An addition algorithm in Jacobian of $C_{ab}$ curves

Seigo Arita

Internet Systems Research Laboratories, NEC Corporation, Miyamae, Kawasaki 216-8555, Japan

Received 23 January 2001; received in revised form 13 November 2001; accepted 7 May 2002

Abstract

Nowadays, elliptic curve cryptosystems receive attention and much effort is being dedicated to make it more and more practical. It is worthwhile to construct discrete logarithm based cryptosystems using more general algebraic curves, because it supplies more security sources for public key cryptosystems. The presented paper introduces $C_{ab}$ curves. Roughly speaking, a curve is $C_{ab}$ if it is non-singular in its affine part and if its singularity at infinity is “nice”. $C_{ab}$ curves compose a large family of algebraic curves, including elliptic, hyperelliptic and superelliptic curves. The paper shows an addition algorithm in Jacobian group of $C_{ab}$ curves in three steps: firstly with a geometrical point of view, which is impractical, secondly by translating the algorithm in the language of ideals, and finally, the final algorithm in which some costly steps are removed. The paper also gives experiments that prove that the algorithm behaves well in practice.

© 2003 Elsevier B.V. All rights reserved.

1. Introduction

Nowadays, elliptic curve cryptosystems (ECC) receive attention and much effort is being dedicated to make it more and more practical. ECC is a public key cryptosystem based on the discrete logarithm problem on a group of points on an elliptic curve. A general algebraic curve also has a group, Jacobian group, roughly speaking, which is a group of point sets on a curve. It is worthwhile to construct discrete logarithm based cryptosystems using Jacobian group of general algebraic curves beyond elliptic or hyperelliptic curves, because it supplies more security sources for public key cryptosystems.

Suppose a family $F$ of algebraic curves (which is bigger than a family of elliptic curves) is given. To construct a discrete logarithm based cryptosystems using the family $F$, we need to solve the following two basic problems.

Problem 1. Find an efficient algorithm for addition in Jacobian group of any curve in the family $F$. 
Problem 2. Find an efficient algorithm to find a curve with Jacobian group of almost prime order in the family $F$.

Problem 1 is solved in the case of hyperelliptic curves [1, 6], and can be dealt with rather easily in superelliptic curves [4]. Problem 2 is partially solved in hyperelliptic curves [7, 11, 2]. Miura [9] has found a family of algebraic curves named “$C_{ab}$ curve” in the development of algebraic geometry codes. Roughly speaking, a curve is $C_{ab}$ if it is non-singular in its affine part and if its singularity at infinity is “nice”, in the sense that there is only one place at infinity and it is of degree 1. $C_{ab}$ curves compose a large family of algebraic curves, including elliptic, hyperelliptic and superelliptic curves.

This paper gives a solution for Problem 1 in the case of $C_{ab}$ curves. An algorithm for addition in Jacobian group of $C_{ab}$ curves is given in three steps: firstly with a geometrical point of view, which is impractical, secondly by translating the algorithm in the language of ideals, and finally, the final algorithm in which some costly steps are removed. This paper also gives experiments that prove that the algorithm behaves well in practice.

2. Preliminaries

This section gives preliminaries for Jacobian group of an algebraic curve and for a Gröbner basis of an ideal in a polynomial ring.

2.1. Jacobian group of an algebraic curve

Take an algebraic curve $C$ defined over a field $K$. Let $\bar{K}$ be its algebraic closure. A divisor $D$ is defined to be a formal sum $D = \sum m_i P_i$ for integers $m_i$ and rational points (strictly, places) $P_i$ of $C$ over $\bar{K}$. When $m_i \geq 0$ for all $i$, the divisor $D = \sum m_i P_i$ is called positive. The integer $\deg(D) = \sum m_i$ is called degree of a divisor $D = \sum m_i P_i$. All divisors on a curve $C$ compose an abelian group $D$ under the formal addition, and all divisors of degree zero become a subgroup $D_0$ of $D$. Let $D_K$ and $D^0_K$ be invariant subgroups of $D$ and $D_0$ under the action of $\text{Gal}(\bar{K}/K)$, respectively. Elements of $D_K$ are called divisors defined over $K$.

For a rational function $f$ on a curve $C$, let $v_P(f) = n(\text{or} \ -n)$ be the order $n$ of zero (or, pole) of $f$ at a point $P$ on $C$. Then, $(f) := \sum_P v_P(f) P$ becomes a divisor of degree 0, called a principal divisor of $f$. The partial sums $(f)_0 := \sum_{P, v_P(f) \geq 0} v_P(f) P$ and $(f)_{\infty} := \sum_{P, v_P(f) < 0} -v_P(f) P$ are called a zero divisor and a pole divisor of $f$, respectively. Note both $(f)_0$ and $(f)_{\infty}$ are positive divisors, and $(f)_0 = (f)_{\infty}$. All principal divisors $\{(f) | f \in \bar{K}(C)\}$ compose a subgroup $P$ of $D^0$. The quotient group $J(C) = D^0/P$ is called Jacobian group of $C$. The invariant subgroup $J_K(C) = D^0_K/P_K$ of $J(C)$ under the action of $\text{Gal}(\bar{K}/K)$ is called Jacobian group of $C$ defined over $K$.

For a divisor $D$ defined over $K$,

$$L(D) = \{ f \in \bar{K}(C) | (f) + D \geq 0 \} \cup \{ 0 \}$$
is a finite dimensional vector space over $K$. By Riemann’s theorem, $\dim L(D) \geq \deg(D) + 1 - g$, where $g$ denotes the genus of $C$.

For details, see [10, Chapter 2].

### 2.2. Monomial order and Groebner bases

Let $\mathbb{N}_0$ denote the set of non-negative integers. For an $n$-variable monomial $x^\alpha = x_1^{\alpha_1} \cdots x_n^{\alpha_n}$, an $n$-tuple of integers $\alpha = (\alpha_1, \ldots, \alpha_n) \in \mathbb{N}_0^n$ is called a multi-degree of $x^\alpha$, denoted by MD($x^\alpha$). A well order $<$ on $\mathbb{N}_0^n$ is called monomial order if $\alpha + \gamma < \beta + \gamma$ holds whenever $\alpha < \beta$ and $\gamma \in \mathbb{N}_0^n$. A monomial order on $\mathbb{N}_0^n$ determines a well order on the set of all monomials through multi-degrees, called a monomial order, too.

Suppose a monomial order is given. For an $n$-variable polynomial $f$, the largest term (monomial) appearing in $f$ with respect to the monomial order is called a leading term (monomial) of $f$, denoted by $LT(f)$ ($LM(f)$).

By using monomial orders, we can describe the division algorithm for $n$-variable polynomials, in which a polynomial $f$ is divided by a set of polynomials $G$.

Fix a monomial order. A subset $G = \{g_1, \ldots, g_m\}$ of an ideal $I$ of $n$-variable polynomial ring $R = K[x_1, \ldots, x_n]$ is called a Groebner basis of $I$ when leading monomials $LM(g_1), \ldots, LM(g_m)$ generate an ideal $LM(I)$ of leading monomials in $I$:

$$LM(I) = (LM(g_1), \ldots, LM(g_m)).$$

Any ideal of $n$-variable polynomial ring $K[x_1, \ldots, x_n]$ has a Groebner basis. If $G = \{g_1, \ldots, g_m\}$ is a Groebner basis of an ideal $I$, $G$ generates $I$.

For an ideal $I$, the set of all of the multi-degrees of monomials outside $LM(I)$ is called $\Delta$-set of $I$, denoted by $\Delta(I)$:

$$\Delta(I) = \{ \alpha \in \mathbb{N}_0^n \mid x^\alpha \not\in LM(I) \}.$$ 

Let $\delta(I)$ denote the number of elements in $\Delta(I)$. Obviously, when $\delta(I)$ is finite, $\delta(I) = \dim_K(R/I)$. So, when $\delta(I)$ is finite, it is equal to the number of points which are zeros of $I$, including multiplicities.

For a polynomial set $G = \{g_1, \ldots, g_m\}$, we set

$$\delta(g_1, \ldots, g_m) := \# \left( \mathbb{N}_0^n - \bigcup_{i=1}^{m} (MD(LM(g_i)) + \mathbb{N}_0^n) \right)$$

($\#S$ denotes the number of elements in the set $S$). Then, for an ideal $I$ satisfying $\delta(I) < \infty$ and for its subset $G = \{g_1, \ldots, g_m\}$, we have

$$G \text{ is a Groebner basis of } I \iff \delta(I) = \delta(g_1, \ldots, g_m).$$  \hspace{1cm} (1)

Use of a Groebner basis justifies the division algorithm for $n$-variable polynomials. That is, a polynomial $f$ is a member of an ideal $I$ if and only if the remainder of $f$ divided by the Groebner basis of $I$ is equal to zero. Although there are several Groebner bases for a given ideal, the reduced Groebner basis is uniquely determined.
up by a given ideal. A Groebner basis $G$ of an ideal $I$ is called reduced when (1) the coefficient of $\text{LT}(p)$ is 1 for all $p \in G$, (2) any term appearing in $p$ does not belong to $(\text{LT}(G - \{ p \}))$ for all $p \in G$. In this paper, a Groebner basis is always considered in the reduced form.

For details, see [3, Chapter 2].

3. $C_{ab}$ curve

This section, after Miura, defines $C_{ab}$ curves and shows some properties of them [9, 8].

Let $C$ be an algebraic curve with a place $P$ of degree one over a perfect field $K$. Take the ring $L(\infty P)$ of functions on $C$ which are holomorphic away from $P$:

$$L(\infty P) = \{ f \in K(C) \mid v_Q(f) \geq 0 \ (\forall Q \neq P) \}.$$  

All of the pole numbers $-v_P(f)$ at $P$ of $f \in L(\infty P)$ become a monoid $M_P$:

$$M_P = \{ -v_P(f) \mid f \in L(\infty P) \}.$$  

Take a minimum system $A = \{ a_1, a_2, \ldots, a_t \}$ ($a_1 < a_2 < \cdots < a_t$) of generators of $M_P$ as a monoid:

$$M_P = N_0a_1 + N_0a_2 + \cdots + N_0a_t = \langle A \rangle,$$

where $N_0$ denotes the set of non-negative integers. Note $\gcd(a_1, \ldots, a_t) = 1$, since $M_P$ is co-finite in $N_0$.

For $A = \{ a_1, \ldots, a_t \}$, define a function $\Psi_A$ on $N_0^t$ as

$$\Psi_A(n_1, \ldots, n_t) = \sum_{i=1}^{t} a_in_i \quad (n = (n_i) \in N_0^t).$$

**Definition 1** ($C_{ab}$ order). For $m = (m_1, \ldots, m_t)$, and $n = (n_1, \ldots, n_t) \in N_0^t$, define an order $\succ_A$ as

$$m \succ_A n \iff \Psi_A(m) > \Psi_A(n)$$

or

$$\Psi_A(m) = \Psi_A(n), \ m_1 = n_1, \ldots, m_{i-1} = n_{i-1},$$

$$m_i < n_i \text{ for } a_i \in [1..t].$$

Then, the order $\succ_A$ becomes a monomial order, called “$C_{ab}$ order of type $A$”.

For each $a \in \langle a_1, \ldots, a_t \rangle$, take the smallest $m$ with respect to $C_{ab}$ order of type $A$ satisfying $\Psi_A(m) = a$, and put those as $B(A)$:

$$B(A) = \{ \text{the smallest } m \in N_0^t \text{ w.r.t } C_{ab} \text{ order of type } A$$

$$\text{satisfying } \Psi_A(m) = a \mid a \in \langle A \rangle \}.$$
Take a set $V(A)$ of ‘minimum’ elements not belonging to $B(A)$:

$$V(A) = \{ l \in N_0^t \setminus B(A) \mid l = m + n, m \in N_0^t \setminus B(A), n \in N_0^t \Rightarrow n = (0,0,\ldots,0) \}.$$ 

$V(A)$ is a finite set as seen later.

**Theorem 2.** Let $C$ be an algebraic curve defined over a perfect field $K$ with a place $P$ of degree one. Suppose $M_P$ has a minimum system $A = \{ a_1, \ldots, a_t \}$ ($a_1 < \cdots < a_t$) of generators as a monoid. Then, the curve $C$ has a non-singular affine model in $t$-dimensional affine space defined by the equations

$$F_m = X^m + a_i X^l + \sum_{n \in B(A), \Psi_A(n) < \Psi_A(m)} a_n X^n \quad (m \in V(A))$$

with a unique $l \in B(A)$ satisfying $\Psi_A(m) = \Psi_A(l)$, and $\mathcal{A}(\neq 0)$, $\mathcal{A} \in K$. The affine model has a unique point $P_\infty$ at infinity, which corresponds to the place $P$.

Conversely, for $A = \{ a_1, \ldots, a_t \}$ such that $\gcd(a_1, \ldots, a_t) = 1$, $a_1 < \cdots < a_t$, if the affine curve defined by Eqs. (2) is non-singular, and Eqs. (2) compose a Groebner basis w.r.t. $C_{ab}$ order of type $A$, putting

$$x_i = X_i \mod \{ F_m \mid m \in V(A) \} \quad (i = 1, \ldots, t),$$

we have

$$-v_{P_\infty}(x_i) = a_i \quad (i = 1, \ldots, t),$$

$$M_{P_\infty} = (A).$$

In particular,

$$\Psi_A(n_1, \ldots, n_t) = -v_{P_\infty}(x_1^{n_1} \cdots x_t^{n_t}).$$

The affine curve $F_m = 0$ ($m \in V(A)$) obtained from $A = \{ a_1, \ldots, a_t \}$ ($\gcd(a_1, \ldots, a_t) = 1, a_1 < \cdots < a_t$), is called a “$C_{ab}$ curve of type $A$”.

It is not trivial to determine $V(A)$ for a given $A$. For $i = 0, 1, \ldots, a_1 - 1$, put

$$b_i = \min \{ b \in (a_2, a_3, \ldots, a_t) \mid b \equiv i \pmod{a_1} \}$$

and put

$$T(A) = \{ \text{the smallest } m \in N_0^t \setminus B(A) \mid \Psi_A(m) = b_i \mid i = 0, 1, \ldots, a_1 - 1 \}.$$ 

The set $V(A)$ is easily determined by the following proposition.

**Proposition 3.** We have

$$V(A) \subset T(A) + \{(0,\ldots,0,\tilde{i},0,\ldots,0) \mid i = 2, \ldots, t \} \setminus T(A).$$

**Proposition 4.** The genus $g(A)$ of a $C_{ab}$ curve of type $A = \{ a_1, \ldots, a_t \}$ is given by

$$g(A) = \sum_{i=1}^{a_1-1} \left[ \frac{b_i}{a_1} \right].$$
In particular, when \( A = \{a, b\} \),
\[
g(a, b) = (a - 1)(b - 1)/2.
\]

**Example.** \( C_{3,4} \) curve. Let \( A = \{3, 4\} \). By Proposition 3, we have
\[
B(A) = \{(0, 0), (1, 0), (0, 1), (2, 0), \ldots\},
\]
\[
T(A) = \{(0, 0), (0, 1), (0, 2)\},
\]
\[
V(A) = \{(0, 3)\}.
\]

Since \( \Psi_A((0, 3)) = 12 \) is equal to the value of \( \Psi_A \) for \((4, 0) \in B(A)\), we see that a \( C_{3,4} \) curve (i.e. \( C_{ab} \) curve of type \( \{3, 4\} \)) is a plane curve defined by a polynomial of the form
\[
Y^3 = a_0X^4 + a_1XY^2 + a_2X^2Y + a_3X^3 + a_4Y^2 + a_5XY
\]
\[
+ a_6X^2 + a_7Y + a_8X + a_9 \quad (5)
\]
by Theorem 2.

Similarly a \( C_{ab} \) curve of type \( A = \{a, b\} \) (\( \gcd(a, b) = 1 \)) is a plane curve defined by a polynomial of the form
\[
F(X, Y) = \sum_{0 \leq i \leq b, 0 \leq j \leq a, ai + bj \leq ab} x_{i, j}X^i Y^j. \quad (6)
\]

Galbraith et al. [4] call a non-singular plane curve with an equation of the form
\[
Y^n = a_0X^n + \cdots + a_0,
\]
superelliptic curves, where \( n \) is coprime with the characteristic of the definition field, and \( n \) and \( \delta \) are prime to each other. Obviously, superelliptic curves are special cases of the plane \( C_{ab} \) curves.

**Example.** \( C_{3,5,7} \) curve. Let \( A = \{3, 5, 7\} \). By Proposition 3, we have
\[
B(A) = \{(0, 0, 0), (1, 0, 0), (0, 1, 0), (2, 0, 0), (0, 0, 1), \ldots\},
\]
\[
T(A) = \{(0, 0, 0), (0, 1, 0), (0, 0, 1)\},
\]
\[
V(A) = \{(0, 2, 0), (0, 1, 1), (0, 0, 2)\}.
\]

\( \Psi_A((0, 2, 0)) = 10 \) is equal to the value of \( \Psi_A \) for \((1, 0, 1) \in B(A)\), \( \Psi_A((0, 1, 1)) \) is equal to the value of \( \Psi_A \) for \((4, 0, 0) \in B(A)\), and \( \Psi_A((0, 0, 2)) \) is equal to the value of \( \Psi_A \) for \((3, 1, 0) \in B(A)\). So, by Theorem 2, \( C_{3,5,7} \) curve is a space curve defined by three equations of the form
\[
Y^2 = a_0XZ + a_1X^3 + a_2XY + a_3Z + a_4X^2 + a_5Y + a_6X + a_7,
\]
YZ = b_0X^4 + b_1X^2Y + b_2XZ + b_3X^3 + b_4XY + b_5Z + b_6X^2
+ b_7Y + b_8X + b_9,

Z^2 = c_0X^3Y + c_1X^2Z + c_2X^4 + c_3X^2Y + c_4XZ + c_5X^3 + c_6XY
+ c_7Z + c_8X^2 + c_9Y + c_{10}X + c_{11}.

(7)

4. An addition algorithm in Jacobian with divisors

Let \( C \) be an algebraic curve of genus \( g \) with a rational point \( P_\infty \) over a perfect field \( K \). As seen in Section 2.1, Jacobian group \( J_K(C) \) of \( C \) over \( K \) is the quotient group \( D_0/K \). When an element \( j \) in \( J_K(C) \) has a representative \( D \) in \( D_0/K \), we denote \( j = [D] \);

\[
J_K(C) = D_0/K, \quad j = [D].
\]

**Definition 5.** We call a divisor \( D \) of the form \( D = E - mP_\infty \) with an effective divisor \( E \) prime to \( P_\infty \) and with some integer \( m \) between 0 and \( g \), a semi-normal divisor.

**Lemma 6.** Every element in \( J_K(C) \) is represented by a semi-normal divisor.

**Proof.** Let \( j = [D] \) be any element in \( J_K(C) \). By Riemann’s theorem,

\[
\dim L(D + gP_\infty) \geq g + 1 - g = 1.
\]

So, with some nonzero function \( f \),

\[
D + gP_\infty + (f) \geq 0.
\]

Letting \( E = D + gP_\infty + (f) \), we have \( j = [E - gP_\infty] \). □

In general, there are several semi-normal divisors, which represent the same element in Jacobian. However, we can determine a unique representative by using the following algorithm.

**Algorithm 1.** Input: a divisor \( D = E - nP_\infty \) of degree 0 with an effective divisor \( E \) prime to \( P_\infty \), Output: a semi-normal divisor \( G \) equivalent to \( -D \).

1. Find \( f \in L(\infty P_\infty) \) satisfying \( (f)_0 \geq E \) with the smallest pole order \( -v_{P_\infty}(f) \) at \( P_\infty \).
2. \( G \leftarrow -D + (f) \).

**Proposition 7.** Algorithm 1 outputs a constant divisor for equivalent divisors.
Proof. Let $D_1 = E_1 - n_1 P_\infty$ and $D_2 = E_2 - n_2 P_\infty$ be equivalent divisors of degree 0 with effective divisors $E_i$ ($i = 1, 2$) prime to $P_\infty$.

With some non-zero function $\lambda$, we have

$$E_1 - n_1 P_\infty = E_2 - n_2 P_\infty + (\lambda).$$

For $D_1$, take the function $f_1 \in L(\infty P_\infty)$ satisfying $(f_1)_0 \geq E_1$ as in Algorithm 1. Then we have

$$(f_1 \lambda^{-1}) = (f_1) - (\lambda)$$

$$= (f_1)_0 - E_1 + E_2 + (n_1 + v P_\infty (f_1) - n_2) P_\infty.$$ 

Since $(f_1)_0 - E_1 + E_2 \geq E_2$, letting $f_2 = f_1 \lambda^{-1}$, we have $f_2 \in L(\infty P_\infty)$, $\quad (f_2)_0 \geq E_2$.

Because $\lambda$ is independent from the choice of $f_1$ and $f_2$, the $f_1$ with the smallest pole number at $P_\infty$ corresponds to the $f_2$ with the smallest pole number at $P_\infty$. Then,

$$-E_2 + n_2 P_\infty + (f_2) = -E_2 + n_2 P_\infty + (f_1) - E_1 + E_2 + (n_1 - n_2) P_\infty$$

$$= -E_1 + n_1 P_\infty + (f_1)$$

shows that outputs of Algorithm 1 for $D_1$ and $D_2$ are the same. \qed

Definition 8. We call divisors obtained as outputs of Algorithm 1 normal divisors.

As Algorithm 1 outputs a divisor equivalent to $-1$ times the input divisor, we can normalize any semi-normal divisor by applying Algorithm 1 twice. So, Lemma 6 and Proposition 7 show that

Theorem 9. Any element in Jacobian is represented by a unique normal divisor.

Now, addition in Jacobian can be done by normalizing the added divisors.

Algorithm 2. Input: (semi-)normal divisors $D_1 = E_1 - n_1 P_\infty$ and $D_2 = E_2 - n_2 P_\infty$.

Output: a normal divisor $D_3$ equivalent to $D_1 + D_2$.

1. Applying Algorithm 1 for $D_1 + D_2 = (E_1 + E_2) - (n_1 + n_2) P_\infty$, get a normal divisor $D'$.
2. Applying Algorithm 1 for $D'$, get a normal divisor $D_3$.

To perform Algorithms 1 and 2 on computers, we need to encode divisors in some way. The most straightforward way is to encode divisors as point sets with multiplicities as in [12]. But to encode divisors involved in our algorithms as point sets, we need to deal with $g$th degree extension field of the definition field $K$, and it hurts the efficiency of the algorithms.

In $C_{ab}$ curves, Jacobian group is naturally isomorphic to the ideal class group of the coordinate ring. The next section, due to this fact, realizes Algorithms 1 and 2 by ideal computations in the coordinate ring.
5. An addition algorithm in Jacobian with ideals

Let \( C \) be a \( C_{ab} \) curve of type
\[
A = \{a_1, \ldots, a_t\} \quad (\gcd(a_1, \ldots, a_t) = 1, a_1 < \cdots < a_t)
\]
defined over a perfect field \( K \) with equations
\[
F_m = X^m + x_i X^l + \sum_{n \in \mathbb{N}, x_i(n) < x_i(m)} x_n X^n \quad (m \in V(A)) \tag{8}
\]
and let \( P_\infty \) be the unique point on \( C \) at infinity.

By the definition of a \( C_{ab} \) curve, we have
\[
L(\infty P_\infty) = K[x_1, x_2, \ldots, x_t] \\
\simeq K[X_1, X_2, \ldots, X_t]/(F_m \mid m \in V(A)).
\]
So, the coordinate ring \( R_K = K[X_1, X_2, \ldots, X_t]/(F_m \mid m \in V(A)) \) is a Dedekind domain.

In general, for a Dedekind domain \( R \), Jacobian group of \( \text{Spec}(R) \) is just the ideal class group \( H(R) \) of \( R \) ([5, Example 6.3.2, p. 132]). In this case, the isomorphism \( \Phi \) is given by
\[
J_K(C) \cong H(R_K) \\
\Phi : \left[ \sum_p n_P P - n_{P_\infty} \right] \mapsto \left[ L \left( \infty P_\infty - \sum_p n_P P \right) \right].
\]

Remember that \( C_{ab} \) order is defined by the order function \( x_i \), and by Theorem 2,
\[
x_i(n_1, \ldots, n_t) = -v_{P_\infty}(x_i^{n_1} \cdots x_i^{n_t}).
\]
So, we see that \( C_{ab} \) order puts monomials in order by pole numbers at \( P_\infty \) of them.

Hence, by applying the isomorphism \( \Phi \) for Algorithm 1, we get

**Algorithm 3.** Input: an ideal \( I \) of the coordinate ring \( R_K \) of a \( C_{ab} \) curve of type \( A \),  
Output: an ideal \( J \) equivalent to the inverse ideal of \( I \)

1. \( f \leftarrow \text{the smallest } f \neq 0 \in I \) with respect to the \( C_{ab} \) order of type \( A \).
2. \( J \leftarrow (f) : I \).

Using Algorithm 3 twice as in the case of Algorithm 2, we get the following addition algorithm in Jacobian of \( C_{ab} \) curve.

**Algorithm 4.** Input: ideals \( I_1 \) and \( I_2 \) of the coordinate ring \( R_K \) of a \( C_{ab} \) curve of type \( A \),  
Output: an ideal \( I_3 \) equivalent to the ideal product \( I_1 \cdot I_2 \)

1. \( I \leftarrow I_1 \cdot I_2 \).
2. \( f \leftarrow \text{the smallest } f \neq 0 \in I \) with respect to the \( C_{ab} \) order of type \( A \).
(3) $J \leftarrow (f) : I$.  
(4) $g \leftarrow$ the smallest $g(\neq 0) \in J$ with respect to the $C_{ab}$ order of type $A$.  
(5) $I_3 \leftarrow (g) : J$.

Algorithm 4 can be implemented using Groebner basis with respect to $C_{ab}$ order. For example, we show a sample code by Mathematica Ver. 3 in Appendix A. Computation of ideal quotients costs Algorithm 4 its efficiency. We remove computation of them from the addition algorithm. In Algorithm 4, we have

$$(f) = I_1 \cdot I_2 \cdot J,$$

$$(g) = J \cdot I_3.$$  

So,

$$I_1 \cdot I_2 \cdot (g) = I_1 \cdot I_2 \cdot J \cdot I_3 = (f) \cdot I_3$$

and we see

$$I_3 = g \cdot I_1 \cdot I_2 / f.$$  

Thus, we get the following addition algorithm without ideal quotients.

**Algorithm 5.** Input: ideals $I_1$ and $I_2$ of the coordinate ring $R_K$ of a $C_{ab}$ curve of type $A$, Output: an ideal $I_3$ equivalent to the ideal product $I_1 \cdot I_2$.

(1) $I \leftarrow I_1 \cdot I_2$.  
(2) $f \leftarrow$ the smallest polynomial $f(\neq 0) \in I$ with respect to the $C_{ab}$ order of type $A$.  
(3) $g \leftarrow$ the smallest polynomial $g(\neq 0)$ with respect to the $C_{ab}$ order of type $A$ s.t. $g \cdot I \subseteq (f)$.  
(4) $I_3 \leftarrow g \cdot I / f$.

Remember $g$ was the smallest member of $J = (f) : I$. So, by the definition of the ideal quotient, $g$ is the smallest polynomial satisfying $gI \subseteq (f)$.

6. Details of implementation of the addition algorithm

This section explains the details of implementation of Algorithm 5, showing an example of performing the algorithm. For a $C_{34}$ curve on the prime field $K = F_{17}$ with equation

$$F = Y^3 + X^4 + 1,$$

we compute the double of

$$I_1 = \{ f_1 = X^2 + 14Y + 4X + 5, f_2 = XY + 3Y + 4X + 9, f_3 = Y^2 + 9Y + 16X + 2 \}$$

in its Jacobian group $J_K(C)$.  

---

22  
In $C_{34}$ order, monomials are put in the ascending order as follows;

\[ 1, X, Y, X^2, XY, Y^2, X^3, X^2Y, XY^2, \ldots . \]  

(9)

In the below, for a polynomial $f$ and an ideal $G$, let $\bar{f}^G$ denote the remainder of a polynomial $f$ divided by an ideal $G$.

1. At the first step, we compute the Groebner basis of the ideal product $I = I_1 \cdot I_1$ with respect to the $C_{34}$ order. Remember that $\delta(I)$ means the number of points which are zeros of $I$, including multiplicities (Section 2.2). So, $\delta(I) = \delta(I_1) + \delta(I_1) = 6$. Then, if the ideal $I$ contains a subset of the form

\[ \{X^3 + \ldots, X^2Y + \ldots, XY^2 + \ldots \} , \]

it must be a Groebner basis of $I$ (see Eq. (1)).

We compute members in $I = I_1 \cdot I_1$ as follows:

\[ g_1 \leftarrow \frac{f_1^{(F)}}{f_2^{(F)}} = X^4 + \ldots \]
\[ g_2 \leftarrow \frac{f_1 \cdot f_2^{(g1,F)}}{f_2^{(g1,F)}} = X^3Y + \ldots \]
\[ g_3 \leftarrow \frac{f_2^{(g2,F)}}{f_2^{(g2,F)}} = X^2Y^2 + \ldots \]
\[ g_4 \leftarrow \frac{f_1 \cdot f_3^{(g1,g2,F)}}{f_3^{(g1,g2,F)}} = XY^2 + \ldots \]
\[ g_5 \leftarrow \frac{f_2 \cdot f_3^{(g2,g3,F)}}{f_3^{(g2,g3,F)}} = X^2Y + \ldots \]
\[ g_6 \leftarrow \frac{f_2 \cdot f_3^{(g3,F)}}{f_3^{(g3,F)}} = X^3 + \ldots \]

Then,

\[ I \leftarrow \{g_6, g_5, g_4\} \]

is a Groebner basis of the ideal $I$ by the above remark.

2. $f \leftarrow g_6 = X^3 + 10Y^2 + 5XY + 7Y + 11X + 4$.

3. We find the smallest polynomial $g(\neq 0)$ such that $g \cdot I \subset (f, F)$.

Computing the remainder of the product of $g_5$ and monomials divided by $\{f, F\}$ in the ascending order, we get

\[ \overline{g_5^{(f,F)}} = X^2Y + \ldots \]
\[ \overline{Xg_5^{(f,F)}} = XY^2 + \ldots \]
\[ \overline{Yg_5^{(f,F)}} = X^2Y^2 + \ldots \]

Moreover, computing $\overline{X^2g_5^{(F,F)}}$, we get the result $4X^2Y^2 + \ldots$, which leading monomial $X^2Y^2$ is equal to the one of $\overline{Yg_5^{(f,F)}}$. So,

\[ X^2g_5 \equiv 4Yg_5 + 12XY^2 + \ldots \pmod{\{f, F\}}. \]
Noting $XY^2$ is a leading monomial of $\overline{Xg_5(f,F)}$, and repeating the similar computation, we get
\[ X^2g_5 \equiv 4Yg_5 + 12Xg_5 + 2g_5 \pmod{\{f,F\}}. \]

So, we have
\[ g \leftarrow X^2 + 13Y + 5X + 15. \]

(4)

\[ (g/f) \cdot J = (g/f) \cdot \{g_6, g_5, g_4\} \]
\[ = \{g, (gg_5)/f, (gg_4)/f\}. \]

Getting the quotient $\{a_5, b_5\}$ and $\{a_4, b_4\}$ by dividing $gg_5$ and $gg_4$ by $\{f,F\}$, respectively, we have
\[ I_3 \leftarrow \{g, (gg_5)/f, (gg_4)/f\} \]
\[ \equiv \{g, a_5, a_4\} \pmod{\{F\}} \]
\[ = \{X^2 + 13Y + 5X + 15, XY + 13Y + 5X + 11, \]
\[ Y^2 + 5Y + 12X + 6\}. \]

The right-hand side is the Groebner basis of $I_3$, the result of the double of $I_1$.

Remark. In the above, a polynomial $g$ is computed such that $gg_5$ is divisible by $f$. Then we get such an $g$ with $\text{LM}(g) = X^2$. In this case, $gg_4$ is automatically divisible by $f$. The reason is as follows. Note that $\delta(J) = \delta(f : I) = -v_\infty(X^3) - \delta(I) = 3$. So the smallest monomial in $\text{LM}(J)$ (w.r.t. $C_{34}$ order) is not greater than $X^2$ since $X^2$ is the fourth smallest monomial (see Eq. (9)). Then, if $gg_4$ is not divisible by $f$, $\text{LM}(g)$ becomes larger than $X^2$, and this is impossible since $g$ must be the smallest member in $J$.

Now we get Algorithm 6 giving the details of Algorithm 5. For simplicity, Algorithm 6 treats only plane $C_{ab}$ curve. In Algorithm 6, “$\{c_1, c_2, \ldots, c_a\}, r \leftarrow \text{Division}(g, G)$” denotes that we get the quotient $\{c_1, c_2, \ldots, c_a\}$ and the remainder $r$ by dividing the polynomial $g$ by the polynomial set $G$ (see [3, Section 3, Chapter 2] for details). “$\{a_1, \ldots, a_i\}, r \leftarrow \text{Coefficients}(f, r_1, \ldots, r_i)$” denotes that we get coefficients $\{a_1, \ldots, a_i\}$ and the remainder $r$ to express $f$ as a linear combination of $r_1, \ldots, r_i$. Mono$_i$ denotes the $i$th monomial in $C_{ab}$ order (Mono$_1 = 1, \text{Mono}_2 = X, \ldots$).

Algorithm 6

algorithm JacobianSum(inputs $I_1, I_2$, output $I_3$)

\[ I_3 \leftarrow \text{Compose}(I_1,I_2) \]
\[ f \leftarrow \text{the smallest element of } I_3 \]
\[ I_3 \leftarrow \text{Reduce}(f, I_3) \]
\[ \text{RETURN } I_3 \]
subroutine Compose(inputs $I_1 = \{f_1, f_2, \ldots, f_a\}$, $I_2 = \{g_1, g_2, \ldots, g_a\}$, output $I_3$)

$I_3 \leftarrow \{F\}$

FOR $i = 1$ TO $a$, $j = 1$ TO $a$ DO

$g \leftarrow f_i \cdot g_j$

$I_3 \leftarrow \{g\} \cup I_3$

IF $\delta(I_3) > \delta(I_1) + \delta(I_2)$ THEN $I_3 \leftarrow$ Buchberger($\delta(I_1) + \delta(I_2), I_3$)

$I_3 \leftarrow$ the set of the smallest $a$ elements of $I_3$

RETURN $I_3$

subroutine Reduce(inputs $f, I = \{f_1, f_2, \ldots, f_a\}$, output $J$)

$G \leftarrow \{f, f \cdot Y^{(F)}, \ldots, f \cdot Y^{a-1}(F), F\}$

LABEL (retry)

$J \leftarrow \{\}$

$h \leftarrow \sum_{i=1}^{a}$ (random number) $\cdot f_i$

$g \leftarrow$ Divide($G, h$)

FOR $i = 1$ TO $a$

$\{c_1, c_2, \ldots, c_a, r\} \leftarrow$ Division($g \cdot f_i, G$)

IF $r \neq 0$ THEN GOTO retry

$k \leftarrow c_1 + c_2 \cdot Y + \cdots + c_a \cdot Y^{a-1}$

$J \leftarrow J \cup \{k\}$

RETURN $J$

subroutine Divide(inputs $G, h$, output $s$)

$r_1 \leftarrow$ Mono $\cdot \hat{h}^G$

$s_1 \leftarrow$ Mono $\cdot \hat{h}^G$

$i \leftarrow 1$

WHILE $r_i \neq 0$ DO

$i \leftarrow i + 1$

$r_i \leftarrow$ Mono $\cdot \hat{h}^G$

$\{A_1, \ldots, A_{i-1}, r_i\} \leftarrow$ Coefficients($r_i, \{r_1, \ldots, r_{i-1}\}$)

$s_i \leftarrow$ Mono $- \sum_{j=1}^{i-1} A_j s_j$

RETURN $s_i$

subroutine Buchberger(inputs $m, I = \{f_1, \ldots, f_s\}$, output $G = \{g_1, \ldots, g_t\}$)

$B \leftarrow \{(i, j) | 1 \leq i < j \leq s\}$

$G \leftarrow F$

$t \leftarrow s$

WHILE $B \neq \emptyset$ AND $\delta(G) > m$ DO

Select $(i, j) \in B$

IF $\text{LCM}($LT($f_i$), LT($f_j$)) $\neq \text{LT}(f_i) \cdot \text{LT}(f_j)$ THEN

$S \leftarrow S(f_i, f_j)^G$

RETURN $S$
\[
\begin{align*}
\text{IF } S \neq 0 \text{ THEN} \\
\quad t &\leftarrow t + 1; f_t \leftarrow S \\
\quad G &\leftarrow G \cup \{f_t\} \\
\quad B &\leftarrow B \cup \{(i, t) \mid 1 \leq i \leq t - 1\} \\
B &\leftarrow B - \{(i, j)\}
\end{align*}
\]

\text{RETURN reduced } G

The subroutine Compose corresponds to the first step of Algorithm 5. It computes the Groebner basis of the ideal product \( I_3 = I_1 \cdot I_2 \) for ideals \( I_1 \) and \( I_2 \) of the coordinate ring \( K[X, Y]/(F(X, Y)) \). In the subroutine, we compute the order of \( \Delta \)-set \( \delta(I) \) of an ideal \( I \) (for a definition of \( \Delta \)-set, see Section 2.2). Only in the case that \( \delta(I_1) + \delta(I_2) \neq \delta(I_3) \), we need the subroutine Buchberger to obtain Groebner basis of \( I_3 \).

In general, the complexity of Buchberger algorithm may be quite huge. However, in our situation, the monomial order \( C_{ab} \) order has the order function \( \Psi_{a,b} \), and forms of ideals involved are quite simple. So, Buchberger algorithm works quite efficiently. Let \( g \) be the genus of \( C_{ab} \) curve defined over a finite field of \( q \) elements. First, \( I_3 \) is composed of polynomials whose leading monomials are the about 3\( g \)th monomials among all the monomials. To obtain Groebner basis of \( I_3 \), we need to find polynomials in \( I_3 \), whose leading monomials are the about 2\( g \)th monomials. In WHILE loop of Buchberger algorithm, using notations in subroutine Buchberger, if we can make an ideal schedule for the choice of polynomials \( f_i, f_j \) for the computation \( S(f_i, f_j) \), \( S = S(f_i, f_j)^G \) is not trivial and its leading monomial is strictly smaller than those of members in \( G \). This means that WHILE loop should finish in \( O(g) \) repeats. So, the complexity of Buchberger algorithm in our situation with the ideal schedule is \( O(q^2 g^3) \).

However, our experimental results show, when the size of the definition field is large enough and the genus is small enough, the complexity of Buchberger algorithm is not far from \( O(q^2 g^3) \) with a hand-made schedule.

The subroutine Divide corresponds to the third step of Algorithm 5. For a random linear combination \( h \) of \( f_i \), it compute the smallest polynomial \( s \) with respect to \( C_{ab} \) order, such that \( hs \in \langle G \rangle \) just as in the above example. The subroutine Divide is essentially same as the Gaussian elimination among \( g \) variables. So, the complexity of Divide is \( O(q^2 g^3) \).

The subroutine Reduce corresponds to the fourth step of Algorithm 5. The reason why we choose a random linear combination of \( f_i \) for \( h \) is just a heuristic. As seen in the above example, it seems that the fact
\[
g \cdot (\text{a random linear combination of } f_i) \in (f)
\]
is sufficient for \( gI \subset (f) \) when the size of the definition field is large enough and the genus is small enough. Our experimental results support the heuristic.

Note that the Groebner basis of the principal ideal \( (f) \) in the coordinate ring \( K[X, Y]/(F) \) is given by
\[
\{f, f \cdot Y^{(F)}, \ldots, f \cdot Y^{a-1[F], F}\},
\]
since the leading term \( Y^a \) of \( F(X, Y) \) is prime to \( X \).
Finally, we show timing results of our implementation of Algorithm 6 by C language. Tables 1–3 show running times on 266MHz Pentium II, for $C_{35}$ curve, $C_{37}$ curve and $C_{2,13}$ curve, respectively. In each case, $C_{ab}$ curve has the real size parameter in cryptographical applications, that is, $C_{ab}$ curves have 160 bits Jacobian groups. In tables, ‘simple’ denotes $C_{ab}$ curves with defining equations of the form $Y^a + aX^b + \beta$, and ‘random’ denotes randomly chosen $C_{ab}$ curves. ‘Sum’, ‘Double’ and ‘Scalar’ denotes addition of two random elements, doubling a random element and multiplication of a random element by a 160 bits random integer, respectively. These results prove that the algorithm behaves well in practice.

Acknowledgements

I wish to thank S. Miura and R. Matsumoto for their helpful and suggesting discussions with me. I also wish to thank anonymous referees for their a lot of efforts to help me arrive at the final version of the paper.

Appendix A. Sample code

Here is a sample code of Algorithm 4 with Mathematica Ver. 3.
(* Addition Algorithm in Jacobian of C_{a,b} curve *)

(* parameter *)

Var = \{X,Y,Z;\}
p = 83;
OrderMatrix = \{(3,5,7),\{-1,0,0\},\{0,-1,0\}\}; (* C_{357} *)
OrderMatrix1 = \{(1,0,0),\{0,3,5,7\},\{0,-1,0,0\},\{0,0,-1,0\}\};
(* C_{357} *)
n = 650496;

DefIdeal = \{64 + 4 X + 30 X^2 + 30 X^3 + 76 Y + 75 X Y + Y^2 + 52 Z + 4 X Z,
10 + 27 X + 16 X^2 + 6 X^3 + 44 X^4 + 69 Y + 16 X Y + 27 X^2 Y
+ 31 Z + X Z + Y Z,
22 + 32 X + 77 X^2 + 30 X^3 + 11 X^4 + 17 Y + 25 X Y + 3 X^2 Y
+ 72 X^3 Y + 45 Z + 32 X Z + 76 X^2 Z + Z^2 \};

G = \{X-2, Y-33, Z+21\}; (* a point on the curve *)

(* polynomial library *)

LT[f_] :=
MonomialList[f, Var, Modulus -> p, MonomialOrder ->
OrderMatrix][[1]]

LC[f_] := LT[f] /. ((#- >1)&/@ Var)

LM[f_] := LT[f] / LC[f]

GBasis[J_List] :=
GroebnerBasis[J, Var, Modulus -> p, MonomialOrder -> OrderMatrix]

GDivision[f_, J_List] :=
PolynomialReduce[f, J, Var, Modulus -> p, MonomialOrder ->
OrderMatrix]

GRemainder[f_, J_List] := GDivision[f, J][[2]]; 

IdealIntersection[I_List, J_List] :=
Module[{t, G},
  G = GroebnerBasis[Union[t* I, (1- t)*J], Prepend[Var,t],
    Modulus -> p, MonomialOrder -> OrderMatrix1];
  G = Select[G, (# == (# /.(t- >0))& ]
]
IdealQuotient[I_List, f_] :=
Module[{G},
  G = IdealIntersection[I, {f}];
  G = GDivision[#, {f}] & /@ G;
  G = First /@ G;
  Union @@ G]

IdealQuotient[I_List, J_List] :=
Module[{R},
  R = IdealQuotient[I, First[J]];
  Do[
    R = IdealIntersection[R, IdealQuotient[I, J[[i]]]],
    {i, 2, Length[J]}
  ];
  R]

(* main *)

JPower[n_Integer, I1_List] :=
Module[{i = n, J1 = I1, R = {1}},
  While[i > 0,
    If[OddQ[i], R = JSum[R, J1]; i = (i - 1)/2, i = i/2];
    If[i > 0, J1 = JSum[J1, J1]]
  ];
  R]

JSum[I1_List, I2_List] :=
Module[{I3, f},
  I3 = JCompose[I1, I2];
  f = First[I3];
  I3 = JReduce[f, I3];
  f = First[I3];
  JReduce[f, I3]]

IdealProduct[I1_List, I1_List] :=
Module[{f, I3 = DefIdeal},
  Do[
    f = I1[[i]] I1[[j]] // PolynomialMod[#, p] &;
    f = GRemainder[f, I3];
  ]
If \( f \neq 0 \), \( I_3 = \text{Prepend}[I_3, f] \),
\[
\{ i, 1, \text{Length}[I_1] \}, \{ j, 1, i \}
\];
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
\]
References