IEICE TRANS. FUNDAMENTALS, VOL.E99–A, NO.6 JUNE 2016

PAPER Special Section on Discrete Mathematics and Its Applications

A Simple Improvement for Integer Factorizations with Implicit Hints

Ryuichi HARASAWA[†a)], ***Member***, **Heiwa RYUTO**[†], ***Nonmember***, ***and*** **Yutaka SUEYOSHI**[†], ***Member***

SUMMARY In this paper, we describe an improvement of integer factorization of k RSA moduli $N_i = p_i q_i$ $(1 \le i \le k)$ with implicit hints, namely all p_i share their t least significant bits. May et al. reduced this problem to finding a shortest (or a relatively short) vector in the lattice of dimension k obtained from a given system of k RSA moduli, for which they applied Gaussian reduction or the LLL algorithm. In this paper, we improve their method by increasing the determinant of the lattice obtained from the k RSA moduli. We see that, after our improvement, May et al.'s method works smoothly with higher probability. We further verify the efficiency of our method by computer experiments for various parameters.

key words: *integer factorization with implicit hints*

1. Introduction

The integer factorization is a fundamental theme of computer algebra and also an important topic of public key cryptography, especially for cryptosystems whose security relies on the infeasibility of integer factorization (e.g., the RSA cryptosystem [15]). So far, many researchers proposed various methods for factoring integers, such as Pollard's rho method [14], the number field sieve [6] and so on. There is another aspect for integer factorization: characterizations of weaker integers, namely studying properties with which we can efficiently factor integers, for example, Pollard's $p-1$ method [13].

In this paper, we investigate the integer factorization with implicit hints. The origin of this topic is Coppersmith's paper [1]: He proposed a method for factoring an RSA modulus $N = pq$, provided we know a half of the most significant bits of p. This is a kind of factorization methods using some explicit hints.

On the other hand, May and Ritzenhofen proposed a method for integer factorization with implicit hints [8]. For an RSA modulus $N_1 = p_1 q_1$ to be factored, we get some hints of $k-1$ RSA moduli $N_i = p_i q_i$ $(2 \le i \le k)$ satisfying $p_1 \equiv p_2 \equiv \cdots \equiv p_k \pmod{2^t}$, that is, all p_i share the t least significant bits. Their method reduces the factorization of N_1, more precisely finding q_1, to finding a short non-zero vector in the lattice associated with a system of congruences obtained from implicit hints. In order to realize that, they used the so-called LLL algorithm [7], which computes a basis in which the length of each component vector is relatively short.

Manuscript received September 18, 2015.
Manuscript revised January 7, 2016.

[†]The authors are with the Graduate School of Engineering, Nagasaki University, Nagasaki-shi, 852-8521 Japan.
a) E-mail: harasawa@cis.nagasaki-u.ac.jp

DOI: 10.1587/transfun.E99.A.1090

Let α be an integer such that $q_i < 2^\alpha$. In the case of $k \ge 3$, under certain assumptions based on the Minkowski theorem and a property of a lattice, we get q_i efficiently if the inquality $t \ge \frac{k}{k-1}\alpha$ holds [8]. In the case of $k = 2$, if the inequality $t \ge 2\alpha + 3$ holds, then we always get q_1 and q_2 in polynomial time of $\log N_1$ and $\log N_2$ by the so-called Gaussian reduction. We remark that, in the case of $k = 2$, the paper [5] showed an improved bound $t \ge 2\alpha + 1$ for the condition above.

Recently, in the case where $k = 2$ and $t \le 2\alpha$, Nuida et al. [11] improved the May-Ritzenhofen method by analyzing the inverse matrix of the matrix consisting of the basis obtained by Gaussian reduction. Their method works efficiently if the quantity $2^{2\alpha - t}$ is sufficiently small. On the other hand, as pointed out in their paper, it seems to be difficult to generalize their method to the case $k \ge 3$.

In this paper, we give an alternative improvement of the May-Ritzenhofen method. Namely, we increase the size of the lattice corresponding to a given system of k RSA moduli with implicit hints. Then the length of the target vector consisting of non-trivial factors of the RSA moduli becomes relatively shorter. Therefore, we expect that, after our lattice construction, the success probability of the method [8] becomes higher.

As in Nuida et al.'s method [11], our method works efficiently if the quantity $2^{2\alpha - t}$ is sufficiently small. On the other hand, unlike the work [11], it is easy to generalize our method to the case $k \ge 3$. We further verify the efficiency of our method by computer experiments for various parameters.

The remainder of this paper is organized as follows: In Sect. 2, we introduce some facts on lattices needed later. In Sect. 3, we describe an overview of previous works. In Sect. 4, we describe our improvement on the factorization with implicit hints. In Sect. 5, we implement our method and verify the efficiency. In Sect. 6, we give the conclusion and future works.

2. Preliminaries

Let n be a positive integer. An integer lattice (resp. a lattice) is a discrete additive subgroup of $\mathbb{Z}^n$ (resp. $\mathbb{R}^n$). More intuitively, we describe it as follows: Let d be a positive integer with $d \le n$, and let $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_d \in \mathbb{Z}^n$ be linearly independent vectors over $\mathbb{Q}$. Then the set of all integer combinations of $\boldsymbol{b}_i$'s forms an integer lattice L, that is,

$$L = \left\{ \sum_{i=1}^{d} a_i \boldsymbol{b}_i \mid a_i \in \mathbb{Z} \right\}.$$

We call $\{\boldsymbol{b}_1, \ldots, \boldsymbol{b}_d\}$ a basis of L, and the value d the rank or dimension of L. A lattice is said to have full rank if $d = n$.

Since we deal with only an integer lattice of full rank in this paper, we use the notation "lattice" for "integer lattice of full rank", unless otherwise specified.

The determinant of a lattice $L \subseteq \mathbb{Z}^n$, denoted by $\det(L)$, is defined by $|\det(B)|$, where B is the square matrix whose column vectors consist of a basis of L. The determinant of L is equal to the volume of the parallelotope spanned by the basis of L. We note that the determinant is invariant under unimodular transformations of B.

Let $\|\boldsymbol{v}\|$ denote the Euclidean norm of a vector $\boldsymbol{v}$.

The following result gives a bound of the determinant of a lattice with respect to the Euclidean norm.

Theorem 1 (Hadamard's inequality [9]): Let $L \subseteq \mathbb{Z}^n$ be a lattice with basis $\{\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n\}$. Then we have

$$\det(L) \leq \prod_{i=1}^{n} \|\boldsymbol{b}_i\|.$$

The successive minima of a lattice L of dimension n, denoted by $\lambda_i(L)$ for $1 \leq i \leq n$, are defined by the minimal radius of a ball containing i linearly independent lattice vectors of L. Especially, a vector $\boldsymbol{v}$ in L with $\|\boldsymbol{v}\| = \lambda_1(L)$ is a shortest (non-zero) vector contained in L. By definition, we see $\lambda_i(L) \leq \lambda_j(L)$ for $i \leq j$.

In the case of a lattice L of dimension two (i.e., $n = 2$) with basis $\{\boldsymbol{b}_1, \boldsymbol{b}_2\}$, we can compute efficiently a basis $\{\boldsymbol{v}_1, \boldsymbol{v}_2\}$ such that $\|\boldsymbol{v}_i\| = \lambda_i(L)$ for $i = 1, 2$ with the running time $O(\log^2(\max\{\|\boldsymbol{b}_1\|, \|\boldsymbol{b}_2\|\}))$ by applying Gaussian reduction. (For Gaussian reduction, we refer the readers to [9].)

In the case of a lattice of dimension $n \geq 3$, there exists no efficient method to guarantee the output of a shortest vector contained in a given lattice. Alternatively, A. K. Lenstra et al. proposed an algorithm (the so-called LLL algorithm) in polynomial running time of input size for computing a basis whose first component vector has a relatively short length.

Theorem 2 (LLL [7]): Let $L \subseteq \mathbb{Z}^n$ be a lattice with basis $\{\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n\}$. Then the LLL algorithm outputs a basis $\{\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n\}$, called a reduced basis, satisfying the following properties:

1. $\|\boldsymbol{v}_1\| \leq 2^{\frac{n-1}{4}} \det(L)^{\frac{1}{n}}$;
2. $\|\boldsymbol{v}_1\| \leq 2^{\frac{n-1}{2}} \lambda_1(L)$.

The runnning time of the algorithm is estimated as $O(n^5(n + \log b_{\max}) \log b_{\max})$, namely polynomial time with respect to the input size, where $b_{\max}$ is the largest absolute value among elements in the basis vectors $\{\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n\}$.

Remark 1: With the same notation as in Theorem 2, let $\boldsymbol{v}$ be a shortest vector in L. We see that both of $\boldsymbol{v}$ and $\boldsymbol{v}_1$ are vectors in the set $M := \{\boldsymbol{x} \in L \mid \|\boldsymbol{x}\| \leq 2^{\frac{n-1}{2}} \lambda_1(L)\}$. So, if the rank of the lattice generated by vectors in M is equal to one, then the vector $\boldsymbol{v}_1$ becomes a shortest vector in L, namely $\boldsymbol{v}_1 = \pm\boldsymbol{v}$.

We introduce the Minkowski bound on a shortest vector of a lattice.

Theorem 3 (Minkowski [10]): Let L be a lattice of dimension n. Then L contains a non-zero vector $\boldsymbol{v}$ with

$$\|\boldsymbol{v}\| = \lambda_1(L) \leq \sqrt{n} \det(L)^{\frac{1}{n}}.$$

3. Previous Works

In this section, we describe May et al.'s method [8] for the factorization of k RSA moduli with implicit hints, and its improvement [11] in the case of $k = 2$. For the case $k = 2$, we use the improved estimation of $\|\boldsymbol{v}_2\| = \lambda_2(L)$ in [5] instead of the original one in [8].

We make the following heuristic assumption based on the Minkowski theorem (Theorem 3).

Assumption 1: Let L be a lattice of dimension n. We assume that, if a non-zero vector $\boldsymbol{b}$ in L satisfies $\|\boldsymbol{b}\| \leq \sqrt{n} \cdot \det(L)^{\frac{1}{n}}$, then $\boldsymbol{b}$ is a shortest vector in L.

Now we are ready to describe the method in [8] for factoring k RSA moduli with implicit hints.

Let $N_i = p_i q_i$ $(1 \leq i \leq k)$ be k RSA moduli. We assume $\gcd(N_i, N_j) = 1$ for $1 \leq i < j \leq k$, and that $q_i < 2^{\alpha}$ for $1 \leq i \leq k$. We further suppose $p_1 \equiv \cdots \equiv p_k \pmod{2^t}$, that is, all p_i share their t least significant bits.

Let

$$L := \left\{ \begin{pmatrix} x_1 \\ \vdots \\ x_k \end{pmatrix} \in \mathbb{Z}^k \mid x_1 - \frac{N_1}{N_i} x_i \equiv 0 \pmod{2^t}, \right.$$
$$\left. 2 \leq i \leq k \right\}, \tag{1}$$

be a lattice with a basis $\left\{ \begin{pmatrix} 1 \\ \frac{N_2}{N_1} \bmod 2^t \\ \frac{N_3}{N_1} \bmod 2^t \\ \vdots \\ \frac{N_k}{N_1} \bmod 2^t \end{pmatrix}, \begin{pmatrix} 0 \\ 2^t \\ 0 \\ \vdots \\ 0 \end{pmatrix}, \ldots, \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 2^t \end{pmatrix} \right\}$.

Then we have $\det(L) = 2^{(k-1)t}$ and $\boldsymbol{q} := \begin{pmatrix} q_1 \\ \vdots \\ q_k \end{pmatrix} \in L$.

Adapting Assumption 1 to integer factorizations with implicit hints above, we obtain the following result. †

†More precisely, Theorem 4 is obtained by applying the LLL algorithm to a result in [8]. We note that, in [8], May et al. developed their theory under the use of the Kannan algorithm [4] and the Helfrich algorithm [3], which are deterministic algorithms for computing a shortest vector in a given lattice. However, in practice, they implemented their method by using the LLL algorithm instead of these deterministic algorithms above.

Theorem 4: Let the notation be as above, and let $\{\boldsymbol{v}_1, \ldots, \boldsymbol{v}_k\}$ be the basis obtained by applying the LLL algorithm to the above basis of L. If the following conditions

1. $\|\boldsymbol{q}\| \leq \sqrt{k} \cdot 2^{(k-1)t/k}$, in particular, $\alpha \leq \frac{k-1}{k}t$;
2. $\boldsymbol{v}_1$ is a shortest vector in L;

hold, then we have $\boldsymbol{q} = \pm\boldsymbol{v}_1$ under Assumption 1. In other words, we get the non-trivial factor q_i of each RSA modulus N_i.

Remark 2: By the second property in Theorem 2, it is rare that the second condition in Theorem 4 holds as k becomes large. So, as our strategy for factorization using the LLL algorithm, we make the first condition stronger (namely, we raise the lower bound of t). This stategy seems to be effective at least from the experimental point of view. In fact, the method in [8] works smoothly for the implementation if $t \geq \frac{k}{k-1}\alpha + \epsilon$ holds, where the value ϵ depends on the parameters k, α and t etc. In the case of the experimental result in [8], they evaluate as $3 \leq \epsilon \leq 7$. On the other hand, if the first condition in Theorem 4 holds, then both of $\boldsymbol{q}$ and $\boldsymbol{v}_1$ are relatively short vectors in L. Therefore, it is likely that $\boldsymbol{q} = \pm\boldsymbol{v}_1$ even if neither $\boldsymbol{q}$ nor $\boldsymbol{v}_1$ is a shortest vecor in L.

By using our method described in the next section, we can get the vector $\boldsymbol{q}$ for the case where the method in [8] does not give it, including the case where Assumption 1 does not necessarily hold.

We next restrict ourselves to the case $k = 2$. In this case, by [5], we get an improved evaluation of the method above with no assumption unlike the cases of $k \geq 3$: For the case $k = 2$, the corresponding basis of the latttice L is $\left\{\begin{pmatrix} 1 \\ \frac{N_2}{N_1} \bmod 2^t \end{pmatrix}, \begin{pmatrix} 0 \\ 2^t \end{pmatrix}\right\}$, and the determinant is equal to 2^t. From Gaussian reduction, we get the basis $\{\boldsymbol{v}_1, \boldsymbol{v}_2\}$ with $\|\boldsymbol{v}_i\| = \lambda_i(L)$ for $i = 1, 2$. Then, from Hadamard's inequality, we have $2^t = |\det(L)| \leq \|\boldsymbol{v}_1\| \cdot \|\boldsymbol{v}_2\| \leq \|\boldsymbol{v}_2\|^2$. Therefore, if $\|\boldsymbol{q}\| < 2^{\frac{t}{2}} \leq \|\boldsymbol{v}_2\|$, then we have $\boldsymbol{q} = a\boldsymbol{v}_1$ for some integer a, by the definition of the successive minima. Since q_1 and q_2 are different primes, we have $a = \pm 1$, namely, $\boldsymbol{q} = \pm\boldsymbol{v}_1$. In particular, if $t \geq 2\alpha + 1$, then we have $\|\boldsymbol{q}\| = \sqrt{q_1^2 + q_2^2} < \sqrt{2 \cdot 2^{2\alpha}} \leq 2^{\frac{t}{2}}$ and $\boldsymbol{q} = \pm\boldsymbol{v}_1$.

We remark that, from the experimental point of view in [5], it seems that the condition $t \geq 2\alpha + 1$ is a tight bound so that $\boldsymbol{q} = \pm\boldsymbol{v}_1$ holds.

Furthermore, in the case of $k = 2$ and $t \leq 2\alpha$, Nuida et al. [11] showed that, the integer coefficients a and b of $\boldsymbol{q} = a\boldsymbol{v}_1 + b\boldsymbol{v}_2$ satisfy the relation $|a|, |b| \leq 2 \cdot 2^{2\alpha - t}$ by analyzing the inverse matrix of the matrix $(\boldsymbol{v}_1, \boldsymbol{v}_2)$. Applying the exhaustive search of the coefficients a and b as above, this result implies that we can efficiently find q_1 and q_2, namely prime divisors of N_1 and N_2, if the quantity $2^{2\alpha - t}$ is sufficiently small (e.g., polynomial size of $\log N_1$ and $\log N_2$).

However, this method seems to be difficult to generalize to the cases $k \geq 3$ because of the difficulty of analyzing the size of elements of the inverse matrix of the $k \times k$ matrix consisting of the basis obtained by the LLL algorithm.

In the next section, we propose an alternative method in the case where the previous work [8] seems to be difficult to work smoothly. Namely, we consider the case $t \leq 2\alpha$ for $k = 2$ and the case $t < \frac{k}{k-1}\alpha$ for $k \geq 3$.

4. Proposed Method

In this section, we propose an improved method of the previous work [8] for factoring k RSA moduli with implicit hints.

For the simplicity of the description, we first consider the case of two RSA moduli (i.e., $k = 2$).

4.1 The Case of Two RSA Moduli

Let the notation be the same as in the previous section, namely, let $N_1 = p_1q_1$ and $N_2 = p_2q_2$ be two RSA moduli with $\gcd(N_1, N_2) = 1$, and let $\boldsymbol{q} = \begin{pmatrix} q_1 \\ q_2 \end{pmatrix}$. We suppose that $p_1 \equiv p_2 \pmod{2^t}$, and that $q_i < 2^\alpha$ for $i = 1, 2$.

We consider the case $t \leq 2\alpha$, otherwise we can easily get q_1 and q_2 from Gaussian reduction.

Our strategy is to construct a lattice which contains the vector $\boldsymbol{q}$ and whose determinant is greater than $\|\boldsymbol{q}\|^2$. Once we have found such a lattice, we can easily get the vector $\boldsymbol{q}$ in the same way as the previous work [8]. Namely, the vector $\boldsymbol{q}$ becomes the first component vector (up to sign) in the basis obtained by applying Gaussian reduction.

In order to do this, we consider the following problem, which is a natural generalization of the previous work: For any positive integer ℓ, we consider a congruence modulo $2^{t+\ell}$ to which the vector $\boldsymbol{q}$ is a solution and of which the determinant of the lattice associated with the solution set is equal to $2^{t+\ell}$.

Let $(p_2 - p_1) \bmod 2^{t+\ell} \equiv \beta \cdot 2^t$ with $0 \leq \beta < 2^\ell$, then we have $p_2 \equiv p_1 + \beta \cdot 2^t \pmod{2^{t+\ell}}$. Multiplying both sides by q_1q_2, we have

$$N_2q_1 \equiv N_1q_2 + (\beta q_2 \cdot 2^t)q_1 \pmod{2^{t+\ell}}.$$

Hence, we obtain the congruence

$$q_1 - \frac{N_1}{N_2 - (\beta q_2) \cdot 2^t} q_2 \equiv 0 \pmod{2^{t+\ell}}.$$

We define $\lambda := \beta q_2 \bmod 2^\ell$. Then we obtain the following result.

Theorem 5: Let $N_1 = p_1q_1$ and $N_2 = p_2q_2$ be two RSA moduli with $\gcd(N_1, N_2) = 1$, and let $\boldsymbol{q} = \begin{pmatrix} q_1 \\ q_2 \end{pmatrix}$. We assume $p_1 \equiv p_2 \pmod{2^t}$. Then, for any positive integer ℓ, there exists an integer λ with $0 \leq \lambda < 2^\ell$ such that

$$\boldsymbol{q} \in L' := \left\{ \begin{pmatrix} x \\ y \end{pmatrix} \in \mathbb{Z}^2 \mid x - \frac{N_1}{N_2 - \lambda \cdot 2^t} y \equiv 0 \pmod{2^{t+\ell}} \right\}.$$

A $\mathbb{Z}$-basis of L' is $\left\{\begin{pmatrix} 1 \\ \frac{N_2-\lambda\cdot 2^t}{N_1} \bmod 2^{t+\ell} \end{pmatrix}, \begin{pmatrix} 0 \\ 2^{t+\ell} \end{pmatrix}\right\}$, and the determinant of L' is equal to $2^{t+\ell}$.

Remark 3: In Theorem 5, when we try to find a λ by exhaustive search, the total number of the candidates is 2^ℓ.

If we choose ℓ so that $\sqrt{2}\cdot 2^\alpha \le 2^{\frac{t+\ell}{2}}$, equivalently $\ell \ge 2\alpha - t + 1$, then we find the vector $\boldsymbol{q}$ in the same way as the previous work [8]. Like the previous method [11], our method also works efficiently if the quantity $2^{2\alpha-t}$ is sufficiently small (see Remark 3).

4.2 The Case of k RSA Moduli with $k \ge 3$

We directly generalize our method described in the previous subsection to a system of k RSA moduli with $k \ge 3$:

Applying our method described in the previous subsection (at most) $k-1$ times, we obtain the following result.

Theorem 6: Let the notation be the same as in Sect. 3. We let $\ell_2, \ldots, \ell_k$ be $k-1$ non-negative integers satisfying $\ell_2 + \cdots + \ell_k = \ell$ [†]. Then, there exists a $(k-1)$-tuple $(\lambda_2, \ldots, \lambda_k)$ with $0 \le \lambda_i < 2^{\ell_i}$ $(2 \le i \le k)$ such that

$$\boldsymbol{q} \in L' := \left\{ \begin{pmatrix} x_1 \\ \vdots \\ x_k \end{pmatrix} \in \mathbb{Z}^k \mid x_1 - \frac{N_1}{N_i - \lambda_i \cdot 2^t} x_i \equiv 0 \pmod{2^{t+\ell_i}},\ 2 \le i \le k \right\}.$$

A $\mathbb{Z}$-basis of L' is

$$\left\{ \begin{pmatrix} 1 \\ \frac{N_2-\lambda_2\cdot 2^t}{N_1} \bmod 2^{t+\ell_2} \\ \frac{N_3-\lambda_3\cdot 2^t}{N_1} \bmod 2^{t+\ell_3} \\ \vdots \\ \frac{N_k-\lambda_k\cdot 2^t}{N_1} \bmod 2^{t+\ell_k} \end{pmatrix}, \begin{pmatrix} 0 \\ 2^{t+\ell_2} \\ 0 \\ \vdots \\ 0 \end{pmatrix}, \ldots, \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 2^{t+\ell_k} \end{pmatrix} \right\},$$

and the determinant of L' is equal to $2^{(k-1)t+\ell}$.

Remark 4: In Theorem 6, if we fix a tuple $(\ell_2, \ldots, \ell_k)$ with $\ell_2 + \cdots + \ell_k = \ell$ when we try to find such a tuple $(\lambda_2, \ldots, \lambda_k)$ by exhaustive search, then the total number of the candidates is 2^ℓ.

Combining our method with Theorem 4, we get the following result.

Theorem 7: Let the notation be as above. We choose the value ℓ so that $\|\boldsymbol{q}\| \le \sqrt{k}\cdot 2^{((k-1)t+\ell)/k}$. In particular, if $\ell \ge k\alpha - (k-1)t$, then this inequality holds. Further, we assume that, for a lattice L, the first component vector of the basis obtained by the LLL algorithm is a shortest vector in L. If the quantity $2^{k\alpha-(k-1)t}$ is sufficiently small, then we get efficiently the vector $\boldsymbol{q}$ by applying the LLL algorithm, under Assumption 1.

[†]For given k and ℓ, the total number of $(k-1)$-tuple $(\ell_2, \ldots, \ell_k)$'s with $\ell_2 + \cdots + \ell_k = \ell$ is equal to $\binom{\ell+k-2}{\ell}$.

5. Experimental Results

We implemented our algorithm on a Pentium G6950 2.8 GHz PC with 2 GB RAM. Our algorithm is written in C++ using NTL [16] with gcc 4.1.2 compiler. Further, we compared our method with Nuida et al.'s method for $k = 2$.

Let p, q and $N = pq$ be integers whose bit-lengths have the same as those of p_i, q_i and $N_i = p_i q_i$ respectively. Let α denote the bit-length of q and suppose all p_i share their t least significant bits. For this experiments, we set the bit-length of N as 1000 bit. We try the factorizations 100 times for each setting of parameters.

We then evaluate the results from the following aspects (as well as the running time and the success rate):

1. the lattice size applied to Gaussian reduction/ the LLL algorithm (i.e., the bit length of the determinant of the lattice);
2. the ratio of the exhaustive serch to the whole.

The former concerns the success rate. The latter is useful for comparing our method with Nuida et al.'s method.

Nuida et al.'s method consists of two parts as follows:

LR: Gaussian reduction for the lattice L given by (1) for $k = 2$;
ES: the exhaustive search of a and b satisfying $\boldsymbol{q} = a\boldsymbol{v}_1 + b\boldsymbol{v}_2$ with $|a|, |b| \le 2^{2\alpha+1-t}$, where $\{\boldsymbol{v}_1, \boldsymbol{v}_2\}$ is the basis obtained by Gaussian reduction in the [LR] part.

According to two parts above for Nuida et al.'s method, we divide our method into two parts as follows:

LR: Gaussian reduction for the lattice L' with $\ell = \lambda = 0$ in Theorem 5, which coincides with the [LR] part in Nuida et al.'s method;
ES: the exhaustive search, that is, Gaussian reduction for the lattices L' with λ $(0 \le \lambda < 2^\ell)$ for a fixed $\ell \ne 0$ in Theorem 5.

We then compare the ratios of the [ES] part to the whole, namely the quotients [ES]/([LR] + [ES]).

We note that the description "the additional bits ℓ is equal to zero" means the previous method [8]. The notation "success" means a trial in which the target vector $\boldsymbol{q}$ becomes the first component vector (up to sign) in the basis obtained by the LLL algorithm or Gaussian reduction. For our evaluations of running times in this paper, we count only success trials.

5.1 Two RSA Moduli

We set the bit-length of q as 250, namely $\alpha = 250$. We implemented the factorization of two RSA moduli with implicit hints $p_1 \equiv p_2 \pmod{2^t}$ for various pairs (t, ℓ) with

Table 1 Attack for factoring k RSA moduli with $(k, \alpha) = (2, 250)$.

number of shared bits t	bound $2\alpha+1$	additional bits ℓ	bits of lattice size: $(k-1)t+\ell$	success rate
501	501	0	501	100 %
500	501	0	500	38 %
		1	501	100 %
491	501	0	491	0 %
		8	499	0 %
		9	500	40 %
		10	501	100 %

additional bits ℓ, by using our method (Table 1). We note that the running time of all experiments is below 1 second.

By the evaluation of the paper [5], we get the target vector $\boldsymbol{q}$ by the original/our methods whenever $\ell \geq 2\alpha - t + 1$. Even if the condition above does not hold, we see that the success rate grows high as the value ℓ grows and tends to $2\alpha - t + 1$. As the value ℓ becomes large, the length of the target vector $\boldsymbol{q}$ becomes relatively short because the determinant of the corresponding lattice becomes large. So the probability that the vector $\boldsymbol{q}$ is a shortest vector in the lattice becomes high, in which case we get the target vector $\boldsymbol{q}$ by Gaussian reduction.

On the other hand, the running time increases as the value ℓ becomes large. By Remark 3, the running time is proportional to 2^ℓ, namely an exponential time with respect to the value ℓ. In other words, the ratio of the exhaustive search part to the whole increases as the value ℓ becomes large. So it is important how to decrease the running time as the value ℓ becomes large.

Next we compare our method with Nuida et al.'s method [11] (Table 2) [†]. We set the value ℓ as $\ell = 2\alpha + 1 - t$, the minimum value which guarantees that the target vector $\boldsymbol{q}$ coincides with the first component vector (up to sign) of the basis obtained by Gaussian reduction (i.e., the success rate is always 100%).

In all cases, Nuida et al.'s method is much faster than our method. This is because the [ES] part in our method includes Gaussian reduction for each candidate λ, while the same part in Nuida et al.'s method consists only of the computation of $a\boldsymbol{v}_1 + b\boldsymbol{v}_2$ for each candidate (a, b). In fact, we can see the difference between two methods from the rightmost two columns in Table 2.

5.2 k RSA Moduli with $k \geq 3$

We implemented the factorization of k RSA moduli with implicit hints $p_1 \equiv \cdots \equiv p_k \pmod{2^t}$ for various pairs (t, ℓ) with additional bits ℓ, by using our method (Tables 3 and 4). We note that the notation "n/a" (resp. "$*$") means "not applicable" (resp. "less than 1 second").

For the partition of additional bits ℓ into a $(k-1)$-tuple $(\ell_2, \ldots, \ell_k)$ with $\ell = \ell_2 + \cdots + \ell_k$, we chose the partition as flat as possible. Namely, dividing ℓ by $k-1$, we let r (resp. s) be the quotient (resp. the remainder). We then set a $(k-1)$-tuple $(\ell_2, \ldots, \ell_k)$ as $(\overbrace{r+1, \ldots, r+1}^{s}, r, \ldots, r)$. In this case, the upper bound of the largest absolute value among elements of the matrix consisting of the basis obtained from the partition is equal to the smallest value, 2^{t+r+1}. (On the other hand, the upper bound is $2^{t+\ell}$ in the case where $(\ell_2, \ldots, \ell_k) = (\ell, 0, \ldots, 0)$.) So our partition above might lead to more efficient computation of the LLL algorithm (recall Theorem 2).

As in the case $k = 2$, we see from Tables 3 and 4 that the success rate grows high as the value ℓ becomes large. This implies that, under Assumption 1, the probability that the first component vector obtained by the LLL algorithm becomes a shortest vector in a given lattice grows high as the value ℓ becomes large. We can explain the situation as follows. For a (fixed) input (k, α) and a chosen value $\ell = \ell_2 + \cdots + \ell_k$, the target vector $\boldsymbol{q}$ is contained in L' associated with some $(\lambda_2, \ldots, \lambda_k)$ as in Theorem 6. On the other hand, the first component vector $\boldsymbol{v}_1$ obtained by the LLL algorithm applied to L' satisfies $\|\boldsymbol{v}_1\| \leq 2^{\frac{k-1}{2}} \lambda_1(L') \leq 2^{\frac{k-1}{2}} \|\boldsymbol{q}\|$. Since the right-hand side of this inequality is constant, the ratio of the volume of the set $\{\boldsymbol{x} \mid \|\boldsymbol{x}\| \leq 2^{\frac{k-1}{2}} \lambda_1(L')\}$ to the volume of L', $2^{(k-1)t+\ell}$, becomes small as ℓ becomes large. Therefore, it is probable that the number of vectors in the set $\{\boldsymbol{x} \in L' \mid \|\boldsymbol{x}\| \leq 2^{\frac{k-1}{2}} \lambda_1(L')\}$ becomes small as ℓ becomes large, which indicates that the probability that $\boldsymbol{v}_1$ becomes a shortest vector in L' becomes large (see Remark 1).

As in the case $k = 2$, the running time increases as the value ℓ becomes large (see Remark 4).

For Assumption 1, we remark that, unlike the case $k = 2$, given a lattice L of dimension more than two, there exists no effective condition for a non-zero vector in L to have the shortest length among $L \backslash \{\boldsymbol{o}\}$. Furthermore, the LLL algorithm does not guarantee the output of a shortest vector in a given lattice. Therefore, from the theoretical point of view, the original/our methods do not guarantee to get the target vector $\boldsymbol{q}$ even if the value ℓ becomes sufficiently large, which is indicated by the experimental results (Tables 3 and 4).

On the other hand, the heuristic bound on the value ℓ, namely $\ell \geq k\alpha - (k-1)t$, seems to be sufficiently reliable. That is, the success rate is nearly equal to 100 % in the case $\ell \geq k\alpha - (k-1)t + \epsilon$ with a few bits ϵ, which is the same consideration as in [8] (or Remark 2). From our experimental results, we guess that the size of ϵ is not negligible as k becomes large. In relation to that, we further see that, for a given tuple (N, k, α), the success rate depends only on the bit size of lattice, $(k-1)t + \ell$, applied to Gaussian reduction/the LLL algorithm. This gives an indication on the optimal value of ℓ. For example, in the case of the input parameter tuple $(N, k, \alpha) = (1000, 3, 250)$, if we want to factor integers with probability 70%, then we choose the parameter ℓ such that $(k-1)t + \ell = 2t + \ell = 753$. To make this assertion more clear, we need to implement our method for more cases. Contrary to the experimental evaluation, we get no

[†]By 100%, in the ratio column, we mean a probability more than or eqaul to 99.5%.

Table 2 Comparison between two methods with $(k, \alpha) = (2, 250)$.

number of shared bits t	additional bits ℓ ($\ell = 2\alpha + 1 - t$)	methods	running time (ave.)	ratio: [ES]/([LR]+[ES]) (ave.)
499	2	Nuida et al. [11]	2.800×10^{-4} s	19%
		our method	1.286×10^{-3} s	52%
496	5	Nuida et al. [11]	2.800×10^{-4} s	45%
		our method	5.880×10^{-3} s	77%
493	8	Nuida et al. [11]	6.000×10^{-4} s	61%
		our method	3.976×10^{-2} s	93%
490	11	Nuida et al. [11]	4.280×10^{-3} s	65%
		our method	3.095×10^{-1} s	99%
487	14	Nuida et al. [11]	4.552×10^{-2} s	94%
		our method	2.696 s	100%

Table 3 Attack for factoring k RSA moduli with $(k, \alpha) = (3, 250)$.

number of shared bits t	bound $\lceil \frac{k}{k-1}\alpha \rceil$	bound on ℓ : $k\alpha - (k-1)t$	additional bits ℓ	bits of lattice size: $(k-1)t + \ell$	success rate	running time		
						ave.	min.	max.
379	375	−8	0	758	99 %	*	*	*
375	375	0	0	750	1 %	*	*	*
			1	751	16 %	*	*	*
			3	753	73 %	*	*	*
			5	755	94 %	*	*	*
			7	757	99 %	*	*	*
368	375	14	0	736	0 %	n/a	n/a	n/a
			14	750	0 %	n/a	n/a	n/a
			15	751	10 %	3.45 s	*	8.09 s
			17	753	70 %	18.71 s	*	34.17 s
			19	755	95 %	64.37 s	*	140.14 s

Table 4 Attack for factoring k RSA moduli with $(k, \alpha) = (10, 350)$.

number of shared bits t	bound $\lceil \frac{k}{k-1}\alpha \rceil$	bound on ℓ : $k\alpha - (k-1)t$	additional bits ℓ	bits of lattice size: $(k-1)t + \ell$	success rate	running time		
						ave.	min.	max.
391	389	−19	0	3519	97 %	*	*	*
390	389	−10	0	3510	1 %	*	*	*
			1	3511	6 %	*	*	*
			3	3513	34 %	*	*	*
			5	3515	72 %	*	*	*
			7	3517	93 %	*	*	1.12 s
			9	3519	99 %	2.22 s	*	4.45 s
389	389	−1	0	3501	0 %	n/a	n/a	n/a
			9	3510	0 %	n/a	n/a	n/a
			10	3511	6 %	4.29 s	*	8.59 s
			12	3513	43 %	12.04 s	*	34.99 s
			14	3515	82 %	52.68 s	*	140.22 s
			16	3517	94 %	218.13 s	*	559.21 s
			18	3519	99 %	992.60 s	*	2203.58 s

theoretical estimation on the optimal value of ℓ.

6. Conclusion and Future Works

We described an improved method for factoring k RSA moduli with implicit hints. More precisely, we proposed a method for raising the modulus of the congruence obtained from the implicit hints. This leads to the increase of the determinant of the corresponding lattice by multiplicative factor 2^ℓ with respect to our setting parameter ℓ. As a result, from the experimental point of view, we see that our method increases the success rate for getting the target vector $\boldsymbol{q}$ consisting of non-trivial factors for a given system of k RSA moduli.

Our future works are as follows:

1. How to decrease the total number of the candidates when we construct the congruence modulo $2^{t+\ell_i}$ in our method;
2. Theoretical and experimental evaluations on the optimal value of ℓ;
3. Construction of an effective condition for a non-zero vector to be shortest among $L \setminus \{\boldsymbol{o}\}$ for a given lattice L of dimension more than two;
4. Construction of a condition for the LLL algorithm to output a shortest vector for a given lattice;
5. Generalization to integer factorizations with other implicit hints, for example implicit hints of most signifi-

cant and middle bits [2];
6. Application to other RSA moduli types, for example Okamoto-Uchiyama's RSA modulus p^2q [12] and Takagi's RSA modulus p^rq [17].

Acknowledgments

The authors thank the reviewers for their valuable comments which make this paper more valuable.

References

[1] D. Coppersmith, "Finding a small root of a bivariate integer equation; Factoring with high bits known," Advances in Cryptology, EUROCRYPT'96, Lecture Notes in Computer Science, vol.1070, pp.178–189, 1996.
[2] J.-C. Faugère, R. Marinier, and G. Renault, "Implicit factoring with shared most significant and middle bits," Public Key Cryptography, PKC 2010, Lecture Notes in Computer Science, vol.6056, pp.70–87, 2010.
[3] B. Helfrich, "Algorithms to construct Minkowski reduced and hermite reduced lattice bases," Theor. Comput. Sci., vol.41, pp.125–139, 1985.
[4] R. Kannan, "Minkowski's convex body theorem and integer programming," Math. Oper. Res., vol.12, no.3, pp.415–440, 1987.
[5] K. Kurosawa and T. Ueda, "How to factor N_1 and N_2 when $p_1 = p_2 \bmod 2^t$," Advances in Information and Computer Security, IWSEC 2013, Lecture Notes in Computer Science, vol.8231, pp.217–225, 2013.
[6] A.K. Lenstra and H.W. Lenstra, Jr., The Development of The Number Field Sieve, Lecture Notes in Mathematics, vol.1554, Springer-Verlag, 1993.
[7] A.K. Lenstra, H.W. Lenstra, and L. Lovász, "Factoring polynomials with rational coefficients," Math. Ann., vol.261, no.4, pp.515–534, 1982.
[8] A. May and M. Ritzenhofen, "Implicit factoring: On polynomial time factoring given only an implicit hint," Public Key Cryptography, PKC 2009, Lecture Notes in Computer Science, vol.5443, pp.1–14, 2009.
[9] C.D. Meyer, Matrix Analysis and Applied Linear Algebra, Cambridge University Press, 2000.
[10] H. Minkowski, Geometrie der Zahlen, Teubner-Verlag, 1896.
[11] K. Nuida, N. Itakura, and K. Kurosawa, "A simple and improved algorithm for integer factorization with implicit hints," Topics in Cryptology, CT-RSA 2015, Lecture Notes in Computer Science, vol.9048, pp.258–269, 2015.
[12] T. Okamoto and S. Uchiyama, "A new public-key cryptosystem as secure as factoring," Advances in Cryptology, EUROCRYPT'98, Lecture Notes in Computer Science, vol.1403, pp.308–318, 1998.
[13] J.M. Pollard, "Theorems on factorization and primality testing," Math. Proc. Camb. Phil. Soc., vol.76, no.3, pp.521–528, 1974.
[14] J.M. Pollard, "A Monte Carlo method for factorization," BIT, vol.15, no.3, pp.331–334, 1975.
[15] R.L. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," Commun. ACM, vol.21, no.2, pp.120–126, 1978.
[16] V. Shoup, NTL: A Library for doing Number Teory, http://www.shoup.net/ntl/
[17] T. Takagi, "Fast RSA-type cryptosystem modulo p^kq," Advances in Cryptology, CRYPTO'98, Lecture Notes in Computer Science, vol.1462, pp.318–326, 1998.

Ryuichi Harasawa received the B.S. degree in mathematics from Osaka City University, Japan, in 1997, and the M.S. and D.S. degrees in mathematics from Osaka University, Japan in 1999 and 2003, respectively. From 2003 to 2007, he was a Reseach Associate in the Department of Computer and Information Sciences at Nagasaki University. Since 2007, he has been an Assistant Professor in the Graduate School of Engineering at Nagasaki University. His research interests are in algebraic curve cryptograpy and its applications, provable security of cryptosystems, and computer algebra. Dr. Harasawa is a member of the Mathematical Society of Japan and the Japan Society for Industrial and Applied Mathematics. Graduate School of Engineering, Nagasaki University, 1-14 Bunkyomachi, Nagasaki-shi, Nagasaki, 852-8521, Japan

Heiwa Ryuto received the B.S. degree in engineering from Nagasaki University, Japan, in 2014. Currently he is in the second year of master course in the Graduate School of Engineering at Nagasaki University. Graduate School of Engineering, Nagasaki University, 1-14 Bunkyomachi, Nagasaki-shi, Nagasaki, 852-8521, Japan

Yutaka Sueyoshi received the B.S., M.S. and D.S. degrees in mathematics from Kyushu University, Japan, in 1975, 1977 and 1992, respectively. From 1979 to 1994, he was with the Department of Mathematics at Kyushu University. From 1994 to 2000, he was a Lecturer in the Graduate School of Mathematics at Kyushu University. From 2000 to 2012, he was an Associate Professor in the Department of Computer and Information Sciences at Nagasaki University. Since 2012, he has been a Professor in the Graduate School of Engineering at Nagasaki University. His research interests are in algebraic number theory, class field theory and algebraic theory of cryptography. Dr. Sueyoshi is a member of the Mathematical Society of Japan, the Japan Society for Industrial and Applied Mathematics, the Information Processing Society of Japan and American Mathematical Society. Graduate School of Engineering, Nagasaki University, 1-14 Bunkyomachi, Nagasaki-shi, Nagasaki, 852-8521, Japan