D_T = \gamma(D)$. The pullback (see Silverman [30] Chapter II page 33) satisfies
$$\gamma^*\left(\sum_P n_P(P)\right) = \sum_P \sum_{S \in \gamma^{-1}(P)} n_P e_\gamma(S)(S) = \sum_P n_P(\gamma^{-1}(P)).$$

Hence
$$\begin{aligned}\gamma^*(f_{T,TD}) &= \gamma^*(TD_T - D_{T^2} - (T-1)d(\infty))\\ &= TD - D_T - (T-1)d(\infty)\\ &= (f_{T,D}).\end{aligned}$$

Also, (Silverman [30] pages 33-34)
$$\gamma^*(f_{T,TD}) = (\gamma^* f_{T,TD}) = (f_{T,TD} \circ \gamma).$$

Hence, we have (up to a scalar multiple in $\mathbb{F}_q^*$)
$$f_{T,TD} \circ \gamma = f_{T,D}.$$

[4] An alternative formulation of this condition is $\gamma\psi^\pi = \psi$ where π is the q-power Frobenius and ψ^π means the map obtained by applying π to the coefficients of the map ψ.

Evaluating at $\psi(D')$ and raising to the power M (which kills $\mathbb{F}_q^*$) we get

$$f_{T,TD}(\gamma(\psi(D')))^M = f_{T,D}(\psi(D')^M.$$

Consider the left hand side of the statement of the Lemma

$$f_{T,D}(\psi(D'))^{TM} = (f_{T,TD}(\gamma(\psi(D')))^{TM}.$$

Now use the fact that $T = q + cN$ and that $NM = (q^k - 1)$ so anything raised to the power NM is 1. We therefore have that the above is equal to

$$(f_{T,TD}(\gamma(\psi(D')))^{qM}.$$

Interpreting a power of q as action by Frobenius and using the fact that $f_{T,TD}$, γ and D' are defined over $\mathbb{F}_q$ gives

$$(f_{T,TD}(\gamma(\psi^q(D')))^M.$$

By condition (3), $\gamma\psi^q = \psi$ so we get

$$f_{T,TD}(\psi(D'))^M$$

which proves the result.

Lemma 2. *With notation as above*

$$(f_{T^a,D}) = (f_{T,D}^{T^{a-1}} f_{T,TD}^{T^{a-2}} \cdots f_{T,T^{a-1}D}).$$

Proof. We have $(f_{T^a,D}) = T^aD - D_{T^a} - (T^a - 1)d(\infty)$. Hence

$$\begin{aligned}(f_{T,D}^{T^{a-1}} f_{T,TD}^{T^{a-2}} \cdots f_{T,T^{a-1}D}) &= T^{a-1}(f_{T,D}) + T^{a-2}(f_{T,TD}) + \cdots + (f_{T,T^{a-1}D})\\ &= T^{a-1}(TD - D_T - (T-1)d(\infty)) + T^{a-2}(TD_T - D_{T^2}\\ &\quad -(T-1)d(\infty)) + \cdots + TD_{T^{a-1}} - D_{T^a} - (T-1)d(\infty)\\ &= T^aD - D_{T^a} - (T^a - 1)d(\infty)\end{aligned}$$

which proves the result.

We can now obtain the statement of the theorem:

Lemma 3. *With notation as above*

$$(f_{N,D}(\psi(D')))^{ML} = (f_{T,D}(\psi(D')))^{MaT^{a-1}}.$$

Proof. Note that $f_{N,D}^L = f_{LN,D} = f_{T^a+1,D}$. Since $T^a+1 = LN$ we know that $(T^a+1)D \equiv 0$, which implies $T^aD \equiv -D$ and so (up to scalar in $\mathbb{F}_q^*$)

$$f_{T^a+1,D} = f_{T^a,D} \cdot v$$

where v is the vertical line through D and $-D$.

Evaluating at $\psi(D')$ and raising to the power M we have (since ψ admits denominator elimination)

$$f_{N,D}(\psi(D'))^{ML} = f_{T^a,D}(\psi(D'))^M \cdot v(\psi(D'))^M = f_{T^a,D}(\psi(D'))^M.$$

By Lemma 2 this is

$$\prod_{j=0}^{a-1} f_{T,T^jD}(\psi(D'))^{MT^{a-1-j}}.$$

Now, substituting T^jD for D in Lemma 1 implies that

$$f_{T,T^jD}(\psi(D'))^{MT^{a-1-j}} = f_{T,D}^{MT^{a-1}}.$$

Hence the result follows.

5 Elliptic curves in characteristic three

We first show how the eta pairing idea explains the loop shortening used by Duursma and Lee.

The elliptic curve of interest[5] is $E : y^2 = x^3 - x + b$ over $\mathbb{F}_{3^m}$ where $b = \pm 1$ and $\gcd(m, 6) = 1$. The number of points on this curve is given in Table 1. The tripling formula (see [11, 3]) is $[3](x, y) = \phi\pi^2(x, y)$ where π is the 3-power Frobenius and $\phi(x, y) = (x - b, -y)$. Note that $\phi^2(x, y) = (x - 2b, y), \phi^3 = -1$ etc. The distortion map is $\psi(x, y) = (\rho - x, \sigma y)$ where $\sigma^2 = -1$ and $\rho^3 = \rho + b$ (and thus $\rho^{3^2} = \rho + 2b, \rho^{3^3} = \rho$).

Table 1. Order of the curve $E : y^2 = x^3 - x + b$ over $\mathbb{F}_{3^m}$, $b = \pm 1$.

$\#E(\mathbb{F}_{3^m})$	condition
$3^m + 1 + b3^{(m+1)/2}$	$m \equiv 1, 11 \pmod{12}$
$3^m + 1 - b3^{(m+1)/2}$	$m \equiv 5, 7 \pmod{12}$

Let $q = 3^m$. It follows that multiplication by $[q]$ is $[q](x, y) = [3^m](x, y) = \phi^m\pi^{2m}(x, y) = \phi^m(x, y)$. Hence we take $\gamma = \phi^m$.

Lemma 4. *With notation as above, condition (3) is satisfied.*

Proof. Let $q = 3^m$ with $m \equiv 1 \pmod 6$. Suppose $(x, y) \in E(\mathbb{F}_q)$. Then $[q](x, y) = \phi(x, y)$ and so $\gamma = \phi$. Now, $\psi^q = \psi^3$ and so

$$\gamma\psi^q(x, y) = \phi(\rho + b - x, -\sigma y) = (\rho + b - x - b, \sigma y) = (\rho - x, \sigma y) = \psi(x, y).$$

[5] Note that all supersingular curves in characteristic three have j-invariant 0 (by Theorem V.4.1 of [30]) and hence are isomorphic over $\overline{\mathbb{F}}_3$ ([30] Theorem A.1.2). It follows that all choices of supersingular equations over $\mathbb{F}_{3^m}$ with fixed embedding degree k are equally secure for pairing applications and so there is no loss of generality from considering just this case.

Similarly, when $m \equiv 5 \pmod 6$ we have $\gamma = \phi^5 = -\phi^2$ and $\psi^q = -\psi^{3^2}$ and so

$$\gamma\psi^q(x,y) = \phi^2(\rho + 2b - x, \sigma y) = (\rho + 2b - x - 2b, \sigma y) = \psi(x,y).$$

This completes the proof. □

Since condition (3) is satisfied we may apply Theorem 1.

The method of Duursma and Lee computes the eta pairing with respect to the value $T = q = 3^m$. In the notation of Theorem 1 we have $N = q^3 + 1$, $M = q^3 - 1$, $a = 3$, $L = 1$ and $c = 0$. Hence, we have

$$\left(\eta_T(P,Q)^M\right)^{3q^2} = \langle P, \psi(Q)\rangle_N^M$$

The formulae given in [10] computes the Tate pairing directly by bringing the powering to $3q^2$ into the formulae. Further efficiency is obtained in [10] by using the other three techniques mentioned in section 3.

5.1 An improvement on Duursma and Lee

The power of the eta pairing approach is that one can immediately improve on the Duursma-Lee method giving a further halving of the length of the loop.

We know that the number of points on $E(\mathbb{F}_q)$ is $N = 3^m \pm 3^{(m+1)/2} + 1$ and we have established that $[3^m]P = \gamma(P)$ for some automorphism γ. If $P \in E(\mathbb{F}_q)$ then, since $[N]P = \infty$, we deduce that

$$[\mp 3^{(m+1)/2} - 1]P = [q - N]P = [q]P = \gamma(P).$$

We can therefore choose $T = q - N = \mp 3^{(m+1)/2} - 1$ (when T is negative we use the relation $TD = (-T)(-D)$). In the notation of Theorem 1 we have $c = -1$. Taking $a = 3$ gives $T^3 + 1 = LN$ where $L = \mp 3^{(m+3)/2}$. We have $M = (3^{6m} - 1)/N$. Theorem 1 implies that the pairing satisfies

$$\left(\eta_T(P,Q)^M\right)^{3T^2} = \left(\langle P, \psi(Q)\rangle_N^M\right)^L$$

and so, since 3, T and L are all coprime to N, it is bilinear and non-degenerate.

With this method we can compute the Tate pairing using an algorithm with roughly half as many iterations as the original Duursma-Lee method. However, the final exponentiation is now more complicated since the value M required to obtain a unique pairing value is $(3^{3m} - 1)(3^m + 1)(3^m \mp 3^{(m+1)/2} + 1)$. Unlike the original Duursma-Lee method, this value has terms which are not powers of 3^m, hence an extra $(m+1)/2$ cubings in the large field are required. Luckily, cubing is faster than a step in the loop of Miller's algorithm, so this approach does give faster code. Notice that the result of raising to $3^{3m} - 1$ produces a unitary value, so that any further inversion reduces to a simple conjugation.

Further exponentiations are also required to transform the value of the eta pairing to a correct Tate pairing value, but the extra cost of these is not very significant (see the end of section 5.2). One possibility is to design cryptosystems using the eta pairing instead of the Tate pairing. On the other hand, for some applications there may be compatibility issues with using a 'non-standard' pairing and so the actual Tate pairing value may be required.

5.2 Implementation details

We now give some of the implementation details for the eta pairing in this case. Recall from [10] that, for any point $V \in E(\mathbb{F}_q)$ the function

$$g_V(x, y) = y_V^3 y - (x_V^3 - x + b)^2$$

has divisor $(g_V) = 3(V) + (-3V) - 4(\infty)$.

Consider the eta pairing of P and Q where $T = q - N = \mp 3^{(m+1)/2} - 1$. If $T < 0$ we first replace P by $-P$ and T by $-T$. From Table 1 it follows that we have $T = 3^{(m+1)/2} + b$ when $m \equiv 1, 11 \pmod{12}$ and $T = 3^{(m+1)/2} - b$ when $m \equiv 5, 7 \pmod{12}$.

We are required to compute

$$f_{T,P}(\psi(Q)) = \left(\prod_{i=0}^{(m-1)/2} g_{3^i P}(\psi(Q))^{3^{(m-1)/2-i}} \right) l(\psi(Q))$$

where l is a function corresponding to addition of $3^{(m+1)/2}P$ with $\pm P$. Note that this final addition cannot be ommited since T is not the order of P.

We now explain that the extra addition can be easily handled.

Lemma 5. *With notation as above, let $l(x, y)$ be the line in the final addition of the algorithm. Then l has slope $\lambda = y_P$ if $m \equiv 7, 11 \pmod{12}$ or $\lambda = -y_P$ if $m \equiv 1, 5 \pmod{12}$.*

If $m \equiv 1, 11 \pmod{12}$ the equation for l is $y = \lambda(x - x_P) + by_P$ and if $m \equiv 5, 7 \pmod{12}$ then the equation for l is $y = \lambda(x - x_P) - by_P$.

Proof. The proof is straightforward. For example, when $m \equiv 1 \pmod{12}$ then $T = 3^{(m+1)/2} + b$ and $[3^{(m+1)/2}]P = \phi(x_P^3, y_P^3) = (x_P^3 - b, -y_P^3)$. The slope is therefore $\lambda = (-y_P^3 - y_P)/(x_P^3 - b - x_P)$. Using $x_P^3 - x_P - b = y_P^2 + b$ gives $\lambda = -y_P$. The addition is of $[3^{(m+1)/2}]P$ with bP, from which the equation for $l(x, y)$ follows.

The other cases are similar. □

The exponent $3^{(m-1)/2}$ is inconvenient and a naive implementation would lead to an unnecessary $(m - 1)/2$ cubings. There are two ways to avoid this problem. One method is to bring the powering $3^{(m-1)/2}$ into the formulae as a Frobenius action. The other method, which we adopt here, is to compute the product in reverse, by setting $j = (m - 1)/2 - i$. We define $P' = 3^{(m-1)/2}P$ which can be efficiently computed as $\phi^{(m-1)/2}\pi^{m-1}P = \phi^{(m-1)/2}(x_P^{1/3}, y_P^{1/3})$. Then the desired product is

$$l(\psi(Q)) \prod_{j=0}^{(m-1)/2} g_{3^{-j}P'}(\psi(Q))^{3^j}.$$

It is then relatively straightforward to obtain an explicit description of the algorithm. We use the notation $a^{(i)}$ for a^{3^i}. For example, in the case $m \equiv 1 \pmod{12}$ one shows that

$$g_{3^{-j}P'}(\psi(Q))^{3^j} = (\sigma y_P^{(-j)} y_Q^{(j)} - u^2) - \rho u - \rho^2$$

where $u = x_P^{(-j)} + x_Q^{(j)} + b$. The algorithm in this case is given in Algorithm 2.

Algorithm 2 Computation of $\eta_T(P, Q)$ on $E(\mathbb{F}_{3^m}) : y^2 = x^3 - x + b,\ m \equiv 1 \pmod{12}$ case

```
INPUT: $P, Q$
OUTPUT: $\eta_T(P, Q)$
 1: $P_0 \leftarrow -P$
 2: **if** $T < 0$ **then** $T \leftarrow -T$, $P \leftarrow -P$
 3: **let** $P = (x_P, y_P)$, $Q = (x_Q, y_Q)$
 4: $l \leftarrow$ the line between $3^{(m+1)/2}P$ and $P_0$
 5: $f \leftarrow l(\psi(Q))$
 6: **for** $j \leftarrow 0$ **to** $(m-1)/2$ **do**
 7:     $u \leftarrow x_P + x_Q + b$
 8:     $g \leftarrow \sigma y_P y_Q - u^2 - \rho u - \rho^2$
 9:     $f \leftarrow f \cdot g$
10:     $x_P \leftarrow x_P^{1/3},\ y_P \leftarrow y_P^{1/3}$
11:     $x_Q \leftarrow x_Q^3,\ y_Q \leftarrow y_Q^3$
12: **end for**
13: **return** $f^{(3^{3m}-1)(3^m+1)(3^m-b3^{(m+1)/2}+1)}$
```

Depending on the choice of basis, we can unroll the loop if necessary to exploit the innate sparseness of g. Cube roots can be calculated quickly using the method described by Barreto [1]. However this is still substantially slower than calculating cubes. Therefore it makes sense to do a precalculation to build a list of all cubes of x_P and y_P, and to get the cube roots by accessing this list in reverse order. Note however that we only need the 'last' half of the cubes.

The final exponentiation can be obtained for the relatively inexpensive cost of $(m + 1)/2$ extension field cubings, plus nine applications of the 3^m-power Frobenius, nine extension field multiplications, one extension field squaring, one more cubing, and one extension field division.

If the Tate pairing is required then we should also power to $3T^2/L$. A sensible strategy seems to be to raise the η_T pairing value to $3T^2q/L$ instead, and then to compute the inverse q-power Frobenius π to get rid of the extra q-th power. Namely, if $u = \eta_T(P, Q)$, then $\hat{e}(P, Q) = \pi^{-1}(u^{\mp 3^{(m-1)/2}})u^{\mp 3^{(m+1)/2}-2}$, which amounts to m extra cubings, one squaring, three products, one inverse Frobenius and at most three conjugations, totalling a small amount of extra work compared to the cost of pairing computation.

6 Elliptic curves in characteristic 2

We now consider the case of the supersingular curve[6] $E : y^2 + y = x^3 + x + b$ over $\mathbb{F}_{2^m}$ where $b = 0, 1$ and m is odd. We will use the ideas presented in section 4. The order of E is given in table 2. It follows that the embedding degree in this case is $k = 4$.

The field $\mathbb{F}_{2^{4m}}$ has elements s, t such that $s^2 = s + 1$ and $t^2 = t + s$; we will represent $\mathbb{F}_{2^{4m}}$ using the basis $\{1, s, t, st\}$. Following [3] we use the distortion map $\psi(x, y) = (x + s^2, y + sx + t)$.

[6] As before, there is no loss of generality from considering just one curve equation, as all superpersingular curves in characteristic two are isomorphic over $\overline{\mathbb{F}}_2$ to one other.

Table 2. Order of the curve $E : y^2 + y = x^3 + x + b$ over $\mathbb{F}_{2^m}$, $b \in \mathbb{F}_2$.

$\#E(\mathbb{F}_{2^m})$	condition
$2^m + 1 + (-1)^b 2^{(m+1)/2}$	$m \equiv 1, 7 \pmod 8$
$2^m + 1 - (-1)^b 2^{(m+1)/2}$	$m \equiv 3, 5 \pmod 8$

Let $P = (x_P, y_P) \in E(\mathbb{F}_{2^m})$. We define $\phi(x, y) = (x + 1, y + x)$. One can verify that $\phi^2(x, y) = (x, y + 1) = -(x, y)$, $\phi^3(x, y) = (x + 1, y + x + 1)$, and $\phi^4(x, y) = (x, y)$. One can show by induction that

$$[2^i]P = \phi^i \left(x_P^{(2i)}, y_P^{(2i)} \right) \tag{4}$$

If $q = 2^m$ it follows that $[q]P = \phi^m(P)$ and we are in the setting of our main result (set $\gamma = \phi^m$).

For any field element a we use the notation $a^{(i)}$ for a^{2^i}. Depending on the field of definition of a we will usually have either $a^{(m)} = a$ or $a^{(4m)} = a$. Hence we can consider the values (i) as being modulo m or $4m$. This allows us to extend to negative values by $a^{(-i)} = a^{(4m-i)}$, which can also be interpreted as the 2^i-th root of a.

For future reference we compute how s and t and ψ behave under powers of the 2-power Frobenius. Element s satisfies $s^{(1)} = s^2 = s + 1$, $s^{(2)} = s$, and thus $s^{(i)} = s + i$ and $s^{(-i)} = s^{(4m-i)} = s + i$. Similarly for t, $t^{(1)} = t + s$, $t^{(2)} = t + 1$, $t^{(3)} = t + s + 1$, $t^{(4)} = t$, and thus $t^{(i)} = t + is + \tau(i)$ where $\tau(i) = 0$ for $i \equiv 0, 1 \pmod 4$ and $\tau(i) = 1$ for $i \equiv 2, 3 \pmod 4$. Hence, $t^{(-i)} = t^{(4m-i)} = t + is + \tau(-i)$.

We now show that the eta pairing can be applied in this case.

Lemma 6. *Let notation be as above, in particular, $q = 2^m$ and $\gamma = \phi^m$. Then condition (3) is satisfied.*

Proof. We must show that $\gamma\psi^q = \psi$. Consider first the case $m \equiv 1 \pmod 4$. We have

$$\begin{aligned}\gamma\psi^{2^m}(x, y) &= \phi\psi^2(x, y) = \phi(x + s, y + s^2 x + (t + s)) \\ &= (x + s + 1, y + s^2 x + t + s + x + s) \\ &= (x + s^2, y + sx + t) \\ &= \psi(x, y).\end{aligned}$$

Similarly, when $m \equiv 3 \pmod 4$ we find that

$$\begin{aligned}\gamma\psi^{2^m}(x, y) &= \phi^3\psi^{2^3}(x, y) = \phi^3(x + s, y + s^2 x + (t + s + 1)) \\ &= (x + s + 1, y + s^2 x + t + s + 1 + x + s + 1) \\ &= \psi(x, y).\end{aligned}$$

This completes the proof.

To generalise the Duursma-Lee idea to characteristic 2 is now straightforward using Theorem 1. Let $N = 2^{2m} + 1$ and $M = 2^{2m} - 1$. We take $T = q = 2^m$ (so that $c = 0$) and take $a = 2$ so that $T^2 + 1 = NL$ where $L = 1$. Then the eta pairing satisfies

$$\left(\eta_T(P, Q)^M \right)^{2q} = \langle P, \psi(Q) \rangle_N^M.$$

Adapting the other methods of [10] gives a very fast pairing computation in characteristic 2.

6.1 A further improvement

As in characteristic 3, we can obtain a further halving of the loop.

Let $N = \#E(\mathbb{F}_{2^m}) = 2^m \pm 2^{(m+1)/2} + 1$. For $P \in E(\mathbb{F}_{2^m})$ we have

$$[\mp 2^{(m+1)/2} - 1]P = [2^m - N]P = \gamma(P).$$

Taking $T = \mp 2^{(m+1)/2} - 1$ we have $T = 2^m - N$ so $c = -1$. Taking $a = 2$ gives $T^2 + 1 = 2N$ so $L = 2$. We have $M = ((2^m)^4 - 1)/N = (2^m \mp 2^{(m+1)/2} + 1)(2^{2m} - 1)$. Theorem 1 therefore implies that

$$\left(\eta_T(P, Q)^M\right)^{2T} = \langle P, \psi(Q)\rangle_N^{2M}$$

from which it follows that

$$\left(\eta_T(P, Q)^M\right)^{T} = \langle P, \psi(Q)\rangle_N^{M}.$$

We can therefore easily compute the exact Tate pairing using the eta approach in this case. As before, the halving of the loop is slightly offset by the extra squarings required for the final exponentiation, but we still see an overall gain in performance.

6.2 Implementation details

Let $P, Q \in E(\mathbb{F}_{2^m})$ be the input points for the eta pairing with $T = \mp 2^{(m+1)/2} - 1$. In the case $T < 0$ we replace P by $-P$ and T by $-T$. Thus we have $T = 2^{(m+1)/2} \pm 1$. To compute the eta pairing we must compute the Miller function $f_{T,P}$, which will require $(m + 1)/2$ doublings and an addition. Note that the addition cannot be ommitted since the point P does not have order T.

Given a point V, it is easy to show that the straight line in doubling V is given by

$$g_V(x, y) = (x_V^2 + 1)(x_V + x) + y_V + y.$$

The function g_V has divisor $2(V)+(-2V)-3(\infty)$. Hence, by a standard argument similar to the proof of Lemma 2 combined with the fact that we can disregard functions of x only, we have

$$f_{T,P}(\psi(Q)) = \left(\prod_{i=0}^{(m-1)/2} \left(g_{[2^i]P}(\psi(Q))\right)^{2^{(m-1)/2-i}}\right) l(\psi(Q))$$

where the function l comes from the elliptic curve addition of $[2^{(m+1)/2}]P$ with $\pm P$.

The power $2^{(m-1)/2}$ is somewhat inconvenient. A naive implementation might involve $(m - 1)/2$ unnecessary squarings because of it. There are two ways around this problem, which both give equally efficient solutions. One solution would be to absorb the powering by $2^{(m-1)/2}$ into the equations. Another solution is to re-write the expression by substituting $j = 2^{(m-1)/2} - i$ and $P' = [2^{(m-1)/2}]P$. This gives

$$f_{T,P}(\psi(Q)) = l(\psi(Q)) \prod_{j=0}^{(m-1)/2} g_{[2^{-j}]P'}(\psi(Q))^{2^j}.$$

Note that, due to our doubling formula, the 'point halving' in $[2^{-j}]P'$ has the same efficiency as point doubling. Also note that $P' = \phi^{(m-1)/2}(\sqrt{x_P}, \sqrt{y_P})$.

We now give some formulae which allow us to present an efficient and general algorithm. The first result gives the equation of the line l for the final addition (note that no inversions are required to compute this). The proof of this result is straightforward.

Lemma 7. *Let m and b be as above. Define $\epsilon = -1$ when $m \equiv 1, 7 \pmod 8$ and $b = 1$ or when $m \equiv 3, 5 \pmod 8$ and $b = 0$. Define $\epsilon = 1$ in all other cases. Then T is taken to be $2^{(m+1)/2} + \epsilon$.*

Let $P = (x_P, y_P)$. Define $\lambda = x_P$ when $m \equiv 1, 5 \pmod 8$ and $\lambda = x_P + 1$ when $m \equiv 3, 7 \pmod 8$. Then the formula for the line $l(x, y)$ through $2^{(m+1)/2}P$ and ϵP is given by $l(x, y) = y + \lambda(x + x_P) + y_P + (1 - \epsilon)/2$.

Lemma 8. *Let notation be as above. Define $v_1 = 1$ if $m \equiv 1, 5 \pmod 8$ and $v_1 = 0$ otherwise. Define $v_2 = 1$ if $m \equiv 5, 7 \pmod 8$ and $v_2 = 0$ otherwise. Define $u = x_P^{(-j)} + v_1$. Then the function $g_{[2^{-j}]P'}(\psi(Q))^{2^j}$ is given by*

$$u(x_P^{(-1-j)} + x_Q^{(j)} + v_1) + y_P^{(-1-j)} + y_Q^{(j)} + (1 - v_1)x_P^{(-1-j)} + s(u + x_Q^{(j)}) + t + v_2.$$

Proof. Note that $[2^{-j}]P' = \phi^{(m-1)/2-j}(x_P^{(-1-2j)}, y_P^{(-1-2j)})$. The result is proved by a tedious case-by-case analysis.

For example, when $m \equiv 3 \pmod 8$ and $j \equiv 2 \pmod 4$ then $(m-1)/2 - j \equiv 3 \pmod 4$ and so $[2^{-j}]P' = (x_P^{(-1-2j)} + 1, y_P^{(-1-2j)} + x_P^{(-1-2j)} + 1)$. The function is therefore

$$\begin{aligned}&\left((x_P^{(-2j)})(x_P^{(-1-2j)} + x_Q + s) + y_P^{(-1-2j)} + x_P^{(-1-2j)} + 1 + y_Q + s x_Q + t\right)^{2^j}\\ &= x_P^{(-j)}(x_P^{(-1-j)} + x_Q^{(j)} + s) + y_P^{(-1-j)} + x_P^{(-1-j)} + 1 + y_Q^{(j)} + s x_Q^{(j)} + t + 1.\end{aligned}$$

Letting $u = x_P^{(-j)}$ this simplifies to

$$u(x_P^{(-1-j)} + x_Q^{(j)}) + y_P^{(-1-j)} + x_P^{(-1-j)} + y_Q^{(j)} + s(u + x_Q^{(j)}) + t$$

as required. The other 15 cases are similar. □

Here we give the algorithm in the case of a curve with $m \equiv 3 \pmod 8$.

In this optimized algorithm the point addition is dealt with first, using the formula of Lemma 7.

Each step in the subsequent loop costs 7 $\mathbb{F}_q$ multiplications (1 to compute g, 6 to accumulate it into f by making use of the sparse structure of g). In practise to obtain this speed-up we might have to unroll the loop times 2 (depending on the basis chosen). The total cost of the loop plus the initial point addition is therefore $7(m + 1)/2 + 1$

Algorithm 3 Computation of $\eta_T(P,Q)$ on $E(\mathbb{F}_{2^m}) : y^2 + y = x^3 + x + b,\ m \equiv 3 \pmod 8$ case

```
INPUT: $P, Q$
OUTPUT: $\eta_T(P,Q)$
 1: let $P = (x_P, y_P)$, $Q = (x_Q, y_Q)$
 2: $u \leftarrow x_P + 1$
 3: $f \leftarrow u \cdot (x_P + x_Q + 1) + y_P + y_Q + b + 1 + (u + x_Q)s + t$
 4: for $i \leftarrow 1$ to $(m+1)/2$ do
 5:     $u \leftarrow x_P$, $x_P \leftarrow \sqrt{x_P}$, $y_P \leftarrow \sqrt{y_P}$
 6:     $g \leftarrow u \cdot (x_P + x_Q) + y_P + y_Q + x_P + (u + x_Q)s + t$
 7:     $f \leftarrow f \cdot g$
 8:     $x_Q \leftarrow x_Q^2$, $y_Q \leftarrow y_Q^2$
 9: end for
10: return $f^{(2^{2m}-1)(2^m-2^{(m+1)/2}+1)}$
```

multiplications. The final exponentiation can be obtained for the relatively inexpensive cost of $(m+1)/2$ extension field squarings, plus three applications of the 2^m-power Frobenius, four extension field multiplications, and one extension field division. To obtain the Tate pairing, the result must be further exponentiated to the power of T.

Unlike the case of characteristic three, there seems to be no reason to precompute and store the square roots. This is because a careful implementation of the technique described in [12] for calculating square roots in the field $\mathbb{F}_{2^m}$ is in fact just as fast as squaring. In fact it may be a little faster, as large precomputed tables can lead to memory cache misses which are detrimental to performance.

6.3 Compression of pairing elements

Pairing values lie in the subgroup of order q^2+1 in $\mathbb{F}_{q^4}^*$, which is the torus $T_2(\mathbb{F}_{q^2})$. Hence it is trivial to compress pairing values by a factor of 2 using standard torus or trace methods.

In the case of the eta pairing with $T = q$, the method of Granger, Page and Stam [17] avoids performing the final exponentiation (though note that their compression method requires an inversion so is of similar complexity to the final exponentiation anyway). Their idea is to note that the pairing value e can be written as $e = gh = e_0 + e_1 t$ where g has order dividing (q^2+1) and where $h, e_0, e_1 \in \mathbb{F}_{q^2}$. Hence, the value e_0/e_1 is a uniquely defined element in $\mathbb{F}_{q^2}$ corresponding to the class of e.

When using the eta pairing with $T = q - N$ the method of Granger, Page and Stam to avoid the final exponentiation cannot be applied. Nevertheless, once the final exponentiation has been performed one can compress by a factor of 2 using traces or tori in the standard way.

7 Genus 2 curves in characteristic 2

We now consider the curve[7] $C_d : y^2 + y = x^5 + x^3 + d$ with $d = 0$ or 1 over $\mathbb{F}_{2^m}$, where m is coprime to 6. This curve is supersingular and has embedding degree 12 (see [16]). The group order is given in Table 3; some examples are listed in Table 4.

In this section we show that the eta pairing approach can be easily applied in this setting. We give an octupling formula which enables fast point exponentiation and give a corresponding function for Miller's algorithm.

A general reduced divisor D on a genus 2 curve has support consisting of two affine points (i.e., $D = (P_1) + (P_2) - 2(\infty)$). Following Duursma and Lee [10] (also see [18, 19]) we exploit the benefits of using *degenerate divisors*[8] of the form $D = (P) - (\infty)$ where possible.

Table 3. Order of Jac(C_d) for the curve $C_d : y^2 + y = x^5 + x^3 + d$ over $\mathbb{F}_{2^m}$, $d \in \mathbb{F}_2$.

$\#\mathrm{Jac}(C_d)(\mathbb{F}_{2^m})$	condition
$2^{2m} + (-1)^d 2^{(3m+1)/2} + 2^m + (-1)^d 2^{(m+1)/2} + 1$	$m \equiv 1,\ 7, 17, 23 \pmod{24}$
$2^{2m} - (-1)^d 2^{(3m+1)/2} + 2^m - (-1)^d 2^{(m+1)/2} + 1$	$m \equiv 5, 11, 13, 19 \pmod{24}$

Table 4. Examples where $\#\mathrm{Jac}(C)(\mathbb{F}_{2^m})$ is equal to a small cofactor times a prime.

field	curve	cofactor
$\mathbb{F}_{2^{79}}$	$y^2 + y = x^5 + x^3 + 1$	151681
$\mathbb{F}_{2^{103}}$	$y^2 + y = x^5 + x^3$	$13 \cdot 1237$
$\mathbb{F}_{2^{127}}$	$y^2 + y = x^5 + x^3 + 1$	198168459411337
$\mathbb{F}_{2^{199}}$	$y^2 + y = x^5 + x^3 + 1$	$2389 \cdot 121789$
$\mathbb{F}_{2^{239}}$	$y^2 + y = x^5 + x^3 + 1$	1
$\mathbb{F}_{2^{313}}$	$y^2 + y = x^5 + x^3 + 1$	1

We note that our special curve has certain properties that lend to faster arithmetic than the explicit formulae given in [22] for general genus 2 curves, namely that the equation is sparse and that all the coefficients are defined over $\mathbb{F}_2$. See [23] for examples of arithmetic on similar (albeit non-supersingular) curves.

7.1 Arithmetic on the curves

We first introduce the representation of $\mathbb{F}_{2^{12}}$ and the distortion map we will be using.

[7] In this case not all supersingular equations are isomorphic over $\overline{\mathbb{F}}_2$. Nevertheless, the Jacobians are isogenous, so there seems to be no good reason to consider other curve equations.

[8] Note that the definition of degenerate divisors in [18, 19] is that they have less than g points in their support, whereas our definition is tougher when $g > 2$ in that we insist on having exactly one point in the support.

Choose $w \in \mathbb{F}_{2^6}$ to be a root of the polynomial

$$x^6 + x^5 + x^3 + x^2 + 1.$$

Note that $w^8 = w + 1$. Define $s_1 = w^2 + w^4$, $s_2 = w^4 + 1$, and let $s_0 \in \mathbb{F}_{2^{12}}$ be a solution of $s_0^2 + s_0 = w^5 + w^3$.

We will represent elements of the field $\mathbb{F}_{2^{12m}}$ as 12-tuples with respect to the basis

$$\{1, w, w^2, w^3, w^4, w^5, s_0, ws_0, w^2s_0, w^3s_0, w^4s_0, w^5s_0\}.$$

We choose the distortion map

$$\psi(x, y) = (x + w, y + s_2x^2 + s_1x + s_0).$$

We now consider the octupling formula and hence determine when $[2^{3m}]D$ can be written as $\gamma(D)$ for some γ.

Consider a divisor of form $D = (P) - (\infty)$. In general, jD is not equivalent to a divisor of the form $(Q) - (\infty)$, but as shown in Appendix A, in this case we have the octupling formula $8D = (P') - (\infty)$ where $P' = \phi\pi^6(P)$, π is the 2-power Frobenius map, and

$$\phi(x, y) = (x + 1, y + x^2 + 1).$$

Note that $\phi^2 = -1$. As a suggestive (but non-standard) notation, we write $[8]P = \phi\pi^6(P)$ and so $8D = ([8]P) - (\infty)$. Koblitz [20] gives a map for $64D$, which is exactly the octupling operation applied twice in succession. Similar results for other supersingular curves were obtained by Duursma and Lee [10].

Since our basic operation is octupling, we are forced to consider the η pairing in the case where we have a power of 2^3. Hence we will work with $q = 2^{3m}$ rather than $q = 2^m$. Nevertheless, since our basic operation is octupling, our loops will still have at most m iterations. It follows that, if D is a divisor class defined over $\mathbb{F}_{2^m}$, then

$$[q]D = [2^{3m}]D = \phi^m(D)$$

hence we define $\gamma = \phi^m$.

We also note that it is possible to use the octupling operation for straightforward scalar multiplication, which yields a simple and speedy implementation.

7.2 Eta pairings in genus 2

We now show that condition (3) is satisfied for our distortion map.

Lemma 9. *Let the notation be as above with $q = 2^{3m}$. Then condition (3) is satisfied.*

Proof. We have $q = 2^{3m}$ where $m \equiv 1, 5, 7 \text{ or } 11 \pmod{12}$ and $\gamma = \phi^m$.

As before, we write $a^{(i)}$ for a^{2^i}. We will repeatedly use the easily checked formulae that $w^{(3)} = w + 1$, $s_0^{(3)} = s_0 + w^2$, $s_1^{(3)} = s_1$ and $s_2^{(3)} = s_2 + 1$.

First suppose $m \equiv 1 \pmod 4$ (and so $3m \equiv 3 \pmod{12}$). Then

$$\begin{aligned}\gamma\psi^q(x,y) &= \phi\psi^{(3)}(x,y)\\ &= \phi(x+w+1, y+(s_2+1)x^2+s_1x+s_0+w^2)\\ &= (x+w, y+s_2x^2+s_1x+s_0)\\ &= \psi(x,y).\end{aligned}$$

Similarly, when $m \equiv 3 \pmod 4$ we have $\gamma = -\phi$ and $3m \equiv 9 \pmod{12}$. Hence

$$\begin{aligned}\gamma\psi^q(x,y) &= -\phi^3\psi^{(9)}(x,y)\\ &= -\phi(x+w+1, y+(s_2+1)x^2+s_1x+s_0+w^2+1)\\ &= (x+w, y+s_2x^2+s_1x+s_0)\\ &= \psi(x,y).\end{aligned}$$

This proves the lemma. □

Having established this, we can now apply Theorem 1. For the basic generalisation of Duursma and Lee we take $T = q = 2^{3m}$ so that $c = 0$. We have $N = 2^{6m}+1$ so that $M = 2^{6m}-1$. We then take $a = 2$ so that $q^2+1 = N$ and $L = 1$. It therefore follows that

$$\left(\eta_T(D,D')^M\right)^{2q} = \langle D, \psi(D')\rangle_N^M.$$

Computing the eta pairing with respect to $T = 2^{3m}$ requires m iterations of the octupling formula. Interestingly, this is not much better than the basic BKLS-GHS method generalised to genus 2 (see Section 7.7).

As before, we can obtain a further halving of the loop. Let $q = 2^{3m}$ and $N = 2^{2m} \pm 2^{(3m+1)/2} + 2^m \pm 2^{(m+1)/2} + 1$. Consider the equation

$$(2^m \mp 2^{(m+1)/2} + 1)N = 2^{3m} \pm 2^{(3m+1)/2} + 1.$$

This suggests taking $T = \mp 2^{(3m+1)/2} - 1$ so that, if D is a divisor defined over $\mathbb{F}_q$ (and hence of order dividing N),

$$[T]D = [q - (2^m \mp 2^{(m+1)/2} + 1)N]D = [q]D = \gamma(D).$$

Then $c = -(2^m \mp 2^{(m+1)/2} + 1)$ in the notation of Theorem 1. Taking $a = 2$ gives $T^2+1 = LN$ where $L = 2^{m+1} \mp 2^{(m+3)/2} + 2$. Theorem 1 therefore implies that

$$\left(\eta_T(D,D')^M\right)^{2T} = \left(\langle D, \psi(D')\rangle_N^M\right)^L.$$

Computing the eta pairing using $T = \mp 2^{(3m+1)/2} - 1$ will require roughly $m/2$ iterations of the octupling formula, which is clearly superior to the BKLS-GHS method.

7.3 Implementation Details

We compute the η_T pairing of divisors D and D' using the order $T = \mp 2^{(3m+1)/2} - 1$. As usual, if $T < 0$ then set $T = -T$ and $D = -D$. Hence we have $T = 2^{(3m+1)/2} \pm 1$.

We will exploit the octupling formula. Let $f_{8,P}$ be a function such that $(f_{8,P}) = 8(P) - ([8]P) - 7(\infty)$. We show in appendix A that,

$$f_{8,P}(x,y) = \frac{(y + b_4(x))^2 (y + b_8''(x))}{a_4'(x)^2 a_8'(x)}$$

where

$$b_4(x) = x^3 + (x_P^8 + x_P^4)x^2 + (x_P^4)x + y_P^4$$

and

$$b_8''(x) = (x_P^{32} + 1)x^2 + (x_P^{32} + x_P^{16})x + (y_P^{16} + x_P^{16} + x_P^{48} + 1).$$

The denominator $a_4'(x)^2 a_8'(x)$ can be ignored for the usual reasons. We will be composing our function with ψ, and so we will use the notation $\alpha\beta$ for the function, where $\alpha = (y + b_4(x))^2 \circ \psi$ and $\beta = (y + b_8'') \circ \psi$.

To compute a function $f_{T,P}$ with divisor $T(P) - (D_{T,P}) - n(\infty)$ where $D_{T,P}$ is a reduced divisor equivalent to TD we want to use the above functions. Since $(3m+1)/2 = 3(m-1)/2 + 2$ we must use $(m-1)/2$ octuplings and two doublings.

In other words, we have

$$f_{T,P}(\psi(Q)) = \left(\prod_{i=0}^{(m-3)/2} f_{8,[8^i]P}(\psi(Q))^{2^{(3m-5)/2-3i}} \right) l_1(\psi(Q))^2 l_2(\psi(Q)) l_3(\psi(Q))$$

where l_1 and l_2 are functions coming from the extra doublings and l_3 is from the final addition of $2^{(3m+1)/2}((P) - (\infty))$ and $\pm((P) - (\infty))$.

Appendix B gives more detail on implementing the pairing. In Appendix B.1 the distortion map is built into the functions, and formulae are derived for $f_{8,[8^i]P}(\psi(Q))$ which do not require the explicit computation of $[8^i]P$. These formulae are computed efficiently by accessing a table of precomputed values for $x_P^{2^i}$ and $y_P^{2^i}$. In Appendix B.4 powers of 8 are absorbed into the formulae, which involves precomputing powers of the second point. Appendix B.7 describes how the final doublings and addition can be simplified for degenerate divisors. Algorithm 4 details the genus 2 η_T pairing for degenerate divisors in the case when $m = 103$ (although only trivial changes are required to modify the algorithm for arbitrary m).

All of these optimisations lead to a very fast pairing implementation on genus 2 curves in characteristic two. Details of timings are given in Section 10.

7.4 Degenerate divisors versus general divisors

We have focussed on the case of degenerate divisors since this gives a convincing example where hyperelliptic curves can be superior to elliptic curves. Nevertheless, most applications will also require pairings to be computed on general divisors.

A general reduced divisor defined over $\mathbb{F}_q$ on a genus 2 curve C is represented by $(P_1) + (P_2) - 2(\infty)$ where either $P_1, P_2 \in C(\mathbb{F}_q)$ or $P_1, P_2 \in C(\mathbb{F}_{q^2})$ are Galois conjugates of each other. The Mumford representation for divisors on hyperelliptic curves essentially gives the symmetric functions of the coordinates of the points P_i in the support of the divisor.

Algorithm 4 The genus 2 η_T pairing when $m = 103$

```
INPUT: $P = (x_P, y_P), Q = (x_Q, y_Q) \in J_C(F_{2^m})$
OUTPUT: $f \in F_{2^{12m}}$
 1: ▹ Initialisation: set $\gamma = 1$ if $m \equiv 1 \bmod 4$, otherwise $\gamma = 0$
 2: ▹ Precompute powers of P and Q
 3: $x_1[i] \leftarrow x_P^{2^i}, y_1[i] \leftarrow y_P^{2^i}, x_2[i] \leftarrow x_Q^{2^i}, y_2[i] \leftarrow y_Q^{2^i}, 0 <= i <= m - 1$
 4: $f \leftarrow 1$
 5:
 6: for $i = 0$ to $(m - 3)/2$ do
 7:     ▹ All $k_*$ in the next 2 lines to be considered modulo $m$
 8:     $k_1 \leftarrow (3m - 9 - 6i)/2, k_2 \leftarrow (k_1 + 1), k_3 \leftarrow (k_2 + 1)$
 9:     $k_4 \leftarrow (3m - 3 + 6i)/2, k_5 \leftarrow (k_4 + 1), k_6 \leftarrow (k_5 + 1)$
10:
11:     ▹ Calculate $\alpha \leftarrow a + bw + cw^2 + dw^4 + s_0$
12:     $d \leftarrow x_1[k_4] + x_1[k_5]$
13:     $a \leftarrow y_2[k_2] + (x_1[k_4] + 1 + x_2[k_3]) \cdot x_2[k_2] + d \cdot x_2[k_3] + y_1[k_4] + \gamma$
14:     $b \leftarrow x_2[k_3] + x_2[k_2]$
15:     $c \leftarrow x_2[k_3] + x_1[k_4] + 1$
16:
17:     ▹ Calculate $\beta \leftarrow e + f_2 w + gw^2 + hw^4 + s_0$
18:     $f_2 \leftarrow x_1[k_5] + x_1[k_6]$
19:     $e \leftarrow y_2[k_1] + f_2 \cdot x_2[k_1] + y_1[k_5] + x_1[k_6] \cdot (x_1[k_5] + x_2[k_2]) + x_1[k_5] + \gamma$
20:     $g \leftarrow x_2[k_1] + x_1[k_6] + 1$
21:     $h \leftarrow x_2[k_2] + x_2[k_1]$
22:
23:     $f \leftarrow f \cdot (\alpha \cdot \beta)$
24: end for
25:
26: ▹ "Extract" current point $(x_P, y_P)$
27: $x_P \leftarrow x_1[100] + 1$
28: $y_P \leftarrow y_1[100] + x_1[101]$
29:
30: ▹ Perform the final doublings/addition
31: $t \leftarrow (y_2[0] + x_2[1] \cdot (1 + x_2[0] + x_P^8 + x_P^4) + x_P^4 \cdot x_2[0] + y_P^4)$
32: $f \leftarrow f^4 \cdot (t, x_2[1] + x_P^4, x_P^8 + x_P^4, 1, x_2[1] + x_2[0], 0, 1, 0, 0, 0, 0, 0)$
33:
34: ▹ Perform the final exponentiation
35: $f \leftarrow f^{(2^{6m}-1)(2^{3m}-2^{4m}2^{(m+1)/2}-1)}$
36:
```

General divisors may appear as either the first or second components of the pairing (or both). Handling the second case (i.e., generalising evaluation of a function at a point to evaluation at a divisor in Mumford representation) is relatively straightforward.

For the first case, bilinearity implies that $\eta_T((P_1) + (P_2) - 2(\infty), D') = \eta_T(P_1, D')\eta_T(P_2, D')$ and so one can compute a pairing on divisors by taking a product of pairings on points. However, in the case where the points P_i are actually defined over $\mathbb{F}_{q^2}$ this will not be the most efficient way to proceed. It is relatively straightforward to obtain the general formulae: just multiply the functions obtained from the single point case and then express the resulting polynomials in terms of the symmetric polynomials in the point coordinates. We leave this as an exercise for the reader; a full discussion will be given in the thesis of the third author.

From a performance point of view, the cost of computing a pairing between general divisors in genus 2 is at worst 4 times the cost of a pairing between single points. Obviously, various optimisations are applicable, including only performing the final exponentiation once, sharing some of the function calculations and only having to precompute squarings of points once.

We now briefly discuss how degenerate divisors can be used to speed up pairing-based cryptosystems. First, note that Katagi *et al.* [19] showed (using the random self-reducibility of the discrete logarithm problem) that there is no loss of security from using degenerate divisors.

In most of the cases we consider, the divisor class group has (nearly) prime order N and we work with pairings of order N. Hence, a randomly chosen degenerate divisor $(P) - (\infty)$ will have order divisible by our large prime and the pairing value will be non-degenerate. In the general case of pairings on higher genus curves this assumption may not hold; we refer to Frey and Lange [13] for a discussion of these issues.

In the case of pairing-based cryptography it is easy to benefit from the use of degenerate divisors. As a case study we consider the Boneh-Franklin identity-based encryption scheme [7] (similar ideas can speed up aspects of other pairing-based cryptosystems, we refer to Frey and Lange [13] for further discussion). The natural generalisation of this system to genus g curves (see [16]) is to have a master public key pair $D, D_{\text{pub}} = [s]D$ of divisors; identities are hashed to obtain divisors $D(ID)$; user private keys are $[s]D(ID)$; encryption involves computing $[r]P$ and the pairing of D_{pub} with $D(ID)$ (and then raising to the power r); decryption involves the pairing of $[s]D(ID)$ with $[r]P$.

Without loss of security, one can choose several of these divisors to be degenerate. For example, one can choose D_{pub} to be degenerate (i.e., choose D_{pub} first and then set $D = [s^{-1}]D_{\text{pub}}$). One can also choose $H(ID)$ to be degenerate, so that we are hashing to points on the curve, rather than general divisors. This simplified hashing process is also easier to implement than the general case. Of course, the user private keys are now general divisors. Encryption therefore involves a pairing of two degenerate divisors and so can be performed very efficiently, while decryption involves a pairing of general divisors. This is similar to RSA with small public exponents, where the public operations are fast compared to the private ones. For some applications this may be a useful feature.

7.5 The Final Exponentiation

Theorem 1 relates the η_T pairing to the Tate pairing in the genus 2 case as follows

$$\left(\eta_T(D, D')^M\right)^{2T} = \left(\langle D, \psi(D')\rangle_N^M\right)^L.$$

where $T = \mp 2^{(3m+1)/2} - 1$, $M = (2^{12m} - 1)/N$, $N = 2^{2m} \pm 2^{(3m+1)/2} + 2^m \pm 2^{(m+1)/2} + 1$ and $L = 2^{m+1} \mp 2^{(m+3)/2} + 2$.

One can compute a bilinear pairing by computing the η_T pairing and raising to the power of M. However, it is actually more efficient to compute the full Tate pairing with $\eta_T^{\frac{M2T}{L}}$. By factoring M we get the product

$$M = (2^{6m} - 1)(2^m \mp 2^{(m+1)/2} + 1)(2^{3m} \mp 2^{(3m+1)/2} + 1)$$

Note that L can be written as $L = 2(2^m \mp 2^{(m+1)/2} + 1)$, which cancels out with the middle factor of M and the squaring of the η_T function. The exponent to compute the Tate pairing is now;

$$(2^{6m} - 1)(2^{3m} \mp 2^{(3m+1)/2} + 1)(\mp 2^{(3m+1)/2} - 1)$$

Some cancellations occur whilst unrolling the multiplication of the second and third factor, and we get

$$(2^{6m} - 1)(2^{3m} \mp 2^{4m} 2^{(m+1)/2} - 1)$$

Note that raising an element to the power of 2^{6m} over $\mathbb{F}_{2^{12m}}$ can be computed with a simple conjugation. Also note that, once the powering to $(2^{6m} - 1)$ has been performed, we have $z^{2^{6m}+1} = 1$ and so $z^{-1} = z^{2^{6m}}$, i.e., computing an inverse is done by simple conjugation. Using these facts, we can compute the final exponentiation in $(m + 1)/2$ squarings, 4 Frobenius actions, 2 multiplications and a division.

7.6 Compression of pairing values

After the final exponentiation our pairing values lie in the subgroup of order $(q^4 - q^2 + 1)$ in $\mathbb{F}_{q^{12}}^*$. This subgroup is the torus $T_6(\mathbb{F}_{q^2})$ so we can represent the field elements using 2 elements of $\mathbb{F}_{q^2}$ rather than 6. This gives compression by a factor of 3. The details are similar to those given by [17].

7.7 Computing BKLS-GHS using octupling

In this section we will briefly look at computing the Tate pairing using the BKLS-GHS algorithm, degenerate divisors and the fast octupling operation defined previously. At first glance, the group order $N \approx 2^{2m}$ requires about $2m/3$ iterations of the octupling formula, so the BKLS-GHS method looks competitive. However, care is needed as the additions will destroy the special form of the divisor. The best approach is to postpone the additions until the end. In other words, compute the appropriate functions for $2^{2m}P$, $2^{(3m+1)/2}P$, $2^m P$ and $2^{(m+1)/2}P$ separately in different loops and then add them up at the end.

Let h_1, h_2, h_3 be the functions that arise from Cantor composition and reduction of the divisors that occur at $2^{2m}P$, $2^{(3m+1)/2}P$, $2^m P$ and $2^{(m+1)/2}P$. The function we desire is then:

$$f = f_{2^{2m}} f_{2^{(3m+1)/2}} f_{2^m} f_{2^{(m+1)/2}} h_1 h_2 h_3$$

This can be computed using $2m/3$ octuplings, as well as a few additions and doublings. The powers of 8 can also be absorbed as is done in Appendix B, however as each function is raised to a different power, we require four different sets of formulae. We can take some advantage of similarities between the four functions however, to speed up the computation. The end result is an efficient if messy pairing computation. See section 10 for timings.

8 The Duursma-Lee hyperelliptic curves

Duursma and Lee [10] also consider the curves $C : y^2 = x^p - x + d$ over $\mathbb{F}_{p^m}$ where $p \geq 5$ and $d \neq 0$. The genus of C is $(p-1)/2$. When $p \equiv 3 \pmod 4$ the embedding degree is $k = 2p$ and $\#\mathrm{Jac}(C)(\mathbb{F}_{p^m}) \mid (p^{mp} + 1)$. The distortion map is $\psi(x, y) = (\rho - x, iy)$ where $i^2 = -1$ and $\rho^p - \rho + 2d = 0$.

Duursma and Lee [10] show that $p((x, y) - (\infty))$ is equivalent to $(x^{p^2} + 2d, -y^{p^2})$. Let $q = p^m$ where m is coprime to $2p$. Let $\phi(x, y) = (x + 2d, -y)$. If $P \in C(\mathbb{F}_{p^m})$ then it follows that $q((P) - (\infty))$ is equivalent to $(\phi^m(P)) - (\infty)$. Hence we set $\gamma = \phi^m$.

We check condition (3) for this case. Note that $\rho^p = \rho - 2d$.

If $p \equiv 3 \pmod 4$ then we have (since m is odd)

$$\begin{aligned} \gamma\psi^q &= \phi^m \psi^{p^m}(x, y) \\ &= \phi^m(\rho^{p^m} - x, i^{p^m} y) \\ &= \phi^m(\rho - 2dm - x, -iy) \\ &= (\rho - x, iy) \\ &= \psi(x, y). \end{aligned}$$

Having established this, we set $N = p^{pm} + 1$, $T = p^m$, $c = 0$, $a = p$ and $L = 1$ and apply Theorem 1 to show that the eta pairing approach recovers the results of Duursma and Lee.

If $p \equiv 1 \pmod 4$ then the embedding degree is $k = p$ and our methods do not immediately apply. Theorem 1 can be generalised so that condition 3 reads $T^a - 1 = LN$, in which case we may choose $a = p$. But when $p^m \equiv 1 \pmod 4$ we have $i^{p^m} = i$ and so condition (3) is not satisfied.

There are two natural open problems for these curves. The first is to develop an eta pairing for the cases where $p \equiv 1 \pmod 4$. The second natural problem is to give the further halving of the loop for these curves. We leave these problems for future research.

9 The Rubin-Silverberg approach

Rubin and Silverberg [25] (also see [26, 29]) have proposed an alternative way to view pairings on Abelian varieties. Their method can be thought of as a method for computing pairings on trace zero subvarieties of Weil restrictions of elliptic curves. A simpler way to think about their method is as a form of point compression for pairings on elliptic curves.

The Jacobian of the supersingular genus 2 curve considered in Section 7 is a 2-dimensional Abelian variety over $\mathbb{F}_{2^m}$ with embedding degree $k = 12$. Another way to get this Abelian variety (up to isogeny) is with the Rubin-Silverberg approach, by taking the $k = 4$ elliptic case and using the Rubin-Silverberg construction with $r = 3$, to obtain a 2-dimensional Abelian variety with embedding degree $3 \times 4 = 12$. We recall the details in this case.

Let $E_b : y^2 + y = x^3 + x + b$ with $b = 0, 1$ over $\mathbb{F}_2$ be the supersingular elliptic curves with embedding degree $k = 4$. The idea of Rubin and Silverberg is to compute pairings with points defined over $\mathbb{F}_{2^{3m}}$ where m is coprime to 12. This means that pairing values lie in $\mathbb{F}_{2^{12m}}$.

To transmit group elements, Rubin and Silverberg propose a compression method so that the element is represented using only 2 elements in $\mathbb{F}_{2^m}$. Hence, the bandwidth is about $2m$ bits but the finite field security is 2^{12m}, which corresponds to 'security multiplier 6'.

The group $E(\mathbb{F}_{2^{3m}})$ clearly has $E(\mathbb{F}_{2^m})$ as a subgroup. Indeed, we can write

$$E(\mathbb{F}_{2^{3m}}) \cong E(\mathbb{F}_{2^m}) \times A$$

where A is a finite group and one can check that the order of A is $2^{2m} \pm 2^{(3m+1)/2} + 2^m \pm 2^{(m+1)/2} + 1$. Note that this agrees with the group orders in Table 3.

The following result is an important classification of A.

Lemma 10. *Let m be coprime to 12. Let* Tr *be the trace map with respect to $\mathbb{F}_{2^{3m}}/\mathbb{F}_{2^m}$. Then $A = \{P \in E(\mathbb{F}_{2^{3m}}) : \mathrm{Tr}(P) = 0\}$.*

The method is to perform pairing computations with the curve E over $\mathbb{F}_{2^{3m}}$ so that the pairing values lie in $\mathbb{F}_{2^{12m}}$. Suitable group orders are the same as in the genus 2 case (e.g. $m = 103$ with a 192-bit subgroup).

Write $\mathbb{F}_{2^{3m}}$ as $\mathbb{F}_{2^m}(\theta)$ where $\theta^3 = \theta + 1$. So points in $E(\mathbb{F}_{2^{3m}})$ are represented as (x, y) where x is represented as a triple (x_0, x_1, x_2) over $\mathbb{F}_{2^m}$ with respect to the basis $\{1, \theta, \theta^2\}$.

To transmit a point we first apply point compression so that we need send only the x-coordinate and a single bit determining the sign (sometimes even this bit can be removed). Then to transmit the x-coordinate just send x_0 and x_1 (and possibly another bit).

To recover (decompress) we must do the following: Given x_0 and x_1 compute an element $x_2 \in \mathbb{F}_{2^m}$ such that there is a point P with x-coordinate (x_0, x_1, x_2) which satisfies $\mathrm{Tr}(P) = 0$.

In this case the trace is $\mathrm{Tr}(P) = P + \pi(P) + \pi^2(P)$ where π is the 2^m-power Frobenius map. So the condition is that $P, \pi(P)$ and $\pi^2(P)$ sum to zero, or in other words, lie on a straight line. The decompression procedure is to deduce which x_2 ensures that there is a line $l(x, y) = 0$ through the three points.

Let $P = (x_P, y_P)$ and write $l(x, y) = y + u_0 x + u_1$ where $u_0, u_1 \in \mathbb{F}_{2^{3m}}$. Define $\bar{l}(x, y) = l(x, y) + 1$. Then

$$\begin{aligned} l(x, y)\bar{l}(x, y) &= y^2 + y + (u_0 x + u_1) + (u_0 x + u_1)^2 \\ &= x^3 + u_0^2 x^2 + (u_0 + 1)x + (u_1^2 + u_1 + b). \end{aligned}$$

Also,

$$l(x,y)\bar{l}(x,y) = (x - x_P)(x - x_{\pi(P)})(x - x_{\pi^2(P)}) = x^3 + Tx^2 + Sx + N$$

from which we deduce that $T = S^2 + 1$. One can check that $T = x_0$ and $S = x_0^2 + x_1^2 + x_1x_2 + x_2^2$. Hence x_2 is a solution to the equation

$$y^4 + x_1y^2 + (x_0^4 + x_0 + 1 + x_1^2) = 0.$$

Solving this equation involves solving a quadratic and then taking square roots. A single bit is needed to distinguish the two roots of the quadratic and to ensure a unique solution to the decompression process.

The total cost is solving a quadratic and then taking a square root, plus solving another quadratic to recover the y-coordinate of the point.

From a performance point of view it is essential to compare the running time of the pairing computation on the genus 2 curve with the Rubin-Silverberg method. In a general implementation, where we may be required to compute the pairing of general divisors on the genus 2 curve, then the Rubin-Silverberg approach may be superior.

10 Experimental results

We have proposed a number of algorithms for pairing computation which apply to different supersingular curves. The only natural way to compare these methods is to give running times for equivalent security levels. A precise formulation of 'equivalent security' is deeply problematic, but a reasonable approach is to consider parameters so that the value q^k is roughly the same. This would mean that the cost of index calculus in the finite field is roughly equal for all examples. Two different field sizes are chosen for testing. Firstly, 950-bit finite fields with (g=1) $q = 2^{239}$, (g=1) $q = 3^{97}$, (g=2) $q = 2^{79}$, and 1230-bit finite fields with (g=1) $q = 2^{307}$, (g=1) $q = 3^{127}$, (g=2) $q = 2^{103}$.

As noted, there are two different ways to view the dimension two case, the first using the genus 2 curve directly, and the second using the Rubin-Silverberg approach. The relative performance is seen by comparing the running time of the pairing on the Jacobian of the curve over $\mathbb{F}_{2^m}$ with the pairing on $E(\mathbb{F}_{2^{3m}})$. The latter will be roughly the same as the cost of a pairing on $E(\mathbb{F}_{2^{m'}})$ where $m' \approx 3m$. Note that this does not take into account the cost of conversion between the Rubin-Silverberg Abelian variety representation and the elliptic curve over the larger field.

Table 5 gives some running times for calculating the 950-bit case, and Table 6 the running times for the 1230-bit case. As noted earlier, we use the notation η for the eta pairing with $T = q$ and write η_T for the faster variant with $T = q - N$. In both tables, cases 1 to 3 illustrate the computation of the η pairing for the elliptic characteristic 2 and 3 cases, as well as the genus 2, characteristic 2 case. Cases 4 to 7 give timings for the computation of the η_T pairing, where the genus 2 "general" case is a general divisor, rather than a divisor with one point on it as is the case for the third and sixth entries in the tables. The final case in Table 6 gives a time for computing the Tate pairing using BKLS-GHS.

The first observation to make from the tables is that the new η_T method is clearly superior to the η generalisation of the Duursma-Lee method for all cases. In Table 5, the

Table 5. Running times for pairing computation (950-bit finite field).

case	curve	optimisation	pairing time (ms)
1	$E(\mathbb{F}_{2^{239}})$	elliptic char 2 η	3.16
2	$E(\mathbb{F}_{3^{97}})$	elliptic char 3 (see [17])	4.05
3	$C(\mathbb{F}_{2^{79}})$	genus 2 η	1.95
4	$E(\mathbb{F}_{2^{239}})$	elliptic char 2 η_T	1.70
5	$E(\mathbb{F}_{3^{97}})$	elliptic char 3 η_T	2.72
6	$C(\mathbb{F}_{2^{79}})$	genus 2 η_T	1.25
7	$C(\mathbb{F}_{2^{79}})$	genus 2 general η_T	4.20

Table 6. Running times for pairing computation (1230-bit finite field).

case	curve	optimisation	pairing time (ms)
1	$E(\mathbb{F}_{2^{307}})$	elliptic char 2 η	5.83
2	$E(\mathbb{F}_{3^{127}})$	elliptic char 3 η	8.42
3	$C(\mathbb{F}_{2^{103}})$	genus 2 η	3.00
4	$E(\mathbb{F}_{2^{307}})$	elliptic char 2 η_T	3.50
5	$E(\mathbb{F}_{3^{127}})$	elliptic char 3 η_T	5.36
6	$C(\mathbb{F}_{2^{103}})$	genus 2 η_T	1.87
7	$C(\mathbb{F}_{2^{103}})$	genus 2 general η_T	6.42
8	$C(\mathbb{F}_{2^{103}})$	genus 2 BKLS-GHS	3.15

elliptic char 2 case is 46% faster, the elliptic char 3 case is 32% faster, and the genus 2 char 2 case is 36% faster.

The second observation is that the genus 2 η_T pairing is considerably faster than either of the elliptic cases for both levels of security. In Table 5, the genus 2 η_T case is 26% faster than the elliptic char 2 case, and 54% faster than the elliptic char 3 case. The difference is even more pronounced in Table 6. The timings for the genus 2 BKLS-GHS method confirm the surprising observation that the BKLS-GHS method is roughly computationally equivalent to the η method in the genus 2 case.

One of the potential advantages of using hyperelliptic curves is that the base field can be much smaller than that required for an elliptic curve, for the same level of security. Great potential savings can be realised if an element of the base field can be represented in a single machine word, rather than using a multi-precision representation, and for comparison with elliptic curves we regard it as quite "fair" to try to exploit this feature.

So in implementing arithmetic in the field $\mathbb{F}_{2^{103}}$ and $\mathbb{F}_{2^{79}}$ we take advantage of the 128-bit registers available to those processors, like the Pentium IV, which support the SSE2 instruction set, and have written a special function to carry out field multiplication using SSE2 instructions. This is twice as fast as a standard multi-precision implementation, and improves the overall timings by about 50%.

All timings were done on a Pentium IV running at 3 GHz.

11 Conclusions

We have presented the eta pairing approach to compute pairings on supersingular curves. This approach generalises and clarifies the Duursma-Lee algorithm. We have provided full examples of the method in characteristic 2 for genus 1 and 2, which turn out to be very efficiently implementable.

12 Acknowledgements

We are grateful to Alice Silverberg for discussions regarding section 9, to Eunjeong Lee for pointing out a missing validity condition for theorem 1, and to Tanja Lange for some comments.

References

1. P. S. L. M. Barreto. A note on efficient computation of cube roots in characteristic 3. Cryptology ePrint Archive, Report 2004/305, 2004. Available from `http://eprint.iacr.org/2004/305`.
2. P. S. L. M. Barreto. The well-tempered pairing. In *8th Workshop on Elliptic Curve Cryptography – ECC'2004*, Bochum, Germany, 2004. Invited talk.
3. P. S. L. M. Barreto, H. Y. Kim, B. Lynn, and M. Scott. Efficient algorithms for pairing-based cryptosystems. In *Advances in Cryptology – Crypto'2002*, volume 2442 of *Lecture Notes in Computer Science*, pages 354–368. Springer-Verlag, 2002.
4. P. S. L. M. Barreto, B. Lynn, and M. Scott. Efficient implementation of pairing-based cryptosystems. *Journal of Cryptology*, 17(4):321–334, 2004.
5. P. S. L. M. Barreto and M. Naehrig. Pairing-friendly elliptic curves of prime order. In *Selected Areas in Cryptography – SAC'2005*, Lecture Notes in Computer Science. Springer-Verlag, 2005. to appear.
6. I. F. Blake, G. Seroussi, and N. P. Smart. *Advances in elliptic curve cryptography*. Cambridge, 2005.
7. D. Boneh and M. Franklin. Identity-based encryption from the Weil pairing. *SIAM Journal of Computing*, 32(3):586–615, 2003.
8. D. G. Cantor. Computing in the jacobian of a hyperelliptic curve. *Math. Comp.*, 48(177):95–101, 1987.
9. R. Dutta, R. Barua, and P. Sarkar. Pairing-based cryptography: A survey. Cryptology ePrint Archive, Report 2004/064, 2004. `http://eprint.iacr.org/2004/064`.
10. I. Duursma and H.-S. Lee. Tate pairing implementation for hyperelliptic curves $y^2 = x^p - x + d$. In *Advances in Cryptology – Asiacrypt'2003*, volume 2894 of *Lecture Notes in Computer Science*, pages 111–123. Springer-Verlag, 2003.
11. I. Duursma and K. Sakurai. Efficient algorithms for the jacobian variety of hyperelliptic curves $y^2 = x^p - x + 1$ over a finite field of odd characteristic p. In *Coding theory, cryptography and related areas (Guanajuato, 1998)*, pages 73–89. Springer-Verlag, 2000.
12. K. Fong, D. Hankerson, J. López, and A. Menezes. Field inversion and point halving revisited. Technical report CORR 2003-18, University of Waterloo, 2002.
13. G. Frey and T. Lange. Fast bilinear maps from the tate-lichtenbaum pairing on hyperelliptic curves, 2005.
14. G. Frey and H.-G. Rück. A remark concerning m-divisibility and the discrete logarithm problem in the divisor class group of curves. *Math. Comp.*, 52:865–874, 1994.

15. S. Galbraith, K. Harrison, and D. Soldera. Implementing the Tate pairing. In *Algorithmic Number Theory – ANTS V*, volume 2369 of *Lecture Notes in Computer Science*, pages 324–337. Springer-Verlag, 2002.
16. S. D. Galbraith. Supersingular curves in cryptography. In *ASIACRYPT 2001*, volume 2248 of *Lecture Notes in Computer Science*, pages 495–513. Springer-Verlag, 2001.
17. R. Granger, D. Page, and M. Stam. On small characteristic algebraic tori in pairing-based cryptography. Cryptology ePrint Archive, Report 2004/132, 2004.
18. M. Katagi, T. Akishita, I. Kitamura, and T. Takagi. Some improved algorithms for hyperelliptic curve cryptosystems using degenerate divisors. In *ICISC 2004*, volume 3506, pages 296–312. Springer-Verlag, 2005.
19. M. Katagi, I. Kitamura, T. Akishita, and T. Takagi. Novel efficient implementations of hyperelliptic curve cryptosystems using degenerate divisors. In *Information Security Applications – WISA'2004*, volume 3325 of *Lecture Notes in Computer Science*, pages 345–359. Springer-Verlag, 2005.
20. N. Koblitz. Hyperelliptic cryptosystems. *Journal of Cryptology*, 1(3):139–150, 1989.
21. S. Kwon. Efficient Tate pairing computation for supersingular elliptic curves over binary fields. Cryptology ePrint Archive, Report 2004/303, 2004. `http://eprint.iacr.org/2004/303`.
22. T. Lange. Formulae for arithmetic on genus 2 hyperelliptic curves. In *Applicable Algebra in Engineering, Communication and Computing*, Online publication. Springer-Verlag, 2004. `http://www.springerlink.com/openurl.asp?genre=article&id=doi:10.1007/s0%0200-004-0154-8`.
23. T. Lange and M. Stevens. Efficient doubling on genus two curves over binary fields. In *Selected Areas in Cryptography – SAC'2004*, volume 3357 of *Lecture Notes in Computer Science*, pages 170–181. Springer-Verlag, 2004.
24. V. S. Miller. Short programs for functions on curves. Unpublished manuscript, 1986. `http://crypto.stanford.edu/miller/miller.pdf`.
25. K. Rubin and A. Silverberg. Supersingular abelian varieties in cryptology. In *Advances in Cryptology – Crypto'2002*, volume 2442 of *Lecture Notes in Computer Science*, pages 336–353. Springer-Verlag, 2002.
26. K. Rubin and A. Silverberg. Using primitive subgroups to do more with fewer bits. In *Algorithmic Number Theory – ANTS VI*, volume 3076 of *Lecture Notes in Computer Science*, pages 18–41. Springer-Verlag, 2004.
27. M. Scott. Faster identity based encryption. *Electronics Letters*, 40(14):861, 2004.
28. M. Scott and P. Barreto. Compressed pairings. In *Advances in Cryptology – Crypto' 2004*, volume 3152 of *Lecture Notes in Computer Science*, pages 140–156. Springer-Verlag, 2004. Also available from `http://eprint.iacr.org/2004/032/`.
29. A. Silverberg. Compression for trace zero subgroups of elliptic curves. Preprint, 2004. Available from `http://www.math.uci.edu/~asilverb/bibliography/compress.pdf`.
30. J. H. Silverman. *The Arithmetic of Elliptic Curves*. Number 106 in Graduate Texts in Mathematics. Springer-Verlag, Berlin, Germany, 1986.

A The hyperelliptic function $f_{8,P}$

We now derive an explicit expression for the function $f_{8,P}$ needed for Miller's algorithm on the supersingular hyperelliptic curve $C_b : y^2 + y = x^5 + x^3 + b$.

Let $P = (x_P, y_P)$. We will consider divisors $D_n = n(P) - n(\infty)$. To achieve this we will consider the **reduced** divisor (via Cantor's algorithm) D'_n which is equivalent to D_n. We will consider functions such that $D_n = D'_n + (f_n)$.

The divisor $D_1 = (P) - (\infty)$ has Mumford representation

$$(a_1(x), b_1(x)) = (x + x_P, y_P).$$

We take the function $f_1 = 1$.

Now consider $D_2 = 2(P) - 2(\infty)$. One can show that this divisor has Mumford representation $(a_2(x), b_2(x)) = (x^2 + x_P^2, (x_P^4 + x_P^2)x + y_P^2)$. This divisor is reduced (so no reduction step in Cantor's algorithm is performed). Hence $D_2' = D_2$ and so the function f_2 may be chosen to be 1.

Now consider $D_4 = 4(P) - 4(\infty)$. The Mumford representation (after performing the composition step of Cantor's algorithm) is

$$(a_4(x), b_4(x)) = (x^4 + x_P^4, x^3 + (x_P^8 + x_P^4)x^2 + (x_P^4)x + y_P^4).$$

This divisor is not reduced. We have $(b_4^2 + b_4 + x^5 + x^3 + b)/a_4(x) = a_4'(x) = x^2 + x + (x_P^{16} + x_P^8)$ and $b_4'(x) := b_4(x) + 1 \pmod{a_4'(x)} = (x_P^{16} + 1)x + (y_P^8 + x_P^8 + x_P^{24} + 1)$.

We must consider functions and divisors. The divisor D_4 is equivalent to the divisor $D_4' = E - 2(\infty)$ where E is effective. The divisor D_4' has the Mumford representation $(a_4'(x), b_4'(x))$ given above. Denote by $\overline{E}$ the 'negative' of E. The function $a_4(x)$ has divisor $4(P) + 4(\overline{P}) - 8(\infty)$ while the function $a_4'(x)$ has divisor $E + \overline{E} - 4(\infty)$. The function $y + b_4(x)$ has divisor $4(P) + \overline{E} - 6(\infty)$ while the function $y + b_4(x) + 1$ has divisor $4(\overline{P}) + E - 6(\infty)$. It follows that

$$((y + b_4(x))/a_4'(x)) = 4(P) - E - 2(\infty).$$

Hence we define $f_4 = (y + b_4(x))/a_4'(x)$ and we have $D_4 = D_4' + (f_4)$.

Now for the final step (thankfully!). We double the divisor D_4' using Cantor's composition rule to obtain $D_8'' = 2E - 4(\infty)$. Note that $D_8 = 2D_4 = 2(D_4' + (f_4)) = D_8'' + (f_4^2)$. One computes the Mumford representation of D_8'' to be

$$(a_8''(x), b_8''(x)) = (a_4'(x)^2, (x_P^{32} + 1)x^2 + (x_P^{32} + x_P^{16})x + (y_P^{16} + x_P^{16} + x_P^{48} + 1))$$

and one can check that $((b_8'')^2 + b_8'' + f(x))/a_8''(x) = a_8'(x) = (x + (x_P^{64} + 1))$. Thus, $b_8'(x) := b_8''(x) + 1 \pmod{a_8'(x)} = y_P^{64} + x_P^{128} + 1$. Define $[8]P = (x_P^{64} + 1, y_P^{64} + x_P^{128} + 1)$. We obtain $D_8' = ([8]P) - (\infty)$ which confirms the octupling formula for the point $[8]P$. Algorithm 5 describes divisor octupling in detail.

We now consider principal divisors. As before, $y + b_8''(x)$ has divisor $2E + (\overline{[8]P}) - 5(\infty)$ and $(a_8'(x)) = (\overline{[8]P}) + ([8]P) - 2(\infty)$. Hence we have $D_8'' = D_8' + (f_8')$ where $f_8' = (y + b_8''(x))/a_8'(x)$.

Putting it all together, we get

$$(f_8) = 8(P) - ([8]P) - 7(\infty)$$

where

$$f_8 = \left(\frac{y + b_4(x)}{a_4'(x)}\right)^2 \frac{y + b_8''(x)}{a_8'(x)}.$$

Algorithm 5 Octupling of a divisor $[u, v]$

```
INPUT: divisor $[u, v]$.
OUTPUT: $[u', v'] = 8[u, v]$.
1: if $\deg(u) = 2$ then ▷ $[u, v] = [x^2 + u_1x + u_0, v_1x + v_0]$
2:     $[u', v'] \leftarrow [x^2 + u_1^{64}x + (u_1 + u_0 + 1)^{64}, (v_1 + u_1)^{64}x + (u_1 + u_0 + v_1 + v_0 + 1)^{64}]$
3: else if $\deg(u) = 1$ then ▷ $[u, v] = [x + u_0, v_0]$
4:     $[u', v'] \leftarrow [x + (u_0 + 1)^{64}, (v_0 + u_0^2 + 1)^{64}]$
5: else ▷ $[u, v] = [1, 0]$
6:     $[u', v'] \leftarrow [1, 0]$
7: end if
```

B Efficient implementation of pairings in genus 2

B.1 Precomputation

We will precompute a table of powers of x_P and y_P (these are the initial input values for the point P) labelled as

$$x_P^{(i)} = \pi^i(x_P) = x_P^{2^i}, \quad \text{and} \quad y_P^{(i)} = \pi^i(y_P) = y_P^{2^i}$$

for $i = 0, 1, \ldots, m - 1$.

We focus on computing the term $f_{8,P}(\psi(Q))$ (i.e. we do not bring Frobenius actions into this computation).

Note that, at loop iteration i, the current value of the x-coordinate of $[2^{3i}]P$ can be written in terms of the precomputed initial values as

$$x_P^{(6i)} + \gamma_1(i)$$

where $\gamma_1(i)$ is 1 when i is odd and 0 otherwise. Similarly, the current value of the y-coordinate of $[2^{3i}]P$ is

$$y_P^{(6i)} + \gamma_1(i)x_P^{(6i+1)} + \gamma_3(i)$$

where $\gamma_3(i) = 1$ when $i \equiv 1, 2 \pmod 4$ and 0 otherwise.

Obviously, in the above the exponents $6i$ in $x_P^{(6i)}$ are taken modulo m. One sees that they wrap around rapidly.

B.2 The α factor

Write $\alpha = (y + b_4(x))^2 \circ \psi$ as a function of (x_Q, y_Q). We have $(y + b_4(x)) \circ \psi =$

$$y + s_2x^2 + s_1x + s_0 + (x + w)^3 + (x_P^8 + x_P^4)(x + w)^2 + (x_P^4)(x + w) + y_P^4$$

and squaring gives

$$y^2 + s_2^2x^4 + s_1^2x^2 + s_0^2 + x^6 + x^4w^2 + x^2w^4 + w^6 + (x_P^{16} + x_P^8)(x^4 + w^4) + (x_P^8)(x^2 + w^2) + y_P^8.$$

Now, $s_2^2 = (w^4 + 1)^2 = w$ and $s_1^2 = (w^2 + w^4)^2 = w^4 + w + 1$. Also, $s_0^2 = s_0 + w^5 + w^3$.

Expressing as a 12-tuple we get α as follows: The first component is

$$y^2 + x^2 + x^6 + 1 + (x_P^{16} + x_P^8)x^4 + x_P^8 x^2 + y_P^8$$

and the remaining components are

$$(x^4 + x^2, x^4 + 1 + x_P^8, 1 + 1, x^2 + x^2 + x_P^{16} + x_P^8, 1 + 1, 1, 0, 0, 0, 0, 0)$$

which can be slightly simplified.

Finally, we want to evaluate this on (x_Q, y_Q) and to replace the current value for x_P with the precomputed values. We obtain the 12-tuple with first component

$$y_Q^2 + x_Q^6 + (x_P^{(6i+4)} + x_P^{(6i+3)})x_Q^4 + (x_P^{(6i+3)} + 1 + \gamma_1(i))x_Q^2 + y_P^{(6i+3)} + \gamma_1(i)x_P^{(6i+4)} + \gamma_3(i) + 1$$

and remaining components

$$(x_Q^4 + x_Q^2, x_Q^4 + x_P^{(6i+3)} + \gamma_1(i) + 1, 0, x_P^{(6i+4)} + x_P^{(6i+3)}, 0, 1, 0, 0, 0, 0, 0).$$

B.3 The β factor

We then do a similar thing for $\beta = (y + b_8'') \circ \psi$. We have

$$\beta = y + s_2x^2 + s_1x + s_0 + (x_P^{32} + 1)(x + w)^2 + (x_P^{32} + x_P^{16})(x + w) + (y_P^{16} + x_P^{16} + x_P^{48} + 1).$$

We expand $s_2 = 1 + w^4$ etc and write $x_P^{16} + x_P^{48} = x_P^{16}(1 + x_P^{32})$. Hence, β can be expressed as a 12-tuple with first component

$$y + (x_P^{32})x^2 + (x_P^{32} + x_P^{16})x + y_P^{16} + x_P^{16}(1 + x_P^{32}) + 1$$

and remaining components

$$(x_P^{32} + x_P^{16}, x + x_P^{32} + 1, 0, x^2 + x, 0, 1, 0, 0, 0, 0, 0).$$

Finally, we substitute $(x, y) = (x_Q, y_Q)$ and insert the precomputed values $x_P = x_P^{(6i)} + \gamma_1(i)$ and $y_P = y_P^{(6i)} + \gamma_1(i)x_P^{(6i+1)} + \gamma_3(i)$. Using the formula $\gamma_1(i)(1 + \gamma_1(i)) = 0$ gives the 12-tuple with first component

$$\begin{gathered} y_Q + (x_P^{(6i+5)} + \gamma_1(i))x_Q^2 + (x_P^{(6i+5)} + x_P^{(6i+4)})x_Q \\ +y_P^{(6i+4)} + x_P^{(6i+4)}\left(x_P^{(6i+5)} + \gamma_1(i) + 1\right) + \gamma_3(i) + 1. \end{gathered}$$

and remaining components

$$(x_P^{(6i+5)} + x_P^{(6i+4)}, x_Q + x_P^{(6i+5)} + \gamma_1(i) + 1, 0, x_Q^2 + x_Q, 0, 1, 0, 0, 0, 0, 0).$$

It remains to multiply the α and β together efficiently. But first we consider how to absorb the powers of 8 into the equations.

B.4 Absorbing powers of 8

We break the computation of the eta pairing $\eta_T(P, Q)$ into two parts, the main part is the loop corresponding to the $(m-1)/2$ octuplings, and the secondary part is the final two doublings and addition. The main part can be expressed as the product

$$\prod_{i=0}^{(m-3)/2} f_{8,2^{3i}P}(\psi(Q))^{2^{3(m-3-2i)/2}}$$

where $f_{8,2^{3i}P} = \alpha\beta$ as described previously. The goal of this section is to write this as

$$\prod_{i=0}^{(m-3)/2} f_i$$

where each f_i is an equation which has the 2-power Frobenius action already brought into the equation. Using the formulae for α and β above we will compute $\alpha^{2^{3(m-3-2i)/2}}$ and $\beta^{2^{3(m-3-2i)/2}}$.

To achieve this efficiently requires precomputation of the 2-power Frobenius orbit of the point Q, so define for $i = 0, 1, \ldots, m-1$

$$x_Q^{(i)} = x_Q^{2^i} \qquad \text{and} \qquad y_Q^{(i)} = y_Q^{2^i}.$$

The most delicate part of the argument is handling how w and s_0 behave under powering by $2^{3(m-3-2i)/2}$. Recall that $w^8 = w + 1$ from which we deduce

$$\begin{aligned} s_0^8 &= s_0 + w^2 \\ s_0^{8^2} &= s_0 + 1 \\ s_0^{8^3} &= s_0 + w^2 + 1 \end{aligned} \tag{5}$$

Note that m is coprime to 12 and so is odd. We have $w^8 = w + 1$ and so, since $(m-3-2i)/2 \equiv i \pmod 2$ we have $w^{2^{3(m-3-2i)/2}} = w + \gamma_1(i)$. The same formula holds when w is replaced by w^2 or w^4. For s_0 note that if $m \equiv 1 \pmod 4$ then

$$s_0^{2^{3(m-3-2i)/2}} = s_0 + \gamma_1(i)w^2 + \gamma_3(i)$$

while if $m \equiv 3 \pmod 4$ then

$$s_0^{2^{3(m-3-2i)/2}} = s_0 + \gamma_1(i)w^2 + \gamma_3(i) + 1.$$

We denote by $\gamma_4(m, i)$ the value $\gamma_3(i)$ when $m \equiv 1 \pmod 4$ and $\gamma_3(i) + 1$ otherwise.

B.5 The α factor

The basic shape of the term α will be similar to previously, except a few extra terms due to equation (5). The process is simple, just bring the 2-power operation into the formula and simplify the 'exponents'.

The "constant" term will be

$$y_Q^{((3m-7-6i)/2)} + (x_Q^{((3m-7-6i)/2)})^3 + (x_P^{((3m-1+6i)/2)} + x_P^{((3m-3+6i)/2)})x_Q^{((3m-6i-5)/2)} +$$
$$(x_P^{((3m-3+6i)/2)} + 1 + \gamma_1(i))x_Q^{((3m-7-6i)/2)} + y_P^{((3m-3+6i)/2)} + \gamma_1(i)x_P^{((3m-1+6i)/2)} + \gamma_3(i) + 1$$

plus when $(m-3-2i)/2$ is odd (i.e., when i is odd) another term must be added (coming from the fact that $(w^{2j})^8 = w^{2j} + 1$ and the s_0 term). We write this other term as

$$\gamma_1(i)\left(x_Q^{(3m-7-6i)/2)} + 1 + \gamma_1(i) + x_P^{((3m-1+6i)/2)}\right) + \gamma_4(m,i).$$

We can apply $\gamma_1(i)(1 + \gamma_1(i)) = 0$, cancel various terms and simplify the cubing of $x_Q^{((3m-7-6i)/2)}$. The expression simplifies to

$$y_Q^{((3m-7-6i)/2)} + (x_P^{((3m-1+6i)/2)} + x_P^{((3m-3+6i)/2)})x_Q^{((3m-5-6i)/2)} +$$
$$(x_P^{((3m-3+6i)/2)} + 1 + x_Q^{((3m-5-6i)/2)})x_Q^{((3m-7-6i)/2)} + y_P^{((3m-3+6i)/2)} + \gamma_5(i)$$

where $\gamma_5(i) = 1$ if $i \equiv 1 \pmod 4$ and 0 otherwise.

The remaining terms are (note that there is an additional $\gamma_1(i)w^2$ term due to the s_0 term):

$$(x_Q^{((3m-5-6i)/2)} + x_Q^{((3m-7-6i)/2)})w + (x_Q^{((3m-5-6i)/2)} + x_P^{((3m-3+6i)/2)} + 1)w^2$$
$$+(x_P^{((3m-1+6i)/2)} + x_P^{((3m-3+6i)/2)})w^4 + s_0$$

As usual, the indices inside round brackets should be reduced modulo m to the range $\{0, 1, \ldots, m-1\}$.

B.6 The β factor

We now consider the β factor. One sees that the "constant term" of $\beta^{2^{3((m-3-2i)/2)}}$ is

$$y_Q^{((3m-9-6i)/2)} + (x_P^{((3m+1+6i)/2)} + \gamma_1(i))x_Q^{((3m-7-6i)/2)} + (x_P^{((3m+1+6i)/2)}$$
$$+x_P^{((3m-1+6i)/2)})x_Q^{((3m-9-6i)/2)} + y_P^{((3m-1+6i)/2)}$$
$$+x_P^{((3m-1+6i)/2)}(x_P^{((3m+1+6i)/2)} + \gamma_1(i) + 1) + \gamma_3(i) + 1$$

plus

$$\gamma_1(i)\left(x_P^{((3m-1+6i)/2)} + x_Q^{((3m-7-6i)/2)} + \gamma_1(i) + 1\right) + \gamma_4(m,i).$$

This simplifies to

$$y_Q^{((3m-9-6i)/2)} + (x_P^{((3m+1+6i)/2)} + x_P^{((3m-1+6i)/2)})x_Q^{((3m-9-6i)/2)} + y_P^{((3m-1+6i)/2)}$$
$$+x_P^{((3m+1+6i)/2)}(x_P^{((3m-1+6i)/2)} + x_Q^{((3m-7-6i)/2)}) + x_P^{((3m-1+6i)/2)} + \gamma_5(i)$$

The remaining terms are (again, including a $\gamma_1(i)w^2$ term)

$$(x_P^{((3m+1+6i)/2)} + x_P^{((3m-1+6i)/2)})w + (x_P^{((3m+1+6i)/2)} + x_Q^{((3m-9-6i)/2)} + 1)w^2$$
$$+(x_Q^{((3m-7-6i)/2)} + x_Q^{((3m-9-6i)/2)})w^4 + s_0$$

B.7 Simplifying the final operations

After the loop of $(m-1)/2$ iterations, it remains to perform two doublings and an addition. For the case in which the input divisors are both of the form $D_i = (P_i) - (\infty)$ the final addition can be skipped as it has no impact on the function. Note that $(2^{(3m+1)/2}+1)D_1 = \phi(D_1)$, where $\phi(D_1) = (x+1, y+x^2+1) - (\infty)$. Denote by D_1' the reduced divisor $2^{(3m+1)/2}D_1$. D_1' is equivalent to $\phi(D_1) - D_1 = (\phi(P)) - (P)$.

Now, let v be the vertical line through P and $-P$. So $(v) = (P) + (-P) - 2(\infty)$. Then

$$(\phi(P)) - (P) + (v) = (\phi(P)) + (-P) - 2(\infty)$$

Hence, by the uniqueness of the reduced divisors in Mumford representation we have

$$D' = (\phi(P)) + (-P) - 2(\infty).$$

As one of the points on D' is $-P$, the composition stage of Cantor's algorithm immediately cancels P and $-P$ using a vertical line function. As we don't need to know what the "current" divisor is before the addition, we can also skip the two doublings. We know from Appendix A that the mumford representation of the divisor $D = 4(P) - 4(\infty)$ is $(a_4(x), b_4(x)) = (x^4 + x_P^4, x^3 + (x_P^8 + x_P^4)x^2 + (x_P^4)x + y_P^4)$. So, we need to extract the point on the divisor after the $(m-1)/2$ octupling phase, square the function twice, and then multiply it by the function defined below;

$$y + b_4(x) = y + x^3 + (x_P^8 + x_P^4)x^2 + (x_P^4)x + y_P^4$$

Building the distortion map into the formula gives us a constant term;

$$y + x^2(1 + x + x_P^8 + x_P^4) + x_P^4 x + y_P^4$$

and the remaining terms are;

$$(x^2 + x_P^4, x_P^8 + x_P^4, 1, x^2 + x, 0, 1, 0, 0, 0, 0, 0).$$

When we are working with the general divisor case, we cannot skip the final addition as we can for the simple divisor case. For both the doublings and the addition, we need to evaluate the points on the second divisor at the function $y + s_1x^3 + l_2x^2 + l_1x + l_0$, where s_1, l_2, l_1, l_0 come from Cantor's algorithm. After building the distortion map into this formula, we get the "constant" term:

$$(y + x^2 + x(s_1x^2 + l_1) + l_2x^2 + l_0)$$

and the remaining terms are;

$$(s_1x^2 + l_1, x + s_1x + l_2, s_1, x^2 + x, 0, 1, 0, 0, 0, 0).$$