An Experiment of The Number Field Sieve for Discrete Logarithm Problem over GF(pn)

早坂, 健一郎

https://doi.org/10.15017/1500512

An Experiment of The Number Field Sieve for Discrete Logarithm Problem over GF(p^n)

Kenichiro HAYASAKA

A thesis submitted in fulfillment of the requirements
for the Ph.D. degree in Functional Mathematics

Supervisor: Prof. Tsuyoshi TAKAGI

Graduate School of Mathematics Kyushu University

Abstract

The security of pairing-based cryptography is based on the hardness of solving the discrete logarithm problem (DLP) over an extension field GF(p^n) of characteristic p and degree n. Joux et al. proposed the asymptotically fastest algorithm for solving DLPs over GF(p^n) (JLSV06-NFS). This algorithm is an extension of the number field sieve over the prime field GF(p) (JL03-NFS).

The lattice sieve is often used in large-scaled experiments on solving DLPs over GF(p). Franke and Kleinjung proposed a two-dimensional lattice sieve that efficiently enumerates all the points in a given sieve region of the lattice. However, we have to consider a sieve region of more than two dimensions in the lattice sieve of JLSV06-NFS.

In this paper, we presented an implementation of the number field sieve for solving the DLP over an extension field GF(p^n) that underpinned the security of pairing-based cryptography. Especially we proposed the implementation of the lattice sieve of more than two dimensions. In our experiment, we discussed the dimension and size of sieve region suitable for the number field sieve over an extension field GF(p^{12}). Finally we have solved the DLP over an extension field GF(p^{12}) of 203 bits using a PC of 16 CPU cores in about 43 hours.

Additionally, we extend the Franke-Kleinjung method to a three-dimensional sieve region. We construct an appropriate basis by using the Hermite normal

form. An experiment on a GF(p^{12}) of 303 bits indicated that we can enumerate more than 90% of the points generated by special-q in the three-dimensional sieve region. Moreover, we implemented a number field sieve using the three-dimensional lattice sieve. Our implementation of JLSV06-NFS over a GF(p^6) of 240 bits was about as efficient as the current record holder, a three-dimensional line sieve by Zajac, over a GF(p^6).

Acknowledgments

First, I appreciate Professor Tsuyoshi Takagi giving me a lot of great suggestions and continued support. It was so fortunate for me that I met him and studied cryptography in his laboratory. Second, I express my gratitude to Kazumaro Aoki and Tetsutaro Kobayashi (NTT Secure Platform Laboratories) for many valuable comments through joint work. Finally, I am thankful for the support by all professors and students in Takagi laboratory.

Contents

List of Tables

List of Figures

List of Algorithms

Chapter 1

Introduction

1.1 Pairing-based cryptography

Nowadays, services on the Internet are ordinary tools in our daily life. For instance, e-mail, banking, shopping and so on. When we use these services, cryptography is utilized in order to hide our secret information such as a password, a credit card number and so on. Especially, public-key cryptography is the fundamental factor to hide the secret information's. Public-key cryptography plays an important role to communicate secretly and verify signature through insecure channel by solving key distribution problem. Recently, in many cases, our several sensitive data is stored on not a private storage but a storage connected to the Internet by growth of cloud computing. Thus, more functional cryptosystems are needed to protect such data effectively.

Pairing-based cryptography is a cryptosystem that attracts due to the novel cryptographic protocols such as ID-based cryptography [12], functional encryption [69], etc. It is expected that these more functional protocols are useful for prospective information security. Many efficient implementations of pairing have

been reported, and one of the most efficient algorithms for computing pairing is the optimal Ate pairing [89] using BN curves [10]. The security of pairing-based cryptography using BN curves is based on the hardness of the discrete logarithm problem over finite field GF(p^{12}). The complexity of solving the discrete logarithm problem over finite field GF(p^{12}) of 3072 bits is estimated as 2^{128} [10].

1.2 The number field sieve over GF(p^n)

In this paper, we deal with the discrete logarithm problems over an extension field GF(p^n) with extension degree $n > 1$ and large characteristic p since we consider the security of pairing-based cryptography.

The number field sieve algorithms belongs to index calculus algorithm class. We usually describe the running time of an algorithm belonging to index calculus algorithm class as follows:

$$L_{p^n}(u, c) = \exp((c + o(1))(\log p^n)^u(\log\log p^n)^{1-u})$$

If the constant c is not given we use $L_{p^n}(u)$. If $u = 0$ holds, the running time of the algorithm is polynomial time. On the other hand, If $u = 1$ holds, the running time of the algorithm is exponential time. Additionally, If $0 < u < 1$ holds, the running time of the algorithm is subexponential time. The number field sieve consists of 4 major steps: polynomial selection, collection of relation, linear algebra, individual logarithms. First, we select two polynomials that defines a number field F in the polynomial selection step. Second, we try to find sufficient pairs of integer that satisfies some conditions in the collection of relation step. Third, we construct a linear system that consists of the relations in linear algebra step. Finally, we compute target logarithm from the solution of the linear system.

At the beginning, we introduce the number field sieve over a prime field

GF(p). The number field sieve over GF(p) was first proposed by Gordon in [29] as an extension of the number field sieve for integer factorization and its running time is $L_p(1/3, 3^{2/3})$. Next, Schirokauer improved the structure of a relation and the running time to $L_p(1/3, (64/9)^{1/3})$ [74]. Furthermore, Joux and Lercier proposed the polynomial selection method and the construction of the linear system by introducing virtual logarithms [37].

The asymptotically fastest algorithm for solving the discrete logarithm problems over an extension field GF(p^n) is the number field sieve proposed by Joux et al. at CRYPTO 2006 [41]. In this paper, we denote the number field sieve method by JLSV06-NFS. There are two experimental reports on the implementation of the number field sieve over an extension field GF(p^n) of $n = 3$ [41] and $n = 6$ [94, 95]. However, to the best of our knowledge, there is no experimental report on the hardness of the DLP over finite field GF(p^{12}) by the number field sieve. In order to correctly estimate the security of the pairing-based cryptography we need some experimental evaluations of number field sieve over an extension field GF(p^{12}).

The number field sieve over an extension field GF(p^n) has a substantially different sieving step from that over a prime field GF(p). There are two sieving algorithms, called the line sieve and lattice sieve [71]. The large-scale implementation of the number field sieve over a prime field GF(p) deploys the lattice sieve of two dimensions, but we have to construct the lattice sieve of more than two dimensions for the number field sieve over an extension field GF(p^{12}). The currently known reports on the multi-dimensional sieving have discussed only the case of dimension three [94, 95].

1.3 Contribution

In this paper, we propose the lattice sieving of more than 2 dimensions for the number field sieve over an extension fields GF(p^n) by naturally extending the lattice sieve of two dimensions. We implemented the proposed multi-dimensional lattice sieve over an extension field GF(p^{12}) of 203 bits, and we show some experimental data for accelerating the number field sieve by choosing the suitable dimensions and sizes of the sieving region. Consequently we have solved the DLP over an extension field GF(p^{12}) of 203 bits by the number field sieve using a PC of 16 CPU cores in about 43 hours.

On the other hand, we extend the Franke-Kleinjung method to a three-dimensional sieve region. We construct an appropriate basis by using the Hermite normal form. An experiment on a GF(p^{12}) of 303 bits indicated that we can enumerate more than 90% of the points generated by special-q in the three-dimensional sieve region. Moreover, we implemented a number field sieve using the three-dimensional lattice sieve. Our implementation of JLSV06-NFS over a GF(p^6) of 240 bits was about as efficient as the current record holder,- a three-dimensional line sieve by Zajac, over GF(p^6).

Chapter 2

The Number Field Sieve over GF(p^n)

In this chapter, we give an overview of the number field sieve over extension field GF(p^n) proposed by Joux et al. at CRYPTO 2006 [41]. The number field sieve consists of four steps: polynomial selection, searching relations, and linear algebra. We explain each step of the number field sieve in the following.

2.1 Discrete Logarithm Problem over GF(p^n)

We denote by GF$(p^n)^*$ the multiplicative group of finite field of cardinality p^n, where p is a prime number and n is an extension degree. Let γ be a generator of GF$(p^n)^*$. The discrete logarithm problem (DLP) over finite field GF(p^n) tries to find the non-negative smallest integer x that satisfies $\gamma^x = \delta$ for a given δ in GF(p^n). This discrete logarithm x is written as $\log_\gamma \delta$ in this paper.

2.2 Polynomial Selection

In the polynomial selection of the number field sieve proposed by Joux et al., we generate two polynomials $f_1, f_2 \in \mathbb{Z}[X]\backslash\{0\}$ that satisfy the following conditions.

1. $f_1 \neq f_2$,
2. $\deg f_1 = n$,
3. f_1 is irreducible in GF(p),
4. $f_1 \mid f_2 \bmod p$.

From the conditions there exists $v \in$ GF(p^n) such that $f_1(v) = f_2(v) = 0$ in GF(p^n). Let α_1 and $\alpha_2 \in \mathbb{C}$ be the root of $f_1(X) = 0$ and $f_2(X) = 0$, respectively. Let $\mathcal{O}_1$ and $\mathcal{O}_2$ be the ring of integers of the number field $\mathbb{Q}(\alpha_1)$ and $\mathbb{Q}(\alpha_2)$, respectively. There are homomorphism maps

$$\begin{aligned} \phi_1 : \mathbb{Z}[\alpha_1] \to \text{GF}(p^n),\ \alpha_1 \mapsto v \\ \phi_2 : \mathbb{Z}[\alpha_2] \to \text{GF}(p^n),\ \alpha_2 \mapsto v. \end{aligned} \tag{2.1}$$

2.3 Searching Relations

In the step of searching relations, we try to find many relations of certain polynomials of degree $t \geq 1$. Let $B_1, B_2 \in \mathbb{R}_{>0}$ be the smoothness bound associated with polynomials f_1, f_2 in Section 2.2. We define the factor base $\mathcal{B}_1, \mathcal{B}_2$ as follows.

$$\mathcal{B}_i = \{(q, g) \mid q : \text{ prime}, q \leq B_i, \text{ and irreducible monic } g \in \mathbb{Z}[X] : g | f_i \bmod q\}$$

In this paper we represent polynomial $h_a(X) = a_0 + a_1 X + \cdots + a_t X^t \in \mathbb{Z}[X]$ as a vector $a = (a_0, a_1, \ldots, a_t)^{\mathrm{T}} \in \mathbb{Z}^{t+1}$. For a given $H = (H_0, H_1, \ldots, H_t) \in \mathbb{R}_{>0}^{t+1}$, we define the $(t+1)$-dimensional region $\mathcal{H}_a(H)$ as

$$\mathcal{H}_a(H) = \{(a_0, a_1, \ldots, a_t)^{\mathrm{T}} \in \mathbb{Z}^{t+1} \mid |a_i| \leq H_i\ (0 \leq i \leq t),\ a_t \geq 0\}.$$

Here H and $\mathcal{H}_a$ are called as the sieving interval and the sieving region, respectively. Next, the norm of $h_a(\alpha_i)$ is defined by $N(h_a(\alpha_i)) = |\mathrm{Res}(h_a, f_i)|$, where $\mathrm{Res}(h_a, f_i)$ is the resultant of $h_a(X)$ and $f_i(X)$ for $i = 1, 2$. In the step of searching relations, for given sieving interval H and smoothness bound B_1, B_2, we try to find $a \in \mathbb{Z}^{t+1}$ (called the hit tuple) that satisfies the following conditions.

1. $N(h_a(\alpha_1))$ is B_1-smooth,
2. $N(h_a(\alpha_2))$ is B_2-smooth,

where B-smooth is the integer whose prime factors are at most B. Denote by S the set of all hit tuples gathered in searching relations. In order to solve the correct discrete logarithm, the size of S is chosen as

$$\sharp S \geq \sharp\mathcal{B}_1 + \sharp\mathcal{B}_2 + 2n. \tag{2.2}$$

2.4 Linear Algebra

Next the hit tuple $a \in S$ has relationship $(h_a(\alpha_i))\mathcal{O}_i = \prod_{\mathfrak{q}\in\mathcal{B}_i} \mathfrak{q}^{\varepsilon_\mathfrak{q}}$ for $i = 1, 2$. We can compute $\varepsilon_\mathfrak{q}$ from the prime decomposition of norm $N(h_a(\alpha_i)) = \prod_{q:\text{prime},q\leq B_i} q^{e_q}$ for $q \nmid [\mathcal{O}_i : \mathbb{Z}[\alpha_i]]$. Let r_i be the torsion-free rank of $\mathcal{O}_i$ for $i = 1, 2$. From $\phi_1(h_a(\alpha_1)) = \phi_2(h_a(\alpha_2))$ using homomorphism map (2.1), we obtain the following relation of the discrete logarithm

$$\sum_{\mathfrak{q}\in\mathcal{B}_1} \varepsilon_\mathfrak{q} \log \phi_1(\mathfrak{q}) + \sum_{j=1}^{r_1} \lambda_j(h_a(\alpha_1)) \log \Lambda_{1,j} \equiv$$
$$\sum_{\mathfrak{q}\in\mathcal{B}_2} \varepsilon_\mathfrak{q} \log \phi_2(\mathfrak{q}) + \sum_{j=1}^{r_2} \lambda_j(h_a(\alpha_2)) \log \Lambda_{2,j} \pmod{p^n - 1},$$

where $\log \mathfrak{q}$ and $\log \Lambda_{i,j}$ are called the virtual logarithms [37, 77] and $\lambda_j(h_a(\alpha_i))$ is the character map proposed by Schirokauer [74]. Consequently, we can compute

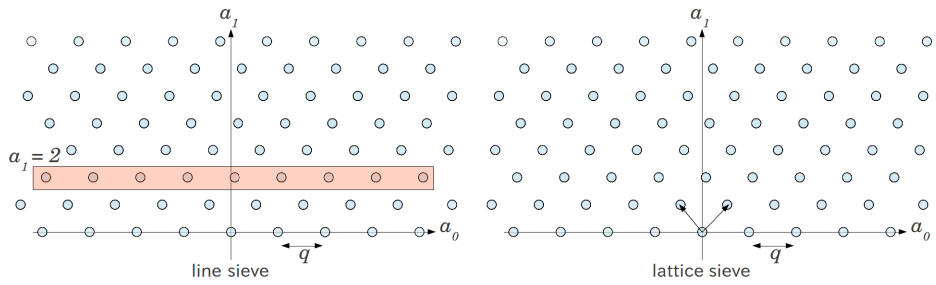

Figure 2.1: Line sieve and lattice sieve in two dimensions

$\log \mathfrak{q}, \log \Lambda_{i,j} \pmod{p^n - 1}$ by solving the linear algorithm obtained from the relations.

2.5 Searching Relations by Sieving in Multi Dimensions

In this section we discuss how to search the hit tuple in the sieving region of $t+1$ dimensions from Section 2.3.

Sieving methods try to find elements $(a_0, a_1, \ldots, a_t)$ in the sieving region $\mathcal{H}_a$ whose norm is divisible by prime number q smaller than the smoothness bound. There are two different sieving methods, called the line sieve and lattice sieve [71, 27]. The line sieve searches elements $(a_0, a_1, \ldots, a_t)$ by repeatedly adding a_0 to q for fixed $(a_1, \ldots, a_t)$. The lattice sieve generates a lattice of elements whose norm is divisible by q, and then finds elements $a \in \mathcal{H}_a$ whose norm is divisible by prime r in the lattice.

The number field sieve for solving the DLP over prime field GF(p) [74, 37] or for factoring integers [50] utilizes the lattice sieve of two dimensions, namely $t = 1$. Fig. 2.1. shows a figure of the line sieve and lattice sieve in two dimen-

sions. Currently the lattice sieve is often used for solving the DLP or factorization problem by the number field sieve of the size of more than 500 bits.

On the other hands, the lattice sieve of two dimensions can not efficiently accumulate sufficient number of smooth elements for the number field sieve for solving the DLP over extension field GF(p^{12}), and thus we have to extends the sieving region to more than two dimensions. Zajac presented an implementation of the line sieve of three dimensions [94, 95], but there is no report on the implementation of the lattice sieve of more than dimension two.

2.5.1 Line Sieve in Multi Dimensions

In the following we describe the line sieve presented by Zajac [95]. If $q \mid N(h_a(\alpha_i))$ holds for a prime $q < B_i$ and $i = 1, 2$, then $q \mid N(h(\alpha_i))$ satisfies for polynomials $h(X) = h_a(X) + kq$ where k is any integer. From this fact, we can search a hit tuple a divisible by q in the sieving region without performing the division of integers. Similarly, for $\mathfrak{q} = (q, g) \in \mathcal{B}_i\,(i = 1, 2)$, we have relationship

$$g \mid h_a \bmod q \;\Rightarrow\; q^{\deg g} \mid N(h_a(\alpha_i)). \tag{2.3}$$

Then we can find a candidate of hit tuple $a \in \mathcal{H}_a$ whose norm $N(h_a(\alpha_i))$ is B_i-smooth by repeatedly adding $L[a]$ to $\deg g \log q$ for the elements which satisfy the left-hand side of equation (2.3) for $\forall \mathfrak{q} \in \mathcal{B}_i\,(i = 1, 2)$. The candidate a is confirmed by checking $N(h_a(\alpha_i))$ is B_i-smooth using the trial division, and then we obtain a hit tuple $a \in \mathcal{H}_a$.

Let I_d be an identity matrix of size $d \times d$. The set of all polynomials in $\mathbb{Z}[X]$ of degree less than $t + 1$ that satisfy the left-hand size of equation (2.3) is generated by the integer linear combination of the columns of the following matrix:

$$\left(\begin{array}{c|ccc} & g_0 & & 0 \\ qI_{\deg g} & g_1 & \ddots & \\ & \vdots & \ddots & g_0 \\ \hline & g_{\deg g} & & g_1 \\ 0 & & \ddots & \vdots \\ & 0 & & g_{\deg g} \end{array}\right), \tag{2.4}$$

where $g_0, g_1, \ldots, g_{\deg g}$ is the coefficient of the polynomial $g = \sum_{j=0}^{\deg g} g_j X^j$, respectively.

From $g_{\deg g} = 1$, we can convert the $(\deg g+1)$-th column to $(t+1)$-th column of this matrix (2.4) by the integer linear combination of columns as follows.

$$M_{\mathfrak{q}} = \left(\begin{array}{c|c} qI_{\deg g} & T_{\mathfrak{q}} \\ \hline 0 & I_{t-\deg g+1} \end{array}\right), \tag{2.5}$$

where $T_{\mathfrak{q}}$ is a $\deg g \times \deg g$ integer matrix. Conversely, for any $c = (c_0, c_1, \ldots, c_t)^{\mathrm{T}} \in \mathbb{Z}^{t+1}$ the relation $a = M_{\mathfrak{q}}\, c$ satisfies the left-hand side of equation (2.3). Therefore, for $\deg g \times (t - \deg g + 1)$ matrix $T_{\mathfrak{q}}$ and $c \in \mathbb{Z}^{t+1}$, we can represent $M_{\mathfrak{q}}$ as follows.

$$\begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_{\deg g-1} \end{pmatrix} = q \begin{pmatrix} c_0 \\ c_1 \\ \vdots \\ c_{\deg g-1} \end{pmatrix} + T_{\mathfrak{q}} \begin{pmatrix} a_{\deg g} \\ a_{\deg g+1} \\ \vdots \\ a_t \end{pmatrix}. \tag{2.6}$$

Here set $(u_0, \ldots, u_{\deg g-1}) = T_{\mathfrak{q}}\ (a_{\deg g}, \ldots, a_t)^{\mathrm{T}}$ for the input $a_{\deg g}, \ldots, a_t$. Then we can search a that satisfies the left-hand size of equation (2.3) by repeatedly adding $u_0, \ldots, u_{\deg g-1}$ to q in sieving region $\mathcal{H}_a$ for $(u_0, \ldots, u_{\deg g-1}, a_{\deg g}, \ldots, a_t)$.

2.5.2 Lattice Sieve in Multi Dimensions

In this section we propose a lattice sieve in the sieving region of multi dimensions by extending the lattice sieve of two dimension used for the number field sieve over prime field GF(p) [71, 6].

The lattice sieve tries to find a candidate of hit tuples in the lattice whose elements are divisible by $\mathfrak{q} \in \mathcal{B}_i$ (called special-$\mathfrak{q}$). For special-$\mathfrak{q}$, $\mathfrak{q} = (q, g) \in \mathcal{B}_i$, let $M_{\mathfrak{q}}$ be the matrix of equation (2.5), and let $M_{\mathfrak{q}}^{\mathrm{LLL}}$ be the matrix generated by LLL reduction algorithm [51] from $M_{\mathfrak{q}}$.

In this paper we call the search space of $t+1$ dimensions for hit tuple $a \in \mathcal{H}_a$ as the a-space. On the other hand, the $(t+1)$-dimensional lattice $M_{\mathfrak{q}}^{\mathrm{LLL}}$, which is generated by $M_{\mathfrak{q}}^{\mathrm{LLL}}\, c$ for $c \in \mathbb{Z}^{t+1}$, is called as the c-space. Moreover, for the sieving interval $H_c \in \mathbb{R}_{>0}$, we define the sieving region over the c-space by

$$\mathcal{H}_c(H_c) = \{(c_0, c_1, \ldots, c_t)^{\mathrm{T}} \in \mathbb{Z}^{t+1} \mid |c_i| \leq H_c (0 \leq i \leq t), c_t \geq 0\}.$$

The lattice sieve for the special-$\mathfrak{q}$ searches the candidates of the hit tuple in the sieving region $\mathcal{H}_c$ in the c-space.

Next we construct the matrix $M_{\mathfrak{r}}$ from the element $\mathfrak{r} = (r, h) \in \mathcal{B}_i$ that is different from $\mathfrak{q}$ in the factor base. By the same method for generating $M_{\mathfrak{q}}$ from $\mathfrak{q}$, we can obtain equation (2.6) corresponding to $M_{\mathfrak{r}}$, and by reducing vector

$r(c_0, \ldots, c_{\deg h-1})^{\mathrm{T}}$ modulo r the next equation yields

$$\begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_{\deg h-1} \end{pmatrix} \equiv T_{\mathfrak{r}} \begin{pmatrix} a_{\deg h} \\ a_{\deg h+1} \\ \vdots \\ a_t \end{pmatrix} \pmod{r}. \tag{2.7}$$

Here we decompose the $(t+1) \times (t+1)$ matrix $M_{\mathfrak{q}}^{\mathrm{LLL}}$ into the $\deg h \times (t+1)$ matrix $M_{\mathfrak{q},1}^{\mathrm{LLL}}$ and the $(t - \deg h + 1) \times (t+1)$ matrix $M_{\mathfrak{q},2}^{\mathrm{LLL}}$ as follows.

$$M_{\mathfrak{q}}^{\mathrm{LLL}} = \begin{pmatrix} M_{\mathfrak{q},1}^{\mathrm{LLL}} \\ \hline M_{\mathfrak{q},2}^{\mathrm{LLL}} \end{pmatrix}. \tag{2.8}$$

The set of all elements a divisible by $\mathfrak{q}$ is represented by $a = M_{\mathfrak{q}}^{\mathrm{LLL}}\, c$ for $c \in \mathbb{Z}^{t+1}$, namely

$$\begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_{\deg h-1} \end{pmatrix} = M_{\mathfrak{q},1}^{\mathrm{LLL}}\, c, \quad \begin{pmatrix} a_{\deg h} \\ a_{\deg h+1} \\ \vdots \\ a_t \end{pmatrix} = M_{\mathfrak{q},2}^{\mathrm{LLL}}\, c. \tag{2.9}$$

Therefore, from equations (2.7) and (2.9) we obtain

$$(M_{\mathfrak{q},1}^{\mathrm{LLL}} - T_{\mathfrak{r}}\ M_{\mathfrak{q},2}^{\mathrm{LLL}})\, c \equiv 0 \pmod{r}. \tag{2.10}$$

Next let $M_{\mathfrak{q},\mathfrak{r}}$ be the lattice generated by c from equation (2.10), namely $M_{\mathfrak{q},\mathfrak{r}}$ is the kernel of linear map $(M_{\mathfrak{q},1}^{\mathrm{LLL}} - T_{\mathfrak{r}}\ M_{\mathfrak{q},2}^{\mathrm{LLL}})$. Note that $a = M_{\mathfrak{q}}^{\mathrm{LLL}} M_{\mathfrak{q},\mathfrak{r}}\ e$ for any $e = (e_0, e_1, \ldots, e_t) \in \mathbb{Z}^{t+1}$ satisfies the left-hand size of equation (2.3) for both $\mathfrak{q}$ and $\mathfrak{r}$. We can compute $M_{\mathfrak{q},\mathfrak{r}}$ from the matrix $M_{\mathfrak{q},1}^{\mathrm{LLL}} - T_{\mathfrak{r}}\ M_{\mathfrak{q},2}^{\mathrm{LLL}}$ corresponding to equation (2.10).

From the above observations, we present the proposed lattice sieve for the special-$\mathfrak{q}$, $\mathfrak{q}$ as Algorithm 1.

Algorithm 1 Proposed Lattice Sieve in Multi Dimensions

Input: special-$\mathfrak{q}$ where $\mathcal{Q} \subset \mathcal{B}_j$, factor base $\mathcal{B}_1, \mathcal{B}_2$, sieving region $\mathcal{H}_c(H_c)$.

Output: S is the set of candidates of hit tuples.

$k \leftarrow (j \bmod 2) + 1$.
for all $\mathfrak{q} = (q, g) \in \mathcal{Q}$ **do**
  Compute matrix $M_{\mathfrak{q}}^{\mathrm{LLL}}$ from $\mathfrak{q}$ using LLL.
  $L[c] \leftarrow 0$ for $\forall c \in \mathcal{H}_c$.
  $D[c] \leftarrow \log N(h_a(\alpha_j))$ where $a = M_{\mathfrak{q}}^{\mathrm{LLL}} c$ for $\forall c \in \mathcal{H}_c$.
  for all $\mathfrak{r} = (r, h) \in \mathcal{B}_j$ s.t. $r < q$ **do**
    Compute lattice $M_{\mathfrak{q},\mathfrak{r}}$ of $\mathfrak{r}$ over c-space from $\mathfrak{q}$.
    $L[c] \leftarrow L[c] + \deg h \log r$ s.t. $c \in \mathcal{H}_c, c \in M_{\mathfrak{q},\mathfrak{r}}$.
  end for
  for all $c \in \mathcal{H}_c$ **do**
    if $L[c] + \deg g \log q - D[c]$ is small **then**
      $L[c] \leftarrow 0$
    else
      $L[c] \leftarrow -\infty$
    end if
  end for
  $D[c] \leftarrow \log N(h_a(\alpha_k))$ where $a = M_{\mathfrak{q}}^{\mathrm{LLL}} c$ for $\forall c \in \mathcal{H}_c$.
  for all $\mathfrak{r} = (r, h) \in \mathcal{B}_k$ **do**
    Compute lattice $M_{\mathfrak{q},\mathfrak{r}}$ of $\mathfrak{r}$ over c-space from $\mathfrak{q}$.
    $L[c] \leftarrow L[c] + \deg h \log r$ s.t. $c \in \mathcal{H}_c, c \in M_{\mathfrak{q},\mathfrak{r}}$.
  end for
  for all $c \in \mathcal{H}_c$ s.t. $L[c] - D[c]$ is small **do**
    $S \leftarrow S \cup M_{\mathfrak{q}}^{\mathrm{LLL}} c$
  end for
end for
return S

Chapter 3

The 3-dimensional Lattice Sieve

3.1 The Franke-Kleinjung method in 2 dimensions

In this section, we explain how to efficiently enumerate the points in the two-dimensional lattice proposed by Franke and Kleinjung [27].

Let $L^2_{\mathfrak{q},\mathfrak{r}}$ be the two-dimensional lattice generated by $M^2_{\mathfrak{q},\mathfrak{r}}$ defined in Section 2.5.2 for the case of 2 dimension i.e. $t = 1$. Let $\mathbf{u} = (u_0, u_1)^{\mathrm{T}}, \mathbf{v} = (v_0, v_1)^{\mathrm{T}}$ be the basis of $L^2_{\mathfrak{q},\mathfrak{r}}$. Let $\mathcal{H}^2_c$ be the sieve region such that

$$\mathcal{H}^2_c = \{(c_0, c_1)^{\mathrm{T}} \in \mathbb{Z}^2 \mid -I/2 \leq c_0 < I/2,\ 0 \leq c_1 < J\},$$

where $I, J \in \mathbb{Z}_{>0}$ and I is even.

The Franke-Kleinjung method enumerates the points in sieve region $\mathcal{H}^2_c$ by a special basis $\mathbf{u}, \mathbf{v}$ of lattice $L^2_{\mathfrak{q},\mathfrak{r}}$, which has the following good properties (See Figure 4.1 for an example: $\mathbf{v} = (27, 1)^{\mathrm{T}}$ and $\mathbf{u} = (-47, 2)^{\mathrm{T}}$ with $I = 64$). (1) We can exhaustively compute all the points in $\mathcal{H}^2_c \cap L^2_{\mathfrak{q},\mathfrak{r}}$ by adding vector $\mathbf{u}$, $\mathbf{v}$, or $\mathbf{u}+\mathbf{v}$ recursively. (2) The second coordinate of the points in sieve region $\mathcal{H}^2_c \cap L^2_{\mathfrak{q},\mathfrak{r}}$ generated by the enumeration algorithm is monotonically increasing. Indeed we

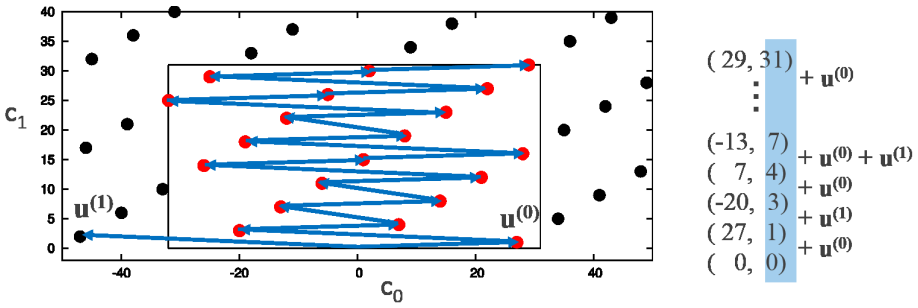

Figure 3.1: An example of an enumeration of lattice points with the generated basis $\mathbf{u}, \mathbf{v}$ by the Franke-Kleinjung method on 2 dimensions

have the following theorem. We also show a proof which will be extended to the case of 3 dimensions in the proposed method in Section 3.2.

Theorem 3.1.1 ((Franke-Kleinjung [27])) *We assume that the basis* $\mathbf{u} = (u_0, u_1)^{\mathrm{T}}, \mathbf{v} = (v_0, v_1)^{\mathrm{T}}$ *of lattice* $L^2_{\mathfrak{q},\mathfrak{r}}$ *satisfies the following conditions:*

A1: $|u_0| < I$ *and* $|v_0| < I$, A2: $|u_0 - v_0| \geq I$, A3: $u_1 > 0$ *and* $v_1 > 0$.

Let $\mathbf{p} = (p_0, p_1)^{\mathrm{T}}$, $\mathbf{q} = (q_0, q_1)^{\mathrm{T}}$ *be points in* $\mathcal{H}^2_c \cap L^2_{\mathfrak{q},\mathfrak{r}}$. *If* $q_1 > p_1$ *holds, then* $\mathbf{q} = \mathbf{p} + i\mathbf{u} + j\mathbf{v}$ *satisfies* $i \geq 0, j \geq 0$ *and* $i + j \neq 0$.

Proof 3.1.1 *From* $\mathbf{p}, \mathbf{q} \in L^2_{\mathfrak{q},\mathfrak{r}}$, *then we have* $\mathbf{q} = \mathbf{p} + i\mathbf{u} + j\mathbf{v}$, $q_0 = p_0 + iu_0 + jv_0$ *and* $q_1 = p_1 + iu_1 + jv_1$.

At first, both $i = 0$ *and* $j = 0$ *can not be satisfied due to* $q_1 > p_1$. *If* $i = 0$ *and* $j \neq 0$ *hold, then we obtain* $0 < q_1 - p_1 = jv_1$ *and thus* $j > 0$ *from Condition A3. The assertion of the theorem is derived. Similarly, if* $i \neq 0$ *and* $j = 0$ *hold, then the theorem holds.*

Next, we assume $i \neq 0$ *and* $j \neq 0$. *We will prove that* $i > 0$ *and* $j > 0$ *hold under the assumption of* $q_1 > p_1$ *and* $\mathbf{p}, \mathbf{q} \in \mathcal{H}^2_c$, *i.e.,* $-I/2 \leq p_0 < I/2$ *and* $-I/2 \leq q_0 < I/2$. *At first, if* $i < 0$ *and* $j < 0$ *hold, then we have* $iu_1 + jv_1 < 0$

from Condition A3. However, it contradicts from the assumption of $q_1 > p_1$ *due to* $q_1 - p_1 = iu_1 + jv_1$. *Next, we consider the case that* i *and* j *have the different sign. Note that if* u_0 *and* v_0 *satisfy Conditions A1 and A2, then* $u_0 v_0 < 0$ *holds. From* $u_0 v_0 < 0$, *we know that* iu_0 *and* jv_0 *have the same sign and* $|u_0| + |v_0| = |u_0 - v_0|$. *Then we obtain* $|q_0 - p_0| = |iu_0 + jv_0| = |iu_0| + |jv_0| \geq |u_0| + |v_0| = |u_0 - v_0| \geq I$ *from Condition A2. However, it contradicts* $|q_0 - p_0| < I$ *from the assumption of* $\mathbf{p}, \mathbf{q} \in \mathcal{H}_c^2$.

In the following, we denote by $M_{\mathfrak{q},\mathfrak{r}}^{\mathrm{FK2}}$ the basis $(\mathbf{u}, \mathbf{v})$ that satisfies Conditions A1, A2 and A3 in Theorem 3.1.1. Franke and Kleinjung showed that the basis that satisfies Conditions A1, A2 and A3 in Theorem 3.1.1 can be generated by the continued fraction method shown in Algorithm 2.

From Theorem 3.1.1 Franke-Kleinjung proved the following theorem [27]. We also show the proof which is extended to the case of 3-dimensions.

Theorem 3.1.2 *Let* $\mathbf{u} = (u_0, u_1)^{\mathrm{T}}, \mathbf{v} = (v_0, v_1)^{\mathrm{T}}$ *be the basis of* $M_{\mathfrak{q},\mathfrak{r}}^{FK2}$. *Let* $\mathbf{p} = (p_0, p_1), \mathbf{q} = (q_0, q_1)$ *be points in* $\mathcal{H}_c^2 \cap L_{\mathfrak{q},\mathfrak{r}}^2$. *If* q_1 *is the smallest among all the points whose second coordinate is larger than* p_1, *then* $\mathbf{q}$ *is one of the points* $\mathbf{p} + \mathbf{u}$, $\mathbf{p} + \mathbf{v}$, *or* $\mathbf{p} + \mathbf{u} + \mathbf{v}$.

Proof 3.1.2 *From Theorem 3.1.1, we know that all the points, whose second coordinate is larger than* p_1 *in* $\mathcal{H}_c^2 \cap L_{\mathfrak{q},\mathfrak{r}}^2$, *can be obtained by repeatedly adding* $\mathbf{u}$ *or* $\mathbf{v}$. *Every time we add point* $\mathbf{u}$ *or* $\mathbf{v}$, *then the second coordinate of the resulting point becomes larger from Condition A3. At first note that if* $\mathbf{p} + \mathbf{u}$ *is contained in* $\mathcal{H}_c^2$, *then* $\mathbf{p} + \mathbf{v} \notin \mathcal{H}_c^2$ *holds from Condition A2. Therefore, if* $\mathbf{p} + \mathbf{u} \in \mathcal{H}_c^2$ *holds, then the second coordinate of* $\mathbf{p} + \mathbf{u}$ *is the smallest among all points whose second coordinate is larger than* p_1 *in* $\mathcal{H}_c^2 \cap L_{\mathfrak{q},\mathfrak{r}}^2$. *Similarly, we can prove the case of* $\mathbf{p} + \mathbf{v} \in \mathcal{H}_c^2$. *Finally, if both* $\mathbf{p} + \mathbf{v}$ *and* $\mathbf{p} + \mathbf{u}$ *are not contained in* $\mathcal{H}_c^2$, *then*

Algorithm 2 Generation of basis $M_{\mathfrak{q},\mathfrak{r}}^{\mathrm{FK2}}$ of Franke-Kleinjung method

Input: bound of the lattice region I, $M_{\mathfrak{q},\mathfrak{r}}^2 = (\mathbf{u}, \mathbf{v}) = ((u_0, u_1)^{\mathrm{T}}, (v_0, v_1)^{\mathrm{T}}) = ((r, 0)^{\mathrm{T}}, (z, 1)^{\mathrm{T}})$, where $r > I$ and $0 < z < r$ (Case 1 of HNF in Section 3.1)

Output: $M_{\mathfrak{q},\mathfrak{r}}^{\mathrm{FK2}}$ that satisfies Conditions A1, A2 and A3 in Theorem 3.1.1

1: $\mathbf{v} \leftarrow \mathbf{v} - \mathbf{u}$
2: **while** $|v_0| \geq I$ **do**
3: $\quad \mathbf{u} \leftarrow \mathbf{u} + a\mathbf{v},\ a = \lfloor -u_0/v_0 \rfloor$
4: $\quad$ SWAP$(\mathbf{u}, \mathbf{v})$
5: $a \leftarrow \lfloor (|u_0| - I)/|v_0| \rfloor + 1$ /* a is the least positive integer s.t. $|u_0 + av_0| < I$ */
6: $\mathbf{u} \leftarrow \mathbf{u} + a\mathbf{v}$
7: **return** $M_{\mathfrak{q},\mathfrak{r}}^{\mathrm{FK2}} = (\mathbf{u}, \mathbf{v})$

$\mathbf{p} + \mathbf{u} + \mathbf{v} \in \mathcal{H}_c^2$ *from Condition A2. Therefore either* $\mathbf{p} + \mathbf{u}$*,* $\mathbf{p} + \mathbf{v}$*, or* $\mathbf{p} + \mathbf{u} + \mathbf{v}$ *is contained in* $\mathcal{H}_c^2 \cap L_{\mathfrak{q},\mathfrak{r}}^2$.

From this monotonically increasing property, we can enumerate all the points in $\mathcal{H}_c^2 \cap L_{\mathfrak{q},\mathfrak{r}}^2$ by Algorithm 3.

We stress that the Franke-Kleinjung method for $M_{\mathfrak{q},\mathfrak{r}}^2$ is not required in some cases. The Hermite normal form (HNF) of the basis of $M_{\mathfrak{q},\mathfrak{r}}^2$ in Section 2.5.2 becomes one of the following:

$$\text{Cases } 1: \begin{pmatrix} r & z \\ 0 & 1 \end{pmatrix},\ 2: \begin{pmatrix} r & 0 \\ 0 & 1 \end{pmatrix},\ 3: \begin{pmatrix} 1 & 0 \\ 0 & r \end{pmatrix}$$

where $z \in \mathbb{Z}_{>0}, z < r$. The basis is orthogonal in Cases 2 and 3, and thus we can use the line sieve on $\mathbf{c}$-space. Therefore we only deal with Case 1 where $r > I$ for the lattices sieve.

Algorithm 3 NEXTFK2($I, M^{\mathrm{FK2}}_{\mathfrak{q},\mathfrak{r}}$, $\mathbf{p}$)

Input: bound of the lattice region I, $M^{\mathrm{FK2}}_{\mathfrak{q},\mathfrak{r}} = (\mathbf{u}, \mathbf{v}) = ((u_0, u_1)^{\mathrm{T}}, (v_0, v_1)^{\mathrm{T}})$, where $u_0 < 0$, point $\mathbf{p} = (p_0, p_1)^{\mathrm{T}} \in L^2_{\mathfrak{q},\mathfrak{r}} \cap \mathcal{H}^2_c$

Output: point $\mathbf{q} = (q_0, q_1)$ s.t. $\mathbf{q} \in L^2_{\mathfrak{q},\mathfrak{r}} \cap \mathcal{H}^2_c$ and $q_1 > p_1$ and $q_1 - p_1$ is the least

1: **if** $-I/2 \leq p_0 + u_0$ **then return** $\mathbf{p} + \mathbf{u}$
2: **if** $p_0 + v_0 < I/2$ **then return** $\mathbf{p} + \mathbf{v}$
3: **return** $\mathbf{p} + \mathbf{u} + \mathbf{v}$

3.2 Proposed extension of the Franke-Kleinjung method to 3 dimensions

In this section, we extend the Franke-Kleinjung method of 2 dimensions in Section 3.1 to that of 3 dimensions. First we give a classification of matrix $M^3_{\mathfrak{q},\mathfrak{r}}$ by Hermite normal form. We then explain the conditions for the proposed basis in 3 dimensions and how to generate such a basis in analogue with Section 3.1. Finally, we present an enumeration algorithm using the proposed basis of the 3-dimensional lattice.

3.2.1 Hermite normal form of $M^3_{\mathfrak{q},\mathfrak{r}}$

Let $L^3_{\mathfrak{q},\mathfrak{r}}$ be the the 3-dimensional lattice generated by the basis $M^3_{\mathfrak{q},\mathfrak{r}}$ of size 3×3 in Section 2.5.2 for the case of $t = 2$. We classify the HNF matrix $M^3_{\mathfrak{q},\mathfrak{r}}$ to exclude

some trivial cases. The HNF of matrix $M^3_{\mathfrak{q},\mathfrak{r}}$ becomes one of the following:

$$\text{Cases } \mathbf{1}: \begin{pmatrix} r & z_1 & z_2 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}, 2: \begin{pmatrix} r & z_1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}, 3: \begin{pmatrix} r & 0 & z_2 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}, 4: \begin{pmatrix} r & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix},$$
$$5: \begin{pmatrix} 1 & 0 & 0 \\ 0 & r & z_2 \\ 0 & 0 & 1 \end{pmatrix}, 6: \begin{pmatrix} 1 & 0 & 0 \\ 0 & r & 0 \\ 0 & 0 & 1 \end{pmatrix}, 7: \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & r \end{pmatrix},$$
$$\mathbf{8}: \begin{pmatrix} r & 0 & z_1 \\ 0 & r & z_2 \\ 0 & 0 & 1 \end{pmatrix}, 9: \begin{pmatrix} r & 0 & 0 \\ 0 & r & z_2 \\ 0 & 0 & 1 \end{pmatrix}, 10: \begin{pmatrix} r & 0 & z_1 \\ 0 & r & 0 \\ 0 & 0 & 1 \end{pmatrix}, 11: \begin{pmatrix} r & 0 & 0 \\ 0 & r & 0 \\ 0 & 0 & 1 \end{pmatrix},$$
$$12: \begin{pmatrix} r & z_1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & r \end{pmatrix}, 13: \begin{pmatrix} r & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & r \end{pmatrix}, 14: \begin{pmatrix} 1 & 0 & 0 \\ 0 & r & 0 \\ 0 & 0 & r \end{pmatrix}.$$

where $z_1, z_2 \in \mathbb{Z}_{>0}, z_1, z_2 < r$.

The basis is orthogonal in Cases 4, 6, 7, 11, 13 and 14, and thus we can efficiently use the line sieve on c-space. Moreover, Cases 2, 3, 5, 9, 10, and 12 contain an orthogonal subspace spanned by the 2-dimensional basis of the Franke-Kleinjung type which are colored by gray. We use the line sieve on the non-colored vector and the 2-dimensional Franke-Kleinjung method for its orthogonal projection. Consequently, we have to consider an HNF matrix $M^3_{\mathfrak{q},\mathfrak{r}}$ that corresponds to one of the Case 1 and 8 in the following.

3.2.2 The proposed conditions for $M^3_{\mathfrak{q},\mathfrak{r}}$

In this section we extend the conditions of Theorem 3.1.1 used in the Franke-Kleinjung method to the lattice of 3 dimensions, and then present how to generate the proposed basis.

Let $\mathcal{H}^3_c$ be the sieve region in $\mathbb{Z}^3$ such that

$$\mathcal{H}^3_c = \{(c_0, c_1, c_2)^{\mathrm{T}} \in \mathbb{Z}^3 \mid -I/2 \le c_i < I/2 \ (i = 0, 1),\ 0 \le c_1 < J\},$$

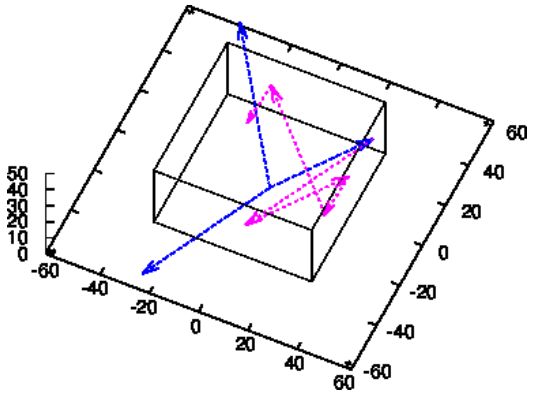

Figure 3.2: An example of an enumeration of lattice points with the generated basis $\mathbf{u}^{(0)}, \mathbf{u}^{(1)}$ and $\mathbf{u}^{(2)}$ by the proposed Franke-Kleinjung method on 3 dimensions

where $I, J \in \mathbb{Z}_{>0}$ is a bound of lattice region and I is even. Our proposed enumeration algorithm can generate all the points in sieve region $\mathcal{H}_c^3$ if we generate an appropriate basis $\mathbf{u}^{(0)} = (u_0^{(0)}, u_1^{(0)}, u_2^{(0)})^{\mathrm{T}}$, $\mathbf{u}^{(1)} = (u_0^{(1)}, u_1^{(1)}, u_2^{(1)})^{\mathrm{T}}$ and $\mathbf{u}^{(2)} = (u_0^{(2)}, u_1^{(2)}, u_2^{(2)})^{\mathrm{T}}$ of lattice $L_{\mathfrak{q},\mathfrak{r}}^3$ with the following properties (See Figure 3.2 for an example: $\mathbf{u}^{(0)} = (22, 39, 1)^{\mathrm{T}}$, $\mathbf{u}^{(1)} = (-63, -12, 7)^{\mathrm{T}}$, and $\mathbf{u}^{(2)} = (45, -49, 11)^{\mathrm{T}}$ with $I = 64$). (1) We can exhaustively compute all the points in $\mathcal{H}_c^3 \cap L_{\mathfrak{q},\mathfrak{r}}^3$ by adding the linear combination of $\mathbf{u}^{(0)}$, $\mathbf{u}^{(1)}$ and $\mathbf{u}^{(2)}$, recursively. (2) The third coordinate of the points in sieve region $\mathcal{H}_c^3 \cap L_{\mathfrak{q},\mathfrak{r}}^3$ generated by the enumeration algorithm is monotonically increasing. Indeed we can prove the following theorem.

Theorem 3.2.1 *We assume that the basis* $\mathbf{u}^{(0)} = (u_0^{(0)}, u_1^{(0)}, u_2^{(0)})^{\mathrm{T}}$, $\mathbf{u}^{(1)} = (u_0^{(1)}, u_1^{(1)}, u_2^{(1)})^{\mathrm{T}}$, $\mathbf{u}^{(2)} = (u_0^{(2)}, u_1^{(2)}, u_2^{(2)})^{\mathrm{T}}$ *of lattice* $L_{\mathfrak{q},\mathfrak{r}}^3$ *satisfies the following conditions:*

B1: $|u_0^{(a)}| < I$ *and* $|u_1^{(a)}| < I$ *for* $a = 0, 1, 2$,

B2: $|u_0^{(b)} - u_0^{(c)}| \geq I$ *or* $|u_1^{(b)} - u_1^{(c)}| \geq I$ *for all pairwise differences* $(b, c) \in \{0, 1, 2\}^2$,

B3: $u_2^{(0)} \geq 0$ *and* $u_2^{(1)} \geq 0$ *and* $u_2^{(2)} \geq 0$ *and* $u_2^{(0)} + u_2^{(1)} + u_2^{(2)} \neq 0$.

B4: $|i_0u_0^{(0)} + i_1u_0^{(1)} + i_2u_0^{(2)}| \geq I$ *or* $|i_0u_1^{(0)} + i_1u_1^{(1)} + i_2u_1^{(2)}| \geq I$*, if one of* $i_0, i_1, i_2 \in \mathbb{Z}$ *is*

negative and the others are positive or equal to 0*.*

Let $\mathbf{p} = (p_0, p_1, p_2)^{\mathrm{T}}$, $\mathbf{q} = (q_0, q_1, q_2)^{\mathrm{T}}$ *be points in* $\mathcal{H}_c^3 \cap L_{\mathfrak{q},\mathfrak{r}}^3$ *with* $\mathbf{p} \neq \mathbf{q}$*. If* $q_2 \geq p_2$ *holds, then* $\mathbf{q} = \mathbf{p} + j_0\mathbf{u}^{(0)} + j_1\mathbf{u}^{(1)} + j_2\mathbf{u}^{(2)}$ *satisfies* $(j_0 \geq 0, j_1 \geq 0, j_2 \geq 0$ *and* $j_0 + j_1 + j_2 \neq 0)$ *or* $(j_0 \leq 0, j_1 \leq 0, j_2 \leq 0$ *and* $j_0 + j_1 + j_2 \neq 0)$*.*

Proof 3.2.1 *From* $\mathbf{p}, \mathbf{q} \in L_{\mathfrak{q},\mathfrak{r}}^3$*, then we have relation* $\mathbf{q} = \mathbf{p} + j_0\mathbf{u} + j_1\mathbf{v} + j_2\mathbf{w}$*,* $q_k = p_k + j_0u_k^{(0)} + j_1u_k^{(1)} + j_2u_k^{(2)}$ *for* $j_0, j_1, j_2 \in \mathbb{Z}$ *and* $k = 0, 1, 2$*. Moreover, from* $-I/2 \leq p_k < I/2$ *and* $-I/2 \leq q_k < I/2$ *for* $k = 0, 1$*, then we have* $|q_k - p_k| < I$ *for* $k = 0, 1$*.*

First of the proof, we prove the theorem in the cases of $q_2 > p_2$*. We first consider the case that some coefficients* j_0, j_1, j_2 *are equal to zero. At first* $j_0 = j_1 = j_2 = 0$ *can not be satisfied due to* $q_2 > p_2$*. If* $j_0 = 0$*,* $j_1 = 0$ *and* $j_2 \neq 0$ *hold, then we obtain* $0 < q_2 - p_2 = j_2u_2^{(2)}$ *and thus* $j_2 > 0$ *from Condition B3. The assertion of the theorem is derived. Similarly, two of coefficients* j_0, j_1, j_2 *are zero, the theorem holds. If we assume that one of coefficients* j_0, j_1, j_2 *is zero. In the case of* $j_0 \neq 0$*,* $j_1 \neq 0$ *and* $j_2 = 0$*. From Conditions B1 and B2. there exists* $k \in \{0, 1\}$ *s.t.* $u_kv_k < 0$*. For such* k*, if* j_0 *and* j_1 *have different sign, we have* $|j_0u_k - j_1v_k| > I$ *in the same manner of Theorem 3.1.1. Then, it contradicts* $|q_k - p_k| < I$ *for* $k = 0, 1$*. On the other hand, if* $j_0 < 0$ *and* $j_1 < 0$ *holds, we have* $j_0u_2 + j_1v_2 \leq 0$ *from Condition B3. Then, it contradicts* $q_2 > p_2$*. Similarly, we can prove that the theorem holds in the case of* $(j_0 \neq 0, j_1 = 0, j_2 \neq 0)$ *or* $(j_0 = 0, j_1 \neq 0, j_2 \neq 0)$*. Next, we consider the case of* $j_0 \neq 0, j_1 \neq 0,$ *and* $j_2 \neq 0$*. We will prove that* $j_0 > 0, j_1 > 0$ *and* $j_2 > 0$ *hold under the assumption of Condition B4,* $q_2 > p_2$*, and* $\mathbf{p}, \mathbf{q} \in \mathcal{H}_c^3$*, i.e.,* $-I/2 \leq p_k < I/2$ *and* $-I/2 \leq q_k < I/2$ *for* $k = 0, 1$*. Recall that Condition B4 assume that* $|u_0| \geq I$ *or* $|u_1| \geq I$ *for* $(u_0, u_1, u_2)^{\mathrm{T}} = i_0\mathbf{u}^{(0)} + i_1\mathbf{u}^{(1)} - i_2\mathbf{u}^{(2)}$ $(i_0, i_1, i_2 \in \mathbb{Z}_{>0})$*. At first,*

if $j_0 < 0, j_1 < 0$ *and* $j_2 < 0$ *hold, then we have* $j_0 u_2^{(0)} + j_1 u_2^{(1)} + j_2 u_2^{(2)} < 0$ *from Condition B3. However, it contradicts from the assumption of* $q_2 > p_2$ *due to* $q_2 - p_2 = j_0 u_2^{(0)} + j_1 u_2^{(1)} + j_2 u_2^{(2)}$*. Next, we assume that one of* j_0, j_1, j_2 *is negative. Here we show the case of* $j_0 < 0, j_1 > 0$ *and* $j_2 > 0$ *(the other cases can be obtained similarly). From Condition B4, we know that* $|j_0 u_0^{(0)} + j_1 u_0^{(1)} + j_2 u_0^{(2)}| \geq I$ *or* $|j_0 u_1^{(0)} + j_1 u_1^{(1)} + j_2 u_1^{(2)}| \geq I$ *holds. However, it contradicts* $|q_0 - p_0| < I$ *and* $|q_1 - p_1| < I$ *from the assumption of* $\mathbf{p}, \mathbf{q} \in \mathcal{H}_c^3$*. Finally, if one of* j_0, j_1, j_2 *is negative, then we can show a contradiction using Condition B4 in the same manner.*

In the following, we prove the case of $q_2 = p_2$*. First, we consider the case that some coefficients* j_0, j_1, j_2 *are equal to zero. From* $\mathbf{p} \neq \mathbf{q}$ *and* $q_2 = p_2$ *we know that* $p_0 \neq q_0$ *or* $p_1 \neq q_1$*. At first* $j_0 = j_1 = j_2 = 0$ *can not be satisfied due to* $q_0 \neq p_0$ *or* $q_1 \neq p_1$*. Second, if two of coefficients* j_0, j_1, j_2 *are zero, the theorem holds. Third, if we assume that one of coefficients* j_0, j_1, j_2 *is zero. We can prove that the theorem in the same manner of the case of* $q_2 > p_2$*. Next, we consider the case of* $j_0 \neq 0, j_1 \neq 0$ *and* $j_2 \neq 0$*. At first we assume that two of* j_0, j_1, j_2 *are negative. Here we show the case of* $j_0 < 0, j_1 > 0$ *and* $j_2 > 0$ *(the other cases can be obtained similarly). From Condition B4, we know that* $|j_0 u_0^{(0)} + j_1 u_0^{(1)} + j_2 u_0^{(2)}| \geq I$ *or* $|j_0 u_1^{(0)} + j_1 u_1^{(1)} + j_2 u_1^{(2)}| \geq I$ *holds. However, it contradicts* $|q_0 - p_0| < I$ *and* $|q_1 - p_1| < I$ *from the assumption of* $\mathbf{p}, \mathbf{q} \in \mathcal{H}_c^3$*. Finally, if one of* j_0, j_1, j_2 *is negative, then we can show a contradiction using Condition B4 in the same manner.*

We propose an algorithm for generating $M_{\mathfrak{q},\mathfrak{r}}^{\mathrm{FK3}}$ that satisfies Conditions B1, B2, B3 and B4. Algorithm 4 presents a procedure to transform $M_{\mathfrak{q},\mathfrak{r}}^{3}$ to $M_{\mathfrak{q},\mathfrak{r}}^{\mathrm{FK3}}$. In Algorithm 4, we first reduce u_0 and u_1-coordinate of $\mathbf{u}^{(0)}, \mathbf{u}^{(1)}$ and $\mathbf{u}^{(2)}$ to satisfy Condition B1 as Steps 1-6, and we adjust the basis in Steps 7 and 8 to satisfy

Algorithm 4 Proposed generation of 3-dimensional basis $M_{\mathfrak{q},\mathfrak{r}}^{\mathrm{FK3}}$

Input: region bound I, integer matrix $M_{\mathfrak{q},\mathfrak{r}}^{3} = (\mathbf{u}^{(0)}, \mathbf{u}^{(1)}, \mathbf{u}^{(2)}) = ((r,0,0)^{\mathrm{T}}, (z_1,1,0)^{\mathrm{T}}, (z_2,0,1)^{\mathrm{T}})$ s.t. $r > I, 0 < z_1 < r$ and $0 < z_2 < r$ (Case 1 of HNF in Section 5.1).

Output: reduced integer matrix $M_{\mathfrak{q},\mathfrak{r}}^{\mathrm{FK3}}$

1: reduce by Algorithm 2 with respect to $\mathbf{u}^{(0)}, \mathbf{u}^{(1)}$.
2: **while** $|u_1^{(2)}| \geq I$ **do**
3: RADIATE($\mathbf{u}^{(0)}, \mathbf{u}^{(1)}, \mathbf{u}^{(2)}$)
4: **if** sign($u_1^{(0)}$) = sign($u_1^{(1)}$) **then do** REDUCE1($\mathbf{u}^{(0)}, \mathbf{u}^{(1)}, \mathbf{u}^{(2)}$)
5: **else do** REDUCE2($\mathbf{u}^{(0)}, \mathbf{u}^{(1)}, \mathbf{u}^{(2)}$)
6: **if** $|u_1^{(0)}| > |u_1^{(1)}|$ **then do** SWAP($\mathbf{u}^{(2)}, \mathbf{u}^{(0)}$) **else do** SWAP($\mathbf{u}^{(2)}, \mathbf{u}^{(1)}$)
7: **if** $\exists a \in \{0,1,2\}$ s.t. $u_2^{(a)} < 0$ **then** $\mathbf{u}^{(a)} \leftarrow -\mathbf{u}^{(a)}$
8: ADJUST($\mathbf{u}^{(0)}, \mathbf{u}^{(1)}, \mathbf{u}^{(2)}$)
9: **return** $M_{\mathfrak{q},\mathfrak{r}}^{\mathrm{FK3}} = (\mathbf{u}^{(0)}, \mathbf{u}^{(1)}, \mathbf{u}^{(2)})$

Conditions B2 and B3. We deal with whether $M_{\mathfrak{q},\mathfrak{r}}^{\mathrm{FK3}}$ generated by Algorithm 4 satisfies Condition B4 in Section 4.2.

In Step 1 of Algorithm 4, we use Algorithm 2 with respect to u_0 and u_1-coordinate of $\mathbf{u}^{(0)}, \mathbf{u}^{(1)}$ and $\mathbf{u}^{(2)}$, then we have $|u_0^{(0)}|, |u_0^{(1)}|, |u_2^{(2)}| < I$. Note that we don't need to care the values $|u_2^{(0)}|$ and $|u_2^{(1)}|$, since $|u_2^{(0)}| = |u_2^{(1)}| = 0$.

In Steps 2-6, we reduce u_1-coordinate of $\mathbf{u}^{(0)}, \mathbf{u}^{(1)}$ and $\mathbf{u}^{(2)}$ with keeping the condition of $|u_0^{(a)}| < I$, where $a = 0, 1$ and 2. At first, Step 3 adjusts $\mathbf{u}^{(0)}, \mathbf{u}^{(1)}$ and $\mathbf{u}^{(2)}$ to reduce $\mathbf{u}^{(2)}$ by adding $\mathbf{u}^{(0)}$ and $\mathbf{u}^{(1)}$ with the subroutine RADIATE. The subroutine RADIATE in Algorithm 7 transforms $\mathbf{u}^{(0)}, \mathbf{u}^{(1)}$ and $\mathbf{u}^{(2)}$ to satisfy that the angle between $\mathbf{x}$ and $\mathbf{y}$ is less than π, where $\mathbf{x}$ and $\mathbf{y}$ are any two of $\mathbf{u}^{(0)}, \mathbf{u}^{(1)}$ and $\mathbf{u}^{(2)}$.

Second, we generate $\mathbf{u}^{(2)}$ s.t. $(u_1^{(2)} \leq u_1^{(0)}) \vee (u_1^{(2)} \leq u_1^{(1)}) \vee (|u_1^{(2)}| < I)$ by adding $\mathbf{u}^{(1)}$ and $\mathbf{u}^{(2)}$ in Steps 4 and 5. If sign($u_1^{(0)}$) = sign($u_1^{(1)}$) holds, we

Algorithm 5 REDUCE1($\mathbf{u}^{(0)}, \mathbf{u}^{(1)}, \mathbf{u}^{(2)}$)

Input: bound of lattice region I, basis $\mathbf{u}^{(0)}, \mathbf{u}^{(1)}, \mathbf{u}^{(2)}$ of the lattice generated by $M^3_{\mathfrak{q},\mathfrak{r}}$ s.t. $\text{sign}(u_1^{(0)}) = \text{sign}(u_1^{(1)})$

Output: reduced basis $\mathbf{u}^{(2)}$ s.t. $|u_1^{(2)}| < |u_1^{(0)}|$ or $|u_1^{(2)}| < |u_1^{(1)}|$

1: $(\mathbf{x}, \mathbf{y}) \leftarrow (\mathbf{u}^{(0)}, \mathbf{u}^{(1)})$ /* $(x_0, x_1, x_2) \leftarrow (u_0^{(0)}, u_0^{(0)}, u_0^{(0)})$, $(y_0, y_1, y_2) \leftarrow (u_0^{(1)}, u_0^{(1)}, u_0^{(1)})$ */
2: **if** $x_2 > y_2$ **then do** SWAP($\mathbf{x}, \mathbf{y}$)
3: **else if** $(x_2 = y_2) \wedge (x_1 > y_1)$ **then do** SWAP($\mathbf{x}, \mathbf{y}$)
4: **while** true **do**
5: **while** $|u_0^{(2)} + x_0| < I$ **do**
6: **if** $(|u_1^{(2)}| < |x_1|) \vee (|u_1^{(2)}| < I)$ **then return** $\mathbf{u}^{(2)}$
7: $\mathbf{u}^{(2)} \leftarrow \mathbf{u}^{(2)} + \mathbf{x}$
8: **if** $(|u_1^{(2)}| < |y_1|) \vee (|u_1^{(2)}| < I)$ **then return** $\mathbf{u}^{(2)}$
9: $\mathbf{u}^{(2)} \leftarrow \mathbf{u}^{(2)} + \mathbf{y}$
10: **return** $\mathbf{u}^{(2)}$

Algorithm 6 REDUCE2($\mathbf{u}^{(0)}, \mathbf{u}^{(1)}, \mathbf{u}^{(2)}$)

Input: bound of lattice region I, basis $\mathbf{u}^{(0)}, \mathbf{u}^{(1)}, \mathbf{u}^{(2)}$ of the lattice generated by $M^3_{\mathfrak{q},\mathfrak{r}}$ $\mathbf{u}^{(0)}, \mathbf{u}^{(1)}, \mathbf{u}^{(2)}$ s.t. $\text{sign}(u_1^{(0)}) \neq \text{sign}(u_1^{(1)})$

Output: reduced basis $\mathbf{u}^{(2)}$ s.t. $|u_1^{(2)}| < |u_1^{(0)}|$ or $|u_1^{(2)}| < |u_1^{(1)}|$

1: **if** $\text{sign}(u_1^{(0)}) \neq \text{sign}(u_1^{(2)})$ **then** $\mathbf{x} \leftarrow \mathbf{u}^{(0)}, \mathbf{y} \leftarrow \mathbf{u}^{(1)}$ **else** $\mathbf{x} \leftarrow \mathbf{u}^{(1)}, \mathbf{y} \leftarrow \mathbf{u}^{(0)}$
2: **while** $|u_1^{(2)}| < I$ **do**
3: $\mathbf{u}^{(2)} \leftarrow \mathbf{u}^{(2)} + \mathbf{x}$
4: **while** $|u_0^{(2)}| \geq I$ **do**
5: $\mathbf{u}^{(2)} \leftarrow \mathbf{u}^{(2)} + \mathbf{y}$
6: **if** $(|u_1^{(2)}| < |u_1^{(0)}|) \vee (|u_1^{(2)}| < |u_1^{(1)}|)$ **then break**
7: **return** $\mathbf{u}^{(2)}$

use the subroutine REDUCE1 presented in Algorithm 5, otherwise we use REDUCE2 presented in Algorithm 6. In Step 6, we swap $\mathbf{u}^{(2)}$ for $\mathbf{u}^{(a)}$ s.t. $|u_1^{(a)}| = \max(|u_1^{(0)}|, |u_1^{(1)}|)$, where $a \in \{0, 1\}$. From Step 6, $|u_1^{(2)}| > |u_1^{(0)}|$ and $|u_1^{(2)}| > |u_1^{(1)}|$ hold at Step 2.

In the following, we explain the subroutine REDUCE1. In Steps 1-3 of REDUCE1, we select two bases $\mathbf{x}, \mathbf{y} \in \{\mathbf{u}^{(0)}, \mathbf{u}^{(1)}\}$ s.t. the elements x_2 (resp. x_1) is less than or equals to y_2 (resp. y_1). In Steps 5-7, we reduce $u_1^{(2)}$ by adding $\mathbf{x}$ with keeping $|u_0^{(2)}| < I$. If $|u_1^{(2)}| < |x_1|$ in Step 6, then $\mathbf{u}^{(2)}$ satisfies $u_1^{(2)} \leq u_1^{(0)}$ or $u_1^{(2)} \leq u_1^{(1)}$. Moreover, if $|u_1^{(2)}| < I$ holds in Step 6, we have $|u_1^{(0)}|, |u_1^{(1)}| < I$, since $|u_1^{(2)}|$ is the largest in $|u_1^{(0)}|, |u_1^{(1)}|, |u_1^{(2)}|$ at beginning of REDUCE1, namely then Condition B1 is satisfied. Therefore, we return $\mathbf{u}^{(2)}$ in Step 6. Similarly, Steps 8-9 reduce $u_1^{(2)}$ by adding $\mathbf{y}$.

In the following, we explain the subroutine REDUCE2. At first, we select two bases $\mathbf{x}, \mathbf{y} \in \{\mathbf{u}^{(0)}, \mathbf{u}^{(1)}\}$ s.t. $\mathrm{sign}(x_1) = \mathrm{sign}(u_1^{(2)})$ in Step 1. From $\mathrm{sign}(x_1) \neq \mathrm{sign}(y_1)$ and $\mathrm{sign}(x_1) \neq \mathrm{sign}(u_1^{(2)})$, we have $\mathrm{sign}(y_1) = \mathrm{sign}(u_1^{(2)})$. Therefore, we are able to reduce $u_1^{(2)}$ by adding only $\mathbf{x}$. In Steps 2-6, we reduce $u_1^{(2)}$ by adding $\mathbf{x}$. However, we use $\mathbf{y}$ if $|u_0^{(2)}| \geq I$ holds in Steps 4 and 5 to satisfy $|u_0^{(2)}| < I$ again. If $\mathbf{u}^{(2)}$ satisfies the condition in Step 6, the termination condition of REDUCE2 holds. Therefore, we break while loop in Step 6 and return $\mathbf{u}^{(2)}$. Moreover, from the same reason in REDUCE1, if $|u_1^{(2)}| < I$ holds in Step 2, we also break the while loop and return $\mathbf{u}^{(2)}$. Therefore, we repeat the procedures in Step 2-6 in Algorithm 4 until $|u_1^{(2)}| < I$ is satisfied, then we obtain $\mathbf{u}^{(0)}, \mathbf{u}^{(1)}$ and $\mathbf{u}^{(2)}$ that satisfy Condition B1.

Finally, Step 7 of Algorithm 4 negates the bases s.t. u_2-coordinate is negative, and Step 8 adjusts $\mathbf{u}^{(0)}, \mathbf{u}^{(1)}$ and $\mathbf{u}^{(2)}$ to satisfy Conditions B1 and B2.

3.2.3 The proposed enumeration algorithm

In this section, we propose an enumeration algorithm which can exhaustively enumerate all the points in the sieve region $\mathcal{H}_c^3$ using the basis $M_{\mathfrak{q},\mathfrak{r}}^{\mathrm{FK3}}$ in the previous section.

Algorithm 7 RADIATE($\mathbf{u}^{(0)}, \mathbf{u}^{(1)}, \mathbf{u}^{(2)}$)

Input: basis $\mathbf{u}^{(0)}, \mathbf{u}^{(1)}, \mathbf{u}^{(2)}$ of the lattice generated by $M^3_{\mathfrak{q},\mathfrak{r}}$

Output: basis $\mathbf{u}^{(0)}, \mathbf{u}^{(1)}, \mathbf{u}^{(2)}$ s.t. the angle of any two of $\mathbf{u}^{(0)}, \mathbf{u}^{(1)}, \mathbf{u}^{(2)}$ is less than π

if IS_OPPOSITE($\mathbf{u}^{(0)}, \mathbf{u}^{(1)}, \mathbf{u}^{(2)}$) is true **then**
  if IS_OPPOSITE($\mathbf{u}^{(1)}, \mathbf{u}^{(2)}, \mathbf{u}^{(0)}$) is false **then** $\mathbf{u}^{(0)} \leftarrow -\mathbf{u}^{(0)}$
else
  if IS_OPPOSITE($\mathbf{u}^{(1)}, \mathbf{u}^{(2)}, \mathbf{u}^{(0)}$) is true **then** $\mathbf{u}^{(1)} \leftarrow -\mathbf{u}^{(1)}$ **else** $\mathbf{u}^{(2)} \leftarrow -\mathbf{u}^{(2)}$
end if
return $\mathbf{u}^{(0)}, \mathbf{u}^{(1)}, \mathbf{u}^{(2)}$

Algorithm 8 IS_OPPOSITE($\mathbf{u}^{(0)}, \mathbf{u}^{(1)}, \mathbf{u}^{(2)}$)

Input: basis $\mathbf{u}^{(0)}, \mathbf{u}^{(1)}, \mathbf{u}^{(2)}$ of the lattice generated by $M^3_{\mathfrak{q},\mathfrak{r}}$

Output: true: if the angle between $\mathbf{u}^{(0)}$ and $\mathbf{u}^{(a)}$ is less than π for $a = 1$ and 2, false: otherwise.

if $u_0^{(0)} = 0$ **then**
  if sign$(u_0^{(1)}) \neq$ sign$(u_0^{(2)})$ **then return** true **else return** false
$g = u_1^{(0)}/u_0^{(0)}$
$y = gu_0^{(1)} - u_1^{(1)}, z = gu_0^{(2)} - u_1^{(2)}$
if sign$(y) \neq$ sign(z) **then return** true **else return** false

Algorithm 9 ADJUST($\mathbf{u}^{(0)}, \mathbf{u}^{(1)}, \mathbf{u}^{(2)}$)

Input: basis $\mathbf{u}^{(0)}, \mathbf{u}^{(1)}, \mathbf{u}^{(2)}$ of the lattice generated by $M^3_{\mathfrak{q},\mathfrak{r}}$

Output: basis $\mathbf{u}^{(0)}, \mathbf{u}^{(1)}, \mathbf{u}^{(2)}$ that satisfies Conditions B1 and B2 in Theorem 3.2.1

for any two $\mathbf{x}, \mathbf{y}$ of $\mathbf{u}^{(0)}, \mathbf{u}^{(1)}$ and $\mathbf{u}^{(2)}$ **do**
  $\mathbf{z} \leftarrow \mathbf{x} - \mathbf{y}$ /* $(z_0, z_1, z_2) \leftarrow (x_0 - y_0, x_1 - y_1, x_2 - y_2)$ */
  if $(|z_0| < I) \wedge (|z_1| < I)$ **then**
    if $|x_2| \geq |y_2|$ **then** $\mathbf{x} = \mathbf{z}$ **else** $\mathbf{y} = \mathbf{z}$
    RADIATE($\mathbf{u}, \mathbf{v}, \mathbf{w}$)
return $\mathbf{u}, \mathbf{v}, \mathbf{w}$

At first, we give an order to all the points in $\mathcal{H}^3_c \cap L^3_{\mathfrak{q},\mathfrak{r}}$ using the property of Theorem 3.2.1. Let $\mathbf{a}, \mathbf{b}$ be two points in $\mathcal{H}^3_c \cap L^3_{\mathfrak{q},\mathfrak{r}}$. From Theorem 3.2.1, if the third coordinate of $\mathbf{a}$ is equal to or larger than that of $\mathbf{b}$, then we can write $\mathbf{a} =$

$\mathbf{b}+j_0\mathbf{u}^{(0)}+j_1\mathbf{u}^{(1)}+j_2\mathbf{u}^{(2)}$ for integers j_0, j_1, j_2 that satisfy ($j_0 \geq 0, j_1 \geq 0, j_2 \geq 0$ and $j_0 + j_1 + j_2 \neq 0$) or ($j_0 \leq 0, j_1 \leq 0, j_2 \leq 0$ and $j_0 + j_1 + j_2 \neq 0$). Note that all integers j_0, j_1, j_2 become zero simultaneously, if and only if $\mathbf{a} = \mathbf{b}$ holds. Here, we define $\mathbf{b} \prec \mathbf{a}$, if $\mathbf{a} - \mathbf{b}$ is equal to $j_0\mathbf{u}^{(0)} + j_1\mathbf{u}^{(1)} + j_2\mathbf{u}^{(2)}$ for some $j_0 \geq 0, j_1 \geq 0, j_2 \geq 0$. Then, $\mathcal{H}_c^3 \cap L_{\mathfrak{q},\mathfrak{r}}^3$ becomes a totally ordered set by order $\prec$, and we can enumerate the points in $\mathcal{H}_c^3 \cap L_{\mathfrak{q},\mathfrak{r}}^3$ by introducing a product order for the pair (j_0, j_1, j_2) of $j_0\mathbf{u}^{(0)} + j_1\mathbf{u}^{(1)} + j_2\mathbf{u}^{(2)}$. Here we define the product order $(j_0, j_1, j_2) \leq (j_0', j_1', j_2')$ for two pairs $(j_0, j_1, j_2), (j_0', j_1', j_2') \in \mathbb{Z}_{>0}^2$, if and only if $j_0 \leq j_0'$, $j_1 \leq j_1'$ and $j_2 \leq j_2'$ hold. In Algorithm 10, we show an algorithm for exhaustively enumerating all points in $\mathcal{H}_c^3 \cap L_{\mathfrak{q},\mathfrak{r}}^3$. Indeed we can prove the following theorem.

Theorem 3.2.2 *Let $\mathbf{u}^{(0)}, \mathbf{u}^{(1)}, \mathbf{u}^{(2)}$ be the basis of $M_{\mathfrak{q},\mathfrak{r}}^{FK3}$. Let $\mathbf{p} = (p_0, p_1, p_2), \mathbf{q} = (q_0, q_1, q_2)$ be points in $\mathcal{H}_c^3 \cap L_{\mathfrak{q},\mathfrak{r}}^3$. If q_2 is the smallest among all the points whose third coordinate is equal to or larger than that of p_2, then $\mathbf{q}$ is computed by Algorithm 10.*

Proof 3.2.2 *From Theorem 3.2.1, we know that all the points, whose third coordinate is is equal to or larger than p_w in $\mathcal{H}_c^3 \cap L_{\mathfrak{q},\mathfrak{r}}^3$, can be obtained by repeatedly adding $\mathbf{u}^{(0)}, \mathbf{u}^{(1)}$, or $\mathbf{u}^{(2)}$.*

Every time we add basis $\mathbf{u}^{(0)}, \mathbf{u}^{(1)}$, or $\mathbf{u}^{(\mathbf{2})}$ to $\mathbf{p}$, then the third coordinate of the resulting point becomes equal to or larger than that of $\mathbf{p}$ from Condition B3. Note that only one of $\mathbf{p} + \mathbf{u}^{(0)}$, $\mathbf{p} + \mathbf{u}^{(1)}$, and $\mathbf{p} + \mathbf{u}^{(\mathbf{2})}$ is contained in $\mathcal{H}_c^3 \cap L_{\mathfrak{q},\mathfrak{r}}^3$. Therefore, Step 1 checks if there exists $a \in \{0, 1, 2\}$ s.t. $\mathbf{p} + \mathbf{u}^{(a)} \in \mathcal{H}_c$, and we return such a point if exists. In Steps 2-6, we deal with the case of adding more than one basis of $\mathbf{u}^{(0)}, \mathbf{u}^{(1)}$, or $\mathbf{u}^{(\mathbf{2})}$ to point $\mathbf{p}$. In this step, we search the smallest pair $(i, j, k) \in \mathbb{Z}_{>0}^3$ in the sense of the above product order that satisfies $\mathbf{p} + j_0\mathbf{u}^{(0)} + j_1\mathbf{u}^{(1)} + j_2\mathbf{u}^{(2)} \in \mathcal{H}_{\mathbf{c}}^3$. Such a point satisfies the assertion of the

Algorithm 10 NEXTFK3($\mathcal{H}^3_{\mathbf{c}}, M^{\mathrm{FK3}}_{\mathfrak{q},\mathfrak{r}}$, **p**)

Input: bound of lattice region I, point $\mathbf{p} = (p_0, p_1, p_2)^{\mathrm{T}} \in L^3_{\mathfrak{q},\mathfrak{r}} \cap \mathcal{H}^3_{\mathbf{c}}$, $M^{\mathrm{FK3}}_{\mathfrak{q},\mathfrak{r}} = (\mathbf{u}^{(0)}, \mathbf{u}^{(1)}, \mathbf{u}^{(2)})$

Output: $\mathbf{q} = (q_0, q_1, q_2)^{\mathrm{T}} \in L^3_{\mathfrak{q},\mathfrak{r}} \cap \mathcal{H}^3_{\mathbf{c}}$, s.t. q_2 is the smallest under the condition of $q_2 > p_1$

```
1:  while true do
2:      r ← p   /* (r_0, r_1, r_2) ← (p_0, p_1, p_2) */
3:      while true do
4:          s ← r   /* (s_0, s_1, s_2) ← (r_0, r_1, r_2) */
5:          while true do
6:              s ← s + u^(0)
7:              if s ∈ H^3_c then return s
8:              if I/2 ≤ s_0 or I/2 ≤ s_1 then break
9:          r ← r + u^(1)
10:         if I/2 ≤ r_0 or I/2 ≤ r_1 then break
11:     p ← p + u^(2)
```

theorem.

From Theorem 5.2, we construct an enumeration algorithm for exhaustively enumerating all the points in $\mathcal{H}^3_c$ with $M^{\mathrm{FK3}}_{\mathfrak{q},\mathfrak{r}}$ by repeatedly adding $\mathbf{u}^{(0)}, \mathbf{u}^{(1)}$, or $\mathbf{u}^{(2)}$.

Example 3.2.1 *We show an example of the basis of the proposed algorithm. We choose an extension field GF(p^n) of size 303 bits s.t.* $p = 38486027, n = 12$*, and choose a polynomial* $f_1(X) = X^{12} + X^2 - 1$*. Additionally, for* f_1*, we obtain* f_2 *by adding* p *to* f_1 *as* $f_2(X) = f_1 + p$*. We take special-*$\mathfrak{q}$ *as* $\mathfrak{q} = (q, g) = (99989, X + 8368)$*, and the other prime ideal* $\mathfrak{r} = (r, h) = (89107, X + 54851)$*.*

Then, we compute the basis as the HNF matrix of $M_{\mathfrak{q}}, M_{\mathfrak{r}}$ as follows:

$$M_{\mathfrak{q}} = \begin{pmatrix} 99989 & 8368 & 0 \\ 0 & 1 & 8368 \\ 0 & 0 & 1 \end{pmatrix}, \quad M_{\mathfrak{r}} = \begin{pmatrix} 89107 & 54851 & 0 \\ 0 & 1 & 54851 \\ 0 & 0 & 1 \end{pmatrix}.$$

We obtain the basis of $M^3_{\mathfrak{q},\mathfrak{r}}$ and the proposed basis $M^{FK3}_{\mathfrak{q},\mathfrak{r}}$ used for the lattice sieve as follows:

$$M^3_{\mathfrak{q},\mathfrak{r}} = \begin{pmatrix} 89107 & 27083 & -50795 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}, \quad M^{FK3}_{\mathfrak{q},\mathfrak{r}} = \begin{pmatrix} 23 & -57 & 35 \\ 23 & -10 & -48 \\ 7 & 28 & 13 \end{pmatrix}.$$

Chapter 4

Experimental Results

4.1 Solving DLP over GF(p^{12}) of size 203 bits

In this section, we report our experiment on solving the discrete logarithm problem over extension field GF(p^{12}) of 203 bits using the number field sieve in Section 2. We chose the characteristic $p = 122663$ of 17 bits, namely the cardinality of the extension field GF(p^{12}) is

$$p^{12} = 116028047901493489912893641612452600729095851402664913077940 81.$$

The computational environment in our experiment is as follows. We used one PC equipped with four CPUs (Intel Xeon X7350 2.93 GHz; Core2 micro architecture; 16 cores in total) and 64 GBytes of RAM. We utilize gmp-5.0.5 for the arithmetic of multi-precision integers, openmpi-1.6 for parallel implementation between processes, pari-2.5.1 [70] for the decomposition of ideals in the number field, and ntl-5.5.2 for the computation of lattice reduction using LLL. We use C++ with compiler gcc-4.7.1 on Linux OS (64 bits).

Table 4.1 presents the experimental data in our implementation and the previous ones of the number field sieve over extension field GF(p^n).

Table 4.1: Comparison of known experiments of the number field sieve over extension field GF(p^n)

Finite Field	GF(p^3)	GF(p^6)	GF(p^{12})
Authors	Joux et al. [41]	Zajac [94]	Ours
Year	2006	2008	2012
CPU	Alpha (1.15GHz) $\times$ 8	Sempron (2.01GHz) $\times$ 8	Xeon (2.93GHz) $\times$ 4
Days	19 days	5 days	2 days
Bit Length	394	242	203
Sieving	2-dim. lattice sieve	3-dim. line sieve	**7-dim. lattice sieve**

4.1.1 How to Select Parameters t, H, B_1, B_2

In this section we explain how to select the parameters of the lattice sieve in Section 2.3 for given two polynomials f_1, f_2 in the polynomial selection in Section 2.2. In particular, we discuss the suitable size of dimension $t+1$, sieving interval H, and the smoothness bound B_1, B_2 that satisfy the equation (2.2) for the number field sieve over extension degree GF(p^{12}). If we select the parameters that accelerate both the searching relation step and the linear algebra step simultaneously, then the total running time of the number field sieve becomes faster.

Selection of t

Denote by V_H the size of sieving region $\mathcal{H}_a(H)$, namely $V_H = 2^t \prod_{j=0}^{t} H_j$. We extend the estimation of the average norm in the two-dimensional lattice sieve [66, 5] to our multi-dimensional case. The average norm $N_{\text{ave}}(h_a(\alpha_i))$ of polynomial $f_i\,(i = 1, 2)$ in the lattice sieve of $t+1$ dimensions is evaluated by the following

equation.

$$N_{\text{ave}}(h_a(\alpha_i)) = \frac{\sqrt{\int_0^{H_t}\int_{-H_{t-1}}^{H_{t-1}}\cdots\int_{-H_0}^{H_0}(\text{Res}(h_a, f_i))^2\, da_0 \ldots da_t}}{V_H}$$

Moreover, we approximate the probability $\rho(x, y)$ that the integers smaller than x are y-smooth as $(\log_y x)^{-\log_y x}$, and we assume that the total size of factor bases $\mathcal{B}_1, \mathcal{B}_2$ is $V_B = \pi(B_1) + \pi(B_2)$ where $\pi(B_i)$ is the number of primes smaller than or equal to $B_i\,(i = 1, 2)$. Let R be the number of relations in the sieving region $\mathcal{H}_a(H)$, and then R is calculated by

$$R = \rho(N_{\text{ave}}(h_a(\alpha_1)), B_1)\rho(N_{\text{ave}}(h_a(\alpha_2)), B_2)V_H. \tag{4.1}$$

Here we have to find the parameters that satisfy (2.2), namely $R > V_B$. Fig. 4.1. shows the minimal V_B that satisfies $R > V_B$ for V_H in the lattice sieve of $t + 1$ dimensions in the extension field GF(p^{12}) of 203, 514 and 3075 bits, respectively. In order to reduce the time of searching such bound V_B we set $H_0 = H_1 = \cdots = H_t$ and $B_1 = B_2$. From Fig. 4.1., we can select smaller sizes of sieving region V_H and factor base V_B that satisfy equation (2.2) using 8 dimension (6 or 4 dimensions) for extension field GF(p^{12}) of 203 bits (514 or 3075 bits), respectively.

Selection of H and B_1, B_2

In the following we discuss the choice of the sieving interval H and the smoothness bound B_1, B_2 so that the running time of the number field sieve becomes faster.

For the fixed size of sieving interval V_H and factor base V_B, we first select the sieving interval H and then the smoothness bound B_1, B_2. The sieving interval H is chosen that the probability of $\rho(N_{\text{ave}}(h_a(\alpha_1)), B_1)\rho(N_{\text{ave}}(h_a(\alpha_2)), B_2)$ arisen

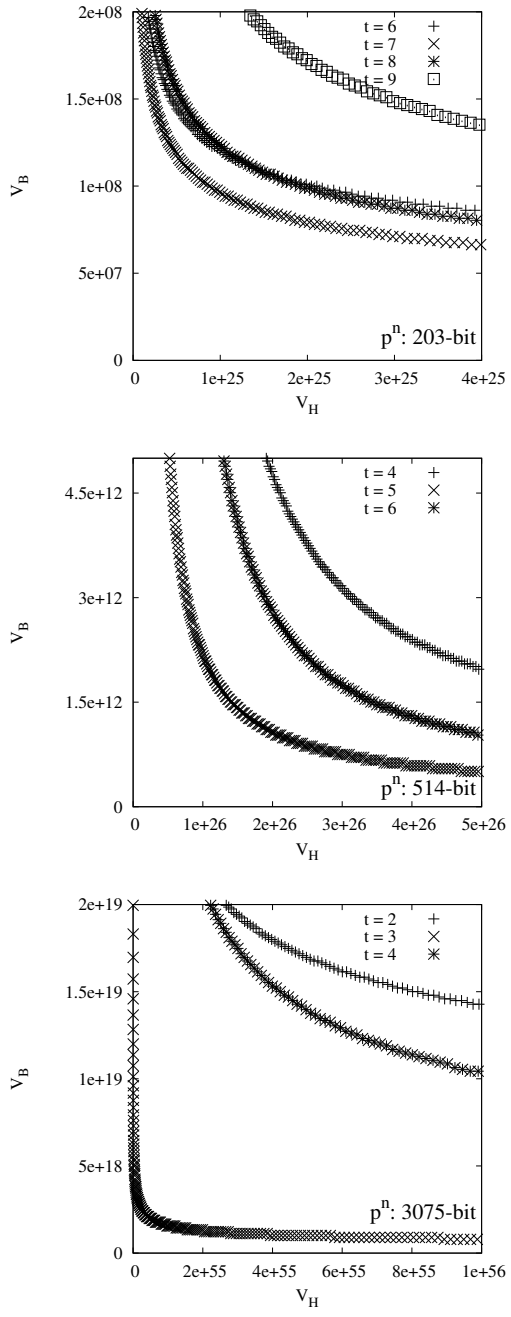

Figure 4.1: V_H and V_B are the size of sieving region and the factor base of multi-dimensional lattice sieve for the number field sieve over extension field GF(p^{12}) of 203 bits (top), 514 bits (middle) and 3075 bits (bottom).

from the hit tuple of equation (4.1) is maximum for fixed B_1, B_2 with $B_1 = B_2$. For the above H, we then select the smoothness bound B_1, B_2 so that the number of relations in equation (4.1) become maximum.

4.1.2 Polynomial Selection

In order to select two polynomials f_1, f_2 in Section 2.2, we use the polynomial selection similar to the previous experiments [41] and [94]. At first an irreducible polynomial $f_1 \in \mathbb{Z}[X]$ of degree 12 with small coefficients is chosen, and then we set $f_2 = f_1 + p$ or $f_2 = f_1 - p$.

In this paper, Murphy's α function [66, 7] is used for selecting a more suitable pair of polynomials f_1, f_2. If the Murphy's α function $f_i\,(i = 1, 2)$ is smaller, then the norm $N(h_a(\alpha_i))\,(i = 1, 2)$ is expected to become smoother, namely it is divisible by small prime divisors with higher probability. The coefficient of polynomial f_1 is searched in the range of ± 10, and then the sum of the Murphy's α of the following polynomials f_1, f_2 is smallest among the range of our search.

$$\begin{aligned} f_1(X) &= X^{12} - 3X^4 + 9X^3 - 9X^2 - 9X + 2, \\ f_2(X) &= X^{12} - 3X^4 + 9X^3 - 9X^2 - 9X - 122661. \end{aligned}$$

4.1.3 Searching Relations

We used the method in Section 4.1.1 to select the dimension $t+1$, sieving interval H and smoothness bound B_1, B_2 for our implementation of the lattice sieve.

In the estimation of Section 4.1.1, the suitable dimension of the lattice sieve for extension field GF(p^{12}) of 203 bits was estimated as eight. We perform some experiments of the lattice sieve of 6, 7 and 8 dimensions for random special-$\mathfrak{q}$ with fixed V_H and V_B. From this experiments, the lattice sieve of 7 dimen-

sions yields the largest number of relations for one special-$\mathfrak{q}$, and then we select $H = (443, 427, 304, 140, 70, 24, 9)$ and smoothness bounds $B_1 = 114547$, $B_2 = 148859$.

We run the lattice sieve using the above polynomials f_1, f_2 and parameters t, H, B_1, B_2. Our experiment has generated 32,241 hit pairs in about 42 hours using only 6 cores in our computational environment. This is about 1.3 times larger than the sufficient number of relations $\sharp\mathcal{B}_1 + \sharp\mathcal{B}_2 + 2n$.

4.1.4 Linear Algebra

From the hit pairs in the searching relations we construct a matrix of linear equations modulo $\ell = 6118607636866573789$ (63 bits) that is the maximum prime divisor of $p^{12} - 1$. The size of the matrix is 32241×24463, and it is shrunk to 16579×15073 by the filter process such as eliminating duplicated relations. Then we solve it by Lanczos method [48, 46].

We found the solutions of the linear equations in about 25 minutes using the 16 cores in our computational environment, and the virtual logarithms $\log \mathfrak{q}$ and $\log \Lambda_{i,j}$ was obtained.

Finally we present an example of the discrete logarithm in finite field GF(p^{12}) of $p = 122663$. Let $g = x^2 + x - 7$ be a generator of the multiplicative group GF$(p^{12})^* = (\text{GF}(p)[X]/f_1(X))^*$. Let $a = x^2 - 5x + 7$ be a target element of solving the discrete logarithm $\log_g a$ in GF(p^{12}). Note that both g and a are B_1-smooth. The above linear equations modulo ℓ yields $\log a = 3540036734608022534$ and $\log g = 3897708711757659596$, and thus the discrete logarithm $\log_g a$ in GF(p^{12}) is computed by $\log a / \log g = 3161374319443177763 \bmod \ell$.

4.2 An experiment on 3-dimensional lattice sieve

In the following, we show some data of the proposed 3-dimensional lattice sieve. We implemented the sieve step of the number field sieve for solving the discrete logarithm problem over finite field GF(p^{12}). The finite field of extension degree $n = 12$ is used for efficient implementation of pairing-based cryptography using BN curves [10].

Table 4.2: Computational environment in our experiment in Section 4.2

CPU	Intel Core i7-3770 3.40 GHz
RAM	8 GBytes
OS	Linux (64 bits)
Language	C++
Compiler	gcc-4.7.2
Library	gmp-5.0.5

Table 4.2 shows the computational environments of our experiment. In order to perform many experiments in this computational resources, we choose a characteristic $p = 38486027$ of 26 bits. In the polynomial selection step, we choose a polynomial $f_1 = x^{12}+x^2-1 \in \mathbb{Z}[X]$, which is irreducible in GF(p) and has small coefficients. Then polynomial f_2 is chosen as $f_2 = f_1+p = x^{12}+x^2+38486026$.

In our experiment of the 3-dimensional lattice sieve generated by $M^3_{\mathfrak{q},\mathfrak{r}}$ defined in Section 2.5.2 for the case of $t = 2$, we set the following parameters. We choose 10 special-$\mathfrak{q}$s s.t. $\mathfrak{q} = (q, g) \in \mathcal{B}_2$ from $99871 \leq q \leq 99989$. The bound of sieve region is $I = 2^k$ $(k = 7, 8, \ldots, 11), J = I/2$. We generate prime ideals $\mathfrak{r} = (r, h) \in \mathcal{B}_2$ s.t. $I < r \leq 85386$ and $\deg h = 1$. The number of such $\mathfrak{r}$ is about 8000 for one special-$\mathfrak{q}$, namely we deal with about 80000 lattices generated

by $M^3_{\mathfrak{q},\mathfrak{r}}$ for one fixed I.

Table 4.3: Rate of $M^3_{\mathfrak{q},\mathfrak{r}}$ that satisfies Condition B4

I	$\sharp M^3_{\mathfrak{q},\mathfrak{r}}$	B1, B2, B3, B4	B1, B2, B4	B1, B2, B3
128	84005	62395 (74%)	3724 (4%)	18765 (22%)
256	83865	61717 (73%)	322 (0%)	21918 (26%)
512	83445	55746 (66%)	2 (0%)	27681 (33%)
1024	82645	54204 (65%)	0 (0%)	28425 (34%)
2048	81185	52446 (64%)	0 (0%)	28725 (35%)

For chosen special-$\mathfrak{q}$, I and $\mathfrak{r}$, we generate the basis $M^3_{\mathfrak{q},\mathfrak{r}}$ by the proposed generation algorithm in Section 3.2. Table 4.3 shows some probabilities related to the basis $M^3_{\mathfrak{q},\mathfrak{r}}$ in our experiments. The first column of Table 4.3 is the number of $M^3_{\mathfrak{q},\mathfrak{r}}$ we generated for each bound I. The second column is the number of $M^3_{\mathfrak{q},\mathfrak{r}}$ that satisfies all conditions in Section 3.2. The third column is the number of $M^3_{\mathfrak{q},\mathfrak{r}}$ that does not satisfies only Condition B3. The fourth column is the number of $M^3_{\mathfrak{q},\mathfrak{r}}$ that does not satisfies only Condition B4. The number of the other types of the basis is less than 2% among the total number of $M^3_{\mathfrak{q},\mathfrak{r}}$. About 74% of the basis $M^3_{\mathfrak{q},\mathfrak{r}}$ fulfill all the conditions for $I = 128$, namely we are able to compute all the points in $\mathcal{H}^3_c$ in the lattice for the basis $M^3_{\mathfrak{q},\mathfrak{r}}$. If the basis $M^3_{\mathfrak{q},\mathfrak{r}}$ does not fulfill them, Condition B4 is critical for all sieve bound in the experiment.

Once the basis $M^3_{\mathfrak{q},\mathfrak{r}}$ does not fulfill all the conditions in Section 3.2, Algorithm 10 not always enumerate all the points in the sieve region $\mathcal{H}^3_c$. Table 4.4 shows the percentages of the points generated by Algorithm 10 over the all points in $\mathcal{H}^3_c$ using the same basis $M^3_{\mathfrak{q},\mathfrak{r}}$ in the previous experiment. Here we assume that the number of all points in $\mathcal{H}^3_c$ as I^2J/r. The proposed enumeration algorithm can enumerate more than 90% of the points in the sieve region using the basis $M^3_{\mathfrak{q},\mathfrak{r}}$.

Table 4.4: Rate of the points that enumerated by $M^3_{\mathfrak{q},\mathfrak{r}}$ that don't satisfies Condition B4

I	rate
128	90%
256	94%
512	96%
1024	98%
2048	99%

4.3 An experiment on the number field sieve over GF(p^6)

In this section, in order to confirm our 3-dimensional lattice sieve works efficiently, we report an experiment on a sieve step of the number field sieve over GF(p^6). The finite field of extension degree r is used for efficient implementation of pairing-based cryptography using MNT curves [65]. Zajac solved the discrete logarithm problem over GF(p^6) of 240 bits [94] (Zaj08-exp), which is the current record of JLSV06-NFS over finite fields of extension degree 6. We perform an experiment on the proposed lattice using same parameter in the experiment of Zaj08-exp.

Our experiment uses the computers in Table 4.5. We deploy the parameters in our experiment as similar as possible those used in Zaj08-exp. Zajac solved the discrete logarithm problem over the extension field whose characteristic $p = 1081034284409$ of 40 bits, namely the cardinality of GF(p^6) is

$$p^6 = 15960144001970777403060723996771756 \setminus$$
$$92025917352715453344036177063352145041$$

Table 4.5: The data of computers that we use in the experiment in Section 4.3

CPU	Intel Core i7-3770 3.40 GHz $\times 8$	Intel Xeon E5-2430L 2.00GHz $\times 24$
RAM	32 GBytes	32 GBytes
OS	Linux (64 bits)	
Language	C++	
Compiler	gcc-4.7.2	
Library	gmp-5.0.5, openmpi-1.6	

of 240 bits. For the extension field, he chose two polynomials

$$\begin{aligned} f_1(X) &= x^6 - 2x^5 + x^3 - x + 2, \\ f_2(X) &= x^6 - 2x^5 + x^3 - x + 1081034284411. \end{aligned}$$

Additionally, he also chose smoothness bounds $B_1 = B_2 = 6532326$ and 3-dimensional sieve region in $\mathbf{a}$-space $\mathcal{H}_a$ s.t. $-2^{18} \leq a_0 \leq 2^{18}$, $-2^{13} \leq a_1 \leq 2^{13}$, $1 \leq a_2 \leq 1149$. In the experiment in Zaj08-exp, he executed 3-dimensional line sieve in $\mathcal{H}_a$ above with elements in factor bases $(q, g) \in \mathcal{B}_1 \cup \mathcal{B}_2$ s.t. $\deg g = 1$.

In our experiment, the same parameter p, n, f_1, f_2, B_1 and B_2 were implemented. Instead the $\mathbf{a}$-space, our 3-dimensional lattice uses a sieve region over $\mathbf{c}$-space $\mathcal{H}_c^3$. Here, we try to chose the sieve region $\mathcal{H}_c^3$ such that the number of hit tuples we can obtain in $\mathcal{H}_c^3$ is larger than $\sharp\mathcal{B}_1 + \sharp\mathcal{B}_2 + 2n = 893773$. In our 3-dimensional lattice sieve, we choose 223595 special-$\mathfrak{q} = (q, g)$ from $\mathcal{B}_2$ s.t. $3112117 \leq q \leq 6532291$ and $\deg g = 1$. We executed our lattice sieve for 10 special-qs that is randomly chosen in $3112117 \leq q \leq 6532291$ with respect to $\mathcal{H}_c^3$ whose bound of lattice region is $I = 2^k$ $(k = 4, 5, \ldots 10)$. Then, we estimated the number of hit tuples we obtain for all special-qs in $3112117 \leq q \leq 6532291$ with respect to $I = 2^k$ $(k = 4, 5, \ldots 10)$. From the estimated number of hit tuples for

$I = 2^k$ $(k = 4, 5, \ldots 10)$, we chosen $I = 2^7$ since $k = 7$ is the least integer s.t. the estimated number of hit tuples is larger than $\sharp\mathcal{B}_1 + \sharp\mathcal{B}_2 + 2n = 893773$.

For example, in the case of special-$\mathfrak{q} = (6532291, X + 1470092)$ and $\mathfrak{r} = (751691, X + 268635)$, the basis of $M^{\mathrm{FK3}}_{\mathfrak{q},\mathfrak{r}}$ becomes

$$M^{\mathrm{FK3}}_{\mathfrak{q},\mathfrak{r}} = \begin{pmatrix} 230 & -6 & -35 \\ -192 & 235 & -42 \\ 27 & 19 & 4 \end{pmatrix}.$$

Then an example of the hit tuple in the lattice generated by $M^{\mathrm{FK3}}_{\mathfrak{q},\mathfrak{r}}$ is $\mathbf{a} = (-63189, 410, 72)^{\mathrm{T}}$, where the norms of $\mathbf{a}$ are

$$\begin{aligned} N_1(\mathbf{a}) &= 625429496719699899560898532 13 \\ &= 41^2 \times 2371 \times 6869 \times 101863 \times 4700621 \times 4771049, \\ N_2(\mathbf{a}) &= 1630525249272665498988847945432 21597 \\ &= 7 \times 17 \times 12953 \times 22271 \times 116461 \times 1344457 \times 4643843 \times 6532291. \end{aligned}$$

In the result, the whole running time of the lattice sieve was 50996 seconds, namely about 14 hours, and we got 1041417 hit tuples. Then, we eliminated 103842 duplicate hit tuples. Therefore, we obtained 937575 hit tuples in contrast to 1077984 obtained in Zaj08-exp.

Table 4.6 shows the experimental data in both our implementation and the previous one in Zaj08-exp. Our experiment deploys about 4 time more CPU cores than Zaj08-exp, but the running time reduces from 3 days to 14 hours.

Table 4.6: Comparison of our experiment with the top record of the number field sieve over GF(p^6)

	Zajac [94]	Ours
Year	2008	2014
CPU and ♯cores	Sempron (2.01GHz) × 8	Core i7 (3.40GHz) × 8 Xeon (2.00GHz) × 24
Timing of Sieve	3 days	14 hours
Sieve	3-dim. line sieve	**3-dim. lattice sieve**

Chapter 5

Conclusion

In this paper we presented an implementation of the number field sieve for solving the DLP over extension field GF(p^n) that underpinned the security of pairing-based cryptography. Especially we proposed the implementation of the lattice sieve of more than two dimensions. In our experiment, we discussed the dimension and size of sieve region suitable for the number field sieve over extension field GF(p^{12}). Finally we have solved the DLP over extension field GF(p^{12}) of 203 bits using a PC of 16 CPU core in about 43 hours.

Additionally, we proposed a three-dimensional lattice sieve as an extension of the Franke-Kleinjung method that is used for enumerating points in a two-dimensional sieve region. First, we gave a natural extension of the basis conditions used in the Franke-Kleinjung method to the 3-dimensional case. We proved that a basis the conditions can exhaustively enumerate all points in the sieve region of the three-dimensional lattice. We then described an enumeration algorithm that can trace all the points in the three-dimensional sieve region if such a basis exists. In an experiment using GF(p^{12}) of 303 bits, our algorithm enumerated more than 90% of all points in the sieve region of the three-dimensional lattices, even though

the basis did not satisfy the above conditions.

Finally, we compared the running times of our three-dimensional lattice sieve and the current record holder of GF(p^6) of 240 bits by Zajac. We found that the sieve step using our extended three-dimensional lattice sieve is about as efficient as that of three-dimensional line sieve used by Zajac.

In the future, we have to discuss how to select more optimal parameters of the number field sieve for the DLPs over an extension field GF(p^{12}) of larger bits. Then, we will estimate the security of pairing-based cryptography more precisely by experiments over an extension field GF(p^{12}) of more than 300 bits with the parameters.

Appendix

PARI/GP script to find singular prime

In this section, we show a sample script code of PARI/GP [70] for finding singular prime with respect to a polynomial f.

```
? fa = 65*x^3 - x - 576
%1 = 65*x^3 - x - 576
? fw = 65^2 * subst( fa, x, x/65 )
%2 = x^3 - 65*x - 2433600
? factor(round(sqrt( poldisc(fw)/nfdisc(fw) )))
%3 =
[ 2 1]

[ 5 1]

[13 1]

? nf = nfinit(fw);
? p2 = idealprimedec( nf, 2 )
%5 = [[2, [1, 0, 1]~, 1, 1,
```

```
        [0, 18720, -18720; 0, -32, 160; 1, -65, 33]],
       [2, [1, 1, 1]~, 1, 1,
        [0, 18720, 0; 1, 33, 128; 1, 65, -32]],
       [2, [2, 1, 0]~, 1, 1,
        [1, 0, 18720; 1, 66, -32; 0, 130, -64]]]
? factor(idealnorm( nf, 1+x ))
%6 =
[ 2 9]

[ 7 2]

[97 1]

? idealval( nf, 1+x, p2[1] )
%7 = 8
? idealval( nf, 1+x, p2[2] )
%8 = 1
? idealval( nf, 1+x, p2[3] )
%9 = 0
? idealfactor( nf, 1+x )
%10 =
[ [2, [1, 0, 1]~, 1, 1, [0, 18720, -18720;
 0, -32, 160; 1, -65, 33]] 8]
[ [2, [1, 1, 1]~, 1, 1, [0, 18720, 0;
 1, 33, 128; 1, 65, -32]] 1]
[ [7, [8, 1, 0]~, 1, 1, [-1, -56160, 74880;
 1, 160, -512; -3, 325, -165]] 2]
```

```
[[97, [1, 1, 0]~, 1, 1, [33, 617760, -1235520;
 -33, -3168, 6336; 33, -6435, 3267]] 1]
```

An example of enumerating lattice points

In this section, we show an example of enumerating lattice points. First, we chose an extension field GF(p^n) with the extension degree $n = 6$ and the characterisitic

$$p^6 = 15960144001970777403060723996771756 \setminus$$
$$92025917352715453344036177063352145041$$

Second, we chose two polynomial f_1, f_2 as follows:

$$f_1(X) = x^6 - 2x^5 + x^3 - x + 2,$$
$$f_2(X) = x^6 - 2x^5 + x^3 - x + 1081034284411,$$

and we set $I = 256, J = 128$ as the bound of region $\mathcal{H}$. Next, we chose special-$\mathfrak{q}$ as $\mathfrak{q} = (6532291, X + 1470092)$, and $\mathfrak{r} = (751691, X + 268635)$. Then, we get

$$M^{\mathrm{FK}}_{\mathfrak{q},\mathfrak{r}} = \begin{pmatrix} 230 & -6 & -35 \\ -192 & 235 & -42 \\ 27 & 19 & 4 \end{pmatrix}.$$

Finally, the lattice points generated by the basis $M^{\mathrm{FK}}_{\mathfrak{q},\mathfrak{r}}$ in the region $\mathcal{H}$ are enumerated as follows:

```
#1:(0, 0, 0)
  step 7: q + w = (-35, -42, 4)
#2:(-35, -42, 4)
  step 7: q + w = (-70, -84, 8)
```

```
#3:(-70, -84, 8)
  step 7: q + w = (-105, -126, 12)
#4:(-105, -126, 12)
  step 5: q + u = (125, -318, 39)
  step 7: q + w = (-140, -168, 16)
  step 9: q + u + w = (90, -360, 43)
  step 16: q + v = (-111, 109, 31)
#5:(-111, 109, 31)
  step 5: q + u = (119, -83, 58)
#6:(119, -83, 58)
    ...
#13:(-97, -100, 101)
```

Bibliography

[1] L.M. Adleman, “A subexponential algorithm for the discrete logarithm problem with applications to cryptography,” in Proc. 20th IEEE Found. Comp. Sci. Symp., pp.55-60, 1979.

[2] L.M. Adleman, “The function field sieve,” ANTS-I, Lecture Notes in Comput. Sci., vol.877, pp.108-121, 1994.

[3] L.M. Adleman and J. DeMarrais, “A subexponential algorithm for discrete logarithms over all finite fields,” Math. Comp., vol.161, pp.1-15, 1993.

[4] L.M. Adleman and M.-D. Huang, “Function field sieve method for discrete logarithms over finite fields,” preprint, 1998

[5] K. Aoki, “Sieving Region, and Relationship between Numbers of Required Relations and Factor Bases on the Number Field Sieve,” Technical Report of IEICE, ISEC, Vol.104, No.53, pp.23-28, 2004.

[6] K. Aoki, Y. Kida and H. Ueda, “A trial of GNFS implementation (Part VI) : lattice sieve,” Technical Report of IEICE, ISEC, Vol.104, No.315, pp.9-14, 2004. (in Japanese)

[7] K. Aoki, H. Ueda and S. Uchiyama, "Evaluation Report on Integer Factoring Problems," Technical Report of CRYPTREC, no.0202-1, 2003. `http://www.cryptrec.go.jp/estimation.html`. (in Japanese)

[8] E. Bach and J. Shallit, "Factoring with cyclotomic polynomials," Math. Comp., vol.52, pp.201-219, 1989.

[9] P. Barreto, S. Galbraith, C. Ó'hÉigeartaigh, and M. Scott, "Efficient Pairing Computation on Supersingular Abelian Varieties," Des. Codes Cryptogr., vol.42, pp.239-271, 2007.

[10] P.S.L.M. Barreto and M. Naehrig, "Pairing-friendly elliptic curves of prime order," SAC 2005, Lecture Notes in Comput. Sci., vol.3897, pp.319-331, Springer, 2006.

[11] E. R. Berlekamp, "Factoring polynomials over large finite fields," Math. Comp., vol.24, pp.713-735, 1970.

[12] D. Boneh and M. Franklin, "Identity Based Encryption From the Weil Pairing," CRYPTO 2001, Lecture Notes in Comput. Sci., vol.2139, pp.213-229, Springer, 2001.

[13] J. Brillhart, M. Filaseta and A. Odlyzko, "On an irreducibility theorem of A. Cohn," Canad. J. Math., vol.33, pp.1055-1059, 1981.

[14] J.A. Buchmann, *Introduction to cryptography,* Undergraduate Texts in Mathmatics, Springer-Verlag, 2001.

[15] J.A. Buchmann and V. Shoup, "Constructing nonresidues in finite fields and the extended Riemann hypothesis," Math. Comp., vol.65, pp.1311-1326, 1996.

[16] J.P. Buhler, H.W. Lenstra and C. Pomerance, "Factoring integers with the number field sieve," pp.50-94, in [50].

[17] E.R. Canfield, P. Erdös and C. Pomerance, "On a problem of Oppenheim concerning 'factorisatio numerorum '," J. Number Theory, vol.17, pp.1-28, 1983.

[18] H. Cohen, *A course in computational algebraic number theory,* Graduate Texts in Math., vol.138, Springer-Verlag, 1993.

[19] A. Commeine and I. Semaev, "An algorithm to solve the discrete logarithm problem with the number field sieve," PKC 2006, Lecture Notes in Comput. Sci., vol.3958, pp.174-190, Springer-Verlag, 2006.

[20] D. Coppersmith, "Fast evaluation of logarithms in fields of characteristic two," IEEE Trans. Inform. Theory, vol.30, pp.587-594, 1984.

[21] D. Coppersmith, A.M. Odlyzko and R. Schroeppel, "Discrete Logarithms in GF(p)," Algorithmica, vol.1, pp.1-15, 1986.

[22] D. Coppersmith, "Modifications to the number field sieve," J. Cryptology, vol.6, pp.169-180, 1993.

[23] A. Devegili, C. Ó'hÉigeartaigh, M. Scott, and R. Dahab, "Multiplication and Squaring on Pairing-Friendly Fields," Cryptography ePrint Archive, Report 2006/471, 2006.

[24] T. Denny, "A Lanczos implementation for GF(p)," Universität des Saarlandes, to appear.

[25] T. ElGamal, "A subexponential-time algorithm for computing discrete logarithms over GF(p^2)," IEEE Trans. Inform. Theory, vol.31, pp.473-481, 1985.

[26] G. Frey, M. Müller and H.-G. Rück, "The Tate pairing and the discrete logarithm applied to elliptic curve cryptosystems," IEEE Trans. Inform. Theory, vol.45, pp.1717-1719, 1999.

[27] J. Franke and T. Kleinjung, "Continued fractions and lattice sieve," Workshop record of SHARCS, 2005, http://www.ruhr-uni-bochum.de/itsc/tanja/SHARCS/talks/FrankeKleinjung.pdf.

[28] S. Gao and J. Howell, "A general polynomial sieve: Designs and codesla memorial tribute to Ed Assmus," Des. Codes Cryptogr., vol.18, pp.149-157, 1999.

[29] D.M. Gordon, "Discrete logarithms in GF(p) using the number field sieve," SIAM J. Discrete Math., vol.6, pp.124-138, 1993.

[30] D.M. Gordon and K. McCurley, "Massively parallel computation of discrete logarithms," CRYPTO '92, Lecture Notes in Comput. Sci., vol.740, pp.312-323, Springer-Verlag, 1993.

[31] R. Granger, A. Holt, D. Page, N. Smart and F. Vercauteren, "Function field sieve in characteristic three," ANTS-VI, Lecture Notes in Comput. Sci., vol.3076, pp.223-234, Springer-Verlag, 2004.

[32] R. Granger and F. Vercauteren, "On the discrete logarithm problem on algebraic tori," CRYPTO '05, Lecture Notes in Comput. Sci., vol.3621, pp.66-85, Springer-Verlag, 2005.

[33] T. Hayashi, N. Shinohara, L. Wang, S. Matsuo, M. Shirase and T. Takagi, "Solving a 676-bit discrete logarithm problem in GF(3^{6n})," PKC2010, Lecture Notes in Comput. Sci., vol.6056, pp.351-367, 2010.

[34] F. Hess, N. Smart, and F. Vercauteren, "The Eta Pairing Revisited," IEEE Trans. on Inform. Theory, vol.52, pp.4595-4602, 2006.

[35] T. Izu, J. Kogure and T. Shimoyama, "Recent Topics on Integer Factorization," Fundamentals Review, vol.1, pp.58-70, 2008. (in Japanese)

[36] A. Joux and R. Lercier, "The function field sieve is quite special," ANTS-V, Lecture Notes in Comput. Sci., vol.2369, pp.431-445, Springer-Verlag, 2002.

[37] A. Joux and R. Lercier, "Improvements to the general number field sieve for discrete logarithms in prime fields. A comparison with the Gaussian integer method," Math. Comp., vol.72, pp.953-967, 2003.

[38] A. Joux and R. Lercier, "Discrete logarithms in GF(2^{607}) and GF(2^{613})," email to the NMBRTHRY mailing list, 23 September, 2005.

[39] A. Joux and R. Lercier, "Discrete logarithms in GF(370801^{30}) | 168digits | 556bits," email to the NMBRTHRY mailing list, 9 November, 2005.

[40] A. Joux, R. Lercier, "The function field sieve in the medium prime case," EUROCRYPTO '06, Lecture Notes in Comput. Sci., vol.4007, pp.254-270, Springer-Verlag, 2006.

[41] A. Joux, R. Lercier, N.P. Smart and F. Vercauteren, "The number field sieve in the medium prime case," CRYPTO '06, Lecture Notes in Comput. Sci., vol.4117, pp.326-344, Springer-Verlag, 2006.

[42] T. Kleinjung, "On polynomial selection for the general number field sieve," Math. Comp., vol.75, pp.2037-2047, 2006.

[43] T. Kleinjung et al., "Discrete logarithms in GF(p) - 160 digits," email to the NMBRTHRY mailing list, 2007. http://listserv.nodak.edu/cgi-bin/wa.exe?A2=ind0702&L=nmbrthry&T=0&P=194.

[44] T. Kleinjung, K. Aoki, J. Franke, A.K. Lenstra, E. Thomé, J.W. Bos, P. Gaudry, A. Kruppa, P.L. Montgomery, D.A. Osvik, H.J.J. te Riele, A. Timofeev and P. Zimmermann, "Factorization of a 768-bit RSA modulus," CRYPTO '10, Lecture Notes in Comput. Sci., vol.6223, pp.333-350, Springer-Verlag, 2010.

[45] N. Koblitz, *A Course in Number Theory and Cryptography*, Graduate Texts in Mathematics, Vol. 114, 1994.

[46] M. LaMacchia and A. Odlyzko, "Solving large sparse linear systems over finite fields," CRYPTO '90, Lecture Notes in Comput. Sci., vol.537, pp.109-133, Springer-Verlag, 1991.

[47] S. Lang, "Algebraic Number Theory," Addison-Wesley, Reading, MA, 1970.

[48] C. Lanczos, "Solution of systems of linear equations by minimized iterations," J. Res. Nat. Bur. Stand., vol.49, pp.33-53, 1952.

[49] E. Lee, H.-S. Lee, and C.-M. Park, "Efficient and Generalized Pairing Computation on Abelian Varieties," IEEE Trans. on Inform. Theory, vol.55, pp.1793-1803, 2009.

[50] A.K. Lenstra and H.W. Lenstra, *The Development of the Number Field Sieve,* Lecture Notes in Math., vol.1554, Springer-Verlag, 1993.

[51] A.K. Lenstra, H.W. Lenstra and L. Lovász, "Factoring polynomials with rational coefficients," Math. Ann., vol.261, pp.515-534, 1982.

[52] A.K. Lenstra, H.W. Lenstra, M.S. Manasse and J.M. Pollard, "The number field sieve," pp.11-42, in [50].

[53] A.K. Lenstra, H.W. Lenstra, M.S. Manasse and J.M. Pollard, "The factorization of the ninth Fermat number," Math. Comp., vol.61, pp.319-349, 1993.

[54] H.W. Lenstra, "Factoring integers with elliptic curves," Ann. of Math., vol.126, pp.649-673, 1987.

[55] H.W. Lenstra, "Algorithms for finite fields," Nmber theory and cryptography, London Math. Soc. Lecture Notes Ser., vol.154, pp.76-85, Cambridge Univ. Press, 1990.

[56] H.W. Lenstra, "Algorithms in algebraic numbeer theory," Bull. Amer. Math. Soc., vol.26, pp.211-244, 1992.

[57] R. Lercier. "Computations - Discrete Logarithms, " available at `http://perso.univ-rennes1.fr/reynald.lercier/plugins/getfilehtml/getfilehtml7d2c.html?lng=en&id=6`, 2009.

[58] R. Lercier and F. Vercauteren, "Discrete logarithms in $\mathbb{F}_{p^{18}}$ - 101digits," email to the NMBRTHRY mailing list, 28 June, 2005.

[59] A. lvić and G. Tenenbaum, "Local densities over integers free of large prime factors," Quart. J. Math. Oxford Ser., vol.37, pp.401-417, 1986.

[60] D.A. Marcus, "Number fields," Springer-Verlag, 1977.

[61] U.M. Maurer and Y. Yacobi, "A non-interactive public-key distribution system," Des. Codes Cryptogr., vol.9, pp.3050316, 1996.

[62] A. Menezes, T. Okamoto and S. Vanstone, "Reducing elliptic curve logarithms to logarithms in a finite field," IEEE Trans. Inform. Theory, vol.39, pp.1639-1646, 1993.

[63] K. McCurley, "The discrete logarithm problem," Proc. Sympos. Appl. Math., vol.42, pp.49-74, 1990.

[64] V. Miller, "The Weil Pairing, and Its Efficient Calculation," J. Cryptology, vol.17, pp.235-261, 2004.

[65] A. Miyaji, M. Nakabayashi and S. Takano, "New explicit conditions of elliptic curve traces for FR-reduction", In IEICE Trans. on Fund., E84-A (5) (2001) 1234-1243.

[66] B. Murphy, "Polynomial selection for the number field sieve integer factorisation algorithm," PhD. thesis, The Australian National University, 1999.

[67] J. Neukirch, *Algebraic number theory,* Grundlehren der mathematischen Wissenschaften, vol.322, Springer-Verlag, 1999.

[68] A. Odlyzko, "Discrete logarithms: the past and the future," Des. Codes Cryptogr., vol.19, pp.129-145, 2000.

[69] T. Okamoto, K. Takashima, "Fully Secure Functional Encryption with General Relations from the Decisional Linear Assumption," CRYPTO 2010, pp.191-208, 2010.

[70] PARI/GP, version `2.5.3`, Bordeaux, 2012. `http://pari.math.u-bordeaux.fr/`.

[71] J.M. Pollard, "The lattice sieve," pp.43-49, in [50].

[72] C. Pomerance, “Elementary thoughts on discrete logarithms,” Algorithmic Number Theory, vol.44, pp.385-396, 2008.

[73] C. Pomerance and J. Smith, “Reduction of huge, sparse matrices over finite fields via created catastrophes,” Experiment. Math., Vol.1, pp.89-94, 1992.

[74] O. Schirokauer, “Discrete logarithms and local units,” Philos. Trans. Roy. Soc. London Ser. A, vol.345, pp.409-424, 1993.

[75] O. Schirokauer, “Using number fields to compute logarithms in finite fields,” Math. Comp., vol.69, pp.1267-1283, 2000.

[76] O. Schirokauer, “The special function field sieve,” SIAM J. Discrete Math., vol.16, pp.81-98, 2002.

[77] O. Schirokauer, “Virtual logarithms,” J. Algorithms, vol.57, pp.140-147, 2005.

[78] O. Schirokauer, “The impact of the number field sieve on the discrete logarithm problem in finite fields,” Algorithmic Number Theory, vol 44, pp.397-420, 2008.

[79] O. Schirokauer, “The number field sieve for integers of low weight,” Math. Comp., vol.79, pp.583-602, 2010.

[80] O. Schirokauer, D. Weber and T. Denny, “Discrete logarithms - the effectiveness of the index calculus method,” ANTS-II, Lecture Notes in Comput. Sci., vol.1122, pp.337-361, Springer-Verlag, 1996.

[81] V. Shoup, “Searching for primitive roots in finite fields,” Math. Comp., vol.58, pp.369-380, 1992.

[82] J. Solinas, "Generalized mersenne numbers," Technical Report CORR 99-39, University of Waterloo, 1999.

[83] P. Stevenhagen, "The arithmetic of number rings," Algorithmic Number Theory, vol.44, pp.209-266, MSRI Publications, 2008.

[84] P. Stevenhagen, "The number field sieve," Algorithmic Number Theory, vol.44, pp.83-100, 2008.

[85] H. Stichtenoth, *Algebraic function fields and codes,* Graduate Texts in Math., vol.254, Springer-Verlag, 1993.

[86] C. Studholme. "The Discrete Log Problem," available at `http://www.cs.toronto.edu/~cvs/dlog/`, 2002.

[87] E. Thomé, "Computation of discrete logarithms in $\mathbb{F}_{2^{607}}$," ASIACRYPTO '01, Lecture Notes in Comput. Sci., vol.2248, pp.107-124, Springer-Verlag,

[88] S. Uchiyama, "The Discrete Logarithm Problem in Finite Fields : The number field sieve and the function field sieve," Transactions of the Japan Society for Industrial and Applied Mathematics, vol.13, pp.245-256, 2003.

[89] F. Vercauteren, "Optimal pairings," IEEE Transactions on Information Theory, vol.56, pp.455-461, 2010.

[90] D. Weber, "An implementation of the number field sieve to compute discrete logarithms mod p," EUROCRYPT '95, Lecture Notes in Comput. Sci., vol.921, pp.95-105, Springer-Verlag, 1995.

[91] D. Weber, "Computing discrete logarithms with the number field sieve," ANTS-II, Lecture Notes in Comput. Sci., vol.1122, pp.391-403, 1996.

[92] D. Weber and T. Denny, "The solution of McCurley's discrete log challenge," CRYPT '98, Lecture Notes in Comput. Sci., vol.1462, Springer-Verlag, 1998.

[93] D.H. Wiedemann, "Solving sparse linear equations over finite fields," IEEE Trans. Inform. Theory, vol.32, pp.54-62, 1986.

[94] P. Zajac, "Discrete logarithm problem in degree six finite fields," PhD thesis, Slovak University of Technology, 2008. http://www.kaivt.elf.stuba.sk/kaivt/Vyskum/XTRDL.

[95] P. Zajac, "On the use of the lattice sieve in the 3D NFS," Tatra Mt. Math. Publ. vol.45, pp.161-172, 2010.

List of Papers and Talks

Journal

1. Kenichiro Hayasaka, Kazumaro Aoki, Tetsutaro Kobayashi and Tsuyoshi Takagi. An Experiment of Number Field Sieve for Discrete Logarithm Problem over GF(p^n). JSIAM Letters, vol.6, pp.53-56, 2014.

Refereed International Conference Papers

1. Kenichiro Hayasaka and Tsuyoshi Takagi. An Experiment of Number Field Sieve over GF(p) of Low Hamming Weight Characteristic. IWCC 2011, LNCS 6639, pp.191-200, Springer, 2011.

2. Kenichiro Hayasaka, Kazumaro Aoki, Tetsutaro Kobayashi and Tsuyoshi Takagi. An Experiment of Number Field Sieve for Discrete Logarithm Problem over GF(p^{12}). Number Theory and Cryptography 2013, Buchmann Festschrift, LNCS 8260, pp.108-120, 2013.

Talks

1. Kenichiro Hayasaka, Kazumaro Aoki, Tetsutaro Kobayashi and Tsuyoshi Takagi. A Construction of 3-dimensional Lattice Sieve for Number Field Sieve over GF(p^n). Computer Security Symposium – CSS 2013, 1C1-3, 2013. (in Japanese).

2. Kenichiro Hayasaka, Kazumaro Aoki, Tetsutaro Kobayashi and Tsuyoshi Takagi. An Experiment of Number Field Sieve for Discrete Logarithm Problem over GF(p^{12}). Symposium on Cryptography and Information Security – SCIS 2013, 4A1-3, 2013. (in Japanese).

3. Kenichiro Hayasaka, Kazumaro Aoki, Tetsutaro Kobayashi and Tsuyoshi Takagi. A Verification of 3-dimensional Lattice Sieve. Computer Security Symposium – CSS 2014, 1E3-3, 2014. (in Japanese).