Minkowski sum based lattice construction for solving simultaneous modular equations and applications to RSA

Yoshinori Aono *

Abstract

We investigate a lattice construction method for the Coppersmith technique for finding small solutions of a modular equation. We consider its variant for simultaneous equations and propose a method to construct a lattice by combining lattices for solving single equations. As applications, we consider (i) a new RSA cryptanalysis for multiple short secret exponents, (ii) its partial key exposure situation, and (iii) investigating the hardness of finding a certain amount of LSBs of the RSA secret exponent. More precisely, our algorithm can factor an RSA modulus from $\ell \geq 2$ pairs of RSA public exponents with the common modulus corresponding to secret exponents smaller than $N^{(9\ell-5)/(12\ell+4)}$, which improves on the previously best known result $N^{(3\ell-1)/(4\ell+4)}$ by Sarkar and Maitra [41, 42]. For partial key exposure situation, we also can factor the modulus if $\beta - \delta/2 + 1/4 < (3\ell-1)(3\ell+1)$, where β and δ are bit-lengths $/n$ of the secret exponent and its exposed LSBs, respectively. Particularly, letting $\beta = 1$, which means that the secret exponent is full-sized, the necessary amount of exposed bits is $[5/2 - 2(3\ell-1)/(3\ell+1)]n$, which is less than n for $\ell \geq 3$. Suppose we have an algorithm that recovers the above amount of d from e and N satisfying $e \approx N$. We showed that N can be factored in polynomial time in $\log N$ under a heuristic assumption that the Coppersmith technique works. When ℓ becomes large, the necessary amount becomes $0.5n$ bits. Hence, we conclude that recovering the lower half of LSBs of d is polynomial time equivalent to the factoring under the heuristic assumption. From the last result, we propose *a half-amount conjecture* that roughly, factoring RSA modulus is polynomial-time equivalent to any continued bits of secret information such as $p, q, d, p+q$ and $p-q$ (or d_p and d_q for RSA-CRT). It is supported from several results, e.g., Coppersmith [12] shows that recovering the upper half of p is equivalent to factoring.

1 Introduction

Since the RSA cryptosystem [38] was proposed, its security has been intensively investigated. In particular, polynomial-time algorithms for recovering short secret exponents have been studied [45, 3].

There are two main strategies for recovering a secret exponent in this situation. The first type of approaches reduce this problem to the problem of finding a Diophantine approximation of a rational number. The continued fraction algorithm was used in this approach [45] and several lattice based extensions exist [25, 27, 22]. The other strategy is reduction to the problem of finding several roots of a polynomial equation. For this, the Coppersmith technique [11, 12] has been used [3, 7, 16, 1]. In this paper, we consider the latter technique.

Using the Coppersmith technique for finding small roots of a modular equation, Boneh and Durfee [3] proposed an algorithm for recovering a small secret exponent from the corresponding public key pair. The outline of their algorithm is (i) reducing the key finding problem to the problem of finding a small integer solution of a modular equation such as $F(x, y) \equiv 0 \pmod{W}$, (ii) using a lattice basis reduction algorithm, compute polynomials whose set of roots over integers contain all the small roots of the original equation, and finally (iii) finding all small integers in the common roots of the computed polynomials and recovering the secret exponent from them. Under several acceptable assumptions,

*National Institute of Information and Communications Technology

the attack is guaranteed to work when the secret exponent is smaller than $N^{0.292}$. This is the basic strategy of the lattice based attack and followed by many works.

Although the original Coppersmith technique was designed to treat a single modular equation, the method can be extended to multivariate simultaneous equations. For example, such situations are considered by Coron et al. [15] in fault attacks on RSA-CRT, by Sarkar and Maitra [41, 42] in attacks on RSA with small secret exponents, and by Herrmann [20] in analysis of the Φ-hiding assumption. Their approaches first construct a single multivariate modular equation whose solutions are also those of the simultaneous equations, and apply the standard Coppersmith technique. This may not be a better strategy from the viewpoint of lattice construction because it does not consider individual equations. May and Ritzenhofen [33] proposed an approach based on the Chinese remainder theorem to solve simultaneous univariate modular equations. In this paper, we study an extension of the Coppersmith technique that directly treats the original simultaneous multivariate equations. We expect that our algorithm will improve several lattice based attacks.

In our formulation of the Coppersmith technique, it requires to construct *a lattice of polynomial* L spanned by basis polynomials $g_1, \ldots, g_c$, and to find "small" elements in the lattice by using a lattice reduction algorithm. Here, the means of constructing better lattices is a key element. Several methods for a single equation have been intensively studied [7, 26, 29]. On the other hand, there has been little research on a general construction method for simultaneous equations [33, 37].

When one wants to solve simultaneous modular equations by the Coppersmith technique, it is expected that, in many situations, better lattices for a single equation are already known. Therefore, it is also likely that a better lattice can be constructed for the simultaneous equations by combining them. We propose this combining method using the Minkowski sum of integer tuple sets.

Related works: Automatic lattice constructions for the Coppersmith technique for a single modular equation have been widely studied. The first work by Coppersmith [11] gave a good lattice construction for any univariate modular equation. Recently, Aono et al. [2] has proven the optimality of this construction. Blömer and May [8] proposed a construction method for bivariate equations, and Jochemsz and May [26] improved this to a method for treating general multivariate equations. Another viewpoint was given by Kunihiro [29], who proposed a method for converting a lattice for an n-variable equation $f(x_1, \ldots, x_n) \equiv 0 \pmod{W}$ into a lattice for a new $(n+1)$-variable equation of the form $x_0 f(x_1, \ldots, x_n) + C \equiv 0 \pmod{W}$ where C is a constant. For simultaneous modular equations, May and Ritzenhofen [33] considered a Chinese remainder theorem based approach. They proposed a method for constructing a lattice in the univariate case and gave an application to RSA. Recently, Ritzenhofen [37] improved this approach to multivariate simultaneous equations and proposed a lattice construction method for equations with the common modulus. However, the case for coprime moduli was not solved (see [37, Section 5.4]). We consider this problem.

On the hardness for finding partial information on d have been considered an important problem. We introduce only two results on this theme. It is folklore that we can easily guess the upper half of d when $e = \mathrm{poly}(\log N)$. Since $p + q + 1 < N - \varphi < 3\sqrt{N}$, we can easily found the upper half of $\varphi(N)$. Thus, by the relation $ed = k\varphi(N) + 1$ holds for some $k < e$, we can guess the polynomial number of candidates of d's upper half (see for example [5]). Therefore, if we have an algorithm for recovering lower half of d, N is easily factored using this and the result by Coron and May [14] who proved the polynomial time equivalence between recovering d and factoring N. Thus, for small e, it can be shown that recovering lower half of d is polynomial time equivalent to factoring N. Boneh, Durfee and Frankel [4] proposed a method for recovering d from a certain amount of d when e is a prime number within the range $[N^{1/4}, N^{1/2}]$, and this attack was improved [16, 26, 1] for several situations. By the same argument, for e of these sizes, the polynomial time equivalence between recovering the amount of information on d and factoring N. In this paper, we prove this type equivalence for full-size e, not necessary to be a prime number, as one of application of our lattice construction.

1.1 Contributions of this work

Minkowski sum based lattice construction: We propose a method to construct a lattice for the Coppersmith technique for simultaneous modular equations. We consider simultaneous equations such as $F_1(x_1, y) \equiv 0 \pmod{W_1}$ and $F_2(x_2, y) \equiv 0 \pmod{W_2}$. Assume that we have better lattices spanned by the sets of polynomials $\{g_1^{(1)}, \ldots, g_{c_1}^{(1)}\}$ and $\{g_1^{(2)}, \ldots, g_{c_2}^{(2)}\}$ for the equations, respectively. Then, we propose the *Minkowski sum based lattice construction*, which is a method for generating a lattice basis

for solving the simultaneous equations, as a set of polynomials of the form $\sum a_\lambda g_\lambda^{(1)} \cdot g_{\lambda'}^{(2)}$. Our method defines the range of suffixes (λ, λ') and the coefficients a_λ of the combination.

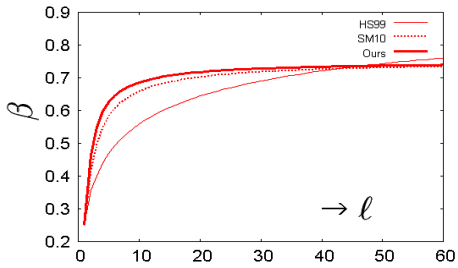
Cryptanalysis of multiple RSA short secret exponents and its partial key exposure situation: The above construction method can easily be extended to multivariate and multi-equation situations. By this, we improve the cryptanalysis of RSA with short secret exponents studied in [25, 22, 41, 42]. In this situation, the attacker has ℓ pairs of RSA public keys (e_k, N) with the common modulus, which correspond to secret exponents smaller than N^β for some $\beta \in (0, 1)$. Then, we prove that the RSA modulus is efficiently factored if

$$\beta < \frac{9\ell - 5}{12\ell + 4}.$$

Here, we assumed that all e_k's are full-sized i.e., they have the same bit sizes. This improves on the previously known best result by Sarkar and Maitra [42], which achieved $\beta < (3\ell - 1)/(4\ell + 4)$. For large ℓ, both values converge to $3/4$, while the result by Howgrave-Graham and Seifert [25] given below becomes one:

$$\beta < \frac{(2n+1)\cdot 2^n - (2n+1)\binom{n}{n/2}}{(2n-2)\cdot 2^n + (4n+2)\binom{n}{n/2}} \text{ if } n \text{ is even, and } \beta < \frac{(2n+1)\cdot 2^n - 4n\binom{n-1}{(n-1)/2}}{(2n-2)\cdot 2^n + 8n\binom{n-1}{(n-1)/2}} \text{ if } n \text{ is odd.}$$

These results are compared in Figure 1. Particularly, for $\ell \leq 46$, our bound is the largest.



ℓ	2	3	4
HS99: [25]	0.357	0.400	0.441
SM10: [42]	0.416	0.500	0.550
Our result	0.464	0.550	0.596

Figure 1: Comparison of the result by Howgrave-Graham and Seifert [25], Sarkar and Maitra [42] and our result

We then extend the attack to a partial key exposure situation studied in [4, 16], in which the attacker has ℓ tuples $(e_k, N, \widetilde{d_k})$ where e_k and N are RSA public keys, and each $\widetilde{d_k}$ is δn LSBs (least significant bits) of the corresponding secret exponent smaller than N^β. Then, we prove that the RSA modulus is efficiently factored if

$$\beta - \frac{\delta}{2} + \frac{1}{4} < \frac{3\ell - 1}{3\ell + 1}.$$

On the hardness of recovering partial information on d: Let $\beta = 1$ in the above result, i.e., the secret exponents are full-sized, do not need to be short. Then the inequality becomes

$$\delta < \frac{5}{2} - \frac{2(3\ell - 1)}{3\ell + 1}.$$

The right-hand size is less than one for $\ell \geq 3$.

We can argue the hardness relationship between factoring N and recovering $\widetilde{d_k}$ for large e. Suppose we have an algorithm that recovers δn LSBs of d from e and N if $\gcd(e, \varphi(N)) = 1$. If otherwise, it returns a random bit string, i.e., an useless string for factoring. Assume that it is not possible to decide whether the returned value is useful or useless. In this situation, we can construct an algorithm for factoring N in polynomial time in $\log N$ using our version of the Coppersmith technique and this recovering algorithm. Therefore, recovering the amount of LSBs of d corresponding to full-size e is polynomial-time equivalent to factoring N. When ℓ becomes large, the necessary amount becomes $0.5n$ bits. Thus, we conclude that recovering the lower half of the LSBs of d is harder than factoring under the heuristic assumption. This improves on the result by Coron and May [14] that proves the

polynomial-time equivalence between the problem of recovering d and that of factoring N. Note that their reduction needs no assumptions.

Following the above result, we propose the *half-amount conjecture* that factoring the RSA modulus is polynomial-time equivalent to recovering any continued bits of secret information such as

$p, q, d, p+q$ or $p-q$ for the original RSA,
d_p or d_q for RSA-CRT, and
$q^2, q^3, \ldots, q^{r-1}$ for Takagi's RSA.

For example, Coppersmith [12] showed that recovering the upper half of p is equivalent to factoring.

Computer experiments: To verify the correctness of our lattice construction, we perform computer experiments of the applications for RSA. Our experiments work well. Interestingly, in the partial key exposure situation, the range of β and δ that we can factor N is slightly larger than that derived by theory. It is expected that there exists a sublattice that achieves such good performance, as Boneh and Durfee [3] extracted a sublattice achieving $d < N^{0.292}$ from a lattice achieving $d < N^{0.284}$ in their cryptanalysis of RSA. However, we observed that the computed vector is a combination of all basis vectors; i.e., the desired sublattice cannot be constructed by simply selecting some vectors from the original basis.

Organization of this paper: Section 2 gives necessary definitions, technical lemmas, and an outline of the Coppersmith technique. In Section 3, we consider the Coppersmith technique for the simultaneous equations and propose our Minkowski sum based lattice construction. Sections 4 and 5 give applications to cryptanalysis of RSA with small secret exponents, its partial key exposure situations, and demonstrate the hardness of finding a certain amount of LSBs of d. Section 6 gives experimental results to verify our lattice construction. In Section 7, we suggest and discuss several open problems. Finally, we give concluding remarks in Section 8.

2 Preliminaries

Here we introduce some definitions and technical lemmas. For any positive integer a and b, let $[a]$ and $[a:b]$ be the set $\{1, \ldots, n\}$ and $\{a, a+1, \ldots, b-1, b\}$, respectively. For natural numbers x, A and N, the notation $|x| < A \pmod N$ means that $0 \le x < A$ or $N - A < x < N$ holds.

We use $\prec$ to denote the lexicographic order between integer tuples. For example, consider two 2-tuples, (i_1, i_2) and (i_1', i_2'), then $(i_1, i_2) \prec (i_1', i_2')$ means that $i_1 < i_1'$ or $[i_1 = i_1'$ and $i_2 < i_2']$ holds. We also use this to order monomials; e.g., $x_1^{i_1} x_2^{i_2} \prec x_1^{i_1'} x_2^{i_2'} \Leftrightarrow (i_1, i_2) \prec (i_1', i_2')$. Here, we neglect the coefficients. These notations are used for general n-tuples and n-variable monomials. We use $x_1, x_2, \ldots, x_{n-1}$ and y to denote the variables, and fix the priority of variables as $y \prec x_{n-1} \prec \cdots \prec x_1$ to order the n-variable monomials. For example, consider four variables, x_1, x_2, x_3, y, and monomials $3x_2^2 x_3$ and $x_1^2 x_2^3 y$. Then, $3x_2^2 x_3 \prec x_1^2 x_2^3 y$ holds since the corresponding tuples are $(0, 2, 1, 0)$ and $(2, 3, 0, 1)$, respectively. Note that for any integer tuples T_1, T_2, S_1, S_2 of the same dimension, $T_1 \prec S_1$ and $T_2 \prec S_2$ implies that $T_1 + T_2 \prec S_1 + S_2$.

With respect to the above order, we can define the maximum element in a polynomial $f(x_1, \ldots, x_\ell, y)$. Let $a x_1^{i_1} \cdots x_\ell^{i_\ell} y^j$ be the non-zero maximum monomial in f. Then, we call it the *head term* of f and denote it by $\mathrm{HT}(f)$. We also call a, $x_1^{i_1} \cdots x_\ell^{i_\ell} y^j$ and $(i_1, \ldots, i_\ell, j)$ *head coefficient*, *head monomial* and *head index*, and denote them by $\mathrm{HC}(f)$, $\mathrm{HM}(f)$ and $\mathrm{HI}(f)$, respectively.

Minkowski sum: Let A and B be finite subsets of $\mathbb{Z}^n$, then their Minkowski sum is defined by the set

$$A \boxplus B = \{(a_1 + b_1, \ldots, a_n + b_n) : (a_1, \ldots, a_n) \in A, (b_1, \ldots, b_n) \in B\}.$$

Note that the sum of three or more sets is similarly defined.

2.1 Overview of the Coppersmith technique

We introduce the Coppersmith technique [11, 12] with necessary definitions and technical lemmas. Our formulation is due to Howgrave-Graham [21] and Aono et al. [2]. See these papers for detailed analysis of the algorithm.

We fix a polynomial $F(x, y) \in \mathbb{Z}[x, y]$ and $X, Y, W \in \mathbb{N}$. Then consider the problem of finding all integer solutions of

$$F(x, y) \equiv 0 \pmod W \tag{1}$$

within the range of $|x| < X$ and $|y| < Y$. While this problem is generally not easy, the Coppersmith technique efficiently solves it if X and Y are much smaller than W. We can easily extend the problem and the technique to three or more variables.

The basic strategy of the Coppersmith technique can be outlined as follows. Fix an integer $m \geq 2$ and consider a set L of polynomials $g(x, y) \in \mathbb{Z}[x, y]$ satisfying

$$\forall x, y \in \mathbb{Z}\ [F(x, y) \equiv 0 \pmod{W} \Rightarrow g(x, y) \equiv 0 \pmod{W^m}]. \tag{2}$$

Note that L forms a lattice, i.e., it can easily see that $g_1, g_2 \in L \Rightarrow g_1 - g_2 \in L$. Next, find polynomials $g(x, y) \in L$ satisfying

$$\forall x, y \in \mathbb{Z}, |x| < X, |y| < Y\ [g(x, y) \equiv 0 \pmod{W^m} \Rightarrow g(x, y) = 0]. \tag{3}$$

Suppose two algebraically independent polynomials are found, then the original equation (1) can be converted to simultaneous equations over integers, which are easily solved by the resultant technique [19] or the Gröbner basis technique [10]. As we explain below, a polynomial with small coefficients satisfies (3). Then, our tasks are to construct a polynomial lattice L, and to find such polynomials in L.

Many algorithms are proposed to find small elements in a lattice, For example, the LLL algorithm [30] is widely used. Unfortunately, most of them are designed for treating lattices in Euclidean spaces $\mathbb{R}^n$ w.r.t. the standard Euclidean norms. To use them as a subroutine, a polynomial lattice needs to be converted to a Euclidean lattice.

Converting polynomials to vectors: For a polynomial $g(x, y) = \sum_{i,j} a_{i,j} x^i y^j$ and parameters X and Y, define the *vectorization* of the polynomial by

$$\mathcal{V}(g; X, Y) = (a_{0,0}, a_{1,0}X, \ldots, a_{i_w,j_w} X^{i_w} Y^{i_w}).$$

Thus, it maps each term $a_{i,j}x^i y^j$ to each coordinate $a_{i,j}X^i Y^j$, respectively. It is a linear mapping with respect to g. We note that the sequence of tuples $\{(i_k, j_k)\}_{k=1}^{w}$ is taken so that it covers all non-zero terms in $g(x, y)$. In the Coppersmith technique, some polynomials need to be converted to vectors. In this situation, we fix a sequence of tuples. We define the *polynomial norm* w.r.t. the parameters X, Y by $|\mathcal{V}(g; X, Y)|$. W.r.t. this norm, the following lemma holds.

Lemma 1. (Howgrave-Graham [21], generalized in [26]) Fix $X, Y, W \in \mathbb{N}$. Let $g(x, y) \in \mathbb{Z}[x, y]$ be a polynomial consisting of w terms, and $|\mathcal{V}(g; X, Y)| < W/\sqrt{w}$ holds. Then we have

$$\forall x, y \in \mathbb{Z}, |x| < X, |y| < Y\ [g(x, y) \equiv 0 \pmod{W} \Leftrightarrow g(x, y) = 0].$$

Hence, if a polynomial lattice L is given, our task is to find independent polynomials satisfying the above lemma, which is performed by finding short vectors in a Euclidean lattice converted from L using certain parameters.

Lattices: We introduce the notion of lattices in Euclidean spaces and polynomial spaces. Consider linearly independent vectors $\mathbf{b}_1, \ldots, \mathbf{b}_c$ in $\mathbb{Z}^{\tilde{c}}$ where $\tilde{c} \geq c$. Then, the *Euclidean lattice* spanned by them is defined by

$$L(\mathbf{b}_1, \ldots, \mathbf{b}_c) = \{a_1\mathbf{b}_1 + \cdots + a_c\mathbf{b}_c\ :\ a_k \in \mathbb{Z} \text{ for } k \in [c]\}.$$

We call $\mathbf{b}_1, \ldots, \mathbf{b}_c$ the basis vectors. Following many papers, we assume that a lattice is represented by its basis vectors.

To find short vectors in a lattice, we use the LLL algorithm [30] which computes an LLL-reduced basis from a given basis. The following theorem bounds the lengths of first vectors in such bases.

Theorem 1. [7] Let L be a Euclidean lattice and $\mathbf{v}_1, \ldots, \mathbf{v}_c$ be its LLL-reduced basis. Then, the following inequality holds for $k \in [c]$.

$$||\mathbf{v}_k|| \leq 2^{\{(c(c-1)+(k-1)(k-2)\}/4(c-k+1)} |\det(L)|^{1/(c-k+1)} \tag{4}$$

Here, $\det(L)$ is the *lattice determinant* that is defined by using the Gram-Schmidt orthogonal basis $\mathbf{v}_1^*, \ldots, \mathbf{v}_c^*$ as $\det(L) = \prod_{i=1}^{c} ||\mathbf{v}_i^*||$.

Input	$F(x,y) \in \mathbb{Z}[x,y]$, $W, X, Y \in \mathbb{N}$; **Parameters** $c \geq 1$ and $m \geq 2$;
Output	All integer solutions of $F(x,y) \equiv 0 \pmod{W}$ satisfying $\|x\| < X$ and $\|y\| < Y$.
Step 1:	W.r.t. $F(x,y)$ and W, define a sequence of independent polynomials $\mathbf{G} = \{g_1, \ldots, g_c\} \subset \mathbb{Z}[x,y]$ satisfying (2).
Step 2:	Convert the polynomial lattice to $L(\mathbf{G}; X, Y)$ using X and Y; Compute its LLL-reduced basis $\mathbf{v}_1, \ldots, \mathbf{v}_k$.
Step 3:	Convert $h_1(x,y)$ and $h_2(x,y)$ from $\mathbf{v}_1$ and $\mathbf{v}_2$ respectively. Find all small integer solutions of $h_1(x,y) = h_2(x,y) = 0$, and output them.

Figure 2: Algorithm of the Coppersmith technique

Next, we define the polynomial lattice. Let $\mathbf{G} = \{g_1, \ldots, g_c\}$ be a sequence of linearly independent polynomials in $\mathbb{Z}[x,y]$. Then, the *polynomial lattice* spanned by them is defined by

$$L(\mathbf{G}) = L(g_1, \ldots, g_c) = \{a_1 g_1 + \cdots + a_c g_c \ : \ a_i \in \mathbb{Z} \text{ for } k \in [c]\}.$$

We also consider the vectorization of polynomial lattices; i.e., for a basis $\mathbf{G} = \{g_1, \ldots, g_c\}$, consider their vectorization $\mathcal{V}(g_1; X, Y), \ldots, \mathcal{V}(g_c; X, Y)$ w.r.t. parameters X and Y. Here, the tuple sequence is assumed to be fixed. Then, define the vectorization of $L(\mathbf{G})$ by the Euclidean lattice spanned by these vectors, and let it be $L(\mathbf{G}; X, Y)$. We use $\det(\mathbf{G}; X, Y)$ to denote the determinant of $L(\mathbf{G}; X, Y)$.

Outline and a working condition for the Coppersmith technique: For fixed X and Y, suppose we have a polynomial lattice $L(\mathbf{G})$ spanned by c independent polynomials satisfying (2), and it holds that

$$2^{c/4} \det(\mathbf{G}; X, Y)^{1/c} < N^m / w. \tag{5}$$

Here, w is the length of tuple sequence used at vectorization, which is equal to the Euclidean dimension of $L(\mathbf{G}; X, Y)$, and bounds upper the number of terms of any polynomials in $L(\mathbf{G})$. Then, compute the LLL-reduced basis of $L(\mathbf{G}; X, Y)$. By Theorem 1, the first two vectors $\mathbf{v}_1$ and $\mathbf{v}_2$ in the reduced basis are shorter than N^m/w. Hence, the corresponding polynomials, i.e., $h_k(x,y)$ satisfying $\mathbf{v}_k = \mathcal{V}(h_k; X, Y)$ for $k = 1, 2$, also satisfy $|\mathcal{V}(h_k; X, Y)| \leq N^m/w$. Thus, by Lemma 1, these polynomials satisfy

$$\forall x, y \in \mathbb{Z}, |x| < X, |y| < Y \ [F(x,y) \equiv 0 \pmod{W} \Rightarrow h_k(x,y) = 0].$$

Finally, finding small integer solutions of $h_1(x,y) = h_2(x,y) = 0$, we obtain the desired solutions. Figure 2 shows the outline of the Coppersmith technique. As in many previous works, we regard the following simplified condition as a working condition.

$$\det(\mathbf{G}; X, Y)^{1/c} < N^m \tag{6}$$

In many applications, the crucial problem is to construct a lattice $\mathbf{G}$ satisfying (6) for X and Y as large as possible.

The algebraic independence of polynomials $h_k(x,y)$ is necessary to solve the final simultaneous equations over the integers. Unfortunately, this is generally not guaranteed. In this paper, again following previous works, we assume this algebraic independence and justify it by computer experiments.

3 Coppersmith technique for simultaneous equations

We consider a variant of the Coppersmith technique for the simultaneous equations, and propose a new method to construct polynomial lattices. For readability, we consider the following three variable simultaneous equations with two equations having the shared variable y. Here, if no variable is shared, the simultaneous equations have no meaning. Again note that the generalization for more variables and more equations is trivial, but the notations become very complicated.

$$\begin{aligned} F_1(x_1, y) &\equiv 0 \pmod{W_1} \\ F_2(x_2, y) &\equiv 0 \pmod{W_2} \end{aligned} \tag{7}$$

Our objective is to find all integer solutions within the range of $|x_1| < X_1$, $|x_2| < X_2$ and $|y| < Y$.

The algorithm is similar to that in Figure 2; only the input and condition in Step 1 are different. The inputs are two polynomials, $F_1(x_1, y)$ and $F_2(x_2, y)$, moduli W_1 and W_2, range of variables X_1, X_2, Y, and parameters c and m. The condition for three variable polynomials $g_i(x_1, x_2, y)$ is

$$\forall x_1, x_2, y \in \mathbb{Z}, \left[\begin{array}{l} F_1(x_1, y) \equiv 0 \pmod{W_1} \\ F_2(x_2, y) \equiv 0 \pmod{W_2} \end{array} \right. \Rightarrow g_i(x_1, x_2, y) \equiv 0 \pmod{(W_1 W_2)^m} \Big]. \tag{8}$$

For a lattice $L(\mathbf{G})$ with basis $\mathbf{G} = \{g_1, \ldots, g_c\}$, compute the LLL-reduced basis of $L(\mathbf{G}; X_1, X_2, Y)$. By the same argument as that in Section 2.1, we can prove the technique works if

$$\det(\mathbf{G}; X_1, X_2, Y)^{1/c} < (W_1 W_2)^m.$$

Hence, the problem is also finding the means of constructing better polynomial lattices.

3.1 Minkowski sum based lattice construction

When we want to solve simultaneous equations by the Coppersmith technique, it can be expected that we already know better lattice constructions for solving individual equations. Even if this is not true, we may expect that lattice construction for a single equation will be easier than for simultaneous equations. We propose a method for constructing a lattice by combining lattices for solving single equations. Again, consider the simultaneous equations (7).

For $k = 1, 2$, let $L(\mathbf{G}_k)$ be a polynomial lattice for solving $F_k(x_k, y) \equiv 0 \pmod{W_k}$ and its basis be $\mathbf{G}_k = \{g_1^{(k)}, \ldots, g_{c_k}^{(k)}\}$. Here we assume that the parameter m is fixed. Then for any $\ell_1 \in [c_1]$ and $\ell_2 \in [c_2]$, the polynomial $g_{\ell_1}^{(1)} \cdot g_{\ell_2}^{(2)}$ satisfies (8). Hence, the set

$$\mathcal{A} = \left\{ \sum_{\ell_1 \in [c_1], \ell_2 \in [c_2]} a_{\ell_1, \ell_2} g_{\ell_1}^{(1)} g_{\ell_2}^{(2)} : a_{\ell_1, \ell_2} \in \mathbb{Z} \right\}$$

is a polynomial lattice for solving the simultaneous equations. Unfortunately, since the polynomials $\{g_{\ell_1}^{(1)} g_{\ell_2}^{(2)}\}_{\ell_1, \ell_2}$ are not generally independent over the integers, it cannot explicitly obtain the basis of $\mathcal{A}$ and its determinant. Instead, we consider a sublattice of $\mathcal{A}$ and define its basis by using the Minkowski sum of indices.

We can assume that each basis $\mathbf{G}_k$ has a strictly increasing degree order, i.e., $\mathrm{HM}(g_1^{(k)}) \prec \cdots \prec \mathrm{HM}(g_{c_k}^{(k)})$ holds for $k = 1, 2$. If this is not true, an equivalent basis having this property can be computed by Gaussian elimination; see [2]. Then, for each k, consider the set of indices $I_k = \{\mathrm{HI}(g_\ell^{(k)}) : \ell \in [c_k]\} \subset \mathbb{Z}^3$ and let their Minkowski sum be I_+. Noting that the elements of I_1 and I_2 have the form $(i_1, 0, j)$ and $(0, i_2, j)$, respectively. For every $(i_1, i_2, j) \in I_+$, define the polynomial $g_{i_1, i_2, j}^+$ to be

$$g_{i_1, i_2, j}^+ = \sum_{(*)} a_\lambda g_\lambda^{(1)} g_{\lambda'}^{(2)}. \tag{9}$$

Here, the range of sum $(*)$ is over all suffix pairs (λ, λ') satisfying

$$\mathrm{HM}(g_\lambda^{(1)} g_{\lambda'}^{(2)}) = x_1^{i_1} x_2^{i_2} y^j$$

and the coefficients a_λ are defined so that

$$\mathrm{HC}(g_{i_1, i_2, j}^+) = \operatorname*{GCD}_{(*)} (\mathrm{HC}(g_\lambda^{(1)} g_{\lambda'}^{(2)})), \tag{10}$$

i.e., the greatest common divisor of all head coefficients within the range. It is easy to see that the polynomial satisfies (8). We define the polynomial basis by $\mathbf{G}_+ = \{g_{(i_1, i_2, j)}^+ : (i_1, i_2, j) \in I_+\}$. Here, it

is clear that the basis polynomials are linearly independent since the head monomials are distinct. We call the polynomial lattice $L(\mathbf{G}_+)$ the *Minkowski sum lattice* of $L(\mathbf{G}_1)$ and $L(\mathbf{G}_2)$. Clearly, $L(\mathbf{G}_+) \subset \mathcal{A}$ holds.

The basic strategy of this construction is to minimize the head coefficient of $g^+_{i_1,i_2,j}$ over all the possible integer combinations. It can be expected that the determinant of the combined lattice is reduced. Note that a combination of a_λ that attains (10) is generally not unique. Hence, care needs to be taken regarding the determinant if the lattice is not triangular. If the lattice is lower triangular, the determinant, which is computed by $\prod |\mathrm{HC}(g^+_{i_1,i_2,j})| X_1^{i_1} X_2^{i_2} Y^j$, is not changed for any allowed combination of a_λ. We consider the latter situation in our applications for RSA cryptanalysis.

Comparison with previous strategy: Many previous works, first consider the single equation by multiplying both sides in (7) as

$$F_1(x_1, y)F_2(x_2, y) \equiv 0 \pmod{W_1W_2}.$$

Then consider a lattice of polynomials $g_i(x_1, x_2, y)$ satisfying

$$\forall x_1, x_2, y \in \mathbb{Z}, [F_1(x_1, y)F_2(x_2, y) \equiv 0 \pmod{W_1W_2} \Rightarrow g_i(x_1, x_2, y) \equiv 0 \pmod{(W_1W_2)^m}]$$

using the lattice construction strategy by Jochemsz and May [26]. Their strategy only considers polynomials of the form $x_1^{i_1} x_2^{i_2} y^j (F_1(x_1, y)F_2(x_2, y))^t (W_1W_2)^{m-t}$, whose variety of polynomial selection is clearly smaller than our Minkowski sum construction. We provide a small example for our attack on RSA with the short secret exponents in Appendix B.

3.2 Minkowski sum of lower triangular lattices

Suppose the lattices for single equations are lower triangular, that is, there exist sequences of tuples $\{(i_1(\ell), j_1(\ell))\}_{\ell=1}^{c_1}$ and $\{(i_2(\ell), j_2(\ell))\}_{\ell=1}^{c_2}$, the polynomials in bases $\mathbf{G}_k$ can be written as

$$g^{(1)}_\ell = \sum_{\ell'=1}^{\ell} a_{\ell,\ell'} x_1^{i_1(\ell')} y^{j_1(\ell')} \text{ and } g^{(2)}_\ell = \sum_{\ell'=1}^{\ell} b_{\ell,\ell'} x_2^{i_2(\ell')} y^{j_2(\ell')}, \text{ where } a_{\ell,\ell} \neq 0 \text{ and } b_{\ell,\ell} \neq 0.$$

In this case, w.r.t. the above sequences of tuples, the Euclidean lattices $L(\mathbf{G}_k; X_k, Y)$ are lower triangular. Since the determinant analysis is easy, lattices of this form are usually used in many applications. We consider the Minkowski sum lattice of them and show that it is also lower triangular.

Theorem 2. For $k = 1, 2$, assume that the polynomial lattice basis $\mathbf{G}_k = \{g^{(k)}_1, \ldots, g^{(k)}_{c_k}\}$ has a strictly increasing degree order, and that they are lower triangular. Then the Minkowski sum lattice $L(\mathbf{G}_+)$ is also lower triangular.

Proof. Recall that $I_k = \{\mathrm{HI}(g^{(k)}_\ell) : \ell \in [c_k]\} \subset \mathbb{Z}^3$ and $I_+ = I_1 \boxplus I_2$. Fix any $(i_1, i_2, j) \in I_+$. We prove that $g^+_{i_1,i_2,j}$ can be written as

$$g^+_{i_1,i_2,j} = a x_1^{i_1} x_2^{i_2} y^j + (\text{terms whose indices are in } I_+ \text{ and smaller than } x_1^{i_1} x_2^{i_2} y^j).$$

By construction, it suffices to show that any $g^{(1)}_\lambda g^{(2)}_{\lambda'}$ in (9) can be written in this form. Fix a suffix pair (λ, λ') and consider the monomial expansion of $g^{(1)}_\lambda g^{(2)}_{\lambda'}$. Then, any term in this polynomial is a sum of the product of terms in $g^{(1)}_\lambda$ and $g^{(2)}_{\lambda'}$. Let them be $A x_1^{\bar{i_1}} y^{\bar{j_1}}$ and $B x_2^{\bar{i_2}} y^{\bar{j_2}}$; i.e., $g^{(1)}_\lambda g^{(2)}_{\lambda'}$ can be written as a sum of $AB x_1^{\bar{i_1}} x_2^{\bar{i_2}} y^{\bar{j_1}+\bar{j_2}}$.

Since $L(\mathbf{G}_k)$ are lower triangular, $(\bar{i_1}, 0, \bar{j_1}) \in I_1$ and $(0, \bar{i_2}, \bar{j_2}) \in I_2$ hold. Thus, $(\bar{i_1}, \bar{i_2}, \bar{j_1} + \bar{j_2}) \in I_+$. By construction, $(\bar{i_1}, 0, \bar{j_1}) \preceq \mathrm{HI}(g^{(1)}_\lambda)$ and $(0, \bar{i_2}, \bar{j_2}) \preceq \mathrm{HI}(g^{(2)}_{\lambda'})$. Hence, we have $(\bar{i_1}, \bar{i_2}, \bar{j_1} + \bar{j_2}) \preceq \mathrm{HI}(g^{(1)}_\lambda) + \mathrm{HI}(g^{(2)}_{\lambda'}) = \mathrm{HI}(g^+_{i_1,i_2,j})$. □

Minkowski sum of three of more lattices: As with the situations of two lattices, the Minkowski sum construction can be extended to three or more lattices. Moreover, we can show that the Minkowski sum lattice of lower triangular lattices is also lower triangular. The detailed construction, theorem, and its proof are left to Appendix A.

4 Cryptanalysis of RSA with short secret exponents

As an application of our Minkowski sum lattice construction, we analyze the RSA with multiple short secret exponents with a common modulus.

Notations: We use the standard notations for the RSA cryptography. That is, p and q are large primes, and let their product be the RSA modulus N. e and d are used to denote the public exponent and secret exponent, respectively. The basic relation $ed \equiv 1 \pmod{\varphi(N)}$ holds. Following [3], we assume that $e \approx N$ and $p + q < 3N^{0.5}$.

In this section, we consider the situation in which the attacker has ℓ pairs of public keys with a common modulus, let them be $(e_1, N), \ldots, (e_\ell, N)$, which correspond to small secret exponents satisfying $d_1, \ldots, d_\ell < N^\beta$ for some $\beta \in (0, 1)$. For simplicity, we assume that e_i and e_j are coprime to each other for $i \neq j$.

4.1 RSA equation and its limit

Following the work of Sarkar and Maitra [41, 42] (see Boneh and Durfee [3] for deriving single equation), it can prove that the simultaneous equations

$$\begin{aligned} F_1(x_1, y) &= -1 + x_1(y + N) \equiv 0 \pmod{e_1} \\ &\vdots \\ F_\ell(x_\ell, y) &= -1 + x_\ell(y + N) \equiv 0 \pmod{e_\ell} \end{aligned} \tag{11}$$

have a small solution satisfying

$$|x_k| < N^\beta, \text{ for } k \in [\ell] \text{ and } |y| < 3N^{0.5}. \tag{12}$$

Hence, our objective here is to find this solution by the Coppersmith technique.

On the other hand, if β is not small, the solution within the range is not unique. In this situation, the number of solutions becomes exponential in $\log N$; thus, no polynomial-time algorithm exists. We derive a heuristic condition in which a polynomial-time algorithm exists.

For any y' such that $y' + N$ is coprime to all e_k, setting $x'_k = (y' + N)^{-1} \pmod{e_l}$, the tuple $(x'_1, \ldots, x'_\ell, y')$ is a solution of (11). Unfortunately, this solution cannot be used to recover the secret keys since it does not generally satisfy (12).

Following the argument in [1], we consider a heuristic condition for β so that a found solution satisfying (12) is expected to be usable for recovering the secret exponents. Assume that the solutions of (11) are random numbers on $\{1, \ldots, N\}^{\ell+1}$. Since the number of solutions is smaller than N, we expect the number of solutions within the range (12) to be smaller than

$$N \cdot \frac{(2N^\beta)^\ell \times (6N^{0.5})}{N^{\ell+1}} \approx N^{0.5+\ell(\beta-1)}.$$

Thus, if $\beta < (\ell - 0.5)/\ell$, then the above is smaller than one. From this observation, we set the following heuristic assumption.

Heuristic assumption: For a natural number ℓ, assume that

$$\beta < \frac{\ell - 0.5}{\ell}. \tag{13}$$

Then, the equation (11) has only one solution $(x'_1, \ldots, x'_\ell, y')$ within the range of (12). By using this solution, we can also recover the corresponding secret keys d_k and can factor N.

4.2 Our lattice construction and bound

Here we give our polynomial lattice to solve the simultaneous equations (11) and a new security analysis of RSA. As mentioned in Section 3.1, assume that lattices for solving single equation $F_k(x_k, y) =$

$-1+x_k(y+N) \equiv 0 \pmod{e_k}$ are given. We follow the work of Boneh and Durfee [3], and employ their simple lower triangular lattice: fix an integer $m \geq 2$ and set

$$g_{i,j}^{(k)}(x_k, y) = x_k^{i-j} F_k(x_k, y) e_k^{m-j} \text{ and } \mathbf{G}_k = \{g_{i,j}^{(k)} : (i,j) \in \mathbb{Z}^2, 0 \leq j \leq i \leq m\} \tag{14}$$

for $k = 1, \ldots, \ell$. It is clear that $g_{i,j}^{(k)}(x_k, y)$ satisfies (2) w.r.t. $F_k(x_k, y) \equiv 0 \pmod{e_k}$ and m.

For each k, ordering its basis in the lexicographic order in suffixes (i,j), the polynomial sequence has strictly increasing order since $\mathrm{HM}(g_{i,j}^{(k)}) = x_k^i y^j$ and $\mathrm{HI}(g_{i,j}^{(k)}) = (0, \ldots, 0, i, 0, \ldots, 0, j) \in \mathbb{Z}^{\ell+1}$ (the k-th and $\ell+1$-th coordinates are i and j, respectively). As shown in [3], the lattice $L(\mathbf{G}_k; X_k, Y)$ is lower triangular. Thus these bases satisfy the assumption of Theorem 4 in Appendix A, and the Minkowski sum lattice $L(\mathbf{G}_+)$ is also lower triangular.

We explicitly give the Minkowski sum lattice. The index set corresponding to $\mathbf{G}_k$ is given by $I_k = \{(0, \ldots, 0, i, \ldots, 0, j) : (i,j) \in \mathbb{Z},\ 0 \leq j \leq i \leq m\}$ and their Minkowski sum is

$$I_+ = I_1 \boxplus \cdots \boxplus I_\ell = \{(i_1, \ldots, i_\ell, j) : 0 \leq i_1, \ldots, i_\ell \leq m \text{ and } 0 \leq j \leq i_1 + \cdots + i_\ell\}.$$

Following Appendix A, for each $(i_1, \ldots, i_\ell, j) \in I_+$, construct a polynomial by

$$g_{i_1,\ldots,i_\ell,j} = \sum_{j_1,\ldots,j_\ell} a_{j_1,\ldots,j_\ell} \cdot g_{i_1,j_1}^{(1)} g_{i_2,j_2}^{(2)} \cdots g_{i_\ell,j_\ell}^{(\ell)}.$$

Following (9), the sum is over indices such that $\mathrm{HM}(g_{i_1,j_1}^{(1)} g_{i_2,j_2}^{(2)} \cdots g_{i_\ell,j_\ell}^{(\ell)}) = x_1^{i_1} \cdots x_\ell^{i_\ell} y^j$. In this situation, i_k are fixed, and $(j_1, \ldots, j_\ell)$ moves over all integer tuples subject to $0 \leq j_k \leq i_k$ and $j_1 + \cdots + j_\ell = j$. Next we consider the coefficients; again as mentioned in Section 3.1, the coefficients $a_{j_1,\ldots,j_\ell}$ are selected so that

$$\mathrm{HC}(g_{i_1,\ldots,i_\ell,j}) = \operatorname*{GCD}_{j_1,\ldots,j_\ell} \left(\mathrm{HC}(g_{i_1,j_1}^{(1)} g_{i_2,j_2}^{(2)} \cdots g_{i_\ell,j_\ell}^{(\ell)}) \right).$$

Note that $\mathrm{HC}(g_{i_1,j_1}^{(1)} g_{i_2,j_2}^{(2)} \cdots g_{i_\ell,j_\ell}^{(\ell)}) = e_1^{m-j_1} \cdots e_\ell^{m-j_\ell}$. Since j_k can move from zero to $\min(i_k, j)$, the greatest common divisor is $e_1^{m-\min(i_1,j)} \cdots e_\ell^{m-\min(i_\ell,j)}$. Thus, we can take $a_{j_1,\ldots,j_\ell}$ so that the head coefficient of $g_{i_1,\ldots,i_\ell,j}$ is this value.

Then, we set the Minkowski sum lattice by $\mathbf{G}_+ = \{g_{i_1,\ldots,i_\ell,j} : (i_1, \ldots, i_\ell, j) \in I_+\}$ and the order is the lexicographic order of suffixes. By Theorem 4, the converted lattice $L(\mathbf{G}_+; X_1, \ldots, X_\ell, Y)$ is lower triangular. The diagonal element corresponding to $(i_1, \ldots, i_\ell, j)$ is

$$\mathrm{HC}(g_{i_1,\ldots,i_\ell,j}) \times X_1^{i_1} \cdots X_\ell^{i_\ell} Y^j = e_1^{m-\min(i_1,j)} \cdots e_\ell^{m-\min(i_\ell,j)} X_1^{i_1} \cdots X_\ell^{i_\ell} Y^j.$$

Therefore, the determinant is

$$\det(\mathbf{G}_+; X_1, \ldots, X_\ell, Y) = \prod_{(i_1,\ldots,i_\ell,j) \in I_+} \left[e_1^{m-\min(i_1,j)} \cdots e_\ell^{m-\min(i_\ell,j)} X_1^{i_1} \cdots X_\ell^{i_\ell} Y^j \right].$$

As with the same argument in Section 2.1, the Coppersmith technique works if

$$\det(\mathbf{G}_+; X_1, \ldots, X_\ell, Y)^{1/|I_+|} < (e_1 \cdots e_\ell)^m.$$

Here, $|I_+|$ denotes the number of elements in I_+. Using approximations $e_k \approx N$ for $k \in [\ell]$, $X_1 = \cdots = X_\ell = N^\beta$ and $Y \approx N^{0.5}$, the condition can be rewritten as

$$\sum_{(i_1,\ldots,i_\ell,j) \in I_+} \left[0.5j + (i_1 + \cdots + i_\ell)\beta - \sum_{k=1}^{\ell} \min(i_k, j) \right] < 0. \tag{15}$$

By computing the left-hand side (see Appendix C), we derive the condition

$$\left(-\frac{3}{16}\ell^2 + \frac{5}{48}\ell + \left(\frac{\ell^2}{4} + \frac{\ell}{12} \right) \beta \right) m^{\ell+2} + o(m^{\ell+2}) < 0.$$

Thus, when m is sufficiently large, this condition is

$$\beta < \frac{9\ell - 5}{12\ell + 4}. \tag{16}$$

Appendix B gives a small example of this construction for $\ell = 2$ and $m = 1$.

Heuristic improvement of lattice: Suppose $\beta > 0.5$, then for $i_1 = \cdots = i_\ell = m$ and $j = \ell m$, which is a corner of the index set I_+, the value $0.5j + (i_1 + \cdots + i_\ell)\beta - \sum_{k=1}^{\ell} \min(i_k, j)$ in the sum (15) is larger than zero. In other words, the corresponding polynomial $g_{m,\ldots,m,\ell m}$ has a negative contribution for the bound. Therefore, considering a new index set $I_+ \setminus \{(m, \ldots, m, \ell m)\}$ and new polynomial lattice, the bound increases. From this observation, we propose a heuristic improvement for lattice construction. Roughly, the strategy is to remove polynomial whose index satisfies both $j > \max\{i_1, \ldots, i_\ell\}$ and $0.5j + (i_1 + \cdots + i_\ell)\beta - \sum_{k=1}^{\ell} \min(i_k, j) > 0$. It can be shown that the new lattice is also lower triangular. Appendix D provides the detailed argument. However, we have never derived an explicit formula of the working condition.

5 Application to partial key exposure attack on RSA and on the hardness of finding LSBs of d

Next, we consider the partial key exposure attack on RSA. Assume that the attacker has ℓ pairs of RSA public keys $(e_1, N), \ldots, (e_\ell, N)$, and δn LSBs of the corresponding d_k. Moreover, each d_k is assumed to be smaller than N^β. Note that this situation may not happen in real life, whereas we can discuss the hardness of revealing some amount of LSBs of secret exponents by considering $\beta = 1$.

Let $M = 2^{\lfloor \delta n \rfloor}$ and the exposed parts be $\widetilde{d_k}$ for $k \in [\ell]$. Then, following the derivation of the single equation for the situation that single $(e, N, \widetilde{d})$ is given [16], we consider the simultaneous equations

$$\begin{aligned} F_1(x_1, y) &= e_1\widetilde{d_1} - 1 + x_1(y + N) \equiv 0 \pmod{e_1 M} \\ &\vdots \\ F_\ell(x_\ell, y) &= e_\ell\widetilde{d_\ell} - 1 + x_\ell(y + N) \equiv 0 \pmod{e_\ell M}. \end{aligned} \tag{17}$$

By the counting argument in Section 4.1, we can assume that if $\beta - \delta < (\ell - 0.5)/\ell$, then the solution satisfying $|x_1|, \ldots, |x_\ell| < N^\beta$ and $|y| < 3N^{0.5}$ is unique, and it can be used to factor N.

The basic lattice construction is the same as in the above section; i.e., we let

$$g_{i,j}^{(k)} = x_k^{i-j}(F_k(x_k, y))^j (e_k M)^{m-j} \text{ and } \mathbf{G}_k = \{g_{i,j}^{(k)} : (i, j) \in \mathbb{Z}^2, 0 \le j \le i \le m\}.$$

Note that only the constant terms and moduli differ between (11) and (17). Thus, $L(\mathbf{G}_k)$ for $k \in [\ell]$ and their Minkowski sum $L(\mathbf{G}_+)$ are also lower triangular. Moreover, the set of indices $I_1, \ldots, I_\ell$ and their Minkowski sum I_+ are also the same as in Section 4.2.

For each $(i_1, \ldots, i_\ell, j) \in I_+$, we give the polynomial $g_{i_1,\ldots,i_\ell,j}$. First note that

$$\mathrm{HC}(g_{i_1,j_1}^{(1)} g_{i_2,j_2}^{(2)} \cdots g_{i_\ell,j_\ell}^{(\ell)}) = e_1^{m-j_1} \cdots e_\ell^{m-j_\ell} M^{\ell m - j_1 - \cdots - j_\ell}.$$

Thus, as Section 4.2, each j_k can move from zero to $\min(i_k, j)$, and we can take the coefficients in (9) so that

$$\mathrm{HT}(g_{i_1,\ldots,i_\ell,j}) = e_1^{m-\min(i_1,j)} \cdots e_\ell^{m-\min(i_\ell,j)} M^{\ell m - j} x_1^{i_1} \cdots x_\ell^{i_\ell} y^j.$$

Hence, we have

$$\det(\mathbf{G}_+; X_1, \ldots, X_\ell, Y) = \prod_{(i_1,\ldots,i_\ell,j) \in I_+} \left[e_1^{m-\min(i_1,j)} \cdots e_\ell^{m-\min(i_\ell,j)} \times M^{\ell m - j} X_1^{i_1} \cdots X_\ell^{i_\ell} Y^j \right].$$

By the approximations $e_k \approx N$, $X_k = N^\beta$, $Y \approx N^{0.5}$ and $M \approx N^\delta$, the attack works if

$$\sum_{(i_1,\ldots,i_\ell,j)\in I} \Big[(0.5-\delta)j + (i_1+\cdots+i_\ell)\beta - \sum_{k=1}^{\ell} \min(i_k, j)\Big] < 0. \tag{18}$$

When m becomes large, the condition is

$$\beta - \frac{\delta}{2} + \frac{1}{4} < \frac{3\ell-1}{3\ell+1}. \tag{19}$$

Here, the detailed computation to derive (19) from (18) is given in Appendix C.

5.1 Hardness of finding partial information of d corresponding to full-size e

We consider the situation $\beta = 1$; i.e., avoid the condition that d is small. Then, the condition (19) is

$$\delta > \frac{5}{2} - \frac{2(3\ell-1)}{3\ell+1}. \tag{20}$$

Note that the right-hand side below than one when $\ell \geq 3$. For example, consider the case $\ell = 3$; then the above inequality is $\delta > 0.9$. Thus, we can factor N when three tuples $(e_k, N, \widetilde{d_k})$ are given, where $\widetilde{d_k}$ is $0.9n$ LSBs of corresponding d_k.

This can demonstrate the difficulty of finding partial information of d from e and N having the same bits. Suppose we have a revealing algorithm that computes LSBs from e and N if $\gcd(e, \varphi(N)) = 1$, and it returns a random bit string, i.e., useless information for factoring if $\gcd(e, \varphi(N)) \neq 1$. For a given N, an n-bit RSA modulus, using Theorem 5 in Appendix E, we can obtain a sequence of n-bit numbers $a_1, \ldots, a_{R-1}$ of length $R = O(\log^2 N)$, in which contains at least three integers coprime to $\varphi(N)$ exist. For three elements in the sequence, let them be e_1, e_2 and e_3, use the recovering algorithm that returns $\widetilde{d_1}, \widetilde{d_2}$ and $\widetilde{d_3}$. If all e_i's are legitimate keys, i.e., they all satisfy $\gcd(e_i, \varphi(N)) = 1$, then the returned values are the lower bits of the corresponding secret keys, and we can factor N in polynomial time in $\log N$ by using our version of the Coppersmith technique. Trying all three elements in the sequence, we can factor N in polynomial time deterministically, since the number of combinations is $O(\log^6 N)$. Hence, under the assumption that the Coppersmith technique works, we conclude that the problem of finding $0.9n$ LSBs of d from (e, N) is polynomial time equivalent to factoring. For the case $\ell = 3$, we verified this result by computer experiments.

Note that when ℓ becomes large, the constant 0.9 decreases to $0.5 + \varepsilon$. Therefore, we have proven the following theorem. Comparing to the result by Coron and May [14], the necessary amount of d is decreased and the heuristic assumption is added.

Theorem 3. The problem of finding $(0.5+\varepsilon)n$ LSBs of d from e and N with the same bits is polynomial time equivalent to factoring N under the assumption that the Coppersmith technique works correctly.

Remark: Consider the situation where the unbalanced prime factor is used, i.e., $p+q < N^\alpha$ holds for some $\alpha \in (0.5, 1]$. Then, the inequality (20) can be rewritten as $\delta > 2 + \alpha - 2(3\ell-1)/(3\ell+1)$, and the necessary amount of LSBs in Theorem 3 becomes α. Here, it could be extract a better sublattice as a similar argument of Appendix D. This is also one of our future work.

6 Experimental results

We carried out computer experiments on our cryptanalyses of RSA presented in Sections 4.2 and 5.

The experiments were conducted on a standard workstation with 16GB of RAM and two Intel Xeon X5675 processors running at 3.07GHz. We wrote our experimental program in the C++ language using the following libraries. To compute the LLL reduced basis, we used Shoup's NTL library [35] version 5.5.2 compiled with the GMP library [18] version 5.0.4. The polynomial computation was performed using the GiNaC library [17] version 1.6.2. We compiled our source code using `g++` version 4.5.4 with the `-O3` option. We also used `Maple 15` to compute the resultant under modulo prime in the final step of the experiments. We performed our experiments on the Windows 7 platform and ran our program in a single thread.

Parameters:	ℓ: Number of RSA keys; n: RSA bit length; β: ratio of secret keys to n
Step 1:	(Generate a sample RSA instance) Randomly choose $\lfloor n/2 \rfloor$-bit pseudoprimes p and q, and let $N = pq$. Randomly choose ℓ $\lfloor \beta n \rfloor$-bit odd integers $d_1, \ldots, d_\ell$ such that $\mathrm{GCD}(d_k, (p-1)(q-1)) = 1$ for all $k \in [\ell]$. Compute the corresponding e_k by $d_k^{-1} \pmod{(p-1)(q-1)}$. For each $k \in [\ell]$, define the RSA polynomial $f_k(x_k, y) = -1 + x_k(N + y)$ and let the solutions $\bar{x_k} = (1 - e_k d_k)/(p-1)(q-1)$ and $\bar{y} = 1 - p - q$ corresponding to the secret keys.
Step 2:	Set the bounds $X_k = \lfloor N^\beta \rfloor$ and $Y = \lfloor 3N^{0.5} \rfloor$. Construct the polynomial lattice $L(\mathbf{G})$ in Section 4.2, and compute the Euclidean lattice $L(\mathbf{G}_+; X_1, \ldots, X_\ell, Y)$. Then, apply the LLL algorithm to $L(\mathbf{G}_+; X_1, \ldots, X_\ell, Y)$.
Step 3:	From the reduced basis, pick the first $\ell + 1$ vectors $\mathbf{v}_1, \ldots, \mathbf{v}_{\ell+1}$. Then, compute the corresponding polynomials $h_k(x_1, \ldots, x_k, y)$, i.e., take polynomials so that $\mathbf{v}_k = \mathcal{V}(h_k; X_1, \ldots, X_k, Y)$ for $k \in [\ell + 1]$.
Step 4:	First check $h_i(\bar{x_1}, \ldots, \bar{x_\ell}, \bar{y}) = 0$ for all $k \in [\ell+1]$. If it is not true, reject the instance. After the polynomials pass the first check, compute the resultant of polynomials modulo prime to check the algebraic independence. If the instance passes two checks, then we regard the experiment as successful.

Figure 3: Procedure of our computer experiments

	m	2	3	4	5	6	7	10	100	limit
$\ell = 2$	β	0.386	0.405	0.416	0.424	0.430	0.434	0.442	0.461	0.464
	dim	27	64	125	216	343	512	1331	1.03×10^6	-

	m	2	3	4	5	6	7	10	100	limit
$\ell = 3$	β	0.464	0.486	0.500	0.508	0.514	0.519	0.527	0.547	0.550
	dim	108	352	875	1836	3430	5888	21296	1.55×10^9	-

Table 1: Theoretical β bound and lattice dimension for small ℓ and several m

6.1 Experiments for short RSA secret exponents

Figure 3 shows the procedure of our computer experiments. This is essentially the same as with the outline in Figure 2. In Step 1, "pseudoprime" means an odd integer that passes the Euler-Jacobi primality testing for bases 2, 3, 5 and 7. In Step 2, we use the command `LLL_XD(L,0.99,0,0,1)`. In the second-half of Step 4, we first generate a random $0.5n$ bit prime number P. Then, we erase the variable x_1 by computing $r_k = \mathrm{Res}_{x_1}(h_1, h_k) \bmod P$ for $k = 2, \ldots, \ell + 1$, and next we compute $\mathrm{Res}_{x_2}(r_2, r_k) \bmod P$ for $k = 3, \ldots, \ell + 1$ modulo P, and repeat this process. Finally, we obtain a univariate polynomial $R(y)$ and check $R(\bar{y}) \equiv 0 \pmod{P}$. We repeat this check for three distinct prime numbers via `Maple 15`.

Parameters and results: Note first that if m and ℓ are fixed, condition (15) is written in a linear function w.r.t. β, and the maximum β satisfying the inequality is easily computed. This β is a theoretical bound when N becomes large along with neglecting several factors as described in Section 2.1. For each m and ℓ we compute the maximum β and this is shown in Table 1. The column "limit" indicates the right-hand side of (16).

We carried out our experiments to search for the practical bound of β for several choices of ℓ, m and n. We executed our procedure for each β at intervals of 0.002. Table 2 shows the experimental results. The column "β_{exp}" indicates the experimental bound of β for parameters (l, m, n); that is, the instance passed the final test at that β and failed at $\beta + 0.002$. The columns "β_{thm}" and "dim" are the

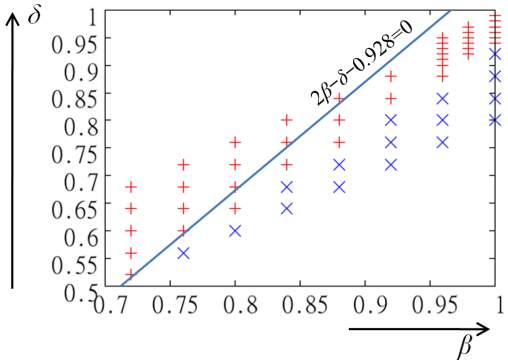
ℓ	m	n	β_{thm}	dim	β_{exp}	LLL-time
2	2	512	0.386	27	0.386	3.2 sec
		1024			0.386	10.55 sec.
2	3	512	0.405	64	0.406	5 min. 33 sec
		1024			0.406	30 min. 44 sec.
2	4	512	0.414	125	0.416	3 hrs. 50 min.
		1024			0.414	20hrs. 26min.
3	2	512	0.464	108	0.464	41 min. 25 sec.
		1024			0.464	3 hrs. 17 min.

Table 2: Experimental results for short secret exponents

theoretical bound of β and the lattice dimension, respectively; which are the same as shown in Table 1. The running time of the LLL algorithm for processing $L(\mathbf{G}_+)$ is given in the column "LLL-time."

We note that for $\ell = 3$ and $m = 2$, the second half of Step 4 is not finished due to computational time. More precisely, `Maple` computed two bivariate polynomials, $r_1(x_3, y)$ and $r_2(x_3, y)$, from $h_1, \ldots, h_4$. It took over 120 hours to compute $\text{Res}_{x_3}(r_1, r_2)$, and we stopped the computation. However, we can observe that $h_1, \ldots, h_4$ are algebraically independent since they are reduced to the bivariate polynomials, and can expect that the final resultant will be computed if more time is permitted. Hence, we regard the experiment as a success. From the observation, we conclude that our algorithm works well.

6.2 Experiments for partial key exposure situation



β	δ	LLL-time	result
1.00	0.96	18 hrs. 59 min.	+ (passed)
1.00	0.94	16 hrs. 13 min.	+ (passed)
1.00	0.92	15 hrs. 46 min.	× (fault)
0.96	0.90	15 hrs. 42 min.	+ (passed)
0.96	0.88	16 hrs. 9 min.	+ (passed)
0.96	0.84	14 hrs. 46 min.	× (fault)

Figure 4: Experimental results for partial key exposure situation for $(\ell, m, n) = (3, 2, 1024)$. The area $2\beta - \delta - 0.928 < 0$ in the left figure is the theoretical bound.

Next we conducted our experiments on the partial key exposure situation. The experimental procedure is similar to in Figure 3. Different points are the definition of $F_k(x_k, y)$, and that $M = 2^{\lfloor \delta n \rfloor}$ and $\widetilde{d}_k = d \bmod M$ are added in Step 1.

We fixed the parameters $\ell = 3$ and $m = 2$ since it could be used to verify the hardness result described in Section 5.1 when $\ell \geq 3$,. Unfortunately, for this ℓ, only the lattice constructed with $m = 2$ can be reduced in reasonable time. The lattice dimension is 108 as in the above subsection. For several choices of β and δ, we generated 1024-bit RSA sample instances and tested them.

Figure 4 shows the result. In the figure, the horizontal and vertical axes are β and δ, respectively. Each mark represents one experiment (β, δ) at the point. The marks "+" and "×" mean that the instance passed and was a fault, respectively. The left table in Figure 4 indicates the running time of the LLL algorithm and experimental results for several β and δ close to $\beta = 1$. Again, note that the final resultant computation was not finished and regard that the experiment is successful if `Maple` computes two bivariate polynomials.

Here, the left/upper area of the diagonal line indicates $\delta > 2\beta - 0.928$, which is derived from (18) for $\ell = 3$ and $m = 2$. Hence, it means that the experimental result is slightly better than the theory. We think this gap is caused by an existence of a better sublattice that improves the original bound (18). Unfortunately, we cannot extract this sublattice. For the constructed lattice basis $\mathbf{B} = (\mathbf{b}_1, \ldots, \mathbf{b}_n)$, write the first vector in the reduced lattice by $\sum_{i=1}^{n} a_i \mathbf{b}_i$. We observed the coefficients are all non-zero in our experiments. Therefore, the sublattice is not spanned by a subset of $\mathbf{B}$. Extracting this sublattice, or finding why the practical result is better than the theory are topics for our future work.

7 Discussion and open problems

Here we suggest and discuss several unsolved problems.

Minkowski sum lattice construction: Although our lattice construction works well, it is not optimal. That is, in Section 3.1, $L(\mathbf{G}_+)$ is a subset of the lattice $\mathcal{A}$ spanned by all possible combination of polynomials. Providing a method to extract the lattice basis of $\mathcal{A}$, and deriving the condition so that $L(\mathbf{G}_+)$ and $\mathcal{A}$ are equivalent are open problems.

A Similar situation is occurred in our partial key exposure attack for RSA. In computer experiments in Section 6.2, for $(\ell, m) = (3, 2)$, we succeeded in factoring N from three pairs of (e, N) and corresponding $0.94n$ LSBs of d by using the Coppersmith technique. This is slightly better than the bound derived by the theory. We expect that a sublattice achieving a better bound exists in the constructed lattice. Note that this is not an unrealistic situation. For instance, in the paper By Boneh and Durfee [3], the lattice achieving $d < N^{0.292}$ is a sublattice of the lattice achieving $d < N^{0.284}$. Hence, as in their paper, extracting a sublattice from the Minkowski sum lattice, i.e., selecting a suitable polynomial basis, is an interesting open problem.

Consider the simultaneous equations $F_1(x_1, y) = Ax_1 + y \pmod{B}$ and $F_2(x_2, y) = Bx_2 + y \pmod{A}$. For $m = 1$, a polynomial $g_i(x_1, x_2, y) = F_1(x_1, y)F_2(x_2, y) = ABx_1x_2 + Ax_1y + Bx_2y + y^2$ can be considered; however, it is equivalent to $Ax_1y + Bx_2y + y^2$ under modulo AB, which is expected to make a better contribution toward lattice construction. In this situation, our Minkowski sum construction does not give a better construction. Adopting our theory to these situations is also a target of future work.

Cryptanalysis of RSA with small secret exponents: Both our bound (16) and that by Sarkar and Maitra converge to $N^{0.75}$ when ℓ becomes large. On the other hand, the bound by Howgrave-Graham and Seifert [25] and that by the counting argument in Section 4.1 converge to N. Filling this gap poses an interesting open problem. We expect that our heuristic improvement shown in Appendix D achives this goal, though this is not proven.

Cryptanalysis of RSA in other situations and the half amount conjecture: The proposed Minkowski sum based lattice construction can be applied to other situations of cryptanalysis of RSA including revealed MSBs [16], RSA-CRT [26], Takagi's RSA [28], small e [4, 7, 31], unbalanced p and q situation [32], and special settings of e [34]. For more information, see [36, Chap. 10]. For these situations, we speculate that recovering half amount of continuous bits is polynomially equivalent to factoring N.

8 Concluding remarks

We proposed a method for constructing a lattice for solving simultaneous equations by combining lattices for a single equation via the Minkowski sum of index sets.

As an application, we gave a new RSA cryptanalysis for two or more short secret exponents. We showed that an RSA modulus can be factored from ℓ pairs of public keys corresponding to ℓ secret exponents smaller than $N^{(9\ell-5)/(12\ell+4)}$, which improves the previously best known bound $N^{(3\ell-1)/(4\ell+4)}$ given by Sarkar and Maitra [42].

Next we considered the partial key exposure situation with small secret exponents. In this situation, from ℓ pairs of RSA public keys and δn LSBs of corresponding secret exponents that are smaller than N^{β}, the attacker can efficiently factor an RSA modulus if $\beta - \frac{\delta}{2} + \frac{1}{4} < \frac{3\ell-1}{3\ell+1}$. Letting $\beta = 1$ and ℓ be a sufficiently large integer, this can provide a factoring algorithm using an algorithm that reveals $(0.5 + \varepsilon)n$ LSBs of the secret exponent from given (e, N). Thus, under the assumption that the

Coppersmith technique works, finding $(0.5+\varepsilon)n$ LSBs of d is polynomial-time equivalent to factoring N in the standard RSA.

References

[1] Y. Aono, A new lattice construction for partial key exposure attack for RSA, in *Proceedings of PKC 2009*, Lecture Notes in Computer Science, vol. 5443, pp. 34-53, Springer, Heidelberg, 2009.

[2] Y. Aono, M. Agrawal, T. Sato and O. Watanabe, On the optimality of lattices for the Coppersmith technique, in *Proceedings of ACISP 2012*, Lecture Notes in Computer Science, vol. 7372, pp. 376-389, Springer, Heidelberg, 2012. The full-version is available online at *Cryptology ePrint Archive*, 2012/134.

[3] D. Boneh and G. Durfee, Cryptanalysis of RSA with private Key d Less Than $N^{0.292}$, in *Proceedings of Eurocrypt 1999*, Lecture Notes in Computer Science, 1592, pp. 389-401, Springer, Heidelberg, 1999.

[4] D. Boneh, G. Durfee and Y. Frankel, Exposing an RSA private key given a small fraction of its bits *IEEE Transactions on Information Theory*, in *Proceedings of Asiacrypt* 1998, Lecture Notes in Computer Science, vol. 1514, pp. 25-34, Springer, Heidelberg, 1998.

[5] D. Boneh, Twenty years of attacks on the RSA cryptsystem, in *Notices of AMS*, Vol. 46, No. 2, pp. 203-213, 1999

[6] J. Blömer and A. May, Low Secret Exponent RSA Revisited in *CaLC 2001*, Lecture Notes in Computer Science, vol. 2146, pp. 4-19, Springer, Heidelberg, 2001.

[7] J. Blömer and A. May, New partial key exposure attacks on RSA, in *Proceedings of Crypto 2003*, Lecture Notes in Computer Science, vol. 2729, pp. 27-43, Springer, Heidelberg, 2003.

[8] J. Blömer and A. May, A tool kit for finding small roots of bivariate polynomials over the integers, *Proceedings of Eurocrypt 2005*, Lecture Notes in Computer Science, vol. 3494, pp. 251-267, Springer, Heidelberg, 2005.

[9] E. Brier, D. Naccache, P. Q. Nguyen and M. Tibouchi, Modulus Fault Attacks Against RSA-CRT Signatures, in *Proceedings of CHES 2011*, Lecture Notes in Computer Science, vol. 6917, pp. 192-206, Springer, Heidelberg, 2011.

[10] D. Cox, J. Little, D. O'Shea, Ideals, Varieties, and Algorithms: An Introduction to Computational Algebraic Geometry and Commutative Algebra, Springer-Verlag, New York, 2007.

[11] D. Coppersmith, Finding a small root of a univariate modular equation, *Proceedings of Eurocrypt 1996*, Lecture Notes in Computer Science, vol. 1070, pp. 155-165, Springer, Heidelberg, 1996.

[12] D. Coppersmith, Finding a small root of a bivariate integer equation; factoring with high bits known, *Proceedings of Eurocrypt 1996*, Lecture Notes in Computer Science, vol. 1070, pp. 178-189, Springer, Heidelberg, 1996.

[13] D. Coppersmith, Finding small solutions to small degree polynomials, *Proceedings of CaLC 2001*, Lecture Notes in Computer Science, vol. 2146, pp. 20-31, Springer, Heidelberg, 2001.

[14] J. S. Coron and A. May, Deterministic polynomial-time equivalence of computing the RSA secret key and factoring, *Journal of Cryptology*, vol. 20,No. 1, pp. 39-50, 2007.

[15] J. S. Coron, D. Naccache and M. Tibouchi, Fault Attacks Against EMV Signatures, *Proceedings of CT-RSA 2010*, Lecture Notes in Computer Science, vol. 2010, pp. 208-220, Springer, Heidelberg, 2010.

[16] M. Ernst, E. Jochemsz, A. May, and B. Weger Partial key exposure attacks on RSA up to full size exponents, in *Proc. of Eurocrypt 2005*, Lecture Notes in Computer Science, vol. 3494, pp. 371-386, Springer, Heidelberg, 2005.

[17] GiNaC is Not a CAS, available online at `http://www.ginac.de/`.

[18] The GNU MP Bignum Library, available online at `http://gmplib.org/`.

[19] A. D. Healy, Resultants, Resolvents and the Computation of Galois Groups, Available online at `http://www.alexhealy.net/papers/math250a.pdf`

[20] M. Herrmann, Improved cryptanalysis of the multi-prime Φ-hiding assumption, *Proceedings of Africacrypt 2011*, Lecture Notes in Computer Science, vol. 6737, pp. 92-99, Springer, Heidelberg, 2011.

[21] N. Howgrave-Graham, Finding small roots of univariate modular equations revisited, *Proceedings of Cryptography and Coding*, Lecture Notes in Computer Science, vol. 1355, pp. 131-142, Springer, Heidelberg, 1997.

[22] M. J. Hinek and C. C. Y. Lam, Common modulus attacks on small private exponent RSA and some fast variants (in practice), in *Journal of Mathematical Cryptology*, vol. 4, Issue 1, pp. 58-93, 2010.

[23] M. Herrmann and A. May, Solving linear equations modulo divisors: On factoring given any bits, in *Proceedings of Asiacrypt 2008*, Lecture Notes in Computer Science, vol. 5350, pp. 406-424, Springer, Heidelberg, 2008.

[24] M. Herrmann and A. May, Attacking power generators using unravelled linearization: When do we output too much? in *Proceedings of Asiacrypt 2009*, Lecture Notes in Computer Science, vol. 5912, pp. 487-504, Springer, Heidelberg, 2009.

[25] N. Howgrave-Graham and J. P. Seifert, Extending Wiener's attack in the presence of many decrypting exponents, *Proceedings of Secure Networking 1999*, Lecture Notes in Computer Science, vol. 1740, pp. 153-166, Springer, Heidelberg, 1999.

[26] E. Jochemsz and A. May, A strategy for finding roots of multivariate polynomials with new applications in attacking RSA variants, in *Proceedings of Asiacrypt 2006*, Lecture Notes in Computer Science, vol. 4284, pp. 267-282, Springer, Heidelberg, 2006.

[27] E. Jochemsz and B. Weger, A partial key exposure attack on RSA using a 2-dimensional lattice, in *Proceedings of ISC 2006*, Lecture Notes in Computer Science, vol. 4176, pp. 203-216, Springer, Heidelberg, 2006.

[28] N. Kunihiro and K. Kurosawa, Deterministic polynomial time equivalence between factoring and key-recovery attack on Takagi's RSA, in *Proceedings of PKC 2007*, Lecture Notes in Computer Science, vol. 4450, pp. 412-425, Springer, Heidelberg, 2007.

[29] N. Kunihiro, Solving generalized small inverse problems, in *Proceedings of ACISP 2010*, Lecture Notes in Computer Science, vol. 6168, pp. 248-263, Springer, Heidelberg, 2010.

[30] A. K. Lenstra, H. W. Lenstra, Jr. and L. Lovász Factoring polynomials with rational coefficients, *Mathematische Annalen*, vol. 261, pp. 515-534, 1982.

[31] P. Luo, H.-J. Zhou, D.-S. Wang and Y.-Q Dai, Cryptanalysis of RSA for a special case with $d > e$, *Science in China Series F: Information Sciences*, vol. 52, num. 4, pp. 609-616, 2009.

[32] A. May, Cryptanalysis of Unbalanced RSA with Small CRT-Exponent, in *Proceedings of Crypto 2002*, Lecture Notes in Computer Science, vol. 2442, pp. 242-256, Springer, Heidelberg, 2002.

[33] A. May and M. Ritzenhofen, Solving systems of modular equations in one variable: How many RSA-encrypted messages does Eve need to know?, in *Proceedings of PKC 2008*, Lecture Notes in Computer Science, vol. 4939, pp. 37-46, Springer, Heidelberg, 2008.

[34] S. Maitra and S. Sarkar, A New Class of Weak Encryption Exponents in RSA, in *Proceedings of Indocrypt 2008*, Lecture Notes in Computer Science, vol. 5365, pp. 337-349, 2008.

[35] V. Shoup, NTL: A Library for doing Number Theory, available online at `http://www.shoup.net/ntl/index.html`

[36] P. Q. Nguyen and B. Vallée, The LLL Algorithm: Survey and Applications, Springer-Verlag, Berlin Heidelberg, 2009.

[37] M. Ritzenhofen, On efficiently calculating small solutions of systems of polynomial equations : lattice-based methods and applications to cryptography, Ph.D. thesis, Ruhr University Bochum, available online at `http://www-brs.ub.ruhr-uni-bochum.de/netahtml/HSS/Diss/RitzenhofenMaike/diss.pdf`

[38] R. L. Rivest, A. Shamir, and L. Adleman, A method for obtaining digital signatures and public-key cryptsystems, *Communications of the ACM*, vol. 21, No. 2, pp. 120-128, 1978.

[39] C. R. Schnorr and M. Euchner, Lattice basis reduction: improved algorithms and solving subset sum problems, in *Mathematics and Programming*, vol. 66, pp. 181-189.

[40] S. Sarkar and S. Maitra, Improved partial key exposure attacks on RSA by guessing a few bits of one of the prime factors, in *Proceedings of ICISC 2008*, Lecture Notes in Computer Science, vol. 5461, pp. 37-51, Springer, Heidelberg, 2009.

[41] S. Sarkar and S. Maitra, Cryptanalysis of RSA with two decryption exponents, in *Information Processing Letter*, vol. 110, pp. 178-181, 2010.

[42] S. Sarkar and S. Maitra, Cryptanalysis of RSA with more than one decryption exponent, in *Information Processing Letter*, vol. 110, pp. 336-340, 2010.

[43] S. Sarkar, S. Maitra and S. Sarkar, RSA cryptanalysis with increased bounds on the secret exponent using less lattice dimension, *Cryptology ePrint Archive*, Report 2008/315, 2008. available online at `http://eprint.iacr.org/2008/315.pdf`.

[44] H.-M. Sum, M.-E. Wu and Y.-H. Chen, Estimating the prime factors of an RSA modulus and an extension of the Wiener attack, in *Proceedings of ACNS 2007*, Lecture Notes in Computer Science, vol. 4521, pp. 116-128, Springer, Heidelberg, 2011.

[45] M. J. Wiener, Cryptanalysis of short RSA secret exponents, *IEEE Transactions on Information Theory* vol. 36, no. 3, pp. 553-558, 1990.

A Minkowski sum of three or more lattices

Here we give the detailed construction of the Minkowski sum of three or more lattices. For fixed integer $\ell \geq 3$, consider polynomial lattice bases $\mathbf{G}_k = \{g_1^{(k)}, \ldots, g_{c_k}^{(k)}\}$ for $k \in [\ell]$. Again, assume that they have strictly increasing degree order. For simplicity, we also assume that a polynomial $g_i^{(k)}$ is bivariate of variables x_k and y; thus, only y is shared. Note that extending to a general case is trivial, but it yields extremely complicated expressions.

For each $k \in [\ell]$, let I_k be the set of indices $\{HI(g_i^{(k)}) : i \in [c_k]\} \subset \mathbb{Z}^{\ell+1}$ and define the Minkowski sum $I_+ = I_1 \boxplus \cdots \boxplus I_\ell$. Then, for every $(i_1, \ldots, i_\ell, j) \in I_+$, define the polynomial

$$g^+_{i_1,\ldots,i_\ell,j} = \sum_{(*)} a_{\lambda_1,\ldots,\lambda_\ell} g_{\lambda_1}^{(1)} \cdots g_{\lambda_\ell}^{(\ell)}$$

where the range of sum $(*)$ is over all suffix tuples $(\lambda_1, \ldots, \lambda_\ell)$ satisfying $HM(g_{\lambda_1}^{(1)} \cdots g_{\lambda_\ell}^{(\ell)}) = x_1^{i_1} \cdot \cdots \cdot x_\ell^{i_\ell} y^j$, and the coefficients are defined so that the head coefficient of the polynomial $g^+_{i_1,\ldots,i_\ell,j}$ is $GCD_{(*)}(HC(g_{\lambda_1}^{(1)} \cdots g_{\lambda_\ell}^{(\ell)}))$. Using the above polynomials, define the Minkowski sum construction $\mathbf{G}^+$ by the lattice spanned by all $g^+_{i_1,\ldots,i_\ell,j}$ for $(i_1, \ldots, i_e ll, j) \in I_+$.

In a similar way to Theorem 2, it can be shown that the Minkowski sum of ℓ lower triangular lattice is also lower triangular.

Theorem 4. Let $\ell \geq 3$ be an integer. For $k \in [\ell]$, assume that the polynomial lattice basis $\mathbf{G}_k = \{g_1^{(k)}, \ldots, g_{c_k}^{(k)}\} \subset \mathbb{Z}[x_k, y]$ has a strictly increasing degree order, and is lower triangular. Then, the Minkowski sum lattice $L(\mathbf{G}_+)$ is also lower triangular.

Proof. Since the bases $\mathbf{G}_k$ are all lower triangular, that is, for each $k \in [\ell]$, there exists a sequence of integer pairs $\{(i_k(1), j_k(1)), (i_k(2), j_k(2)), \ldots, (i_k(c_k), j_k(c_k))\}$ and each polynomial $g_i^{(k)}(x_k, y)$ for $i \in [c_k]$ can be written as

$$g_i^{(k)}(x_k, y) = \sum_{i'=1}^{i} d_{i,i'}^{(k)} x_k^{i_k(i')} y^{j_k(i')}, \text{ where } d_{i,i} \neq 0.$$

Fix any $(i_1, \ldots, i_\ell, j) \in I_+$. We show that $g^+_{i_1,\ldots,i_\ell,j}$ can be written as

$$H \cdot x_1^{i_1} \cdots x_\ell^{i_\ell} y^j + (\text{terms whose indices are in } I_+ \text{ and smaller than } x_1^{i_1} \cdots x_\ell^{i_\ell} y^j).$$

for any $(i_1, \ldots, i_k, j) \in I_+$. Here, H is the heading coefficient. Thus, as the proof of Theorem 2, we show any $g_{\lambda_1}^{(1)} \cdots g_{\lambda_\ell}^{(\ell)}$ can be written as the above form for any allowed combination of $(\lambda_1, \ldots, \lambda_\ell)$. Fix a suffix tuple $(\lambda_1, \ldots, \lambda_\ell)$ and consider the monomial expansion of $g_{\lambda_1}^{(1)} \cdots g_{\lambda_\ell}^{(\ell)}$.

Pick any term in the polynomial $g_{\lambda_k}^{(k)}$ for $k \in [\ell]$ and let it be $A_k x_k^{\bar{i_k}} y^{\bar{j_k}}$. The product of these is $A_1 \cdots A_k x_1^{\bar{i_1}} \cdots x_\ell^{\bar{i_\ell}} y^{\bar{j_1}+\cdots+\bar{j_\ell}}$. Thus, it is clear that the corresponding index $(\bar{i_1}, \ldots, \bar{i_\ell}, \bar{j_1} + \cdots + \bar{j_\ell})$ is in I_+. By the relation $(0, \ldots, 0, \bar{i_k}, 0, \ldots, 0, \bar{j_k}) \prec HI(g_k)$, we have

$$\begin{aligned} & (\bar{i_1}, \ldots, \bar{i_\ell}, \bar{j_1} + \cdots + \bar{j_\ell}) \\ = \; & (\bar{i_1}, 0, \ldots, 0, \bar{j_1}) + (0, \bar{i_2}, 0, \ldots, 0, \bar{j_2}) + \cdots + (0, \ldots, \bar{i_\ell}, \bar{j_\ell}) \\ \prec \; & HI(g_{\lambda_1}^{(1)}) + \cdots + HI(g_{\lambda_\ell}^{(\ell)}) = HI(g^+_{i_1,\ldots,i_\ell,j}). \end{aligned}$$

Therefore, the lattice $L(\mathbf{G}_+)$ spanned by all $g^+_{i_1,\ldots,i_\ell,j}$ for all $(i_1, \ldots, i_\ell, j) \in I_+$ is lower triangular. □

B Small example of Minkowski sum construction for RSA with short secret exponents

Here we give an example of the lattice construction in Section 4.2 for $m = 1$ and $\ell = 2$.

For $k = 1$ and 2, the polynomials satisfying (2) are $g_1^{(k)}(x_k, y) = e_k$, $g_2^{(k)}(x_k, y) = e_k x_k$ and $g_3^{(k)}(x_k, y) = F_k(x_k, y) = -1 + x_k(y + N)$. The index sets are $I_1 = \{(0,0,0), (1,0,0), (1,0,1)\}$ and $I_2 = \{(0,0,0), (0,1,0), (0,1,1)\}$. Then, the Minkowski sum of these sets is

$$I_+ = I_1 \boxplus I_2 = \{(0,0,0), (0,1,0), (0,1,1), (1,0,0), (1,0,1), (1,1,0), (1,1,1), (1,1,2)\}.$$

For all elements in this set, construct the polynomials satisfying (8) via (9):

$$\begin{aligned} g^+_{0,0,0} &= e_1 e_2, & g^+_{0,1,0} &= e_1 e_2 x_2, \\ g^+_{0,1,1} &= e_1 F_2(x_2, y), & g^+_{1,0,0} &= e_1 e_2 x_1, \\ g^+_{1,0,1} &= e_2 F_1(x_1, y), & g^+_{1,1,0} &= e_1 e_2 x_1 x_2, \\ g^+_{1,1,1} &= a_1 e_1 x_1 F_2(x_2, y) + a_2 e_2 x_2 F_1(x_1, y), \text{ and} \\ g^+_{1,1,2} &= F_1(x_1, y) F_2(x_2, y). \end{aligned}$$

Here, for $(i_1, i_2, j) = (1,1,1)$, there are two polynomial pairs $(g_\lambda^{(1)}, g_{\lambda'}^{(2)})$ such that $\mathrm{HM}(g_\lambda^{(1)} g_{\lambda'}^{(2)}) = x_1 x_2 y$ in (9); that is, both $g_2^{(1)} g_3^{(2)}$ and $g_3^{(1)} g_2^{(2)}$ satisfy it. Hence, the coefficients of the integer combination are taken so that $\mathrm{HC}(g^+_{1,1,1}) = \mathrm{GCD}(\mathrm{HC}(g_2^{(1)} g_3^{(2)}), \mathrm{HC}(g_3^{(1)} g_2^{(2)})) = \mathrm{GCD}(e_1, e_2) = 1$. Thus, set integers a_1 and a_2 satisfying $a_1 e_1 + a_2 e_2 = 1$, and $g^+_{1,1,1} = a_1 g_2^{(1)} g_3^{(2)} + a_2 g_3^{(1)} g_2^{(2)} = a_1 e_1 x_1 F_2(x_2, y) + a_2 e_2 x_2 F_1(x_1, y)$.

Therefore, the lattice $L(G^+; X_1, X_2, Y)$ is given by

$$\begin{bmatrix} e_1e_2 & & & & & & & \\ & e_1e_2X_2 & & & & & & \\ -e_1 & e_1NX_2 & e_1X_2Y & & & & & \\ & & & e_1e_2X_1 & & & & \\ -e_2 & & & e_2NX_1 & e_2X_1Y & & & \\ & & & & & e_1e_2X_1X_2 & & \\ & -a_2e_2X_2 & & -a_1e_1X_1 & & NX_1X_2 & X_1X_2Y & \\ 1 & -NX_2 & -X_2Y & -NX_1 & -X_1Y & N^2X_1X_2 & 2NX_1X_2Y & X_1X_2Y^2 \end{bmatrix}. \tag{21}$$

The determinant of this lattice is $e_1^5 e_2^5 X_1^5 X_2^5 Y^5 \approx N^{12.5+10\delta}$. Then, the condition (15) is $12.5 + 10\delta < 16 \Leftrightarrow \delta < 0.35$.

On the other hand, the previous works consider the single equation

$$F_1(x_1, y) F_2(x_2, y) = (-1 + x_1(y + N))(-1 + x_2(y + N)) \equiv 0 \pmod{e_1 e_2}$$

and then construct a polynomial lattice using the strategy by Jochemsz and May [26], which selects a polynomial basis as

$$\begin{array}{ll} g_{0,0,0} = e_1e_2, & g_{0,1,0} = e_1e_2x_2, \\ g_{0,1,1} = e_1e_2x_2y, & g_{1,0,0} = e_1e_2x_1, \\ g_{1,0,1} = e_1e_2x_1y, & g_{1,1,0} = e_1e_2x_1x_2, \\ g_{1,1,1} = e_1e_2x_1x_2y, \text{and} & g_{1,1,2} = F_1(x_1,y)F_2(x_2,y). \end{array}$$

The determinant of the constructed lattice is $e_1^7e_2^7X_1^5X_2^5Y^5 \approx N^{16.5+10\delta}$, which is slightly higher than our construction, and it does not satisfy the condition (15) for any $\delta > 0$.

C Deriving bounds in Sections 4.2 and 5

We give a detailed explanation for evaluating

$$\sum_{(i_1,\ldots,i_\ell,j)\in I_+} \left[0.5j + (i_1+\cdots+i_\ell)\beta - \sum_{k=1}^{\ell}\min(i_k,j)\right] < 0 \tag{15}$$

in Section 4.2, and

$$\sum_{(i_1,\ldots,i_\ell,j)\in I_+} \left[(0.5-\delta)j + (i_1+\cdots+i_\ell)\beta - \sum_{k=1}^{\ell}\min(i_k,j)\right] < 0. \tag{18}$$

in Section 5.

First note that the sum can explicitly be written as $\displaystyle\sum_{(i_1,\ldots,i_\ell,j)\in I} = \sum_{i_1=0}^{m}\cdots\sum_{i_\ell=0}^{m}\sum_{j=0}^{i_1+\cdots+i_\ell}$, and the following formulas hold for any $\ell, m \in \mathbb{N}$ and $a, b \in [\ell]$.

$$\sum_{i_1=0}^{m}\cdots\sum_{i_\ell=0}^{m} i_ai_b = \begin{cases} \dfrac{m^{\ell+2}}{3} + o(m^{\ell+2}) & (a=b) \\ \dfrac{m^{\ell+2}}{4} + o(m^{\ell+2}) & (a\neq b) \end{cases} \tag{22}$$

By this, we have

$$\sum_{i_1=0}^{m}\cdots\sum_{i_\ell=0}^{m}(i_1+\cdots+i_n)^2 = \left(\ell^2\cdot\frac{m^{\ell+2}}{4} + \ell\cdot\frac{m^{\ell+2}}{12}\right) + o(m^{\ell+2}). \tag{23}$$

Thus,

$$\begin{aligned} \sum_{(i_1,\ldots,i_\ell,j)\in I_+} j &= \sum_{i_1=0}^{m}\cdots\sum_{i_\ell=0}^{m}\sum_{j=0}^{i_1+\cdots+i_\ell} j = \sum_{i_1=0}^{m}\cdots\sum_{i_\ell=0}^{m}\left\{\frac{(i_1+\cdots+i_\ell)^2}{2} + o(m)\right\} \\ &= \left(\ell^2\cdot\frac{m^{\ell+2}}{8} + \ell\cdot\frac{m^{\ell+2}}{24}\right) + o(m^{\ell+2}). \end{aligned}$$

and

$$\begin{aligned} \sum_{(i_1,\ldots,i_\ell,j)\in I_+} (i_1+\cdots+i_\ell) &= \sum_{i_1=0}^{m}\cdots\sum_{i_\ell=0}^{m}\sum_{j=0}^{i_1+\cdots+i_\ell}(i_1+\cdots+i_\ell) = \sum_{i_1=0}^{m}\cdots\sum_{i_\ell=0}^{m}\left\{(i_1+\cdots+i_\ell)^2 + o(m)\right\} \\ &= \left(\ell^2\cdot\frac{m^{\ell+2}}{4} + \ell\cdot\frac{m^{\ell+2}}{12}\right) + o(m^{\ell+2}). \end{aligned}$$

Next consider the sum $\sum_{(i_1,\ldots,i_\ell,j)\in I}\Big[\sum_{k=1}^{\ell}\min(i_k,j)\Big]$. By symmetry, it suffices to calculate $\sum_{(i_1,\ldots,i_\ell,j)\in I}\min(i_1,j)$. Using the relation

$$\sum_{j=0}^{i_1+\cdots+i_\ell}\min(i_1,j)=\sum_{j=0}^{i_1}j+\sum_{j=i_1+1}^{i_1+\cdots+i_\ell}i_1=\frac{i_1(i_1+1)}{2}+(i_2+\cdots+i_\ell)i_1,$$

we have

$$\begin{aligned}\sum_{(i_1,\ldots,i_\ell,j)\in I}\sum_{k=1}^{\ell}\min(i_k,j) &= \ell\sum_{(i_1,\ldots,i_\ell,j)\in I}\min(i_1,j)\\ &= \ell\sum_{i_1=0}^{m}\cdots\sum_{i_\ell=0}^{m}\Big[\frac{i_1(i_1+1)}{2}+(i_2+\cdots+i_\ell)i_1\Big] \qquad (24)\\ &= \left(\frac{\ell^2}{4}-\frac{\ell}{12}\right)m^{\ell+2}+o(m^{\ell+2}).\end{aligned}$$

C.1 Condition for RSA short secret exponents

Using the above formulas, the left-hand side of (15) is

$$\sum_{(i_1,\ldots,i_\ell,j)\in I_I}\left[0.5j+(i_1+\cdots+i_\ell)\beta-\sum_{k=1}^{\ell}\min(i_k,j)\right]=\left(-\frac{3}{16}\ell^2+\frac{5}{48}\ell+\left(\frac{\ell^2}{4}+\frac{\ell}{12}\right)\beta\right)m^{\ell+2}+o(m^{\ell+2}).$$

Thus, the condition is to be

$$\beta<\frac{9\ell-5}{12\ell+4}, \qquad (16)$$

when m is sufficiently large.

C.2 Condition for RSA partial key exposure situation

The left-hand side of (18) is

$$\begin{aligned}&\sum_{(i_1,\ldots,i_\ell,j)\in I}\Big[(0.5-\delta)j+(i_1+\cdots+i_\ell)\beta-\sum_{k=1}^{\ell}\min(i_k,j)\Big]\\ &=\left(\frac{1}{4}-\frac{\delta}{2}+\beta\right)\left(\frac{\ell^2}{4}+\frac{\ell}{12}\right)m^{\ell+2}-\left(\frac{\ell^2}{4}-\frac{\ell}{12}\right)m^{\ell+2}.\end{aligned}$$

Hence, the condition is to be

$$\beta-\frac{\delta}{2}+\frac{1}{4}<\frac{3\ell-1}{3\ell+1}, \qquad (19)$$

when m is sufficiently large.

D Heuristic improvement of lattices for small short secret exponents

Here we give the detailed argument for our improvement of lattices mentioned in Section 4.2. Fix the parameters $\beta>0.5$ and m. Consider a subset of I_+:

$$I_{++}=\left\{(i_1,\ldots,i_\ell,j)\in I_+ : j\le\max\{i_1,\ldots,i_\ell\}\text{ or }0.5j+(i_1+\cdots+i_\ell)\beta-\sum_{k=1}^{\ell}\min(i_k,j)<0\right\} \qquad (25)$$

Thus, it deletes tuples such that j is large and terms in (15) are greater than zero. Then, construct the lattice with basis $\mathbf{G}_{++} = \{g^+_{i_1,\ldots,i_\ell,j} : (i_1,\ldots,i_\ell,j) \in I_{++}\} \subset \mathbf{G}_+$.

Proposition 1. The lattice $L(\mathbf{G}_{++})$ is lower triangular.

Proof. As the proof of Theorem 2, it suffices to show for every $(i_1,\ldots,i_\ell,j) \in I_{++}$,

$$g^+_{i_1,\ldots,i_\ell,j} = ax_1^{i_1}\cdots x_\ell^{i_\ell}y^j + (\text{terms whose indices are in } I_{++} \text{ and smaller than } x_1^{i_1}\cdots x_\ell^{i_\ell}y^j).$$

Recall that $g^{(k)}_{i_k,j_k} = x^{i_k-j_k}(-1+x_k(y+N))^{j_k}e^{m-j_k}$. Thus, it is easy to see that the polynomial is a linear combination of $x^{i'_k}y^{j'_k}$ such that the following three inequalities hold.

$$0 \le j'_k,\ i'_k \le i_k \text{ and } i'_k - j'_k \ge i_k - j_k \tag{26}$$

Since each term in $g^+_{i_1,\ldots,i_\ell,j}$ is a sum of the product of terms in $g^{(1)}_{i_1,j_1},\ldots,g^{(\ell)}_{i_\ell,j_\ell}$, it can be written as $A\cdot x_1^{i'_1}\cdots x_\ell^{i'_\ell}y^{j'_1+\cdots+j'_\ell}$. Thus, it suffices to show $(i'_1,\ldots,i'_\ell,j') \in I_{++}$, where we let $j' = j'_1+\cdots+j'_\ell$.

First, we consider the situation that $(i_1,\ldots,i_\ell,j)$ satisfies the first inequality in the definition of I_{++}; that is, $j \le \max\{i_1,\ldots,i_\ell\}$. In this situation, we have

$$\begin{aligned}
j' &= j'_1+\cdots+j'_\ell \\
&\le \sum_k (i'_k - i_k + j_k) \le \sum_k (i'_k - i_k) + \max\{i_1,\ldots,i_\ell\} \\
&= \max\left\{ i_1 + \sum_k (i'_k - i_k), \ldots, i_\ell + \sum_k (i'_k - i_k) \right\} \\
&\le \max\{i_1 + (i'_1 - i_1), \cdots, i_\ell + (i'_\ell - i_\ell)\} \quad (\text{since } i'_k \le i_k) \\
&= \max\{i'_1,\ldots,i'_\ell\}.
\end{aligned}$$

Thus, $(i'_1,\ldots,i'_\ell,j') \in I_{++}$.

Next we consider the situation that $(i_1,\ldots,i_\ell,j)$ satisfies $0.5j+(i_1+\cdots+i_\ell)\beta-\sum_{k=1}^{\ell}\min(i_k,j) < 0$. By Theorem 2, $(i'_1,\ldots,i'_\ell,j') \in I_+$ is already shown; thus, for the case $j' \le \max\{i'_1,\ldots,i'_\ell\}$, the claim is proved. Then consider another case. First note that $j' > \max\{i'_1,\ldots,i'_\ell\}$ implies $\min(i'_k,j') = i'_k$ for all $k \in [\ell]$. Thus, we have,

$$0.5j' + (i'_1+\cdots+i'_\ell)\beta - \sum_{k=1}^{\ell}\min(i'_k,j') = 0.5(j'_1+\cdots+j'_\ell) + (i'_1+\cdots+i'_\ell)(\beta-1).$$

Then, by inequality (26),

$$\begin{aligned}
&0.5(j'_1+\cdots+j'_\ell) + (i'_1+\cdots+i'_\ell)(\beta-1) \\
&\le 0.5(j_1+\cdots+j_\ell) - 0.5(i_1+\cdots+i_\ell) + (\beta-0.5)(i'_1+\cdots+i'_\ell) \\
&\le 0.5(j_1+\cdots+j_\ell) - 0.5(i_1+\cdots+i_\ell) + (\beta-0.5)(i_1+\cdots+i_\ell) \\
&= 0.5(j_1+\cdots+j_\ell) + (\beta-1)(i_1+\cdots+i_\ell) < 0.
\end{aligned}$$

The last equality holds from $(i_1,\ldots,i_\ell,j) \in I_{++}$ Therefore, for both situations the tuple $(i'_1,\ldots,i'_\ell,j')$ is in I_{++} and $L(\mathbf{G}_{++})$ is lower triangular. □

As the same argument in Section 4.2, the Coppersmith technique works if

$$\sum_{(i_1,\ldots,i_\ell,j)\in I_{++}} \left[0.5j + (i_1+\cdots+i_\ell)\beta - \sum_{k=1}^{\ell}\min(i_k,j) \right] < 0. \tag{27}$$

The explicit formula has never been given. In Table 3, we show the comparison between the bounds of β from (27) and (15), for several choices of m and ℓ. Interestingly, for $\ell = 14$, $m = 5$ and $\beta = 0.752$, it can see that (27) holds by a naive computation. Thus, it exceeds the limit of $3/4$ given by our Minkowski sum construction.

$\ell = 5$

m	3	4	5	6
β_{improve}	0.582	0.596	0.604	0.611
β_{original}	0.571	0.583	0.590	0.596

$\ell = 9$

m	3	4	5	6
β_{improve}	0.677	0.688	0.697	0.702
β_{original}	0.641	0.649	0.655	0.659

$\ell = 14$

m	3	4	5
β_{improve}	0.735	0.745	0.752
β_{original}	0.678	0.683	0.686

Table 3: Comparison of β bounds between the Minkowski sum construction and heuristic improvement

E Generating sequence that contains ℓ natural numbers coprime to $\varphi(N)$

Let N be an n-bit RSA modulus, that is $2^{n-1} < N < 2^n$ holds. To prove Theorem 3 in Section 5, it needs to give an efficient way to generate a sequence of distinct natural numbers $a_0, \ldots, a_{R-1} \in [2^{n-1} : 2^n - 1]$ of length $R = \text{poly}(\log N)$ such that at least $\ell = O(\log N)$ elements are coprime to $\varphi(N)$.

Here, we give a method to compute natural numbers K and P so that the sequence $a_i = K + iP$ for $i = 0, 1, \ldots$ satisfies this condition by the following theorem. Note that the theorem holds for $n \geq 385$. For small n, the table of prime numbers near 2^{n-1} can be used for this purpose. Here, such primes are coprime to $\varphi(N)$; this is because the maximum prime factor of $\varphi(N)$ is smaller than $N/3 < 2^{n-1}$ since $4|\varphi(N)$ holds. Noting that the base of logarithm is two throughout this section.

Theorem 5. Let N be an n-bit RSA modulus whose factor is unknown, and assume that $n \geq 385$. For fixed integer ℓ, we can easily compute natural numbers K and P so that the sequence $a_i = K + iP$ for $i = 0, \ldots, R - 1 = 4\ell \log N + \log^2 N$ satisfies conditions

(i) $2^{n-1} < a_i < 2^n - 1$ for all i, and

(ii) at least ℓ elements of the sequence are coprime to $\varphi(N)$.

Proof. Let B be the minimum integer so that $B \log B - 1 > \log N$ holds. Note that we have $\log N > \frac{B \log B - 1}{2}$ and $B \geq 64$ from $\log N \geq 384$.

Write $p_1, p_2, \ldots, p_b$ be all odd prime numbers smaller than B. Then, define our P by their product $\prod_{3 \leq p_i < B} p_i$. By the Chebychev inequality on prime numbers, we have

$$P < 4^B < 4^{\frac{2 \log N}{\log B}} < N^{2/3} < 2^{n-1}.$$

Next we define K. First set $k_3 \in \{0, 1\}$ so that $2^{n-1} + 1 + 2k_3$ is coprime to 3. For this k_3, next set $k_5 \in \{0, 1\}$ so that $2^{n-1} + 1 + 2 \cdot k_3 + 2 \cdot 3 \cdot k_5$. Repeating this process, we find the natural number defined by using integers $k_{p_i} \in \{0, 1\}$,

$$K = 2^{n-1} + 1 + 2 \cdot k_3 + 2 \cdot 3 \cdot k_5 + \cdots + 2 \cdot 3 \cdots p_{b-1} k_{p_b} \tag{28}$$

which is coprime to P.

Using the above K and P, define the sequence $a_i = K + iP$ for $i = 0, 1, \ldots, R - 1 = 4\ell \log N + \log^2 N$. We prove that this sequence satisfies our claims. First, since $1 + 2 + 2 \cdot 3 + \cdots + 2 \cdot 3 \cdots p_{b-1} < P$ holds, we have $a_i < 2^{n-1} + R \cdot P < 2^n$ which is the claim (i).

Next, we prove the claim (ii). Let us write the factoring of $\varphi(N)$ as

$$\varphi(N) = S \cdot \prod_{j=1}^{L} q_j^{e_j},$$

where S is the product of prime factors smaller than B, and q_j be the prime numbers $\geq B$. Here, note that $L < (\log N)/(\log B)$ holds since $N > \varphi(N) = S \cdot \prod_{j=1}^{L} q_j^{e_j} > B^L$. By the construction, all a_i are coprime to $P = \prod_{3 \leq p_i < B} p_i$. Thus, $q_j \not| a_i$ for all j implies that a_i is coprime to $\varphi(N)$. We count such a_i.

Since the number of a_i divisible by q_i in the sequence is bounded upper by $\lceil R/q_i \rceil$, the number of a_i coprime to all $q_1, \ldots, q_L$ is bounded lower by

$$\begin{aligned}
R - \lceil R/q_1 \rceil - \cdots - \lceil R/q_L \rceil &\geq R\left(1 - \tfrac{1}{q_1} - \cdots \tfrac{1}{q_L}\right) - L \\
&> R\left(1 - \tfrac{L}{B}\right) - L \\
&> R\left(1 - \tfrac{\log N}{B \log B}\right) - \tfrac{\log N}{\log B} \\
&= R\left(1 - \tfrac{\log N}{B \log B - 1} + \tfrac{\log N}{B \log B - 1} - \tfrac{\log N}{B \log B}\right) - \tfrac{\log N}{\log B} \\
&> R \cdot \tfrac{\log N}{B \log B (B \log B - 1)} - \tfrac{\log N}{\log B} \quad (\text{since we set} B \log B - 1 > \log N) \\
&> R \cdot \tfrac{\log N}{(B \log B - 1)^2} - \tfrac{\log N}{\log B} \\
&= \tfrac{4\ell \log^2 N + \log^3 N + \log N}{(B \log B - 1)^2} - \tfrac{\log N}{\log B} \\
&> \ell + \tfrac{\log N}{4} - \tfrac{\log N}{\log B} \quad (\text{since we have} \log N > (B \log B - 1)/2).
\end{aligned}$$

Finally, since $B \geq 64$, the last term is larger than ℓ. This is our claim (ii). $\square$