On the Complexity of Hyperelliptic Discrete Logarithm Problem

Hiroki Shizuya

Department of Electrical Communications,
Faculity of Engineering, Tohoku University
Aramaki-Aza-Aoba, Aoba-ku, Sendai, 980 Japan
shizuya@jpntohok.bitnet

Département d'I.R.O., Université de Montréal
C.P.6128, Succ.A, Montréal, Québec, CANADA, H3C 3J7
shizuya@iro.umontreal.ca

Toshiya Itoh

Department of Information Processing,
The Graduate School at Nagatsuta,
Tokyo Institute of Technology,
4259 Nagatsuta, Midori-ku,
Yokohama 227, Japan
titoh@cc.titech.ac.jp

Kouichi Sakurai

Computer & Information Systems Laboratory,
Mitsubishi Electric Corporation,
5-1-1 Ofuna, Kamakura 247, Japan
sakurai@isl.melco.co.jp

Abstract

We give a characterization for the intractability of hyperelliptic discrete logarithm problem from a viewpoint of computational complexity theory. It is shown that the language of which complexity is equivalent to that of the hyperelliptic discrete logarithm problem is in $\mathcal{NP} \cap$ co-$\mathcal{AM}$, and that especially for elliptic curves, the corresponding language is in $\mathcal{NP} \cap$ co-$\mathcal{NP}$. It should be noted here that the language of which complexity is equivalent to that of the discrete logarithm problem defined over the multiplicative group of a finite field is also characterized as in $\mathcal{NP}$ $\cap$ co-$\mathcal{NP}$.

1 Introduction

In the early times when Diffie and Hellman [DH] proposed a public key-distribution system based on the discrete logarithm problem over the multiplicative group of a finite field,

D.W. Davies (Ed.): Advances in Cryptology - EUROCRYPT '91, LNCS 547, pp. 337-351, 1991.

the intractability of the problem was not exactly characterized. However, Brassard [Br] soon pointed out that the language of which complexity is equivalent to that of the discrete logarithm problem is in $\mathcal{NP}\cap$co-$\mathcal{NP}$. Since then, the discrete logarithm problem associated with a finite group has been well studied, but the problem is not known to be solved in polynomial time.

In 1985, Miller [Mil] showed that there is an alternative for the finite group over which the discrete logarithm problem can be defined, the abelian group of points on an elliptic curve over a finite field. The same idea was also proposed by Koblitz [Kol] independently of Miller's work, and the notion of the elliptic curve discrete logarithm problem was clarified by Koblitz and Kaliski [Kal, Ka2, Ko2]. Informally, the elliptic curve discrete logarithm problem is, given two points X and B on an elliptic curve E over a finite field, to find an integer m such that $X = mB$. In 1989, Koblitz [Ko3] extended the elliptic curve discrete logarithm problem to cover hyperelliptic curves, which is the hyperelliptic discrete logarithm problem we discuss in this paper.

Among those works of forerunners, Miller, Kaliski, and Koblitz, what we should recognize is that they have a common observation on the intractability of the hyperelliptic discrete logarithm problem (including the elliptic curve discrete logarithm problem), i.e., the hyperelliptic discrete logarithm problem seems to be more difficult than the discrete logarithm problem defined over the multiplicative group of a finite field. It is remarkable that Menezes, Okamoto, and Vanstone [MOV] announced that if the elliptic curve is supersingular, the elliptic curve discrete logarithm problem is probabilistic polynomial time reducible to the discrete logarithm problem defined over the multiplicative group of the finite field. Although the reduction is restricted to specific curves, this is the first result concerning the relationship between two distinct kind of discrete logarithm problems. However, in general, the intractability of the hyperelliptic discrete logarithm problem is not yet exactly characterized. So we challenge to this work just as Brassard did for the discrete logarithm problem.

In this paper, it is shown that the language of which complexity is equivalent to that of the hyperelliptic discrete logarithm problem is in $\mathcal{NP}\cap$co-$\mathcal{AM}$, and that especially for elliptic curves, the corresponding language is in $\mathcal{NP} \cap$ co-$\mathcal{NP}$, where $\mathcal{AM}$ denotes the set of languages that have constant round Arthur-Merlin games [Ba]. This is the first characterization for the intractability of hyperelliptic discrete logarithm problem from a viewpoint of structural complexity theory. Note that $\mathcal{NP}$ is contained in $\mathcal{AM}$, but the converse inclusion is not known to hold. It should also be noted here that the language of which complexity is equivalent to that of the discrete logarithm problem defined over the multiplicative group of a finite field is characterized as in $\mathcal{NP} \cap$ co-$\mathcal{NP}$.

To our best knowledge, unlike other languages known to be in $\mathcal{NP} \cap$ co-$\mathcal{AM}$ (such as graph isomorphism [GMW2, Sc2]), the hyperelliptic discrete logarithm problem is the first candidate of *number-theoretic* problems that are characterized as in $\mathcal{NP} \cap$ co-$\mathcal{AM}$ but not known to be in $\mathcal{NP} \cap$ co-$\mathcal{NP}$.

2 Preliminaries

2.1 The Mathematical Background

We start with the definitions of notions and notations related to hyperelliptic curves [Ca, Ko3].

Let K be an arbitrary field, and $\overline{K}$ denote its algebraic closure. A hyperelliptic curve C of genus g over K is the set of solutions $(u, v) \in K^2$ to an equation of the form $v^2 + h(u)v = f(u)$, where $h(u)$ is a polynomial of degree at most g and $f(u)$ is a monic polynomial of degree $2g + 1$. We require that the curve has no singular points.

Let L be a field containing K. By an L-point $P \in C$, we mean either the symbol ∞ or else a finite point, that is a solution $u = x \in L$, $v = y \in L$ of the equation $v^2 + h(u)v = f(u)$. Given a finite point $P = P_{x,y} \in C$, we define its opposite $\tilde{P}$ to be $\tilde{P} = (x, -y - h(x))$.

To introduce the jacobian of the curve C, we define in advance a divisor on C. A divisor is a finite formal sum of $\overline{K}$-points $D = \sum m_i P_i$. The degree of D is defined to be the integer $\sum m_i$, and denoted by deg D. The divisors form an additive group $\mathbf{D}$, and the divisors of degree 0 form a subgroup $\mathbf{D}^0 \subseteq \mathbf{D}$. Given $D \in \mathbf{D}$, we set $D^0 = D - (\deg D)\infty$ so that $D^0 \in \mathbf{D}^0$. Given two divisors $D_1 = \sum m_i P_i$ and $D_2 = \sum n_i P_i$ in $\mathbf{D}^0$, we define g.c.d.$(D_1, D_2) \in \mathbf{D}^0$ to be $\sum \min(m_i, n_i) P_i - (\sum \min(m_i, n_i))\infty$.

For a polynomial $q(u, v)$ with coefficients in $\overline{K}$, the discrete valuation for $q(u, v)$ at a point $P \in C$ can be defined, which is called the order and denoted by $\mathrm{ord}_P\, q$. The divisor $\sum(\mathrm{ord}_P\, q)P$ is denoted by (q), where the summation is taken over all points P on the curve (including ∞). It can be shown that $(p) \in \mathbf{D}^0$. For polynomials p and q, a divisor of the form $(p) - (q)$ is called principal, and such divisors form a subgroup $\mathbf{P}$ of $\mathbf{D}^0$. The jacobian $\mathbf{J}$ of the curve C is the quotient group $\mathbf{D}^0/\mathbf{P}$. If $D_1, D_2 \in \mathbf{D}^0$, we write $D_1 \sim D_2$ if $D_1 - D_2 \in \mathbf{P}$, i.e., if D_1 and D_2 are equal when considered as elements of $\mathbf{J}$. We let $\mathbf{J}(C; L)$ denote the set of L-points of $\mathbf{J}$ associated with the curve C.

A divisor $D = \sum m_i P_{x_i,y_i} - (\sum m_i)\infty$ can be uniquely respresented as the g.c.d. of two principal divisors of polynomials of the form $a(u)$ and $b(u) - v$, that is, g.c.d.$((a(u)), (b(u) - v))$. We write $D = \mathrm{div}(a, b)$ in short to denote such D.

Let $K = \mathbf{F}_q$, a finite field of q elements, and let $\mathbf{F}_{q^n}$ be its extension. We regard them as fixed. Given a divisor D, the unique representation $\mathrm{div}(a, b)$ can be obtained in $O(n^2)$ bits operations. Furthermore, given $D \in \mathbf{J}(C; \mathbf{F}_{q^n})$, the multiples of the divisor, denoted by mD, can be computed efficiently by the repeated doubling method, which takes $O(n^3)$ bits operation. Informally, the hyperelliptic discrete logarithm problem is, given two divisors X and B in $\mathbf{J}$ associated with a hyperelliptic curve C over a finite field, to find an integer m such that $X \sim mB$.

2.2 HEDL and the Related Languages

Throughout this paper, all strings will be over the finite alphabet $\Sigma = \{0, 1\}$. We use $|x|$ to represent the length of string x. We let Σ^* designate the set of all possible strings including zero-length string λ. A language is a set of strings. A class is a set of languages. For a language L, we use $\overline{L}$ to denote $\Sigma^* \setminus L$. For a class $\mathcal{C}$, we use co-$\mathcal{C}$ to denote its class of complements, i.e. the set of any L such that $\overline{L}$ is in $\mathcal{C}$. For any finite set A, we let $\sharp A$ designate its cardinality.

We now define HEDL, the hyperelliptic discrete logarithm problem on $\mathbf{J}(C; \mathbf{F}_{q^n})$.

Definition 1 (HEDL) :
HEDL(q, n, C, X, B) is a computing problem, where q is prime power, n is a positive integer, C is a hyperelliptic curve (with no singular points) defined over $\mathbf{F}_q$, and X and B are divisors in $\mathbf{J}(C; \mathbf{F}_{q^n})$. If there exists an integer m such that $X \sim mB$ and $0 \leq m < \sharp\mathbf{J}(C; \mathbf{F}_{q^n})$, then the answer is the smallest m, and if such m does not exist, the answer is a special string "$\perp$".

Given q, n, and C, we can check in probabilistic polynomial time that the curve C has a singular point. Note that, by the definition, HEDL is the elliptic curve discrete logarithm problem when the genus of the curve is 1.

Two languages L_s and L_ℓ are also introduced to explore the intractability of this problem. The language L_s is the set of instances of *solvable* hyperelliptic discrete logarithm problem, of which membership problem is to answer *yes* if the input causes HEDL to return a non-negative integer and *no* otherwise. The language L_ℓ is the set of instances of *location* problem associated with hyperelliptic discrete logarithms, of which membership problem is to answer *yes* if the input causes HEDL to return an integer $\geq k$ and *no* otherwise.

Definition 2 (L_s):
$L_s = \{< q, n, C, X, B > \mid (\exists m \geq 0)[\mathrm{HEDL}(q, n, C, X, B) = m]\}$.

Definition 3 (L_ℓ) :
$L_\ell = \{< q, n, C, X, B, k > \mid (k \in \mathbf{Z}_{\geq 0}) \wedge (\mathrm{HEDL}(q, n, C, X, B) \geq k)\}$.

Obviously, L_s is deterministic polynomial time Turing reducible to L_ℓ. Furthermore, it is easy to see that the complexity of the language L_ℓ is equivalent to the complexity of the problem HEDL.

2.3 The Order of Jacobian

It is important to note that $\mathbf{J}(C; \mathbf{F}_{q^n})$ is not necessarily a cyclic group but a (finite) abelian group. We also define the problem OrdJ and the language L_{NJ} to investigate the complexity of computing the exact order of $\mathbf{J}(C; \mathbf{F}_{q^n})$.

Definition 4 (OrdJ) :
OrdJ(q, n, C) is a counting problem, where q is prime power, n is a positive integer, C is a hyperelliptic curve (with no singular points) defined over $\mathbf{F}_q$. If the input is valid, the answer is the exact order of $\mathbf{J}(C; \mathbf{F}_{q^n})$, and if invalid, the answer is "$\perp$".

Definition 5 (L_{NJ}) :
$L_{NJ} = \{< N, q, n, C > \mid (N$ is a positive integer$)$
$\wedge\ (N = \mathrm{OrdJ}(q, n, C))\ \}$.

Clearly, the language L_{NJ} is in P if there exists a deterministic polynomial time algorithm fo computing OrdJ. Pila [20] showed the following theorem as an extension of Schoof's result [Sch].

Theorem A (Pila [Pi]): Let A be an abelian variety over a finite field $\mathbf{F}_q$. Then one can compute the characteristic polynomial of the Frobenius endomorphism of A in time $O((\log q)^\Delta)$ where Δ and the implied constant depend only on the form of the equations defining A.

Theorem A implies that we can compute the order of $\mathbf{J}(C; \mathbf{F}_{q^n})$ in polynomial time. Thus, we have the following theorem, which will later become important.

Theorem B: The language L_{NJ} is in $\mathcal{P}$.

3 Main Results

Recall that the complexity of the language L_ℓ, the set of instances of *location* problem associated with HEDL, is equivalent to that of HEDL. This implies that L_ℓ completely characterizes the complexity of HEDL. We show in this section the following results.

Theorem 1 : L_ℓ is in $\mathcal{NP} \cap$ co-$\mathcal{AM}$.

Theorem 2 : For any elliptic curve E, let L_ℓ be denoted by L_ℓ^E. Then, L_ℓ^E is in $\mathcal{NP} \cap$ co-$\mathcal{NP}$.

Whereas other complexity-theoretic properties of HEDL and L_s are investigated in the appendix, where we show the followings as well as some immediate corollaries.

Theorem A1: The problem HEDL is random self-reducible in the sense of the definition in [TW].

Theorem A2: There exists a perfect zero-knowledge interactive proof system for the language L_s.

Theorem A3: There exists a perfect zero-knowledge interactive proof system for the language $\overline{L}_s$.

We now restrict ourselves to the discussion on the complexity of L_ℓ. In 1988, Goldreich and Kushilevitz [GK] showed a perfect zero-knowledge interactive proof for the language of which complexity is equivalent to that of the discrete logarithm problem over a multiplicative group of a finite field, and they mentioned that their protocol would be extended to cover the general discrete logarithm problem defined over a finite abelian group. However, they assume in [GK] that the structure of finite abelian group is known, whereas we do not. To investigate the complexity of HEDL without such assumption, we take into account the complexity of determining the structure of finite abelian group. Thus, the context in this paper is crucially different from that in [GK].

Proof of Theorem 1 :

L_ℓ is in $\mathcal{NP}$: It is easily seen that L_ℓ is in $\mathcal{NP}$ if N =OrdJ(q, n, C) is given. In fact, a nondeterministic polynomial time Turing machine can guess $m =$ HEDL(q, n, C, X, B) among positive integers less than N, and then check in a straightforward manner that $m \geq k$. Here, by Theorem B, L_{NJ} is in $\mathcal{P}$. Thus, L_ℓ is in $\mathcal{NP}$.

L_ℓ is in co-$\mathcal{AM}$: We show that $\overline{L}_\ell$ is in $\mathcal{AM}$. $\overline{L}_\ell$ is expressed as follows:

$$\begin{aligned}\overline{L}_\ell = \ & \{x \mid x \text{ does not satisfy at least one of the specifications for } q, n, C, X, B, \text{ and } k\} \\ & \cup \{< q, n, C, X, B, k > \mid (\exists m \geq 0)[\text{HEDL}(q, n, C, X, B) = m < k]\} \cup L_{us},\end{aligned}$$

where

$$L_{us} = \{< q, n, C, X, B > \mid \mathrm{HEDL}(q, n, C, X, B) = \text{“ }\perp\text{”}\},$$

that is, the set of instances of *unsolvable* hyperelliptic discrete logarithm problem.

For $\overline{L}_\ell$, the first two sets are both in $\mathcal{NP}$. Thus, it suffices to show that there exists a constant round interactive proof system for the language L_{us}, because Goldwasser and Sipser showed in [GS] that any language having a constant round interactive proof system can be simulated by a constant round Arthur-Merlin game.

The interactive protocol over P and V on input $< q, n, C, X, B >$ consists of three parts, where we use P and V to designate the all-powerful prover and the probabilistic polynomial time bounded verifier, respectively. Informally saying, in Part 1, P and V share a set of points on $\mathbf{J}(C; \mathbf{F}_{q^n})$ that are seemingly the generators of $\mathbf{J}(C; \mathbf{F}_{q^n})$. Note that $\mathbf{J}(C; \mathbf{F}_{q^n})$ is generated by at most $2g$ cyclic groups, where g is the genus of curve C (the proof for the case $g = 1$ will be found in [Sil]). In Part 2, P shows V that the set is actually the set of generators of $\mathbf{J}(C; \mathbf{F}_{q^n})$. This part is inspired by the constant round interactive protocol for graph non-isomorphism [GMW1]. In Part 3, P shows V that there exists no m such that $X \sim mB$.

The protocol works as follows:

Input to (P,V) : $< q, n, C, X, B >$

Part 1:

V: does nothing.

P: chooses $G = (\xi_1, \ldots, \xi_\ell)$, the tuple of generators of abelian decomposition of $\mathbf{J}(C; \mathbf{F}_{q^n})$. That is, each $\xi_i \in G$ $(1 \leq i \leq \ell)$ has the order of prime power, namely $\mathrm{ord}(\xi_i) = p_i^{n_i}$, and G generates $\mathbf{J}(C; \mathbf{F}_{q^n})$ itself: $\mathbf{J}(C; \mathbf{F}_{q^n}) \cong \langle\xi_1\rangle \oplus \cdots \oplus \langle\xi_\ell\rangle$, where $\oplus$ denotes the direct sum, and $N = \Pi_{i=1}^{\ell} p_i^{n_i} = \mathrm{OrdJ}(q, n, C)$.

P→V: G, $\{p_i\}$, $\{n_i\}$, and $\mathcal{NP}$-proofs [Pr] for the fact that p_i is prime $(1 \leq i \leq \ell)$.

V: continues if $\mathrm{ord}(\xi_i) = p_i^{n_i}$ and $N = \Pi_{i=1}^{k} p_i^{n_i}$ with p_i prime $(1 \leq i \leq \ell)$ else rejects and halts.

Part 2:

V: randomly picks $\sigma \in S_\ell$ and $r_i \in \mathbf{Z}_{\mathrm{ord}(\xi_{\sigma(i)})} \setminus \{0\}$, and computes $T_i \sim r_i \xi_{\sigma(i)}$ $(1 \leq i \leq \ell)$, where S_ℓ denotes the symmetric group of degree ℓ.

V→P: $T_1, \ldots, T_\ell$

P: computes $\tau \in S_\ell$ such that $T_i \sim \tilde{r}_i \xi_{\tau(i)}$ $(1 \leq i \leq \ell)$.

P→V: τ

V: continues if $\tau = \sigma$ else rejects and halts.

Part 3:

P: computes $(x_1, \ldots, x_\ell)$ and $(b_1, \ldots, b_\ell)$ such that

$$X = [\xi_1^{x_1}, \ldots, \xi_\ell^{x_\ell}] \text{ and } B = [\xi_1^{b_1}, \ldots, \xi_\ell^{b_\ell}].$$

P→V: $(x_1, \ldots, x_\ell)$ and $(b_1, \ldots, b_\ell)$.

V: accepts if

$$X = [\xi_1^{x_1}, \ldots, \xi_\ell^{x_\ell}], B = [\xi_1^{b_1}, \ldots, \xi_\ell^{b_\ell}],$$

and there exists no m satisfying the linear equations

$$(\forall j)[x_j \equiv b_j m \bmod \operatorname{ord}(\xi_j)],$$

else rejects and halts. (End of Protocol)

Note that in the last step, V checks that $\neg(\exists m)[X \sim mB]$. Because

$$\neg(\exists m)[X \sim mB] \Leftrightarrow \neg(\exists m)[(\forall j)[\xi_j^{x_j} = (\xi_j^{b_j})^m]]$$
$$\Leftrightarrow \neg(\exists m)[(\forall j)[x_j \equiv b_j m \bmod \operatorname{ord}(\xi_j)]].$$

This protocol constitutes a constant round interactive proof system for L_{us}. Thus, it is immediate from the result in [GS] that L_{us} is in $\mathcal{AM}$. □

It is clearly seen that determining the structure of $\mathbf{J}(C; \mathbf{F}_{q^n})$ is in $\mathcal{AM}$, hence L_{us} is in $\mathcal{AM}$. In other words, if the structure is determined in nondeterministic polynomial time, then L_{us} is in $\mathcal{NP}$, and consequently L_ℓ is in $\mathcal{NP} \cap$ co-$\mathcal{NP}$. To prove Theorem 5, we show that for an elliptic curve E, the structure of $\mathbf{J}(E; \mathbf{F}_{q^n})$ is determined in nondeterministic polynomial time. The idea is based on Miller's algorithm [Mi2] to compute the value of Weil e_m-pairing, which also plays an important role in [MOV] to prove the reduction from the (supersingular) elliptic curve discrete logarithm problem to the discrete logarithm problem defined over the multiplicative group over the finite filed.

Proof of Theorem 2:

It suffices to show that for any elliptic curve E, L_{us}^E is in $\mathcal{NP}$, where L_{us}^E is a subset of L_{us} for the case of genus $g = 1$. We use the notion of Weil e_m-pairing defined as follows (see also [Sil, p.95]).

> **Weil e_m-pairing** [Mi2]: Given an elliptic curve E and a non-negative integer m, there is a unique function e_m such that
>
> $$e_m : E[m] \times E[m] \rightarrow \overline{\mathbf{F}}_{q^n},$$
>
> where
>
> $$E[m] = \{S \in \mathbf{J}(E; \overline{\mathbf{F}}_{q^n}) \mid m \neq 0, \ mS = O\}.$$
>
> Here, we use O to denote the identity element in $\mathbf{J}(E; \mathbf{F}_{q^n})$.

Weil e_m-pairing has the following properties:

1. $e_m(S,T)$ is an m-th root of unity for all $S,T \in E[m]$.
2. Identity: $e_m(S,S)=1$ for all $S \in E[m]$.
3. Skew-symmetry: $e_m(S,T)=e_m(T,S)^{-1}$ for all $S,T \in E[m]$.
4. Linearity: $e_m(S+U,T)=e_m(S,T)e_m(U,T)$ for all $S,T,\, U \in E[m]$.
5. Non-degeneracy: If $e_m(S,T)=1$ for all $S \in E[m]$, then $T=O$.

To determine the structure of $\mathbf{J}(E;\mathbf{F}_{q^n})$, the following facts are essential:

1. $\mathbf{J}(E;\mathbf{F}_{q^n})$ is always either cyclic, or the direct sum of two cyclic groups of orders α and β where $\alpha|\beta$.

2. Let ξ_1 and ξ_2 be points on $\mathbf{J}(E;\mathbf{F}_{q^n})$ of orders α and β respectively. If $\alpha|\beta$, $\alpha\beta=N=\mathrm{Ord}\mathbf{J}(q,n,E)$, and $e_\beta(\xi_1,\xi_2)\neq 1$, then $\mathbf{J}(E;\mathbf{F}_{q^n})$ is the direct sum of two cyclic groups $\langle\xi_1\rangle$ and $\langle\xi_2\rangle$. In fact, by the properties 1 and 4 of Weil pairing, if $\xi_1=t\xi_2$ for some $t\in\mathbf{Z}_N$, $e_\beta(\xi_1,\xi_2)=e_\beta(t\xi_2,\xi_2)=e_\beta(\xi_2,\xi_2)^t=1^t=1$.

3. Miller [Mi2] showed an algorithm that on input an elliptic curve E over $\mathbf{F}_{q^n}$, a natural number m, and two points $P,Q\in E[m]$, outputs the value $e_m(P,Q)$, which runs in expected polynomial (in $\log q$) time. (This is now converted into a deterministic polynomial time algorithm by V. S. Miller.) To compute $e_m(P,Q)$, the algorithm first picks additional points $T,U\in\mathbf{J}(E;\mathbf{F}_{q^n})$ at random repeatedly until T and U satisfy the specific conditions [Ka2, Mi2]. The conditions for the choice depends only on inputs m,P,Q, and they can be checked in deterministic polynomial time. Note that such points T,U always exist for any m,P,Q. Once T and U are appropriately fixed, the subsequent steps are executed in deterministic polynomial time. The reason why the running time is *expected* polynomial is explained by the random choice of the additional points T,U.

By the above facts, we show that L_{us}^E is in $\mathcal{NP}$.

We guess $\xi\in\mathbf{J}(E;\mathbf{F}_{q^n})$ and the factorization of $N=\mathrm{Ord}\mathbf{J}(q,\,n,E)$, and check in deterministic polynomial time that $\mathrm{ord}(\xi)=N$. If the check is passed, we determine that $\mathbf{J}(E;\mathbf{F}_{q^n})=\langle\xi\rangle$. Otherwise, we guess $\xi_1,\xi_2\in\mathbf{J}(E;\mathbf{F}_{q^n})$ and the additional points $T,U\in\mathbf{J}(E;\mathbf{F}_{q^n})$ to be used in Miller's algorithm. Then, we check in deterministic polynomial time that $\mathrm{ord}(\xi_1)\cdot\mathrm{ord}(\xi_2)=\alpha\beta=N$, $\alpha|\beta$, and $e_\beta(\xi_1,\xi_2)\neq 1$. If the checks are passed, we determine that $\mathbf{J}(E;\mathbf{F}_{q^n})\cong\langle\xi_1\rangle\oplus\langle\xi_2\rangle$. At this time, the structure of $\mathbf{J}(E;\mathbf{F}_{q^n})$ has been determined in nondeterministic polynomial time. Next, we check that $\neg(\exists m)[X \sim mB]$, which is an $\mathcal{NP}$-statement as shown in Part 3 in the interactive protocol for L_{us}. Thus, L_{us}^E is in $\mathcal{NP}$, and consequently L_t^E is in $\mathcal{NP} \cap$ co-$\mathcal{NP}$. □

4 Concluding Remarks

We showed that for curves with genus $g \geq 2$, the complexity of hyperelliptic discrete logarithm problems is characterized as in $\mathcal{NP} \cap$ co-$\mathcal{AM}$. Whereas, for curves with $g = 1$, the complexity is characterized as in $\mathcal{NP} \cap$ co-$\mathcal{NP}$. The latter is the same characterization for the discrete logarithm problem defined over the multiplicative group of a finite field.

To characterize the complexity of hyperelliptic discrete logarithm problem as in $\mathcal{NP} \cap$ co-$\mathcal{NP}$ for any g, it suffices to positively solve the question: Is L_{us}, the set of instances of ***unsolvable*** hyperelliptic discrete logarithm problem, in $\mathcal{NP}$ for any $g \geq 2$? However, we have not yet solved it. If Miller's algorithm can be used to determine the structure of jacobian $\mathbf{J}(C; \mathbf{F}_{q^n})$ for C with any g, L_{us} is in $\mathcal{NP}$. But this requires an extension and redefinition of Weil pairing to cover curves with $g \geq 2$. It is worth noting that the attempt to define such extended Weil pairing and to positively solve this open question has already been started by Okamoto and Sakurai [OS].

Acknowledgements

We wish to thank Neal Koblitz of Washington University, Tatsuaki Okamoto of NTT, and Jonathan S. Pila of Columbia University for their invaluable suggestions on our work. Eyal Kushilevitz of Technion gave us helpful comments on [GK]. Victor S. Miller of IBM kindly informed us of the existence of deterministic polynomial time version of his algorithm for computing Weil pairing. We are also grateful to Mitsuru Matsui of Mitsubishi Electric Corporation for his helpful advice on algebraic geometry and number theory.

References

[AH] W. Aiello and J. Håstad, "Statistical zero-knowledge languages can be recognized in two rounds," Proc. 28th FOCS, pp.439-448 (1987).

[Ba] László Babai, "Trading group theory to randomness," Proc. 17th STOC, pp.421-429 (1985).

[Br] Gilles Brassard, "A note on the complexity of cryptography," IEEE Trans. Inf. Theory, vol.IT-25, no.2, pp.232-233 (1979).

[Ca] David G. Cantor, "Computing in the Jacobian of a hyperelliptic curve," Math. Comp., vol.48, no.177, pp.95-101 (1987).

[DH] W. Diffie and M. Hellman, "New directions in cryptography," IEEE Trans. Inf. Theory, vol.IT-22, no.6, pp.644-654 (1976).

[Fo] Lance J. Fortnow, "The complexity of perfect zero-knowledge," Proc. 19th STOC, pp.204-209 (1987).

[GK] O. Goldreich and E. Kushilevitz, "A perfect zero-knowledge interactive proof for a problem equivalent to discrete logarithm," Proc. CRYPTO'88 (1988).

[GMR] S. Goldwasser, S. Micali, and C. Rackoff, "The zero-knowledge complexity of interactive proof-systems," Proc. 17th STOC, pp.291-304 (1985).

[GMW1] O. Goldreich, S. Micali, and A. Wigderson, "Proofs that yield nothing but their validity and a methodology of cryptographic protocol design," Proc. 27th FOCS, pp.174-187 (1986).

[GMW2] O. Goldreich, S. Micali, and A. Wigderson, "Proofs that yield nothing but their validity or All languages in $\mathcal{NP}$ have zero-knowledge proofs," Technical Report 554, Technion (1989).

[GS] S. Goldwasser and M. Sipser, "Private coins versus public coins in interactive proof systems," Proc. 18th STOC, pp.59-68 (1986).

[Ka1] Burton S. Kaliski, Jr., "A pseudo-random bit generator based on elliptic logarithms," Proc. CRYPTO'86, pp.84-103 (1986).

[Ka2] Burton S. Kaliski, Jr., "Elliptic curves and cryptography: a pseudorandom bit generator and other tools," MIT/LCS/ TR-411, MIT (1988).

[Ko1] Neal Koblitz, "Elliptic curve cryptosystems," Math. Comp., vol.48, no.177, pp.203-209 (1987).

[Ko2] Neal Koblitz, "A Course in Number Theory and Cryptography," GTM114, Springer-Verlag, New York (1987).

[Ko3] Neal Koblitz, " Hyperelliptic cryptosystems," J. Cryptology, vol.1, no.3, pp. 139-150 (1989).

[Mi1] Victor S. Miller, "Use of elliptic curves in cryptography," Proc. CRYPTO'85, pp.417-426 (1985).

[Mi2] Victor S. Miller, "Short programs for functions on curves," manuscript (1986).

[MOV] A. Menezes, T. Okamoto, and S. Vanstone, "Reducing elliptic curve logarithms to logarithms in a finite field," announced at CRYPTO'90 rump session (1990) (to appear in Proc. STOC'91).

[OS] T. Okamoto and K. Sakurai, "On the complexity of problems associated with hyperelliptic curves," Proc. SCIS91, 9C (1991).

[Pi] Jonathan S. Pila, "Frobenius maps of abelian varieties and finding roots of unity in finite fields," Ph.D Thesis, Stanford University (to appear in Math. Comp.) (1988).

[Pr] Pratt,V., "Every Prime has a succinct certificate," SIAM J. COMPUT. vol.4, pp.214-220 (1975).

[Sc1] Uwe Schöning, "A low and high hierarchy within $\mathcal{NP}$," J. Comp. Syst. Sci., vol.27, pp.14-28 (1983).

[Sc2] Uwe Schöning, "Graph isomorphism is in the low hierarchy," J. Comp. Syst. Sci., vol.37, pp.312-323 (1988).

[Sch] René Schoof, "Elliptic curves over finite field and the computation of square roots mod p," Math. Comp., vol.44, pp.483-494 (1985).

[Shi] Hiroki Shizuya, "Zero-knowledge interactive proofs for hyper- and elliptic-discrete logarithm problems," Proc. WCIS'89, pp.143-152 (1989).

[SI] H. Shizuya, and T. Itoh, "A group-theoretic interface to random self-reducibility," Trans. IEICE, vol.E-73, no.7, pp.1087-1091 (1990).

[Sil] Joseph H. Silverman, "The Arithmetic of Elliptic Curves," GTM 106, Springer-Verlag, New York (1986).

[TW] M. Tompa and H. Woll, "Random self-reducibility and zero knowledge interactive proofs for possession of information," Proc. 28th FOCS, pp.472-482 (1987).

Appendix

A.1 Random Self-Reducibility of HEDL

HEDL reduces to the elliptic curve discrete logarithm problem when $g = 1$, and we know that the elliptic curve discrete logarithm is random self-reducible [Shi] in the sense of definition in [TW]. We show here that HEDL is also random self-reducible.

Theorem A1 : HEDL is random self-reducible.

A lemma is required for the proof. For a finite group G under a binary operation, we mean by the accessibility of G that any element in G can be picked randomly and uniformly in time polynomial in $|\sharp G|$, and that the binary operation for any pair of elements is computed in time polynomial in $|\sharp G|$.

Lemma A1 ([SI]) :
Let G_1 and G_2 be accessible finite groups, respectively, and φ be a homomorphism from G_1 onto G_2. For any $\xi \in G_1$, let $\xi^{-1} \in G_1$ and $\varphi(\xi)$ be computed in time polynomial in $|\sharp G_1|$. Then, given $x \in G_2$, the problem to compute some $y \in G_1$ such that $x = \varphi(y)$ is random self-reducible.

Proof of Theorem A1 :
Let φ be the homomorphism from the finite abelian group $\mathbf{Z}_N$ onto $\langle B \rangle$ such that $\varphi(\xi) = \xi B$, where $N =$OrdJ(q, n, C), and $\langle B \rangle$ is the group of divisors that consists of any multiple of B. By Pila's theorem, N is computed in polynomial time. This implies that we can determine the range of elements in $\mathbf{Z}_N$ to pick, thus $\mathbf{Z}_N$ is accessible. In addition, given any element in $\mathbf{Z}_N$, its inverse is computed in a straightforward manner. Since φ is computed in polynomial time by using the repeated doubling algorithm, any element in $\langle B \rangle$ can be picked randomly by computing $\varphi(r)$ with r chosen randomly from $\mathbf{Z}_N$. This implies that $\langle B \rangle$ is also accessible. Thus, by Lemma A1, HEDL is random self-reducible. □

A.2 Perfect Zero-Knowledge Interactive Proof for L_s

Theorem A2 : There exists a perfect zero-knowledge interactive proof system for L_s.

Proof : The computing model is based on that in [TW], i.e., we consider the interaction between the prover (P) of unbounded power and the verifier (V) of probabilistic polynomial time bounded power. The construction of protocol is almost the same as shown in [TW] for the language membership. A perfect zero-knowledge protocol for L_s on input (q, n, C, X, B) works as follows:

P: picks $r \in [0, N)$ randomly and computes R such that $R \sim rB$, where $N =$ OrdJ (q, n, C).

P→V: R

V→P: $e \in \{0, 1\}$ chosen at random.

P→V: σ such that $R \sim \sigma B - eX$.

By [TW], the above protocol forms a perfect zero-knowledge interactive proof system for the language membership in L_s. □

A.3 Perfect Zero-Knowledge Interactive Proof for $\overline{L}_s$

We mean by $\overline{L}_s$ the complement of L_s. More precisely,

$\overline{L}_s = \{x \mid x$ does not satisfy at least one of the specifications for q, n, C, X, and $B\}$
$\cup \{< q, n, C, X, B > \mid X \in \mathbf{J}(C; \mathbf{F}_{q^n}) \setminus \langle B \rangle\}$.

The latter set is equivalent to L_{us} which we discussed in Section 3.

Theorem A3 : There exists a perfect zero-knowledge interactive proof system for $\overline{L}_s$.

Proof : The protocol is essentially the same as the statistical zero-knowledge interactive proof for the language membership in $\{< p, a, b > \mid b \in \mathbf{Z}_p^* \setminus \langle a \rangle_p\}$ shown in [TW]. However, the following construction of the protocol is converted into perfect zero-knowledge, based on the idea of perfect zero-knowledge interactive proof for graph non-isomorphism [GMW2]. Note that in the protocol, steps to check the validity of input are omitted since V of probabilistic polynomial time power can check it without interaction.

Input to (P,V) : $< q, n, C, X, B >$

V: chooses $r \in \mathbf{Z}_N \setminus \{0\}$ and $\alpha \in \{0, 1\}$ randomly and uniformly, and computes $Z \sim rB + \alpha X$, where $N =$ OrdJ (q, n, C). V also generates $T_i = (T_{i0}, T_{i1})$ $(1 \leq i \leq t = 2|N|)$ such that $T_{i0} \sim Z + s_{i0}B - \beta_i X$ and $T_{i1} \sim Z + s_{i1}B - (1 - \beta_i)X$, where $s_{ij} \in \mathbf{Z}_N \setminus \{0\}$ $(j = 0, 1)$ and $\beta_i \in \{0, 1\}$ are chosen randomly, uniformly and independently.

V→P: $Z,\ T_i(1 \le i \le t)$

P: chooses at random, a subset $I \subseteq \{1, 2, \dots, t\}$.

P→V: I

V: rejects and halts if $I \not\subseteq \{1, 2, \dots, t\}$. Otherwise, V generates $w_i(1 \le i \le t)$ such that if $i \in I$, $w_i = (\beta_i, s_{i0}, s_{i1})$; if $i \notin I$, $w_i = (\gamma_i, u_i)$ where $\gamma_i \equiv \alpha + \beta_i \pmod 2$ and $u_i \equiv r + s_{i\gamma_i} \pmod N$.

V→P: $w_i(1 \le i \le t)$

P: checks that for $w_i(1 \le i \le t)$, $((i \in I) \wedge (T_{i0} \sim Z + s_{i0}B - \beta_i X) \wedge (T_{i1} \sim Z + s_{i1}B - (1 - \beta_i)X)) \vee ((i \notin I) \wedge (T_{i\gamma_i} \sim u_i B))$. If either condition is violated, P stops. Otherwise, P computes δ such that $\delta = 0$ if $Z \in \langle B \rangle$; $\delta = 1$ if $Z \notin \langle B \rangle$.

P→V: δ

V: accepts if $\alpha = \delta$. Otherwise, V rejects and halts.

Completeness, soundness, and perfect zero-knowledgeness of the protocol are proven in a way like [GMW2]. In the proof of perfect zero-knowledgeness, it is essential that $t = 2|N|$, because $2^{-t} \cdot 2N < 1$. □

A.4 Immediate Corollaries

Combining Theorem A2 and the results of Fortnow [Fo] and Aiello-Håstad [AH], we can show that L_s is in $\mathcal{AM} \cap$ co-$\mathcal{AM}$. However, L_s is polynomial time Turing reducible to L_ℓ, and our main results show that L_ℓ is in $\mathcal{NP} \cap$ co-$\mathcal{AM}$, and that for elliptic curves L_ℓ^E is in $\mathcal{NP} \cap$ co-$\mathcal{NP}$. Thus,

Corollary A1: L_s is in $\mathcal{NP} \cap$ co-$\mathcal{AM}$, and L_s^E is in $\mathcal{NP} \cap$ co-$\mathcal{NP}$, where L_s^E designates L_s for the case of genus $g = 1$.

It immediately follows from [Fo] that neither L_s nor L_s^E will be $\mathcal{NP}$-complete unless the polynomial time hierarchy collapses to the second level. Furthermore, by Schöning's results on his low and high hierarchies within $\mathcal{NP}$ [Sc1, Sc2], we can show

Corollary A2: Neither L_s nor L_ℓ will be $\mathcal{NP}$-complete unless the polynomial time hierarchy collapses to the second level.

Corollary A3: Neither L_s^E nor L_ℓ^E will be $\mathcal{NP}$-complete unless the polynomial time hierarchy collapses to the first level.