種数 2 の超楕円曲線を用いた高速暗号系

松尾 和人[*]　　趙 晋輝[†]　　辻井 重男[‡]

[*] 東洋通信機 (株) R&D センター
〒 253-0192
神奈川県高座郡寒川町小谷 2-1-1

[†] 中央大学理工学部
電気電子情報通信工学科
〒 112-0003
東京都文京区春日 1-13-27

[‡] 中央大学理工学部
情報工学科
〒 112-0003
東京都文京区春日 1-13-27

あらまし　Cantor アルゴリズムを用いた超楕円曲線暗号系は, 同一の安全性を持つ楕円曲線暗号系と比較し数倍低速であることが知られている. 従って, 高速な超楕円曲線暗号系の構成は重要な研究課題である.
最近, Harley によって種数 2 の超楕円曲線上の高速な加算アルゴリズムが提案された. 本論文では, まず Harley アルゴリズムの改良を示し, 次にこのアルゴリズムの適用により, 楕円曲線暗号と比較し, より高速な超楕円曲線暗号を構成可能であることを示す. 更に, 実際に楕円曲線暗号系と同等速度の超楕円曲線暗号系を構成可能であることを実装実験により示す.

キーワード　超楕円曲線暗号系, 種数 2 の超楕円曲線, 因子加算アルゴリズム, Cantor アルゴリズム, Harley アルゴリズム

Fast Genus Two Hyperelliptic Curve Cryptosystems

Kazuto Matsuo[*]　　Jinhui Chao[†]　　Shigeo Tsujii[‡]

[*] R&D Center,
Toyo Communication Equipment
1-1, Koyato 2, Samukawa,
Koza, Kanagawa,
253-0192, Japan

[†] Dept. of Electrical, Electronic,
and Communication Eng.,
Chuo University
1-13-27,
Kasuga, Bunkyo, Tokyo,
112-0003, Japan

[‡] Dept. of Information
and System Eng.,
Chuo University
1-13-27,
Kasuga, Bunkyo, Tokyo,
112-0003, Japan

Abstract:　Most current hyperelliptic curve cryptosystems (HECC) use the Cantor algorithm in hyperelliptic additions for encryption and decryption. It has been reported that hyperelliptic curve cryptosystems are more than several times slower than elliptic curve cryptosystems (ECC). It is then an interesting and challenging question that if the HECC could be faster than the ECC. Recently, Harley proposed a faster algorithm of addition on genus two hyperelliptic curves. In the first part of this paper, we show an improvement of the Harley algorithm which needs less computation then faster than the elliptic addition. The second part of the paper is a tentative implementation of the genus two HECC using the improved Harley algorithm, comparing with the ECC, which shown that the performance of the HECC is actually equal to that of the ECC.

Keywords:　Hyperelliptic curve cryptosystems, Genus two hyperelliptic curves, Hyperelliptic addition algorithm, Cantor algorithm, Harley algorithm.

1 Introduction

Hyperelliptic curve cryptosystems (HECC for short) are known to be secure if general curves of small genus (more precisely, genus two or three) are used. In particular, only efficient attacks such as function field sieves and their variants are known for curves of moderately large genus [ADH94][Pau96][FP97][Gau00][EG00]. For the small genus curves whose Jacobian varieties contain a large prime factor in their orders can be used to define discrete logarithm of fully exponential complexity. Moreover, numerous studies about construction of such secure curves have been carrying out, e.g. [Kob89][Pil90][Kam91][Spa94][AH96][FM98][SSI98][SS98][GHS00][GH00][CMT00][CMKT00][Wen00][Ked01], and it is anticipated that we will be able to construct enough many secure hyperelliptic curves efficiently in the near future.

Besides, in hyperelliptic curve cryptosystems, efficient encryption and decryption requires fast addition algorithms for the divisor group of the hyperelliptic curve. Currently the standard addition algorithm for hyperelliptic curves is proposed by Cantor [Can87]. The Cantor algorithm carries out fast addition by using Mumford's representation of divisors, and fast composition and reduction of quadratic forms related hyperelliptic function fields. A large number of researches have been reported on improvement of the Cantor algorithm [Kob89][PS98][SSI98][SS98][Sma99][Nag00]. However, it seems that they are still not fast enough comparing with elliptic curve cryptosystems (ECC for short). It has been reported that HECC based on the Cantor algorithm are more than several times slower than ECC in practice [Sma99]. This seems to be the main reason preventing HECC from practical usage.

In fact, the addition of divisors on hyperelliptic curves is more complicated than the addition of points on elliptic curves. On the other hand, however, according to the Hasse-Weil theorem an HECC needs only a definition field of g times smaller size than that of an ECC with the same security. It is then an interesting and challenging question that how to take this advantage on field sizes and overcome the difficulty in divisor additions, and could the HECC surpass the speed of the standard ECC.

A major recent progress in fast addition algorithms on genus two hyperelliptic curves is due to Harley [GH00][Har00a][Har00b]. The Harley algorithm carries out the addition based on the theory and tools in Mumford's textbook [Mum84]. Instead of computing quadratic forms related hyperelliptic function fields, they developed a variation of the chord-tangent law on elliptic curves. Besides, they applied the Chinese remainder theorem and Newton's iteration to divisor additions and evenmore, they used the Karatsuba multiplication explicitly in the algorithm. Consequently, their algorithm is much faster than the Cantor algorithm and its variations. Specifically, it takes $2I + 27M$ for an addition and $2I + 30M$ for a doubling in general, much faster than the original Cantor algorithm which takes $3I + 70M$ and $3I + 76M$ respectively [Nag00], where I and M denote the costs of an inversion and a multiplication respectively over the definition finite field.

Although the Harley algorithm is potentially powerful to speed up the HECC, since the algorithm is only mentioned in their work on point counting of hyperelliptic curves, and the details are not enough explicitly given, it seems still being neither widely known by general cryptologists nor implemented for performance comparison with the ECC .

In this paper, we present an improvement of the Harley algorithm. The improved algorithm takes $2I + 23M$ for an addition and $2I + 25M$ for a doubling in general. It is shown that, by applying the algorithm, the additions of genus two hyperelliptic curves are faster than addition of elliptic curves with the same keylength. Then we show a tentative implementation of the algorithm which shown that the performance of genus two HECC is actually equal to that of ECC.

The paper is organized as follows. In Section 2, we give preliminaries on the hyperelliptic curves, their divisors, and the Jacobian varieties. In Section 3, we review the Harley algorithm. A detailed procedure of the algorithm is given here for reference of general researchers. In Section 4, we present an improvement of the Harley algorithm. In Section 5, we discuss the performance of genus two HECC with the improved Harley algorithm comparing with that of ECC. Conditions are given for genus two HECC to surpass the ECC using the fast algorithm of hyperelliptic addition. In Section 6, we show a tentative implementation of both the improved Harley algorithm and elliptic curve arithmetics.

2 Preliminary

Let p be an odd prime, n a positive integer and $q = p^n$. Then a hyperelliptic curve C of genus g defined over $\mathbb{F}_q$ is defined as follows:

$$\begin{aligned} C :& Y^2 = F(X), \\ & F(X) = X^{2g+1} + f_{2g}X^{2g} + \cdots + f_0, \end{aligned} \tag{1}$$

where $f_i \in \mathbb{F}_q$ and $\mathrm{disc}\,(F) \neq 0$.

Let $P = (x, y)$ be a point on C. Then its opposite is $-P = (x, -y)$. For the point at infinity, denoted P_∞, we define $-P_\infty = P_\infty$ and call the points of the form $(x, 0)$ ramification points.

A divisor $\mathcal{D}$ on C is defined as a formal sum

$$\mathcal{D} = \sum_{P_i \in C} \mathrm{ord}_{P_i}(\mathcal{D})P_i,\ \mathrm{ord}_{P_i}(\mathcal{D}) \in \mathbb{Z} \tag{2}$$

of points P_i on C. Obviously, the set $\mathfrak{D}$ of the divisors on C is an Abelian group. Any two divisors $\mathcal{D}_1$ and $\mathcal{D}_2$ are said to be coprime, if $\mathrm{ord}_P(\mathcal{D}_1) \neq 0$ implies that $\mathrm{ord}_P(\mathcal{D}_2) = 0$ for any $P \in C$. The degree of a divisor $\mathcal{D}$ is defined as

$$\deg \mathcal{D} = \sum_i \mathrm{ord}_{P_i}(\mathcal{D}). \tag{3}$$

The degree zero divisors $\mathfrak{D}^0$ forms a subgroup of $\mathfrak{D}$. For a rational function f on C, we define the divisor of f, called the principal divisor, as

$$(f) = \sum_i v_{P_i}(f) P_i, \tag{4}$$

where P_i are zeros and poles of f on C with the multiplicities $v_{P_i}(f)$. The set $\mathfrak{D}^l$ of principal divisors is a subgroup of $\mathfrak{D}^0$. The Jacobian variety $\mathcal{J}_C$ of C is defined as

$$\mathcal{J}_C = \mathfrak{D}^0 / \mathfrak{D}^l. \tag{5}$$

If both $\mathcal{D}_1$ and $\mathcal{D}_2$ belong to the same class in $\mathcal{J}_C$, they are said to be linearly equivalent and denoted as $\mathcal{D}_1 \sim \mathcal{D}_2$. Each element $\mathcal{D}$ of $\mathcal{J}_C$ can be represented in the form

$$\mathcal{D} = \sum_i \mathrm{ord}_{P_i}(\mathcal{D}) P_i - (\sum_i \mathrm{ord}_{P_i}(\mathcal{D})) P_\infty \tag{6}$$

with no two points are opposite, where $\mathrm{ord}_{P_i}(\mathcal{D}) \geq 0$. We call such a divisor semi-reduced, and $\sum_i \mathrm{ord}_{P_i}(\mathcal{D})$ the weight of $\mathcal{D}$ [GH00]. A semi-reduced divisor of weight $\leq g$ is called reduced in particular. For each element of $\mathcal{J}_C$, there is a unique reduced divisor. Therefore, we can identity an element of $\mathcal{J}_C$ with its reduced divisor, and define addition operation over $\mathcal{J}_C$ by using the reduced divisors.

Every semi-reduced divisor given by (6) can be represented as follows [Mum84]:

$$\mathcal{D} = (U, V), \tag{7}$$

where $U, V \in \bar{\mathbb{F}}_q[X]$ such that $U = \prod (X - x_i)^{\mathrm{ord}_{P_i}(\mathcal{D})}$ with $P_i = (x_i, y_i)$, $\deg V < \deg U$, and

$$F - V^2 \equiv 0 \bmod U. \tag{8}$$

which is sometime called Mumford's representation [Har00a]. Obviously, $\deg U \leq g$, if $\mathcal{D}$ is reduced.

Thus we can define the set $\mathcal{J}_C(\mathbb{F}_q)$ of divisors $\mathcal{D} = (U, V)$ by Mumford's representation with $U, V \in \mathbb{F}_q[x]$, which is a finite Abelian group. One can then define discrete logarithm problems suitable for cryptosystems on it. Hereafter, only the divisors belonging to $\mathcal{J}_C(\mathbb{F}_q)$ will be considered. It is well known that the order of $\mathcal{J}_C(\mathbb{F}_q)$ falls in the following Hasse-Weil range.

$$(q^{1/2} - 1)^{2g} \leq \# \mathcal{J}_C(\mathbb{F}_q) \leq (q^{1/2} + 1)^{2g} \tag{9}$$

which implies the field size advantage of HEC comparing with the ECC.

3 Harley algorithm

The Harley algorithm for addition on genus two hyper elliptic curves appeared in [GH00] firstly, then Harley shown a detailed document of the addition [Har00a] and sample C codes of the doubling [Har00b] on his home page. Their algorithm based on theory and tools developed in Mumford's textbook [Mum84]. Instead of computing quadratic forms related hyperelliptic function fields, they extended the so-called chord-tangent law for point addition on elliptic curves. The strategy is to classify the divisors in the addition carefully into different types then optimize the procedures for every type.

In particular, to compute $\mathcal{D}_3 = (U_3, V_3) = \mathcal{D}_1 + \mathcal{D}_2$ for given reduced divisors $\mathcal{D}_1 = (U_1, V_1)$ and $\mathcal{D}_2 = (U_2, V_2)$, one checks the weights of the given divisors by the degrees of polynomials U_1 and U_2 at first, and executes the procedures for different weights. Moreover, the other subprocedures are also adopted according to whether the divisors have common points, which can be checked by the resultant of polynomial U_1 and U_2.

In each procedure, first, one computes U for the semi reduced $\mathcal{D} \sim \mathcal{D}_3$, e.g. $U = U_1 U_2$ when $\mathcal{D}_1$ and $\mathcal{D}_2$ are coprime. Then one computes V satisfied (8) by Chinese remainder theorem (Newton's iteration) for addition (doubling, respectively). E.g. in the case that $\mathcal{D}_1, \mathcal{D}_2$ are coprime, V satisfy the following condition:

$$V \equiv V_1 \bmod U_1, \tag{10}$$

$$V \equiv S U_1 + V_1 \bmod U, \; S \in \mathbb{F}_q[X], \tag{11}$$

and we can find S as

$$S \equiv \frac{V_2 - V_1}{U1} \bmod U_2 \tag{12}$$

by CRT. Finally, one reduces $(U, -V)$ and obtain the reduced divisor (U_3, V_3). For the detailed usage of the CRT and Newton's iterations, see e.g. [GCL92], [GG99].

Beside of using CRT and Newton's iterations, the algorithm uses the Karatsuba multiplication explicitly step by step. Consequently, the algorithm is much faster than the other addition algorithms which have been proposed until now.

While [Har00a] shown the algorithm, it seems that details are not explicitly enough to fill up the procedures. Here, we show a procedure in the Table 1 of weight two coprime addition according to [Har00a], which is generally the most case. The computational cost of the algorithm is also evaluated. We considered only the costs for inversions and multiplications over definition finite fields, because they usually dominate the other operations.

In the Table 1, the second column shows the procedure for all steps, and the third column shows the cost of each step and the total cost. Capital and small letters with subscripts denote variables for polynomial and finite field elements respectively in the second column. The coefficient of X^i in a polynomial, e.g. T_j, are denoted t_{ij}.

$$T_j = t_{j \deg T_j} X^{\deg T_j} + \cdots + t_{ji} X^i + \cdots + t_{j0}. \tag{13}$$

"(Karatsuba)" means using the Karatsuba multiplication. In the third column, I and M denote the cost for an inversion operation and a multiplication operation respectively over the definition finite field.

One sees that the algorithm takes $2I + 27M$ for an addition and $2I + 30M$ for a doubling in the most case from the Table 1 and [Har00b] respectively.

4 Improvement

In this section, we present an improvement to speed up the Harley algorithm.

This improvement is to follow the same strategy of Harley but go further to optimize the procedures for different cases.

Input	Weight two coprime reduced divisors $\mathcal{D}_1 = (U_1, V_1)$ and $\mathcal{D}_2 = (U_2, V_2)$	
Output	A weight two reduced divisor $\mathcal{D}_3 = (U_3, V_3) = \mathcal{D}_1 + \mathcal{D}_2$	

Step	Procedure	Cost
1	Compute the resultant r of U_1 and U_2. $w_1 \leftarrow u_{11}u_{21};\ w_2 \leftarrow u_{10} + u_{21}^2 - u_{20} - w_1;$ $r \leftarrow u_{10}(w_2 - u_{20}) + u_{20}(u_{11}^2 + u_{20} - w_1);$	$5M$
2	If $r = 0$ then $\mathcal{D}_1$ and $\mathcal{D}_2$ have a linear factor in common, and call the exclusive procedure.	—
3	Compute $I_1 = i_{11}X + i_{10} \equiv 1/U_1 \bmod U_2$. $w_1 \leftarrow r^{-1};\ I_1 \leftarrow (w_1(u_{21} - u_{11}))X + w_1w_2;$	$I + 2M$
4	Compute $S = s_1X + s_0 \equiv (V_2 - V_1)I_1 \bmod U_2$. (Karatsuba) $w_1 \leftarrow v_{20} - v_{10};\ w_2 \leftarrow v_{21} - v_{11};\ w_3 \leftarrow i_{10}w_1;\ w_4 \leftarrow i_{11}w_2;$ $w_5 \leftarrow (i_{10} + i_{11})(w_1 + w_2) - w_3 - w_4;$ $S \leftarrow (w_5 - u_{21}w_4)X - u_{20}w_4 + w_3;$	$5M$
5	If $s_1 = 0$ then $\mathcal{D}_3$ should be weight one, and call the exclusive procedure.	—
6	Compute the coefficient k_1 of X in $K = (F - V_1^2)/U_1$. $k_1 \leftarrow f_4 - u_{11};$	—
7	Compute $T_1 = s_1X^3 + t_{12}X^2 + t_{11}X + t_{10} = SU_1$. (Karatsuba) $w_1 \leftarrow s_1u_{11};\ t_{10} \leftarrow s_0u_{10};$ $t_{11} \leftarrow (s_0 + s_1)(u_{10} + u_{11}) - w_1 - t_{10};\ t_{12} \leftarrow w_1 + s_0;$	$3M$
8	Compute $U_3 = (S(T_1 + 2V_1) - K)/U_2$. (Karatsuba) $u_{32} \leftarrow s_1^2;$ $w_1 \leftarrow s_1(s_0 + t_{12}) - 1;\ w_2 \leftarrow s_1(t_{11} + 2v_{11}) + s_0t_{12} - k_1;$ $u_{31} \leftarrow w_1 - u_{21}u_{32};\ u_{30} \leftarrow w_2 - u_{20}u_{32} - u_{21}u_{31};$	$7M$
9	Make U_3 monic $w_1 \leftarrow u_{32}^{-1};\ u_{30} \leftarrow u_{30}w_1;\ u_{31} \leftarrow u_{31}w_1;\ u_{32} \leftarrow 1;$	$I + 2M$
10	Compute $V_3 \equiv -(T_1 + V_1) \bmod U_3$. (Karatsuba) $w_1 \leftarrow t_{11} + v_{11};\ w_2 \leftarrow t_{10} + v_{10};$ $w_3 \leftarrow s_1u_{31};\ w_4 \leftarrow t_{12} - w_3;\ w_5 \leftarrow w_4u_{30};$ $w_6 \leftarrow (u_{30} + u_{31})(s_1 + w_4) - w_3 - w_5;$ $v_{30} \leftarrow w_6 - w_1;\ v_{31} \leftarrow w_5 - w_3;$	$3M$
Total		$2I + 27M$

Table 1: Addition for weight two coprime divisors on a genus two HEC

We take the addition of weight two coprime divisors $\mathcal{D}_1 = (U_1, V_1), \mathcal{D}_2 = (U_2, V_2)$ described in the Table 1 as an example to show the improvement. In the step 8 and 9 of the original algorithm (described in the Table 1), one computes monic U_3 for given U_2, V_1, S, T_1, and K by using Karatsuba multiplication, and the computation takes $I + 9M$. Since the leading coefficient of the result of the step 6 is s_1^2, one can obtain monic U_3 by computing

$$U_3 = s_1^{-2}(S(T_1 + 2V_1) - K)/U_2. \tag{14}$$

Then, by careful manipulation on (14) substituting SU_1 for T_1, one obtains

$$\begin{aligned} U_3 = X^2 &+ (w_1(2s_0 - w_1) - w_2)X + \\ &w_1(w_1(s_0^2 + u_{11} + u_{21} - f_4) + 2(v_{11} - s_0w_2)) \\ &\qquad + u_{21}w_2 + u_{10} - u_{22}, \end{aligned} \tag{15}$$

where $w_1 = s_1^{-1}$ and $w_2 = u_{21} - u_{11}$.

Computing U_3 according to (15) takes only $I + 6M$. Thus, one can reduce the costs of the step 8 and 9 in the Table 1 by $3M$ using these techniques. The similar improvements are applied to the whole original Harley algorithm.

We will not be able to show the whole improved algorithm here due to space limitation, but only show the procedures of the algorithm for the most case addition i.e. adding weight two coprime divisors, and the most case doubling, i.e. doubling a weight two divisor without ramification points, in the Table 2 and the Table 3 respectively. Notations in both tables are the same as in the Table 1.

We also show the costs of the whole of the algorithm in the Table 4.

The Table 4 shows the costs of $\mathcal{D}_1 + \mathcal{D}_2$ where the first row and the first column show the forms of divisors $\mathcal{D}_1$ and $\mathcal{D}_2$ respectively without P_∞ terms. P_i denotes the point on C with $P_i \neq \pm P_j, j \neq i$.

From the Table 4, we known the cost of the most case addition, which is the case of adding $\mathcal{D}_1 = P_1 + P_2 - 2P_\infty$ and $\mathcal{D}_2 = P_3 + P_4 - 2P_\infty$ with $P_1, P_2 \neq \pm P_3, \pm P_4$, is $2I + 23M$. The cost of the worst case, which is the case of adding $\mathcal{D}_1 = P_1 + P_2 - 2P_\infty$ and $\mathcal{D}_2 = P_1 + P_3 - 2P_\infty$ with $P_3 \neq P_1, P_2$, is $4I + 33M$. Moreover, for doubling, the cost is $2I + 25M$ in both the most case and the worst case, which is doubling of $\mathcal{D}_1 = P_1 + P_2 - 2P_\infty$.

Remark 1. In fact, only upper bounds are shown the Ta-

Input	Weight two coprime reduced divisors $\mathcal{D}_1 = (U_1, V_1)$ and $\mathcal{D}_2 = (U_2, V_2)$	
Output	A weight two reduced divisor $\mathcal{D}_3 = (U_3, V_3) = \mathcal{D}_1 + \mathcal{D}_2$	
Step	Procedure	Cost
1	Compute the resultant r of U_1 and U_2.	$4M$
	$w_1 \leftarrow u_{21} - u_{11};\ w_2 \leftarrow u_{21}w_1 + u_{10} - u_{20};$	
	$r \leftarrow u_{10}(w_2 - u_{20}) + u_{20}(u_{20} - u_{11}w_1);$	
2	If $r = 0$ then $\mathcal{D}_1$ and $\mathcal{D}_2$ have a linear factor in common,	
	and call the exclusive procedure.	—
3	Compute $I_1 \equiv 1/U_1 \bmod U_2$.	$I + 2M$
	$w_3 \leftarrow r^{-1};\ I_1 \leftarrow w_1w_3X + w_2w_3;$	
4	Compute $S \equiv (V_2 - V_1)I_1 \bmod U_2$. (Karatsuba)	$5M$
	$w_1 \leftarrow v_{20} - v_{10};\ w_2 \leftarrow v_{21} - v_{11};\ w_3 \leftarrow i_{10}w_1;\ w_4 \leftarrow i_{11}w_2;$	
	$w_5 \leftarrow (i_{10} + i_{11})(w_1 + w_2) - w_3 - w_4;$	
	$S \leftarrow (w_5 - u_{21}w_4)X - u_{20}w_4 + w_3;$	
5	If $s_1 = 0$ then $\mathcal{D}_3$ should be weight one,	
	and call the exclusive procedure.	—
6	Compute $U_3 = s_1^{-2}((S^2U_1 + 2SV_1)/U_2 - (F - V_1^2)/(U_1U_2))$.	$I + 6M$
	$w_1 \leftarrow s_1^{-1};\ w_2 \leftarrow u_{21} - u_{11};$	
	$u_{30} \leftarrow w_1(w_1(s_0^2 + u_{11} + u_{21} - f_4) + 2(v_{11} - s_0w_2))+$	
	$\quad u_{21}w_2 + u_{10} - u_{20};$	
	$u_{31} \leftarrow w_1(2s_0 - w_1) - w_2;\ u_{32} \leftarrow 1;$	
7	Compute $V_3 \equiv -(SU_1 + V_1) \bmod U_3$.	$6M$
	$w_1 \leftarrow u_{30} - u_{10};\ w_2 \leftarrow u_{11} - u_{31};$	
	$v_{30} \leftarrow s_1u_{30}w_2 + s_0w_1 - v_{10};$	
	$v_{31} \leftarrow s_1(u_{31}w_2 + w_1) - s_0w_2 - v_{11};$	
Total		$2I + 23M$

Table 2: Improved addition for weight two coprime divisors on a genus two HEC

Input	A weight two reduced divisor $\mathcal{D}_1 = (U_1, V_1)$ without ramification points	
Output	A weight two reduced divisor $\mathcal{D}_2 = (U_2, V_2) = 2\mathcal{D}_1$	
Step	Procedure	Cost
1	Compute the resultant r of U_1 and V_1.	$4M$
	$w_1 \leftarrow v_{11}^2;\ w_2 \leftarrow u_{11}v_{11};\ r \leftarrow u_{10}w_1 + v_{10}(v_{10} - w_2);$	
2	If $r = 0$ then $\mathcal{D}_1$ is with a ramification point,	
	and call the exclusive procedure.	—
3	Compute $I_1 \equiv 1/(2V_1) \bmod U_1$.	$I + 2M$
	$w_3 \leftarrow (2r)^{-1};\ I_1 \leftarrow -v_{11}w_3X + (v_{10} - w_2)w_3;$	
4	Compute in $T_1 \equiv (F - V_1^2)/U_1 \bmod U_1$.	$4M$
	$w_2 \leftarrow u_{11} - f_4;\ w_3 \leftarrow 2u_{10};$	
	$t_{10} \leftarrow u_{11}(4u_{10} - u_{11}w_2 - f_3) - f_4w_3 + f_2 - w_1;$	
	$t_{11} \leftarrow u_{11}(2w_2 + u_{11}) + f_3 - w_3$	
5	Compute $S \equiv I_1T_1 \bmod U_1$. (Karatsuba)	$5M$
	$w_1 \leftarrow i_{10}t_{10};\ w_2 \leftarrow i_{11}t_{11};$	
	$w_3 \leftarrow (i_{10} + i_{11})(t_{10} + t_{11}) - w_1 - w_2;$	
	$S \leftarrow (w_3 - u_{11}w_2)X - u_{20}w_2 + w_1;$	
6	If $s_1 = 0$ then $\mathcal{D}_2$ should be weight one,	
	and call the exclusive procedure.	—
7	Compute $U_2 = s_1^{-2}((SU_1 + V_1)^2 - F)/U_1^2$.	$I + 4M$
	$w_1 \leftarrow s_1^{-1};\ u_{30} \leftarrow w_1(w_1(s_0^2 + 2u_{11} - f_4) + 2v_{11});$	
	$u_{31} \leftarrow w_1(2s_0 - w_1);\ u_{32} \leftarrow 1;$	
8	Compute $V_2 \equiv -(SU_1 + V_1) \bmod U_2$.	$6M$
	$w_1 \leftarrow u_{11} - u_{21};$	
	$v_{20} \leftarrow u_{20}(s_1w_1 + s_0) - s_0u_{10} - v_{10};$	
	$v_{21} \leftarrow s_1(u_{21}w_1 + u_{20} - u_{10}) + s_0w_1 - v_{11};$	
Total		$2I + 25M$

Table 3: Improved doubling for a weight two divisor without ramification points on a genus two HEC

	P_1	$2P_1$	P_1+P_2
P_1	$I+5M$	$I+11M$	$2I+17M$
$-P_1$	0	$3M$	$3M$
P_2	$I+\ 3M$	$I+10M$	$2I+17M$
$2P_1$	$I+11M$	$2I+25M$	$4I+33M$
P_1+P_2	$2I+17M$	$4I+33M$	$2I+25M$
$-P_1+P_2$	$3M$	$2I+13M$	$2I+\ 7M$
P_1+P_3	$2I+17M$	$4I+33M$	$4I+33M$
$-P_1+P_3$	$3M$	$2I+13M$	$2I+13M$
P_3+P_4	$I+10M$	$2I+23M$	$2I+23M$

Table 4: Costs of the improved algorithm

Input	Two divisors $\mathcal{D}_1 = P_1+P_2-2P_\infty, \mathcal{D}_2 = P_1+P_3-2P_\infty$ with $P_3 \neq P_1, P_2$
Output	A divisor $\mathcal{D}_3 = \mathcal{D}_1 + \mathcal{D}_2$
Step	Procedure
1	Compute $\mathcal{D}_4 = P_1 - P_\infty$ and $\mathcal{D}_5 = P_3 - P_\infty$.
2	Compute $\mathcal{D}_6 = \mathcal{D}_1 + \mathcal{D}_5$.
3	Compute $\mathcal{D}_3 = \mathcal{D}_4 + \mathcal{D}_6$.

Table 5: The worst case computation

ble 4. E.g. in the case of adding two divisors $\mathcal{D}_1 = P_1+P_2-2P_\infty, \mathcal{D}_2 = P_3+P_4-2P_\infty$, it only takes $I+15M$, when $s_1 = 0$ in the step 5 of the Table 2.

Remark 2. We have slightly modified the procedure for the worst case computation as in the Table 5. This strategy give us less cost of the computation, i.e. $4I + 33M$. The original procedure described in [Har00a] using improved computation in each step costs $6I + 47M$.

5 Application to cryptosystems

In this section, we discuss the performance of genus two HECC using the improved Harley algorithm and compare it with the performance of ECC.

When an elliptic curve of order N over $\mathbb{F}_{q_E}$ is used in ECC, $q_E \approx N$ from (9). On the other hand, when a genus two hyperelliptic curve of order N is used in HECC with the same security as the ECC, the curve can be chosen over $\mathbb{F}_{q_H}$ with $q_H \approx \sqrt{N}$ again from (9). Therefore, genus two HECC which have the same sequrity as ECC over $\mathbb{F}_{q_E}$ can be constructed over $\mathbb{F}_{q_H}$ with $q_H \approx \sqrt{q_E}$.

Now, we denote the cost of a multiplication over $\mathbb{F}_{q_H}$ and $\mathbb{F}_{q_E}$ as M_H and M_E respectively. Then, $M_E \approx 4M_H$, because $q_H \approx \sqrt{q_E}$. Here we suppose that classical multiplication methods take time of $2(\log_2 q)^2$ for a multilication over large $\mathbb{F}_q$ [GG99] is usually used for the multiplication over the definition finite fields, except special implementations such as using large size hardware multipliers and bit-slice implementation [KAHO01]. Therefore, an addition and a doubling on the elliptic curves take $16M_E \approx 64M_H$ and $10M_E \approx 40M_H$ respectively in general assuming Jacobian projective coordinates [IEE99] are used for the elliptic arithmetic.

On the other hand, a genus two hyperelliptic addition takes $4I + 33M$ in the worst case, which is the case of adding two divisors $\mathcal{D}_1 = P_1 + P_2 - 2P_\infty$ and $\mathcal{D}_2 = P_1 + P_3 - 2P_\infty$ with $P_3 \neq P_1, P_2$. In this case, $P_1 = P_2$ induces $P_1, P_2 \in C(\mathbb{F}_{q_H})$, because no points in $C(\mathbb{F}_{q_H^2})\backslash C(\mathbb{F}_{q_H})$ conjugate themselves under the Galois action over $\mathbb{F}_{q_H}$. Also $P_3 \in C(\mathbb{F}_{q_H})$ because $P_1 \in C(\mathbb{F}_{q_H})$. Moreover, if $P_1 \neq P_2$, we also have $P_1, P_2, P_3 \in C(\mathbb{F}_{q_H})$, because both P_2 and P_3 are conjugates of P_1 and uniquely determined by the Galois action over $\mathbb{F}_q$, and $P_2 \neq P_3$. Consequently, we have $P_1, P_2, P_3 \in C(\mathbb{F}_{q_H})$ in the worst case. Therefore, there are $O(q_H^3)$ such pairs ($\mathcal{D}_1$, $\mathcal{D}_2$) among all the $O(q_H^4)$ pairs of divisors in $\mathcal{J}_C(\mathbb{F}_{q_H})$. Then, we can neglect the cost for the worst case when q_H is enough large. Thus, we consider the costs of an addition and a doubling on the hyperelliptic curves to be the costs in the most case, which are $2I_H + 23M_H$ and $2I_H + 25M_H$ respectively where I_H denotes the cost of inversion over $\mathbb{F}_{q_H}$.

In discrete logarithm based cryptosystems, such as ECC and HECC, the cost of scalar multiplications dominates the other operations. Hence we can compare the performance of genus two HECC with that of ECC using the costs of scalar multiplicatioin on a genus two hyperelliptic curve over $\mathbb{F}_{q_H}$ and on a ellptic curve over $\mathbb{F}_{q_E}$.

Considering that several methods are known for such scalar multiplication [Knu98][MvOV97], we will assume that an addition and a doubling occur in the same frequency among a scalar multiplication for simplicity. Then, using the improved Harley algorithm, genus two HECC do surpass ECC in the speed performance if $I_H < 14M_H$, which is usually satisfied.

6 Implementation

In this section, we show an implementation of both the improved Harley algorithm and elliptic curve arithmetic for comparison. Each was implemented over optimal extension fields [BP98], a 93-bit OEF $\mathbb{F}_{q_H} = \mathbb{F}_p(\alpha)$ for hyperelliptic curves and a 186-bit OEF $\mathbb{F}_{q_E} = \mathbb{F}_p(\beta)$ for elliptic curves, where $p = 2^{31} - 1$, α is a root of $X^3 - 5$,

Genus Two Hyperelliptic Curve			Elliptic Curve		
addition	doubling	scalar mul.	addition	doubling	scalar mul.
8.32μs.	8.74μs.	1.98ms.	11.6μs.	6.58μs.	1.76ms.

Table 6: Timings of implementations

and β is a root of $X^6 - 5$. Then, both $\#\mathcal{J}_C(\mathbb{F}_{q_H})$ and $\#E(\mathbb{F}_{q_E})$ are of 186-bit.

We used GNU G++-2.95.2 for each implementation. The Jacobian projective coordinate described in IEEE P1363 draft [IEE99] was used for elliptic addition, and a sliding-window [MvOV97] of width four was applied to each scalar multiplication without precomputation. For inversions over finite fields, the linear algebraic method described in [KMKH99] is used.

The timings of the implementations are given in the Table 6. They were obtained on a Linux PC with Pentium III 866MHz.

From the Table 6, one sees that the performance of genus two HECC is roughly equal to that of ECC. This is somewhat disappointing, as in this implementation $M_E \approx 3.8M_H$ and $I_H \approx 6.4M_H$, according to our complexity analysis of both the algorithms the performance of genus two HECC was expected to surpass ECC. This result is mainly due to difficulty to implement the Harley algotithm, which is much more complicated than elliptic addition algorithms. It is expected that further research on implementation of Harley-like algorithm will actually build genus two HECC faster than ECC.

7 Conclusion

In this paper, we presented an improvement of the Harley algorithm and analysis of complexity shows that the addition algorithm on genus two HECC has lower cost therefore is faster than addition on ECC. Then we shown a tentative implementation of the improved Harley algorithm, in which the performance of genus two HECC is equal to that of ECC. It seems that the overall performance of HECC depends strongly on efficiency of implementation. We believe further improvement on both algorithm and implementation is possible to eventually construct HECC surpassing ECC in the near future.

Acknowledgement

The first author would like to thank Mr. Jun-ichi Kuroki and Mr. Yosuke Miyamoto for their fruitful discussions on the Harley algorithm.

References

[ADH94] L. M. Adleman, J. DeMarrais, and M. D. Huang, *A subexponential algorithm for discrete logarithms over the rational subgroup of the Jacobian of large genus hyperelliptic curves over finite fields*, ANTS-I (L. M. Adleman and M. D. Huang, eds.), Lecture Notes in Computer Science, no. 877, Springer-Verlag, 1994, pp. 28–40.

[AH96] L. M. Adleman and M. D. Huang, *Counting rational points on curves and Abelian varieties over finite fields*, ANTS-II (H. Cohen, ed.), Lecture Notes in Computer Science, no. 1122, Springer-Verlag, 1996, pp. 1–16.

[BP98] D. V. Bailey and C. Paar, *Optimal extension fields for fast arithmetic in public-key algorithms*, Advances in Cryptology - CRYPTO'98 (H. Krawczyk, ed.), Lecture Notes in Computer Science, no. 1462, Springer-Verlag, 1998, pp. 472–485.

[BSS99] I. Blake, G. Seroussi, and N. Smart, *Elliptic curves in cryptography*, London Mathematical Society Lecture Note Series, no. 265, Cambridge U. P., 1999.

[Can87] D. G. Cantor, *Computing in the Jacobian of hyperelliptic curve*, Math. Comp. **48** (1987), 95–101.

[CF96] J. W. S. Cassels and E. V. Flynn, *Proglegomena to middlebrow arithmetic of curves of genus 2*, London Mathematical Society Lecture Note Series, no. 230, Cambridge U. P., 1996.

[CMKT00] J. Chao, K. Matsuo, H. Kawashiro, and S. Tsujii, *Construction of hyperelliptic curves with CM and its application to cryptosystems*, Advances in Cryptology - ASIACRYPT2000 (T. Okamoto, ed.), Lecture Notes in Computer Science, no. 1976, Springer-Verlag, 2000, pp. 259–273.

[CMT00] J. Chao, K. Matsuo, and S. Tsujii, *Fast construction of secure discrete logarithm problems over Jacobian varieties*, Information Security for Global Information Infrastructures: IFIP TC 11 16th Annual Working Conference on Information Security (S. Qing and J.Eloff, eds.), Kluwer Academic Pub., 2000, pp. 241–250.

[EG00] A. Enge and P. Gaudry, *A general framework for subexponential discrete logarithm algorithms*, Reserch Report LIX/RR/00/04, LIX École polytechnique, 2000.

[FM98] G. Frey and M. Müller, *Arithmetic of modular curves and applications*, preprint, 1998.

[FP97] R. Flassenberg and S. Paulus, *Sieving in function fields*, preprint, 1997.

[Gau00] P. Gaudry, *An algorithm for solving the discrete log problem on hyperelliptic curves*, Advances in Cryptology - EUROCRYPT2000 (B. Preneel, ed.), Lecture Notes in Computer Science, no. 1807, Springer-Verlag, 2000, pp. 19–34.

[GCL92] K. O. Geddes, S. R. Czapor, and G. Labahn, *Algorithms for computer algebra*, Kluwer Academic Pub., 1992.

[GG99] J. Gathen and J. Gerhard, *Modern computer algebra*, Cambridge U. P., 1999.

[GH00] P. Gaudry and R. Harley, *Counting points on hyperelliptic curves over finite fields*, ANTS-IV (W. Bosma, ed.), Lecture Notes in Computer Science, no. 1838, Springer-Verlag, 2000, pp. 297–312.

[GHS00] P. Gaudry, F. Hess, and N. Smart, *Constructive and destructive facets of Weil descent on elliptic curves*, to appear in J. Cryptology, 2000.

[Har00a] R. Harley, *adding.text*, `http://cristal.inria.fr/~harley/hyper/`, 2000.

[Har00b] R. Harley, *doubling.c*, `http://cristal.inria.fr/~harley/hyper/`, 2000.

[IEE99] IEEE Standards Dept., *IEEE P1363/D13*, 1999.

[KAHO01] T. Kobayashi, K. Aoki, F. Hoshino, and H. Oguno, *Software implementation of parallel elliptic curve cryptosystem*, Proc. of SCIS2001, vol. I, 2001, (in Japanese), pp. 299–303.

[Kam91] W. Kampkötter, *Explizite gleichungen für Jacobische varietäten hyperelliptischer kurven*, Ph.D. thesis, GH Essen, 1991.

[Ked01] K. S. Kedlaya, *Counting points on hyperelliptic curves using Monsky–Washinitzer cohomology*, preprint, 2001.

[KMKH99] T. Kobayashi, H. Morita, K. Kobayashi, and F. Hoshino, *Fast elliptic curve algorithm combining frobenius map and table reference to adapt to higher characteristic*, Advances in Cryptology - EUROCRYPTO'99, Lecture Notes in Computer Science, no. 1592, Springer-Verlag, 1999, pp. 176–189.

[Knu98] D. E. Knuth, *The art of computer programming*, 3rd ed., vol. 2 Seminumerical Algorithms, Addison Wesley, 1998.

[Kob89] N. Koblitz, *Hyperelliptic curve cryptosystems*, J. Cryptology **1** (1989), 139–150.

[Kob98] N. Koblitz, *Algebraic aspects of cryptography*, Algorithms and Computation in Mathematics, no. 3, Springer-Verlag, 1998.

[Mum84] D. Mumford, *Tata lectures on theta II*, Progress in Mathematics, no. 43, Birkhäuser, 1984.

[MvOV97] A. Menezes, P. van Oorschot, and S. Vanstone, *Handbook of applied cryptography*, CRC Press, 1997.

[Nag00] K. Nagao, *Improving group law algorithms for Jacobians of hyperelliptic curves*, ANTS-IV (W. Bosma, ed.), Lecture Notes in Computer Science, no. 1838, Springer-Verlag, 2000, pp. 439–448.

[Pau96] S. Paulus, *An algorithm of sub-exponential type computing the class group of quadratic orders over principal ideal domains*, ANTS-II, Lecture Notes in Computer Science, no. 1122, Springer-Verlag, 1996, pp. 243–257.

[Pil90] J. Pila, *Frobenius maps of Abelian varieties and finding roots of unity in finite fields*, Math. Comp. **55** (1990), 745–763.

[PS98] S. Paulus and A. Stein, *Compring real and imaginary arithmetics for divisor class groups of hyperelliptic curves*, ANTS-III, Lecture Notes in Computer Science, no. 1423, Springer-Verlag, 1998, pp. 576–591.

[Sma99] N. Smart, *On the performance of hyperelliptic cryptosystems*, Advances in Cryptology - EUROCRYPTO'99, Lecture Notes in Computer Science, no. 1592, Springer-Verlag, 1999, pp. 165–175.

[Spa94] A. M. Spallek, *Kurven vom geshlcht 2 und ihre anwendung in public-key-kryptosystemem*, Ph.D. thesis, GH Essen, 1994.

[SS98] Y. Sakai and K. Sakurai, *Design of hyperelliptic cryptosystems in small characteristic and a software imprementation over F_{2^n}*, Advances in Cryptology - ASIACRYPT'98 (K. Ohta and D. Pei, eds.), Lecture Notes in Computer Science, no. 1514, Springer-Verlag, 1998, pp. 80–94.

[SSI98] Y. Sakai, K. Sakurai, and H. Ishizuka, *Secure hyperelliptic cryptosystems and their performance*, Public Key Cryptography (H. Imai and Y. Zheng, eds.), Lecture Notes in Computer Science, no. 1431, Springer-Verlag, 1998, pp. 164–181.

[Wen00] A. Weng, *Constructing hyperelliptic curves of genus* 2 *suitable for cryptography*, preprint, 2000.