PAPER

Gaudry's variant against C_{ab} curves

Seigo ARITA[†], *Member*

SUMMARY Gaudry has described a new algorithm (Gaudry's variant) for the discrete logarithm problem (DLP) in hyperelliptic curves. For a hyperelliptic curve of a small genus on a finite field GF(q), Gaudry's variant solves for the DLP in time $O(q^{2+\epsilon})$. This paper shows that C_{ab} curves can be attacked with a modified form of Gaudry's variant and presents the timing results of such attack. However, Gaudry's variant cannot be effective in all of the C_{ab} curve cryptosystems. This paper also provides an example of a C_{ab} curve that is unassailable by Gaudry's variant.

key words: discrete logarithm, hyperelliptic curve, superelliptic curve, C_{ab} curve

1. Introduction

Gaudry[7] has described a new algorithm (Gaudry's variant) for the discrete logarithm problem (DLP) in hyperelliptic curves. Gaudry's variant uses the method for Pollard's rho algorithm [12] with the function field sieving algorithm of Adleman,DeMarrais, and Huang [1]. Gaudry's variant solves the DLP in a hyperelliptic curve of a genus g defined on a finite field F_q in time $O(q^{2+\epsilon})$ when the genus g is sufficiently small in comparison to the order q of the definition field.

Arita[3] and Galbraith et al.[6] described addition algorithms on the Jacobian group of C_{ab} and superelliptic curves respectively, and demonstrated algorithm applications in discrete-log-based public key cryptosystems. This paper shows that C_{ab} and superelliptic curves can be attacked by a modified Gaudry's variant and presents timing results from the attack.

With hyperelliptic or C_{ab} curve cryptosystems, one usually selects a curve of a sufficiently large genus so that definition fields are less than one word in size to hasten computations [3], [14]. Gaudry's variant has excluded out this conventional hastening method. However, Gaudry's variant cannot be effective in all of the non-elliptic algebraic curve cryptosystems. This paper also provides an example of a C_{ab} curve that is unassailable by Gaudry's variant.

2. Gaudry's variant

Take a hyperelliptic curve $C : y^2 = x^{2g+1} + a_1x^{2g} + \cdots a_{2g+1}$ of genus g defined on a finite filed F_q. Suppose its genus g is sufficiently small in comparison to the order q of the definition field. Let J_C denote the Jacobian group of the hyperelliptic curve C. To handle with the DLP, we look for an integer λ that satisfies $D_2 = \lambda D_1$ for two elements D_1 and D_2 in J_C.

In Pollard's rho algorithm, we calculate random linear sums $R_i = \alpha_i D_1 + \beta_i D_2 \ \ (i = 1, 2, \cdots)$ of D_1 and D_2 step by step through a random walk, and wait for a collision $R_i = R_j$. Once it occurs, from $\alpha_i D_1 + \beta_i D_2 = \alpha_j D_1 + \beta_j D_2$, we get $\lambda = (\alpha_i - \alpha_j)/(\beta_j - \beta_i)$.

In Gaudry's variant[7], just as in the case of rho algorithm, we calculate random linear sums $R_i = \alpha_i D_1 + \beta_i D_2 \ \ (i = 1, 2, \cdots)$ of D_1 and D_2. However, we gather smooth R_i value's instead of waiting for a collision. An element D in J_C is called smooth when D is a sum of F_q rational points on C. Let all of the F_q rational points on C be $\{P_1, P_2, ..., P_w\}$,where the number w is approximately equal to q. Then, every smooth R_i corresponds to a w-dimensional vector $M_i = (m_{i,1}, \cdots, m_{i,w})$ through the relation $R_i = \sum_k m_{i,k} P_k$. Using the polynomial expression $R_i = [u_i(x), v_i(x)]$ from Cantor's algorithm [4], R_i is smooth if and only if the polynomial $u_i(x)$ is factored into the linear product $\prod_k (x - c_k)$ over F_q, and then we get $R_i = \sum_k (c_k, v(c_k))$. Therefore, by calculating all of the F_q rational points $\{P_1, P_2, \cdots, P_w\}$ in advance at a complexity of $O(q)$, the vector M_i is easily obtained.

When g is sufficiently small in comparison to q, about $1/g!$ of all of the elements in J_C are smooth ([7]Prop.4). So, we obtain w' ($\geqq w$) smooth R_i values after $g! \cdot w'$ steps. w' vectors $M_i = (m_{i,1}, \cdots, m_{i,w})$ $(i = 1, \cdots, w')$ must be linearly dependent. By solving the w-dimensional linear equation, we obtain values for γ_i $(i = 1, \cdots, w')$ satisfying

$$\sum_{i=1}^{w'} \gamma_i M_i = 0.$$

These values for γ_i produce λ through the equation $R_i = \sum_k m_{i,k} P_k$;

$$\lambda = -\sum_i \gamma_i \alpha_i / \sum_i \gamma_i \beta_i.$$

In Gaudry's variant, the most complex step is solving the w-dimensional linear equation $\sum_i \gamma_i M_i = 0$ $(i = 1, 2, \cdots, w')$. The matrix $(m_{i,k})$ has a size equal to

Manuscript received
Manuscript revised
[†]The author is with NEC Co.,Kawasaki-shi,216 Japan.

about $q \times q$ and is sparse since there are only g nonzero elements in each row. So the linear equation can be solved in q^2 steps. Hence, the complexity of Gaudry's variant is $O(q^2 \log^\gamma(q))$ for some constant γ, so it is $O(q^{2+\epsilon})$.

When C has a non-trivial automorphism ϕ, Gaudry's variant becomes more powerful. Let m denote the order of ϕ. In this case, all of the F_q rational points are unnecessary. Only the representatives of orbits in F_q rational points for the action of ϕ are needed. The number of the orbits is q/m, so the complexity of Gaudry's variant becomes $O((q/m)^{2+\epsilon})$.

However,

Theorem 1: [2] The order of the automorphism group of a smooth algebraic curve of genus $\geqq 2$ is at most $84(g-1)$.

So, the effect of automorphisms can be ignored by expanding the size of the definition field with $\log_2(84(g-1))$ more bits.

3. C_{ab} and superelliptic curves

A C_{ab} curve is a nonsingular affine curve with the equation

$$\sum_{0 \leqq i \leqq b, 0 \leqq j \leqq a, ai+bj \leqq ab} \alpha_{i,j} x^i y^j = 0,$$

where both $\alpha_{b,0}$ and $\alpha_{0,a}$ are not equal to zero [10]. Arita[3] described an addition algorithm for the Jacobian group of a C_{ab} curve in terms of ideals of the coordinate ring, aiming at cryptosystems using C_{ab} curves. For a C_{ab} curve with 160 bits of a Jacobian group, timing results from the addition algorithm are listed in Tables 1,2, and 3.

Table 1 Performance for the C_{35} curve in ms at 266 MHz, using a Pentium II chip.

	simple	random
sum	3.39	3.65
double	3.76	4.21
scalar	862	958

Table 2 Performance for the C_{37} curve in ms at 266 MHz, using a Pentium II chip.

	simple	random
sum	1.15	1.24
double	1.15	1.28
scalar	273	300

Table 3 Performance for the $C_{2,13}$ curve in ms at 266 MHz, using a Pentium II chip.

	simple	random
sum	0.70	0.73
double	0.65	0.68
scalar	158	167

In the tables, "simple" denotes a C_{ab} curve with a equation of the form $Y^a + \alpha X^b + \beta$, and "random" denotes a randomly chosen C_{ab} curve. "Sum", "double", and "scalar" denote addition, doubling, and scalar multiplication of random elements, respectively.

A superelliptic curve is a nonsingular affine curve of the equation

$$y^n = a_\delta x^\delta + \cdots + a_0,$$

where n is prime to the characteristics of the definition field, and n and δ are prime to each other [6]. Galbraith et al.[6] described an addition algorithm in the Jacobian group of a superelliptic curve in terms of lattice computations.

Given these definitions, C_{ab} curves clearly include superelliptic curves. Only C_{ab} curves will be examined.

4. Application of Gaudry's variant to C_{ab} curves

We modify Gaudry's variant to apply it to C_{ab} curves. The problems we encounter are how to decide if a given element in a Jacobian group is smooth or not, and, if it is, how to represent it as a sum of F_q rational points. Note that about $1/g!$ of all of the elements in J_C are smooth when g is sufficiently small just as in the case of hyperelliptic curves (See the proof of [7]Prop.4).

For example, take a C_{37} curve. The genus is six. An element R in the Jacobian group of C_{37} curve is expressed as an ideal of the coordinate ring. The ideal has the following form of Gröbner basis with respect to the C_{37} order [3]:

$$\begin{aligned} R = \{&a_0 + a_1 x + a_2 x^2 + a_3 y + a_4 x^3 + a_5 xy + x^4, \\ &b_0 + b_1 x + b_2 x^2 + b_3 y + b_4 x^3 + b_5 xy + x^2 y, \\ &c_0 + c_1 x + c_2 x^2 + c_3 y + c_4 x^3 + c_5 xy + x^2 y\}. \end{aligned}$$

Here, a_i, b_i, and c_i are elements in the definition field. The common zeroes of these three equations are six points on the C_{37} curve, that comprise R.

When the definition field F_q is large enough, the six points comprising R have distinct x-coordinates to each other for almost any R in the Jacobian group. So, almost any R can be expressed as common zeroes of two polynomials:

$$\begin{aligned} R = \{&\text{sixth degree polynomial of } x, \\ &y + (\text{fifth degree polynomial of } x)\}, \end{aligned}$$

just as in the case of hyperelliptic curves. The expression is nothing but the Gröbner basis of (the ideal corresponding to) R with respect to the lexicographic order.

Therefore, in Gaudry's variant against C_{ab} curves, for almost any R_i, the decision regarding the R_i value's smoothness and representation as a sum of F_q rational points follows the same pattern as for hyperelliptic curves, after translating the Gröbner basis of R_i w.r.t.

C_{ab} order to another Gröbner basis w.r.t. lexicographic order. It is well known that Gröbner bases can be effectively translated between distinct monomial orders. Note that if exceptional values for R_i are found, we can simply discard them. Thus, now we know that Gaudry's variant can also be applied to C_{ab} curves.

finite field	F_{84211}
defining equation	$1 + 24740x^7 + 32427y^3 = 0$
genus	6
order of Jacobian	$43 \cdot 80689706230162396053189866 17$
order of automorphism	$3 \cdot 7$

Table 4 93 bits of a C_{37} curve over 17 bits of a prime field

We implemented the modified Gaudry's variant with C language and the PARI-GP [11], and then tested it against the C_{37} curve in Table 4. The curve has 93 bits of a Jacobian group over 17 bits of a prime field. Let ζ_3 and ζ_7 denote the primitive third and the seventh root of unity, respectively. Since the curve has the automorphism $\phi(x, y) = (\zeta_7 x, \zeta_3 y)$ of order 21, the number of the orbits of ϕ on the rational points should be $84211/21 = 4010.\cdots$ or more. We needed to collect 4011 or more smooth elements and then solve about 4011 dimensional sparse linear equations. For solving these linear equations, we also used the Lanczos algorithm [9].

Two elements D_1 and D_2 in the Jacobian group were randomly generated:

$$\begin{aligned} D_1 = \{&x^4 + 77465x^3 + 75875x^2 + (37117y + 57992)x \\ &+ (42876y + 21588), \\ &5485x^3 + (y + 79222)x^2 + (4298y + 50456)x \\ &+ (36882y + 81869), \\ &41971x^3 + 64608x^2 + (26263y + 16207)x \\ &+ (y^2 + 42778y + 62216)\}, \\ D_2 = \{&x^4 + 64296x^3 + 44620x^2 + (29434y + 15779)x \\ &+ (61013y + 42557), \\ &51156x^3 + (y + 32172)x^2 + (62401y + 22153)x \\ &+ (78055y + 13056), \\ &79116x^3 + 5028x^2 + (69977y + 21979)x \\ &+ (y^2 + 75761y + 2009)\}. \end{aligned}$$

Then, running the modified Gaudry's variant, we obtained

$$\lambda = 40822718041348743469836704 15$$

for $D_2 = \lambda D_1$ in the time given in Table 5. The average and variance of the number of steps needed to produce a smooth element were 848.265 and 680786, respectively, reasonable results compared to Gaudry's theoretical estimate $1/6! = 1/720$ of the probability for an element to be smooth[7].

Collection of rational points (PARI-GP)	5 m 13 s
Collection of smooth elements (C)	2 h 33 m 6 s
Solving linear equation (C)	32 m 2 s
Total	3 h 11 m 21 s

Table 5 Timing results of modified Gaudry's variant against the C_{37} curve from Table 4 using a 266-MHz Pentium II chip.

5. C_{ab} curves that are unassailable by Gaudry's variant

Let a and b be distinct prime numbers. Let $C(p, \alpha, \beta)(= C(p, a, b, \alpha, \beta))$ denote a C_{ab} curve over a prime field F_p with the equation

$$\alpha Y^a + \beta X^b + 1 = 0.$$

In this section, we will construct a C_{ab} curve $C(p, \alpha, \beta)$ of a small genus, that is secure against Gaudry's variant. Let h be the order of the Jacobian group of $C(p, \alpha, \beta)$. Conditions for security are:

Condition 1 The order h has 160 or more bits of prime factor l [12],

Condition 2 l does not divide $p^k - 1$ for small values of k [5],

Condition 3 l is not equal to p [13], and

Condition 4 p has $(40 + \log_2(84(g-1)))$ or more bits.

The fourth condition secures a curve against Gaudry's variant (Theorem 1). Because we are handling a curve with a small genus, we do not need to consider Adleman,DeMarrais,and Huang's algorithm [1] or its extension to superelliptic curves [6].

Koblitz[8] used Jacobi sums to calculate the order of Jacobian groups of hyperelliptic curves. Jacobi sums can also be used for a curve $C(p, \alpha, \beta)$. For simplicity, $p \equiv 1 \pmod{\mathrm{lcm}(a, b)}$ has been assumed.

Fix the generator w of the multiplicative group F_p^*. For a rational number s such that $(p-1)s$ is an integer, character χ_s of F_p^* is defined by

$$\chi_s(w) = e^{2\pi i s}.$$

Extend χ_s to the whole F_p by setting $\chi_s(0) = 0$ when s is not an integer and setting $\chi_s(0) = 1$ when it is.

For integers $l = 1, 2, \ldots, a-1$ and $m = 1, 2, \ldots, b-1$,

$$j_p(l, m) = \sum_{1+v_1+v_2=0} \chi_{l/a}(v_1)\chi_{m/b}(v_2) \qquad (1)$$

is called a Jacobi sum. Here, v_1 and v_2 run over F_p under the condition $1 + v_1 + v_2 = 0$.

Weil[16] has demonstrated that the L-polynomial $L_p(U)$ of $C(p, \alpha, \beta)$ can be expressed by Jacobi sums:

$$L_p(U) = \prod_{\substack{l=1,2,\ldots,a-1 \\ m=1,2,\ldots,b-1}} (1 + \overline{\chi_{l/a}}(\alpha)\overline{\chi_{m/b}}(\beta) j_p(l,m) U),$$

where $\overline{\chi_s}$ denotes the complex conjugate of χ_s.

Generally, the order of a Jacobian group of a curve is equal to the value of the L-polynomial $L(U)$ of the curve at $U = 1$ [15], so the order h of the Jacobian group of $C(p, \alpha, \beta)$ is given by

$$\begin{aligned} h &= L_p(1) \qquad (2) \\ &= \prod_{\substack{l=1,2,\ldots,a-1 \\ m=1,2,\ldots,b-1}} (1 + \overline{\chi_{l/a}}(\alpha)\overline{\chi_{m/b}}(\beta) j_p(l,m)) \\ &= \mathrm{Norm}_{\boldsymbol{Q}(\zeta)|\boldsymbol{Q}}(1 + \overline{\chi_{1/a}}(\alpha)\overline{\chi_{1/b}}(\beta) j_p(1,1)) \end{aligned}$$

Thus, to know the order h of the Jacobian group, it is sufficient to know the values of the Jacobi sums $j_p(l,m)$. However, direct computation of those values using the formula (1) are infeasible from the computational point of view. We use the Stickelberger element to know the values of the Jacobi sums $j_p(l,m)$.

Let $[\lambda]$ denote the largest integer under the rational number λ, and $< \lambda >$ denote $\lambda - [\lambda]$. Take a cyclotomic field $\boldsymbol{Q}(\zeta)$ with a primitive ab-th root ζ of unity. Let σ_t denote the Galois map $\zeta \mapsto \zeta^t$ of $\boldsymbol{Q}(\zeta)$. An element $\omega(a,b)$ in the group ring $\boldsymbol{Z}[Gal(\boldsymbol{Q}(\zeta)|\boldsymbol{Q})]$ defined by

$$\omega(a,b) = \sum_t [< \frac{t}{a} > + < \frac{t}{b} >] \sigma_{-t}^{-1}, \qquad (3)$$

where t runs over reduced residue classes mod ab, is called a Stickelberger element. As an ideal of $\boldsymbol{Q}(\zeta)$, it is known that

$$(j_p(l,m)) = P^{\omega(a,b)}, \qquad (4)$$

where P denotes a prime ideal of $\boldsymbol{Q}(\zeta)$ lying over p [17]. Suppose that the prime ideal P is principal:

$$P = (\gamma).$$

Then, we have

$$(j_p(l,m)) = (\gamma^{\omega(a,b)}). \qquad (5)$$

From this, we know that $j_p(l,m)$ and $\gamma^{\omega(a,b)}$ differ by the product of some unit of $\boldsymbol{Q}(\zeta)$. Moreover, $j_p(l,m)$ and $\gamma^{\omega(a,b)}$ has the same absolute value $p^{1/2}$ in any embedding into the complex number field [17]. So, $j_p(l,m)$ and $\gamma^{\omega(a,b)}$ differ only by the power of $-\zeta$. Now we see that Equation (5) can be used to determine $j_p(l,m)$ up to the power of $-\zeta$ by knowing the prime p and the principal prime ideal $P = (\gamma)$ in advance.

We arrive at the following algorithms:

Algorithm 1 (Search for a secure $C(p, \alpha, \beta)$)**:**
Input: a, b
Output: p, α, β, and the order h of the Jacobian group

1. $g \leftarrow (a-1)(b-1)/2$
2. $m \leftarrow Max(\lceil 160/g \rceil, \lceil 40 + log_2(84(g-1)) \rceil)$
   $\lceil n \rceil$ denotes the least integer over n.
3. Search the candidate j for the value of a Jacobi sum for some prime p of m or more bits by using Algorithm 2:

   $(p, j) \leftarrow$ Algorithm2(m).
4. For every $k = 0, 1, \ldots, ab-1$,

   $h_k \leftarrow \mathrm{Norm}_{\boldsymbol{Q}(\zeta)|\boldsymbol{Q}}(1 + (-\zeta)^k j)$.
5. Check that there is a value h_k that satisfies Conditions 1,2, and 3 for security in the set $\{h_0, h_1, \ldots, h_{ab-1}\}$. If there is not, go to step 3. If it does, $h \leftarrow h_k$.
6. Let ζ_a and ζ_b denote the primitive a-th and b-th root of unity modulo p, respectively. For every h_k obtained at step 5, do as follows. For every $l = 0, 1, \cdots, a-1$ and $m = 0, 1, \cdots, b-1$, check if the order of the Jacobian group of $C(p, \zeta_a^l, \zeta_b^m)$ is equal to h or not; for example, check if h times the random element is equal to the unit element, or not. If the order is equal, output $p, \alpha = {\zeta_a}^l, \beta = {\zeta_b}^m$, and h. If there is no such l and m, go to step 3.

As per step 2, p has $40 + \log_2(84(g-1))$ or more bits, so the curve obtained by Algorithm 1 is secure against Gaudry's variant. At step 6, we check only the necessary condition for the group order to be h. So strictly the order may be different from h. But, the test at step 6 ensures that the order has the same large prime factor l as the one of h. This is sufficient for our purpose.

Algorithm 2, contained in Algorithm 1, makes use of Equation (3) and (5) for the Stickelberger element to find the candidate value of the Jacobi sum.

Algorithm 2 (Finding the candidate of the Jacobi sum)**:**
Input: m
Output: p and j

1. $\omega \leftarrow \sum_t [< \frac{t}{a} > + < \frac{t}{b} >] \sigma_{-t}^{-1}$
2. Randomly generate
   $\gamma_0 = \sum_{l=0}^{(a-1)(b-1)-1} c_l \zeta^l$ $(-20 \leqq c_l \leqq 20)$.
3. For every $i = 1, 2, \ldots,$

   $\gamma \leftarrow \gamma_0 + i$
   $p \leftarrow \mathrm{Norm}_{\boldsymbol{Q}(\zeta)|\boldsymbol{Q}}(\gamma)$
   If $p < 2^m$, then try the next i.
   If $p >> 2^m$, then go to step 2.
   '$>>$' means 'sufficiently larger than.'
   If p is not prime, then try the next i.
4. $j \leftarrow \gamma^\omega$
   Output p and j.

Note that p obtained by Algorithm 2 satisfies the condition $p \equiv 1 (\mathrm{mod}\, \mathrm{lcm}(a,b))$, because the prime ideal (γ) has the residue degree one over the prime p.

Experimentally, we observed that numbers of repeating steps 2 and 3 depends heavily on the range of c_l at step 2. We take the range $-20 \leqq c_l \leqq 20$, so that Algorithm 2 stops after repeating steps 2 and 3 for several times, for the input m of the size used in the real world.

Example

Algorithm 1 was run for $a = 3$ and $b = 5$.

1. $g \leftarrow (3-1)(5-1)/2 = 4$
2. $m \leftarrow \max(160/4, \lceil 40 + \log_2(84 \cdot 3) \rceil) = 48$
3. Run Algorithm 2 for $m = 48$.
   a. $\omega \leftarrow \sum_t [< \frac{t}{3} > + < \frac{t}{5} >] \sigma_{-t}^{-1} = \sigma_1 + \sigma_7 + \sigma_{11} + \sigma_{13}$
   b. $\gamma_0 \leftarrow 20 - 3\zeta - 12\zeta^2 + 20\zeta^3 - 11\zeta^4 - 4\zeta^5 + 3\zeta^6 - 16\zeta^7$ was randomly generated.
   For $\gamma \leftarrow \gamma_0 + 60 = 80 - 3\zeta - 12\zeta^2 + 20\zeta^3 - 11\zeta^4 - 4\zeta^5 + 3\zeta^6 - 16\zeta^7$,
   $p \leftarrow \mathrm{Norm}(\gamma) = 581929936583251$ is a prime of 50 bits. Now, the candidate value j of the Jacobi sum for p is
   $$\begin{aligned} j &\leftarrow \gamma\gamma^{(7)}\gamma^{(11)}\gamma^{(13)} \\ &= 18682331 + 1900434\zeta + 3903200\zeta^2 \\ &\quad + 735220\zeta^3 + 2683534\zeta^4 - 6028054\zeta^5 \\ &\quad - 1372490\zeta^6 + 3103044\zeta^7 \end{aligned}$$
4. For every $k = 0, 1, \ldots, 29$, compute $h_k \leftarrow \mathrm{Norm}(1 + (-\zeta)^k \mathrm{j})$:
   $$\begin{aligned} h_0 &\leftarrow 1146786729413035548075542797106 \\ &\quad 7128382725740410373604788 2496 \\ &\ldots \\ h_6 &\leftarrow 1146786991386963085872739586632 \\ &\quad 6581132398842272782691512 2881 \\ &\ldots \\ h_{29} &\leftarrow 1146787464216129098443264922470 \\ &\quad 0754765063809498595535465 2416 \end{aligned}$$
5. h_6 satisfies Condition 1,2 and 3. In fact,
   $$h \leftarrow h_6 = 2511 \cdot l$$
   Here,
   $$\begin{aligned} l = {} & 4567052932644217785235 92 \\ & 0297222851904579387603656985 8671 \end{aligned}$$
   is a prime of 185 bits, distinct from p. Condition 2 is satisfied for $k \leqq 1000$.
6. For $\alpha = 579364850535396$ and $\beta = 289406374935593$, it is verified that the order of the Jacobian group of $C(p, \alpha, \beta)$ is equal to h.

Thus, a C_{35} curve is obtained with

$$579364850535396y^3 + 289406374935593x^5 + 1 = 0$$

over the prime field GF(581929936583251) with the Jacobian group of the order

$$\begin{aligned} & 2511 \cdot 4567052932644217785235 92 \\ & \quad 0297222851904579387603656985 8671, \end{aligned}$$

which is secure against Gaudry's variant.

6. Conclusion

We demonstrated that C_{ab} and superelliptic curves can be attacked by a modified Gaudry's variant, and reported timing results for the attack.

However, Gaudry's variant cannot be effective in all of the non-elliptic algebraic curve cryptosystems. We provided an example of a C_{35} curve that is unassailable by Gaudry's variant.

References

[1] L.M.Adleman,J.DeMarrais, and M.D.Huang, "A Subexponential Algorithm for Discrete Logarithms over the Rational Subgroup of the Jacobians of Large Genus Hyperelliptic Curves over Finite Fields," ANTS-I, LNCS 877, pp.28-40,Springer, 1994.

[2] E.Arbarello, M.Cornalba, P.A.Griffiths, and J.Harris, "Geometry of Algebraic Curves Volume I," Springer-Verlag, 1984.

[3] S. Arita, "Algorithms for computations in Jacobian group of C_{ab} curve and their application to discrete-log-based public key cryptosystems," Conference on The Mathematics of Public Key Cryptography, Toronto, 1999.

[4] D.G.Cantor, "Computing in the Jacobian of a hyperelliptic curve," Mathematics of Computation, 48(177), pp.95-101,1987.

[5] G.Frey and H.-G.Rück, "A remark concerning m-divisibility and the discrete logarithm in the divisor class group of curves," Math. Comp.,62(206),pp.865-874,1994.

[6] S.D.Galbraith, S.Paulus, and N.P.Smart "Arithmetic on Superelliptic Curves," preprint,1999.

[7] P.Gaudry, "A variant of the Adleman-DeMarris-Huang algorithm and its application to small genera," Conference on The Mathematics of Public Key Cryptography, Toronto, 1999.

[8] N.Koblitz, "A very easy way to generate curves over prime fields for hyperelliptic cryptosystems," Rump-session Crypto'97, 1997.

[9] B.A.LaMacchia and A.M.Odlyzko, "Solving large sparse linear systems over finite fields," Crypto '90, LNCS 537, pp.109-133,Springer, 1990.

[10] S. Miura, "Linear Codes on Affine Algebraic Curves," Transactions of IEICE, Vol. J81-A, No 10, pp.1398-1421,1998.

[11] PARI-GP, ftp://megrez.math.u-bordeaux.fr/pub/pari

[12] J.M.Pollard, "Monte Carlo methods for index computation mod p," Math. Comp.,32(143),pp.918-924,1978.
[13] H.-G.Rück, "On the discrete logarithm in the divisor class group of curves," Math. Comp.,68(226),pp.805-806,1999.
[14] Y.Sakai and K.Sakurai, "Design of hyperelliptic cryptosystems in small characteristic and a software implementation over F_{2^n}," Asiacrypt '98, Advances in Cryptology, LNCS 1514, pp.80-94,Springer, 1998.
[15] H.Stichtenoth, "Algebraic Function Fields and Codes", Springer-Verlag, 1993.
[16] A.Weil "Numbers of solutions of equations in finite fields," Bull.Amer.Math.Soc.,55,pp.497-508,1949.
[17] A.Weil "Jacobi Sums as "Grössencharaktere"," Trans.Amer.Math.Soc.,73,pp.487-495,1952.

Seigo Arita Since 1990, he has been with C&C Media Research Laboratories, NEC. He is a member of IEICE and JMS.