On the Optimality of Lattices for the Coppersmith Technique

Yoshinori Aono * Manindra Agrawal † Takakazu Satoh ‡ Osamu Watanabe §

December 18, 2015

Abstract

We investigate a method for finding small integer solutions of a univariate modular equation, that was introduced by Coppersmith [9] and extended by May [26]. We will refer this method as the *Coppersmith technique*.

This paper provides a way to analyze a general limitations of the lattice construction for the Coppersmith technique. Our analysis upper bounds the possible range of U that is asymptotically equal to the bound given by the original result of Coppersmith and May. This means that they have already given the best lattice construction. In addition, we investigate the optimality for the bivariate equation to solve the small inverse problem, which was inspired by Kunihiro's [22] argument. In particular, we show the optimality for the Boneh-Durfee's equation [4] used for RSA cryptoanalysis,

To show our results, we establish framework for the technique by following the relation of Howgrave-Graham [18], and then concretely define the conditions in which the technique succeed and fails. We then provide a way to analyze the range of U that satisfies these conditions. Technically, we show that the original result of Coppersmith achieves the optimal bound for U when constructing a lattice in the standard way. We then provide evidence which indicates that constructing a non-standard lattice is generally difficult.

Coppersmith technique, Lattice construction, Impossibility result, RSA cryptanalyses

1 Introduction

Let N and $F(x)$ be an integer whose factoring is not known, and a univariate polynomial of degree D. Consider the problem to find solutions to the modular equation

$$F(x) = x^D + a_{D-1}x^{D-1} + \cdots + a_0 \equiv 0 \pmod{N} \tag{1}$$

within a range of $|x| < U$. Based on preliminary work [37, 17], Coppersmith [9] introduced a general polynomial-time algorithm, which we refer to as the *Coppersmith technique*, to solve this problem with the range parameter $U = N^{1/D-\varepsilon}$. (Here, for any A, $0 < A < N$, the notation $|x| < A$ under modulo N means that x is an integer satisfying $0 \leq x < A$ or $N - A < x < N$.) Following his work

*National Institute of Information and Communications Technology, Japan, aono@nict.go.jp
†Dept. of Computer Science and Engineering, Indian Institute of Technology, Kanpur, India.
‡Dept. of Mathematics, Tokyo Institute of Technology, Tokyo, Japan.
§Dept. of Mathematical and Computing Sciences, Tokyo Institute of Technology, Tokyo, Japan.

and the reformulation by Howgrave-Graham [18], this technique has gained attention in relation to the analysis of several cryptographies; (e.g., [4, 5, 7, 34]). The technique has also been generalized for multivariate cases in a natural way.

One natural extension of the technique is given by May [26] within the context of RSA cryptanalyses. With a large composite number N for which we only know the magnitude of a prime factor defined by $P \approx N^{\beta}$, he introduced the generalized problem to find a small solution to the following equation:

$$F(x) = x^D + a_{D-1}x^{D-1} + \cdots + a_0 \equiv 0 \pmod{P}. \tag{2}$$

May also propose a polynomial-time algorithm for the range parameter $U = N^{\beta^2/D-\varepsilon}$. For the rest of this paper, we will use the words "Coppersmith technique" to refer the the general algorithm used to solve both problems.

The outline of the Coppersmith technique is (i) converting a given modular equation to a certain algebraic equation keeping the same small solutions by using a lattice reduction algorithm and (ii) finding an integer solutions of the algebraic equation by a numerical method. Because the latter problem is solved in polynomial time in n and the bit-size of coefficients [1], we consider the former problem.

One key point of this technique is to choose a good lattice for the lattice reduction algorithm, in other words, construct a lattice having short vectors. For this purpose, a lattice basis with small determinant is usually used. For instance, considering a bivariate modular equation for RSA cryptanalysis the original result of [5] has essentially been improved by defining better lattices [14, 1]. There should clearly be some limit to these improvements. Here, we mainly focus on the univariate case and investigate the optimality of the lattice construction for the Coppersmith technique.

Remarks on the problem setting:
We consider the problem for finding all the solutions for (1) within the range of $|x| < U$ for a given parameter U. We call (1) the *target equation* and the range $|x| < U$ the *target range*. Throughout this paper, the usage of symbols F, D, N, and U is fixed. We use the standard unit cost time complexity, and we evaluate complexity measures in terms of $\log N$, because we can assume that $D \leq \mathrm{poly}(\log N)$ and $U < N$. Thus, by "polynomial-time" we mean a time polynomial in $\log N$ unless otherwise stated.

We assume that N is a large composite number whose non-trivial factor cannot be found during the computation that we investigate. This is because the factor of N would provide the complete solution to the original problem in almost all applications of the Coppersmith technique. Thus, we can assume that all numbers that appear during the computation are coprime to N. This is used in the argument of Section 5. Because of this, we can assume that the coefficient a_D of x^D of $F(x)$ is one as stated in (1) since otherwise we can "divide" it by multiplying a_D^{-1} modulo N because a_D must be coprime to N.

[1] The real root isolation can be performed in polynomial time [11]. Then, rounding the approximate solutions found by using Newton's method can recover the integer solutions.

1.1 Our Results

Defining the framework and conditions for success and failure:
We investigate the conditions in which the Coppersmith technique works and fails. Roughly speaking, the conditions are defined as the input parameters D and N, the determinant of the constructed lattice basis, and the univariate polynomial converted from the output of an algorithm to find a short vector.

The former condition says that if the range parameter U is small enough, the determinant is small, and it derives that a "norm" of polynomial is also small. Then we can solve the problem by using numerical methods. This argument has been used within the context to show the possibility of the technique.

On the other hand, to discuss the impossibility of technique, we give the formal definition of the conditions in which it fails. In this situation, for a large U, any polynomials generated by "standard" integer lattices must have large norms, and it fails to solve the problem. An overview of these conditions is shown in Figure 2 in Section 3.2.

Defining the notion of (non-)standard polynomials:
Fixing the input parameters $(F(x), N, U)$, the univariate polynomials for constructing lattice bases share the same roots under certain moduli. Some of these polynomials are easily generated from the inputs, while others are not easy to find. To explicitly separate these polynomials, this paper establishes the notion of (non-)standard polynomials. The standard polynomials are a natural generalization of the way for selecting polynomials that are normally called "shift polynomials" [20].

Limitation of the technique in univariate cases:
We showed that Coppersmith's bound $U = N^{1/D-\varepsilon}$ for equation (1), and May's bound $U = N^{\beta^2/D-\varepsilon}$ for equation (2) are optimal (except for the choice of ε) if we use only standard polynomials.

We also show that a non-standard construction does indeed lead to either (i) a reduction of the original equation (1) to a strictly simpler one, or (ii) derives a non-trivial factor of the modulus N, that are assumed to be difficult to compute. Moreover, neither reduction requires the Coppersmith technique. That is, we show that such this type of non-standard construction provides for a better way to solve the original problem than the Coppersmith technique.

Thus, from these results, we can claim that the standard Coppersmith technique cannot extend the bounds by changing a lattice construction.

Limitations of the technique in solving the small inverse problem:
Based on the previous argument, we provide the limitations on the technique in solve the small inverse problem. In particular, the famous Boneh-Durfee bound [4], which claims that we can easily solve the RSA cryptosystem when the private exponent is smaller than $N^{1-1/\sqrt{2}} \approx N^{0.292}$, is optimal.

Improvements to the conference version:
This paper has been modified from the ACISP conference version [3], as follows:

- Changed the formulation of the Coppersmith technique from lattices over integers [18] to lattices over polynomials [2, 12] to enhance readability.
- Organized relationships between the conditions for when the technique succeeds and fails, and the norm of polynomials and vectors.

- Added a new technical section (Section 6) to discuss the small inverse problems.
- Omitted the part of the extended framework which uses the integer-valued polynomials because it is not essential, makes argument complicated, and affects only in a constant factors.

1.2 Related Work and Discussion

A short overview of the univariate Coppersmith technique:
We first briefly survey applications of the univariate case of the Coppersmith technique in cryptography. Before Coppersmith's work, there were similar ideas for attacking cryptographic schemes. Vallée-Girault-Toffin [37] proposed a lattice based attack for the Okamoto-Shiraishi signature scheme [29] that uses a quadratic inequality. Håstad [17] also proposed a procedure for solving simultaneous modular equations by converting them to a single modular equation. Unfortunately, its solvable range is slightly weaker because lattice construction is extremely simple.

Building on this previous work, Coppersmith [9] proposed his method for solving general univariate equations and its application for recovering an RSA message with the $(1 - 1/e)$-fraction of MSBs. After his work, Howgrave-Graham [18] reformulated the technique, and many applications have been proposed in the field of cryptography. Shoup [34], for instance, presented interesting application for proving the security of the RSA-OAEP encryption scheme with $e = 3$. Another application is discovered by Boneh [7], which proposes a modification of the Coppersmith technique to find small integers x such that $\gcd(x, N)$ is large, where N is a given integer. He also provide an application of his technique to CRT list decoding.

Our results show the limitations of these approaches. For example, the RSA message cannot be recovered from MSBs that have a length of less than $(1 - 1/e) \log_2 N$ by a single usage of the Coppersmith technique.

Previous results to show the limitations of the Coppersmith technique:
Some investigations have shown the limit of polynomial-time algorithms. Konyagin and Steger [23] present an upper bound of the number of roots of (1) within the range of $|x| < U$, which becomes exponential in $\log N$ when $U = N^{1/D+\varepsilon}$. It is also shown [28] that the bound can be attained by an equation of the form $x^r \equiv 0 \pmod{p^r}$ for a prime number p and integer r. This clearly provides the limit of an algorithm that runs in polynomial-time in $\log N$. In short, there are *some* equations for which there are no polynomial-time algorithm can be used to find *all* solutions within the range of $|x| < N^{1/D+\varepsilon}$. However, their example is somewhat extreme and the modulus is easily factored unlike our problem setting. Hence, it is insufficient to show the hardness of solving the particular equations defined for attacking cryptographies. In fact, for most of these equations, the number of solutions can be easily shown to be quite small. Our objective is to provide a technique for analyzing the limitations of the Coppersmith technique that is applicable for equations for attacking cryptographies.

In addition to univariate cases, a few papers [19, 22] discuss multivariate cases. They have tried to prove the optimality of the technique for small inverse problem. However, the drawback of these arguments is that they do not define the framework concretely.

Our framework comared to previous work for selecting polynomials:
Polynomial selection is essential for constructing the lattices used in the Coppersmith technique. The family of polynomials employed in our framework explicitly larger than that used in previous work.

Here we provide a rough sketch of this family. Our polynomials, which we named "canonical polynomials," have the form $\sum_{i=0}^{m} q_i(x)N^{m-i}(f(x))^i$ in which $q_i(x)$ is an integer coefficient polynomial. In contrast, the polynomials used in certain instances of previous work, which are normally referred to as "shift polynomials," have the form $x^j \cdot N^{m-i}(f(x))^i$. Thus, our polynomial selection allows for combinations of shift-polynomials. Note that the notion of the canonical polynomial can be used for multivariate cases, and to generalize our impossibility results is an interesting future work.

Non-single usage of Coppersmith technique:

While we investigate the limit of *the direct usage* of the Coppersmith technique, it should be noted here that several extensions of the technique have been proposed for extending the range $|x| < N^{1/d-\varepsilon}$. Examples of this can be seen in the survey papers by Coppersmith [10] and by May [28, Chapter 10]. Unfortunately, though, the improvements to these extensions are relatively mild.

One natural way is to solve multiple equations, which means solving the equations $f(x + i\lfloor N^{1/d-\varepsilon} \rfloor) \equiv 0 \pmod{N}$ for $i = 0, \pm 1, \ldots$, by applying the Coppersmith technique to each equation. This provides an algorithm that achieves the range $x < N^{1/d+O(\log\log N)}$ in polynomial-time in $\log N$. Unfortunately, this method is not very efficient in practice, but it may be still useful in a parallel computing environment. Another example is to use small solutions obtained by the Coppersmith technique to create a more powerful equation. Coppersmith [10] showed that by using two small solutions within the range of $|x| < N^{1/d}$, one can construct new simultaneous modular equations whose solution derives new solutions within $|x| < N^{1/(d-1)}$ if these solutions exist.

1.3 Structure of this Paper

The rest of this paper is as follows. Section 2 provides some necessary technical background. Section 3 precisely defines the Coppersmith technique, our framework, and the conditions for the success and failure of the technique. Section 4 defines the standard polynomials and derives the condition for the solution range that satisfies the failure condition under standard lattice construction. Section 5 discusses what we are able to compute from a non-standard construction. Section 6 proves the optimality of previous results for the small inverse problem and RSA cryptanalysis in a short secret exponent.

2 Preliminaries

Here we introduce the definitions and technical lemmas. For any positive integer n, we let $[n]$ denote the set $\{1, \ldots, n\}$. A vector consisting of $s \geq 2$ coordinates $a_1, \ldots, a_s$ is denoted as $[a_1, \ldots, a_s]$. On the other hand, for polynomials $f_1(x), \ldots, f_s(x)$, we use $(f_1, \ldots, f_s)$ to denote their sequence.

Let $\mathbb{Z}[x]$ denote the ring of integer coefficient univariate polynomials. The ring $\mathbb{Z}/N\mathbb{Z}$ is denoted by $\mathbb{Z}_N$, and $\mathbb{Z}_N[x]$ denotes the ring of polynomials whose coefficients are in $\mathbb{Z}_N$. Use $\mathbb{Z}_N^\times$ to denote the set of units; i.e., elements that have inverses, in $\mathbb{Z}_N$. Based on this, we denote the set of units in $\mathbb{Z}_N[x]$ by $\mathbb{Z}_N[x]^\times$. We also use $\mathbb{M}_N[x]$ to denote the set of *monic polynomials* in $\mathbb{Z}_N[x]$; that is, the polynomials whose leading coefficient is one.

By $\equiv_N$ we denote the equivalence between two polynomials under modulo N; that is, for two polynomials $f(x) = \sum_{i=0}^{d} a_i x^i$ and $g(x) = \sum_{i=0}^{e} b_i x^i$, we write $f(x) \equiv_N g(x)$ if $a_i \equiv b_i \pmod N$ for any i, $0 \leq i \leq \max(d, e)$. Here we understand that $a_i = 0$ (resp. $b_i = 0$) for $i > d$ (resp. $i > e$.)

For any polynomial $f(x)$, we use $\deg(f)$ and $\mathrm{lc}(f)$ to denote the degree and the leading coefficient, respectively. For any positive integer c and any polynomial $f(x)$, we define $\mathrm{ord}_c(f)$ by the largest integer r such that $f(x) \equiv_{c^r} 0$ holds.

2.1 Lattices over Euclidean Spaces and Polynomials

For a vector space V and its elements $z_1, \ldots . z_m$, define the lattice spanned by them be

$$L(z_1, \ldots, , z_k) := \{a_1 z_1 + \cdots + a_k z_k : a_1, \ldots, a_k \in \mathbb{Z}\}.$$

We call k the dimension and $\{z_1, \ldots, , z_k\}$ the basis. We sometimes omit the basis if it is clear from the context. This paper considers lattices over polynomials and Euclidean lattices. The former is used to analyse lattice construction, and the latter is used to analyse Euclidean length of short vectors.

Euclidean lattices. Fix any $w \geq k$. Consider the situation where $V = \mathbb{R}^w$. The basis consists of Euclidean vectors $(\mathbf{b}_1, \ldots, \mathbf{b}_k)$. The *determinant*, or the lattice volume, defined by $\det(L) = \prod_{i=1}^{k} |\mathbf{b}_i^*|$, where $\{\mathbf{b}_1^*, \ldots, \mathbf{b}_k^*\}$ is the Gram-Schmidt basis. Here, the notation $|\cdot|$ denotes the standard Euclidean norm.

We need to compute a non-zero short vector in a lattice in a reasonable time which can be done by an approximate shortest vector algorithm. For details, see [28]. For a given lattice L, a lattice reduction algorithm [24, 35] can find a vector $\mathbf{v}_1$ such that

$$|\mathbf{v}_1| \leq A(k) \det(L)^{1/k}, \tag{3}$$

where $A(k) < 2^{(k-1)/4}$. On the other hand, it has been observed [15] that for many polynomial-time lattice reduction algorithm with fixed parameters, we have $\delta > 1$ such that $|\mathbf{v}_1| \approx \delta^k \det(L)^{1/k}$ holds. (Here "polynomial-time" means polynomial-time w.r.t. k and $\log B \stackrel{\mathrm{def}}{=} \log \max_i |\mathbf{b}_i|$.)

If we use a stronger algorithm, which requires an over-polynomial time, the bound $A(k)$ grows smaller. Here, we remark that even if we use the shortest vector oracle, there perhaps exists a lower bound of $A(k)$. For the k-dimensional random lattices of volume $V_k(1)$, Rogers [30] proved that the distribution of $\lambda_1(L)^k$ goes to the exponential distribution of average 2 when n is sufficiently large, where $\lambda_1(L)$ is the shortest vector length of L. Thus, there exists a constant $c < 1$, the shortest vector of L is longer than $c \cdot V_k(1)^{-1/k} \det(L)^{1/k}$ for most of instances. In the rest of this paper, we assume that the univariate Coppersmith type lattices, which we will define in Section 4.2, can be regarded as the set of random lattices. A detailed explanation of this assumption is given in Assumption 1.

Polynomial lattice: For the Euclidean lattice, we use the polynomial lattice $L(g_1, \ldots, g_k)$ for $g_i \in \mathbb{Z}[x]$. We sometimes use $L(\mathbf{G})$ to denote the polynomial lattice by denoting $\mathbf{G} = (g_1, \ldots, g_n)$ the basis. Let w be the maximum degree of the polynomials. Then, the mapping $\mathcal{V}$ parametrized by the constant U is defined by

$$\mathcal{V} : \ \mathbb{Z}[x] \mapsto \mathbb{R}^{w+1} \ (f = \sum_{i=0}^{w} a_i x^i \to \mathcal{V}(f, U) := (a_0, a_1 U, \ldots, a_w U^w)),$$

which we refer to as the *vectorization*. With this, we define the parametrized norm by $||f||_U := |\mathcal{V}(f,U)|$. The parametrized determinant $\det L(\mathbf{G})$ of the basis $\mathbf{G} = (g_1, \ldots, g_k)$ is also defined by the determinant of the Euclidean lattice spanned by $\mathcal{V}(g_i, U)$ for $i \in [k]$.

On the other hand, as its inverse transformation, we define the *functionalization* of $\mathbf{v}$ for a given vector $\mathbf{v}$, as a unique polynomial $f(x)$ such that $\mathbf{v} = \mathcal{V}(f,U)$ holds, and denote it by $\mathcal{F}(\mathbf{v},U)$. Note that this is undefined if $f(x)$ does not exist. $\mathcal{V}(f,U)$ and $\mathcal{F}(\mathbf{v},U)$ are clearly linear mapping w.r.t. polynomials and vectors, respectively.

2.2 Properties of Polynomial Norm

The following lemma is used to connect the output of the lattice reduction algorithm to the solvable range of the target equation.

Lemma 1. *(***Howgrave-Graham** *[18])* *Consider a polynomial $f(x) \in \mathbb{Z}[x]$ consisting of w monomials. Let W be a non-negative integer satisfying*

$$||f||_U < W/\sqrt{w}. \tag{4}$$

Then we have

$$\forall v,\ |v| < U\ [\ f(v) \equiv 0 \pmod{W} \Leftrightarrow f(v) = 0\ ]. \tag{5}$$

Note that (4) implies $|f(x)| < W$ for $x \in [-U, U]$ in the proof. We state the contrapositive to argue the condition of the failure for the Coppersmith technique.

Corollary 1. *The notations are the same as Lemma 1. Suppose there exists a real number $x \in [-U, U]$ so that $|f(x)| > W$. Then it must be $||f||_U \geq W/\sqrt{w}$.*

We also show the converse type claim of Lemma 1 to discuss a condition which implies that the Coppersmith technique fails to work. We start at the standard consequence of Chevychev's theorem.

Lemma 2. *For any polynomial $h(x)$ having leading coefficient l and degree d,*

$$l \leq 2^{d-1} \max_{x \in [-1,1]} |h(x)|.$$

Let $T_n(x) := \cos(n \cos^{-1} x)$ be the Chebyshev polynomial of degree n. It holds that $\mathrm{lc}(T_n) = 2^{n-1}$ and

$$\max_{x \in [-1,1]} |T_n(x)| = 1.$$

Lemma 3. *For any polynomial $h(x) = h_d x^d + \cdots + h_0$ of degree d, an integer W and a range parameter U, suppose that $\max_{x \in [-U,U]} |h(x)| < 1$ holds. Then, we have $||h||_U < d\sqrt{d+1} \cdot 3^{d+1}$.*

Proof. We first consider the simple case $U = 1$. By Lemma 2, $\max_{x \in [-1,1]} |h(x)| < 1$ implies $|h_d| \leq 2^{d-1}$. Let us fix this polynomial as $H_d(x) := h(x)$ and define $H_{d-1}(x) := H_d(x) - h_d 2^{1-d} T_d(x)$ whose degree is $d-1$. Now we have

$$\max_{x \in [-1,1]} |H_{d-1}(x)| \leq \max_{x \in [-1,1]} |H_d(x)| + \max_{x \in [-1,1]} |h_d 2^{1-d} T_d(x)| < 1 + |h_d| 2^{1-d} \leq 2$$

and $|\mathrm{lc}(H_{d-1})| \le 2^{d-2} \cdot \max_{x\in[-1,1]} |H_{d-1}(x)| \le 2^{d-1}$.

Define $H_{d-i}(x) := H_d(x) - \sum_{j=d-i+1}^{d} lc(H_j)2^{1-j}T_j(x)$ recursively. We show by induction that $\max_{x\in[-1,1]} |H_{d-i}(x)| \le 2^i$ and $|\mathrm{lc}(H_{d-i})| \le 2^{d-1}$ for $i = 1, \ldots, d$, starting from the base case $i = 1$.

Suppose the claim holds for $i < k$. Then, we have for $i = k$,

$$\begin{aligned}\max_{x\in[-1,1]} |H_{d-k}(x)| &\le \max_{x\in[-1,1]} |H_d(x)| + \sum_{j=d-k+1}^{d} \max_{x\in[-1,1]} |\mathrm{lc}(H_j)2^{1-j}T_j(x)| \\ &\le 1 + \textstyle\sum_{j=d-k+1}^{d} 2^{d-j} = 2^k\end{aligned}$$

Since $\deg(H_{d-k}) = d - k$, we have $|\mathrm{lc}(H_{d-k})| \le 2^{d-k-1} \cdot \max_{x\in[-1,1]} |H_{d-k}(x)| \le 2^{d-1}$.

Next we bound $|h_i|$ by using the relation $h(x) = \sum_{j=0}^{d} \mathrm{lc}(H_j)2^{1-j}T_j(x)$. By the standard formula for Chebychev polynomials,

$$T_n(x) = \frac{n}{2} \sum_{k=0}^{\lceil n/2 \rceil} \frac{(-1)^k}{n-k} \cdot \binom{n-k}{k} (2x)^{n-2k}$$

holds. Thus, the absolute value of each coefficient in $T_n(x)$ above is bounded by

$$\begin{aligned}&\frac{n}{2(n-k)}\binom{n-k}{k}2^{n-2k} < n2^{n-1} \sum_{k=0}^{\lceil n/2 \rceil} \binom{n-k}{k} \left(\frac{1}{4}\right)^k \\ &= n2^{-5/2}\left[(1+\sqrt{2})^{n+1} - (1-\sqrt{2})^{n+1}\right] < n \cdot 3^n.\end{aligned}$$

Here, the equation is from [31, Formula 4.2.3-6] with $x = 1/4$. Hence we have the following:

$$|h_i| \le \sum_{j=i}^{d} |\mathrm{lc}(H_j)2^{1-j}| \cdot j \cdot 3^j \le \sum_{j=i}^{d} 2^{d-1}2^{1-j} \cdot j \cdot 3^j = 2^d \sum_{j=i}^{d} j \cdot (3/2)^j < d \cdot 3^{d+1}.$$

Therefore, for any polynomial $h(x) = h_d x^d + \cdots + h_0$, we have $|h_i| < d \cdot 3^{d+1} \max_{x\in[-1,1]} |h(x)|$.

Considering a scaled polynomial $h(Ux) = h_d U^d x^d + \cdots h_0$, we have

$$|U^i h_i| < d \cdot 3^{d+1} \max_{x\in[-1,1]} |h(Ux)| = d \cdot 3^{d+1} \max_{x\in[-U,U]} |h(x)| < d \cdot 3^{d+1}.$$

Finally, we obtain the upper bound of polynomial norm as

$$||h||_U^2 = \sum_{i=0}^{d} h_i^2 U^{2i} < d^2(d+1) \cdot 3^{2(d+1)} \max_{x\in[-U,U]} |h(x)|,$$

which derives the claim. □

The contrapositive of this lemma will play a key role in showing the impossibility of the Coppersmith technique.

Claim 1.

$$||h||_U > d\sqrt{d+1} \cdot 3^{d+1} N^m \tag{6}$$

implies $\max_{x\in[-U,U]} |h(x)| > N^m$ *where* $d = \dim(h)$. *Thus, there exists a case in which a solution of* $h(x) \equiv 0 \pmod{N^m}$ *is not the root over integer.*

We will define the "success" and "failure" of the Coppersmith technique in the next section.

3 Framework for the Coppersmith Technique

This section introduces our framework for discussing the Coppersmith technique for a univariate equation. We also precisely define the conditions for success and failure.

As mentioned in the previous section, for a given target equation (1) and a target range specified by U, our task is to find all the solutions within the target range. For this task, we formulate the Coppersmith technique as an algorithm stated as Figure 1. Our formulation is mainly from Howgrave-Graham [18] and its algebraic extension by Heninger and Cohn [12].

Input	$F(x)$, N, U; **Parameters** $k \geq 1$, $m \geq 2$;
Output	All solutions of $F(x) \equiv 0 \pmod{N}$ satisfying $\|x\| < U$;
Step 1:	Based on the input, define a sequence of linearly independent polynomials $\mathbf{G} = (g_1, \ldots, g_k)$ that satisfy (7);
Step 2:	Find a polynomial $h(x) \in L(g_1, \ldots, g_k)$ having small $\|\|h\|\|_U$ by using the LLL algorithm.
Step 3:	Solve the equation $h(x) = 0$ numerically; Output all integer roots within the target range satisfying (1);

Figure 1: Outline of Coppersmith technique

Remarks may be necessary for some steps of the algorithm. First note that the algorithm is given two parameters $k \geq 1$ and $m \geq 2$, which are chosen (often heuristically) for the target equation. The parameters are usually chosen as small because the time complexity of the original LLL algorithm [24] is $O(k^5 u \log^3 B)$ (e.g., [28, Chapter 5]), where u is the dimension of each vector and $B = \log \max_i ||\mathbf{b}_i||$. We can at least assume that these parameters related to time complexity are $\mathrm{poly}(\log N)$, and thus this assumption is sufficient for our analysis. Therefore, throughout the following discussion, we will consider any $k, m, u, \log B \leq (\log N)^c$ for $c > 0$ and let them be fixed.

In Step 1, we define linearly independent polynomials $g_1(x), \ldots, g_k(x) \in \mathbb{Z}[x]$, which we call *initial polynomials*, that satisfy

$$\forall v \ [\ F(v) \equiv 0 \pmod{N} \Rightarrow g_i(v) \equiv 0 \pmod{N^m}\ ]. \tag{7}$$

As we will see, the choice of these polynomials determines the lattice used in the algorithm. This is crucial for the performance of the algorithm. Again, they are defined somewhat heuristically in each application of the technique. We can at least assume that their degrees are bounded by $\mathrm{poly}(\log N)$. From the role of the parameter m in the above, we refer to m as an *initial exponent*.

In Step 2, define vectors $\mathbf{b}_1, \ldots, \mathbf{b}_k$ by $\mathbf{b}_i = \mathcal{V}(g_i, U)$ for $i \in [k]$ to carry out the LLL algorithm. Let us denote $\mathbf{v}_1, \ldots, \mathbf{v}_k$ as the output of the LLL algorithm on $L(\mathbf{b}_1, \ldots, \mathbf{b}_k)$. Then we bring the first vector back to the polynomial by $h(x) = \mathcal{F}(\mathbf{v}_1, U)$.

In Step 3, we enumerate all the roots of $h(x)$. By using a numerical method, this task can be done in polynomial time in $\deg(h)$ and $\log C$, both of which are bound in polynomial in $\log N$, where C is the absolute maximum coefficient in $h(x)$. Note also that the number of roots is polynomially bound. Finally, from among the roots obtained, output integers within the target range which satisfy (1).

In designing an algorithm based on the outline in Figure 1, the key point is the choice of initial polynomials that determine the lattice $L(\mathbf{G})$ used to compute a final $h(x)$.

3.1 Sufficient Condition for Success

Clearly, the algorithm works correctly when

$$\forall v,\ |v| < U\ [\ F(v) \equiv 0 \pmod N \Rightarrow h(v) = 0\ ].$$

Since $h(x) \in L(g_1, \ldots, g_k)$, and the requirement (7) for the initial polynomials, it follows that

$$\forall v\ [\ F(v) \equiv 0 \pmod N \Rightarrow h(v) \equiv 0 \pmod{N^m}\ ].$$

Thus, our above goal is satisfied if we have

$$\forall v,\ |v| < U\ [\ h(v) \equiv 0 \pmod{N^m} \Leftrightarrow h(v) = 0\ ]. \tag{8}$$

By Lemma 1, we see that

$$||h||_U < N^m/\sqrt{d_{\max}+1}\ (< N^m/\sqrt{\deg(h)+1}) \tag{9}$$

is sufficient for (8), where $d_{\max}$ is the largest degree of the initial polynomials. Then, the condition sufficient to show the algorithm works is derived by evaluating $||h||_U$. First, by definition of $h(x)$ and $\mathcal{F}(\cdot, U)$, and the bound (3), we have $||h||_U = |\mathbf{v}_1| \leq A(k)\det(L(\mathbf{G}))^{1/k}$, Therefore, (8) is implied by

$$\det(L)^{1/k}/N^m < \left(A(k)\sqrt{d_{\max}+1}\right)^{-1}. \tag{10}$$

Note that this condition is decided by the selection of the initial polynomials and range factor U. In fact, by using the initial polynomials derived from the original work by Coppersmith [9], we can show that this condition is satisfied if $U \leq N^{1/D-\varepsilon}$ (for any $\varepsilon > 0$ if N and m are large enough), thereby confirming in our framework that the original method [9] works for this range of U.

3.2 Conditions for Failure

In contrast to the case of success, which was defined for the bound U and lattice L, the condition for failure is debatable because how to define the notion that "Coppersmith technique is failure to work" was unclear in literature. We start by defining it.

For the given problem set $(F(x), N, U)$, we define that the *Coppersmith technique fails* if for any exponent m, polynomial lattice L and $h(x) \in L$, there exists *a real number* $v \in [-U, U]$ such that $|h(v)| > N^m$. By Claim 1, it suffices that $||h||_U$ is large for any $h \in L$. Thus, the problem is to evaluate the parametrized norm of polynomials in a polynomial lattice spanned by the initial vectors.

Figure 2 shows the relationship between the conditions for success and failure. We will discuss when condition (6) holds.

Note that (10′) is the inverse of (10). Now, we cannot theoretically guarantee that (10 ′) implies (6). To our best knowledge, evidence to connect the two conditions can be obtained from the random lattice theory by Rogers [30]. For details, see our discussion in Section 4.2.

4 Analysis of Canonical Initial Polynomials

We consider standard lattice construction and investigate its properties. Based on this investigation, we derive a lower bound for U such that the condition for failure (6) holds. We may regard this as the limit of U so that the Coppersmith technique works.

(Algorithm works)	(Algorithm fails to work)	
$\Uparrow$	$\Updownarrow$ (def)	
$\max_{x\in[-U,U]} \lvert h(x)\rvert < N^m$	$\max_{x\in[-U,U]} \lvert h(x)\rvert > N^m$	
$\Uparrow$	$\Uparrow$ (Claim 1)	
$\lvert\lvert h\rvert\rvert_U < N^m/\sqrt{d_{\max}+1}$	$\lvert\lvert h\rvert\rvert_U > d_{\max}\sqrt{d_{\max}+1}\cdot 3^{d_{\max}+1}N^m$	(6)
$\Uparrow$	$\Uparrow$ (?)	
(10)	$\det(L)^{1/k}/N^m > (A(k)\sqrt{d_{\max}+1})^{-1}$ (10′)	

Figure 2: Relationship between the conditions for success and failure

4.1 Defining Standard Polynomials and a Simple Bound

Since the initial polynomials need to satisfy the condition (7), one trivial way to define the polynomials $g(x)$ is by

$$g(x) = \sum_{i=0}^{m} q_i(x) N^{m-i} (F(x))^i, \quad \text{where } q_i(x) \in \mathbb{Z}[x]. \tag{11}$$

This is an integer linear combination of polynomials that were usually referred as "shift polynomials" in previous work. Formally, we introduce the following notion.

Definition 1. *Consider the ideal $\mathfrak{a} = \langle F(x), N\rangle_{\mathbb{Z}[x]}$ in the polynomial ring $\mathbb{Z}[x]$. For any non-zero polynomial $f(x) \in \mathbb{Z}[x]$, let $\nu(f)$ be the $\mathfrak{a}$-adic order of $f(x)$, that is, an integer s that satisfies $f(x) \in \mathfrak{a}^s$ and $f(x) \notin \mathfrak{a}^{s+1}$. For the zero polynomial, define $\nu(0) = \infty$. We say $f(x)$ is an s-canonical polynomial if $\nu(f) \geq s$.*

We simply say that $f(x)$ is *canonical* if $\nu(f) \geq m$ for the initial exponent m. Initial polynomials (or similar ones) used in previous work are all canonical and we can consider that using canonical polynomials to be the standard means for defining initial polynomials. This section discusses the case in which initial polynomials are all canonical.

Consider a sequence of any linearly independent initial polynomials $\mathbf{G} = (g_1, \ldots, g_k)$ of initial exponent m, and the polynomial lattice $L(\mathbf{G})$. Any polynomial $h(x) \in L(\mathbf{G})$ is canonical and $\nu(h) \geq m$ again. It is clear that $\deg(h) \geq mD$ by (11). Thus, we have the following theorem.

Theorem 1. *Fix the initial exponent and polynomial lattice spanned by canonical polynomials. If $U \geq 4N^{\beta/D}$, the technique fails. In particular, letting $\beta = 1$, the original lattice construction of the Coppersmith technique is optimal up to the constant.*

Proof. We fix a polynomial $h(x)$ in the lattice and let its degree $d(\geq mD)$. Since $h(x) = h_d x^d + \cdots + h_0$ and $|h_d| \geq 1$, we have $||h||_U^2 = \sum_{i=0}^{d} h_i^2 U^{2i} > U^{2d}$. By assumption we have

$$||h||_U > U^d \geq 4^d N^{\beta\cdot d/D} \geq \frac{1}{4}\left(\frac{4}{3}\right)^{d+1} \cdot 3^{d+1} \cdot N^{\beta m} \geq d\sqrt{d+1}\cdot 3^{d+1} N^{\beta m}.$$

The last inequality holds for integers $d \geq 19$. Thus, by Claim 1, the technique fails. □

For $\beta < 1$, there is a gap from May's bound $N^{\beta^2/D}$. This is because our bounding $||g||_U \geq U^d$ is not sharp enough.

4.2 A Better Bound using a Heuristic Assumption

We improve the norm bound using the heuristic assumption between the shortest vector lengths and the determinant of the lattices. By definition, any initial polynomial of degree d is contained in the polynomial lattice

$$L_d := \{x^{j'} N^{m-i'} (F(x))^{i'}\}_{i=0,1,\ldots,d} \text{ where } i' = \min(m, \lfloor i/D \rfloor) \text{ and } j' = i - Di',$$

which we call the *Coppersmith type lattice*. Hence, a polynomial $h(x)$ found by a lattice reduction algorithm that is also contained in the lattice. To bind the norm of the polynomials, we set the following assumption.

Assumption 1. *For Coppersmith type lattices constructed from univariate polynomials, assume that $\lambda_1(L_d) \geq \frac{1}{2} GH(L_d)$ holds with high probability. The probability is considered over the choice of $(F(x), N, U)$ and d.*

Theorem 2. *Fix the initial exponent and polynomial lattice spanned by canonical polynomials. Under Assumption 1, if $U \geq 4^2 N^{\beta^2/D}$, the technique fails in an overwhelming proportion of $(F(x), N, U)$.*

Proof. Fix the problem set $(F(x), N, U)$ and let $h(x)$ be a polynomial found in Step 2. Denote $d = \deg(h)$. Since $h(x) \in L_d$, its norm is bounded by $||h||_U \geq \lambda_1(L_d)$. For the range parameter U, the determinant $\det(L_d))$ is the product of diagonal elements:

$$\prod_{i=0}^{d} U^{j'+i'D} N^{m-i'} = \prod_{i=0}^{d} U^{i} N^{m-i'} = U^{d(d+1)/2} N^{\eta}$$

$$\text{where } \eta = (d+1)m + \frac{D}{2} \left\lfloor \frac{d}{D} \right\rfloor^2 + \left(\frac{D}{2} - d - 1 \right) \left\lfloor \frac{d}{D} \right\rfloor \quad (d < D(m+1))$$

$$\eta = \frac{Dm(m+1)}{2} \quad (d \geq D(m+1))$$

Thus, using Assumption 1, and the inequality

$$\frac{1}{2} \cdot V_d(1)^{-1/d} = \frac{1}{2\sqrt{\pi}} \Gamma(n/2+1)^{1/n} > \frac{1}{\sqrt{\pi}},$$

we have

$$\lambda_1(L(\mathbf{G})) \geq (1/2) \cdot V_d(1)^{-1/d} U^{(d+1)/2} N^{\eta/d} > \pi^{-1/2} U^{(d+1)/2} N^{\eta/d}. \tag{12}$$

We show the last factor is larger than $d\sqrt{d+1} \cdot 3^{d+1} N^{\beta m}$ when $U = 4^2 N^{\beta^2/D}$ by separating the situation that $d < D(m+1)$ and otherwise.

For the former situation, we have

$$\lambda_1(L(\mathbf{G}))/N^{\beta m} \geq \pi^{-1/2} 4^{d+1} N^{E} \geq d\sqrt{d+1} \cdot 3^{d+1} N^{E}.$$

Here, $\pi^{-1/2} 4^{d+1} \geq d\sqrt{d+1}$ holds for integers $d \geq 21$, and the exponential part is bound as follows

by using $\left\lfloor \frac{d}{D} \right\rfloor \leq m$:

$$\begin{aligned} E &= \frac{\beta^2(d+1)}{2D} - \beta m + \frac{d+1}{d} \cdot m + \frac{D}{2d} \left\lfloor \frac{d}{D} \right\rfloor^2 + \frac{1}{d} \left(\frac{D}{2} - d - 1 \right) \left\lfloor \frac{d}{D} \right\rfloor \\ &\geq \frac{d+1}{2D} \left(\beta - \frac{Dm}{d+1} \right)^2 + \frac{Dm^2}{2(d+1)} + \frac{D}{2d} \left\lfloor \frac{d}{D} \right\rfloor^2 + \frac{d+1}{d} \cdot m \\ &\quad + \frac{1}{d} \left(\frac{D}{2} - d - 1 \right) m \\ &= \frac{d+1}{2D} \left(\beta - \frac{Dm}{d+1} \right)^2 + \frac{Dm^2}{2(d+1)} + \frac{D}{2d} \left\lfloor \frac{d}{D} \right\rfloor^2 + \frac{Dm}{2d} > 0. \end{aligned}$$

Thus, we get $\lambda_1(L(\mathbf{G})) \geq N^{\beta m}$.

On the other hand, for $d \geq D(m+1)$, we have

$$\lambda_1(L(\mathbf{G})) \geq \pi^{-1/2} 4^{d+1} N^{\frac{d+1}{2} \cdot \frac{\beta^2}{D} + \frac{\eta}{d}} \geq \sqrt{d+1} \cdot 3^{d+1} N^{\frac{d+1}{2} \cdot \frac{\beta^2}{D} + \frac{\eta}{d}}.$$

The last exponent of N is larger than βm because

$$\frac{d+1}{2} \cdot \frac{\beta^2}{D} + \frac{\eta}{d} - \beta m = \frac{d+1}{2D} \left(\beta - \frac{Dm}{d+1} \right)^2 + \frac{Dm}{2} \left(-\frac{m}{d+1} + \frac{m+1}{d} \right) > 0.$$

Therefore, the technique fails in both cases. □

5 Computation from Non-canonical Polynomials

This section considers the possibility of using non-canonical initial polynomials, that is, a given polynomial $F(x)$ and an initial exponent m, an initial polynomial $g(x)$ satisfying (7) and $\nu(g) < m$. Hewe we discuss what we are able to compute if we can indeed construct a non-canonical polynomial. We show technical evidence supporting that there is no polynomial-time algorithm computing such a non-canonical polynomial for any $F(x)$, N, and m.

Technically, we show that if $F(x)$ and its derivative have no common factor (a property we call "separability" following the polynomial theory over a field; e.g., [16, Def. 2]), then by using such a non-canonical initial polynomial, it is possible to compute either a non-trivial factor of N or a polynomial $G(x)$ with $\deg(G) \leq \deg(F) - 2$ that keeps the same set of solutions. This computation can also be done in polynomial-time. One can also show that a simpler polynomial $G(x)$ (if it is obtained) also has separability. Thus, if there were a general polynomial-time algorithm for computing a non-canonical polynomial, we would be able to continue to create simpler polynomials (unless a factor of N is obtained). We would then eventually create a linear equation, which derives either a contradiction if $F(x)$ has more than one solution, or a way to compute $F(x)$'s unique solution in polynomial-time. Thus, if we had such a general algorithm, we would be able to use it to compute either the unique solution of $F(x)$ or a factor of N. Note that this does not rule out the possibility of having *a specific* $F(x)$ and m for which non-canonical initial polynomials are easy to compute. However, we believe that the one-step reduction itself yields some remarkable consequences for many concrete cases.

5.1 Algebraic Properties of Polynomials

Our investigation is based on arithmetic computations under modulo N. As explained in the introduction, we can assume that no factor of N appears during these computations; that is, we can treat N as a prime number in the following analysis. Below, we clarify the points where careful arguments are necessary.

We use standard arithmetics in $\mathbb{Z}_N[x]$. There is no problem with addition, subtraction, and multiplication, which can be defined the same as in $\mathbb{Z}[x]$. On the other hand, the division is defined as follows. For any $f(x), g(x) \in \mathbb{Z}_N[x]$, $g(x) \not\equiv_N 0$, consider polynomials $q(x) \in \mathbb{Z}_N[x]$ and $r(x) \in \mathbb{Z}_N[x]$ such that satisfy $f(x) \equiv_N q(x)g(x)+r(x)$ and $\deg(r) < \deg(g)$ (recall that $\equiv_N$ means the polynomial equivalence under modulo N). Note that $q(x)$ and $r(x)$ are unique when the leading coefficient of $g(x)$ is coprime to N. Thus, under our assumption, we can consider $q(x)$ and $r(x)$ the *quotient* and the *remainder* of $f(x)$ divided by $g(x)$, and denote them by $\mathrm{quo}(f, g)$ and $\mathrm{rem}(f, g)$, respectively. We say that $g(x)$ *divides* $f(x)$ under modulo N or $g(x)$ an *N-divisor* of $f(x)$ (and write it as $g(x)|_N f(x)$) if $r(x) \equiv_N 0$ in the above.

For any two polynomials $f(x)$ and $g(x)$, we say they are N-coprime to each other if $h(x)|_N f(x)$ and $h(x)|_N g(x)$ implies that $h(x) \in \mathbb{Z}_N[x]^\times$. On the other hand, for a monic polynomial $h(x) \in \mathbb{M}_N[x]$ satisfying $h(x)|_N f(x)$ and $h(x)|_N g(x)$, we say it is a *greatest monic common N-divisor* if the quotients $\mathrm{quo}(f, h)$ and $\mathrm{quo}(g, h)$ are N-coprime to each other. We say that $f(x)$ and $g(x)$ are *strictly coprime* if $f(x)$ and $g(x)$ generate the unit ideal of $\mathbb{Z}_N[x]$. The following example described in [25, p.32] would illustrate this: $f(x)$ and $x - a$ are N-coprime if and only if $f(a) \neq 0$ while they are strictly coprime if and only if $f(a) \in \mathbb{Z}_N^\times$. Recall also that a polynomial $f(x)$ is said to be separable if $f(x)$ and its derivative are N-coprime to each other. We can use the standard Euclidean algorithm to compute this divisor of given $f(x)$ and $g(x)$. This computation yields the *unique* greatest monic common N-divisor unless a non-trivial factor of N is computed during the computation.

5.2 Extended Euclidean Algorithm for Polynomials under Modulo N

We first give the algorithm shown in Figure 3, which is a modification of the extended Euclidean algorithm for polynomials. For given N-coprime polynomials $f(x)$ and $g(x) \in \mathbb{M}_N[x]$, it computes a pair of polynomials stated in the following lemma if the polynomial division under modulo N is always defined throughout the execution of the algorithm. If otherwise, i.e., the division is not defined in iteration, it returns a non-trivial factor of the modulo.

Lemma 4. *Let $f(x)$ and $g(x) \in \mathbb{M}_N[x]$ be N-coprime. Then, the algorithm in Figure 3 computes either a non-trivial factor of N, or a pair of polynomials $(u(x), v(x))$ satisfying $u(x)f(x) + v(x)g(x) \equiv_N 1$ in polynomial-time w.r.t. $\deg(f)$, $\deg(g)$, and $\log N$. In particular, in the latter case, $f(x)$ and $g(x)$ are strictly coprime.*

Proof. As in the case of the standard Euclidean algorithm, the ideal $\langle r_0(x), r_1(x)\rangle_{\mathbb{Z}_N[x]}$ is equal to $\langle f(x), g(x)\rangle_{\mathbb{Z}_N[x]}$ at the beginning of the while loop. Suppose the algorithm terminated with returning a pair $(u_1(x), v_1(x))$ of polynomials. Then, $\langle f(x), g(x)\rangle_{\mathbb{Z}_N[x]} = \langle r_1(x)\rangle_{\mathbb{Z}_N[x]}$; hence, both $r_1(x)|_N f(x)$ and $r_1(x)|_N g(x)$ hold. Since $f(x)$ and $g(x)$ are N-coprime, $r_1(x) \in \mathbb{Z}_N[x]^\times$. Moreover, it is monic by the construction. Hence $r_1(x) \equiv_N 1$ and $(u_1(x), v_1(x))$ has the required property. The assertion on time complexity follows from the same argument as the one for the standard Euclidean algorithm. □

Input: $f(x)$, $g(x) \in \mathbb{M}_N[x]$ which are N-coprime;
Output: Either a non-trivial factor of N or $u(x), v(x) \in \mathbb{Z}_N[x]$ satisfying $u(x)f(x) + v(x)g(x) \equiv_N 1$;
Step 1: Initialize as $r_0(x) \leftarrow f(x)$; $u_0(x) \leftarrow 1$; $v_0(x) \leftarrow 0$;
$r_1(x) \leftarrow g(x)$; $u_1(x) \leftarrow 0$; $v_1(x) \leftarrow 1$;
while $(\mathrm{rem}(r_0, r_1) \not\equiv_N 0)$ {
Step 2: $q(x) \leftarrow \mathrm{quo}(r_0, r_1)$;
$r_2(x) \leftarrow r_0(x) - q(x)r_1(x)$; $u_2(x) \leftarrow u_0(x) - q(x)u_1(x)$;
$v_2(x) \leftarrow v_0(x) - q(x)v_1(x)$;
Step 3: **if** $(\mathrm{lc}(r_2) \notin \mathbb{Z}_N^\times$ **then return** $\gcd(N, \mathrm{lc}(r_2))$; // a non-trivial factor of N
Step 4: $r_0(x) \leftarrow r_1(x)$; $u_0(x) \leftarrow u_1(x)$; $v_0(x) \leftarrow v_1(x)$;
$c \leftarrow \mathrm{lc}(r_2)^{-1} \pmod N$; $r_1(x) \leftarrow cr_2(x)$; $u_1(x) \leftarrow cu_2(x)$; $v_1(x) \leftarrow cv_2(x)$;
}
Step 5: **return** $(u_1(x), v_1(x))$;

Figure 3: Extended Euclidean algorithm for two polynomials in $\mathbb{M}_N[x]$

The following proposition shows that the algorithm in Figure 3 always outputs a non-trivial factor of N from such a polynomial pair. As we show later, such pair is derived from a non-canonical polynomials.

Proposition 1. *Let $f(x)$ and $g(x) \in \mathbb{M}_N[x]$ be N-coprime. Fix an integer $s \geq 1$. Then, by the algorithm in Figure 3, we can obtain either a non-trivial factor of N, or a pair of polynomials $(u(x), v(x))$ satisfying $u(x)f(x) + v(x)(g(x))^s \equiv_N 1$ in polynomial-time w.r.t. $\deg(f)$, $\deg(g)$, s, and $\log N$. Particularly, if $f(x)$ and $(g(x))^s$ are not N-coprime, a factor of N is always obtained.*

Proof. Consider the execution of the algorithm on $f(x)$ and $g(x)$. In the case we obtain a non-trivial factor of N, we have finished. Otherwise, we obtain $u(x), v(x) \in \mathbb{Z}_N[x]$ satisfying $u(x)f(x) + v(x)g(x) \equiv_N 1$. Hence, we have

$$\begin{aligned} 1 &\equiv_N \big(u(x)f(x) + v(x)g(x)\big)^s \\ &\equiv_N \left(\sum_{t=1}^{s} \binom{s}{t} (u(x))^t (f(x))^{t-1} (v(x)g(x))^{s-t}\right) f(x) + (v(x))^s (g(x))^s \end{aligned}$$

which implies $f(x)$ and $(g(x))^s$ are strictly coprime. A fortiori, $f(x)$ and $(g(x))^s$ are N-coprime. Thus, if $f(x)$ and $(g(x))^s$ are not N-coprime, it must return a non-trivial factor of N. $\square$

5.3 Derivatives of Polynomials

We use the derivative of polynomials that are defined in the standard way in $\mathbb{Z}[x]$ (even if they are sometimes treated as polynomials in $\mathbb{Z}_N[x]$). For $s \in \mathbb{N}$, use $f^{(s)}(x)$ to denote the s-th derivative of a polynomial $f(x)$.

Proposition 2. *For $f(x), g(x) \in \mathbb{Z}[x]$ and $s \in \mathbb{N}$ satisfying $\gcd(s!, N) = 1$, if it holds that*

$$\forall v \left[f(v) \equiv 0 \pmod N \Rightarrow g(v) \equiv 0 \pmod{N^{s+1}} \right], \tag{13}$$

we then have

$$\forall v \left[f(v) \equiv 0 \pmod{N} \Rightarrow g^{(s)}(v) \equiv 0 \pmod{N} \right].$$

Proof. Considering the Taylor expansion of $g(x)$, we have for each $i \in [s+1]$,

$$g(x+iN) = g(x) + (iN) \cdot g^{(1)}(x) + (iN)^2 \cdot \frac{1}{2!} \cdot g^{(2)}(x) + \cdots .$$

Thus, under modulo N^{s+1}, the following polynomial relation holds.

$$\begin{bmatrix} g(x+N) \\ g(x+2N) \\ g(x+3N) \\ \vdots \\ g(x+(s+1)N) \end{bmatrix} \equiv_{N^{s+1}} \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & 2 & 3 & \cdots & s+1 \\ 1 & 2^2 & 3^2 & \cdots & (s+1)^2 \\ & \vdots & & \ddots & \vdots \\ 1 & 2^s & 3^s & \cdots & (s+1)^s \end{bmatrix} \begin{bmatrix} g(x) \\ N \cdot g^{(1)}(x) \\ (N^2/2!) \cdot g^{(2)}(x) \\ \vdots \\ (N^s/s!) \cdot g^{(s)}(x) \end{bmatrix} \tag{14}$$

We can easily see the matrix is invertible under modulo N^{s+1} since it is the transpose of a van der Monde matrix and $\gcd(s!, N) = 1$. Then, for constants $c_i \in \mathbb{Z}/N^{s+1}\mathbb{Z}$, we have

$$\left(N^s/s!\right) \cdot g^{(s)}(x) \equiv_{N^{s+1}} \sum_{i=1}^{s+1} c_i g(x+iN).$$

On the other hand, we have for any integer i,

$$\forall v \Big[f(v) \equiv 0 \pmod{N} \Rightarrow f(v+iN) \equiv 0 \pmod{N} \Rightarrow g(v+iN) \equiv 0 \pmod{N^{s+1}} \Big].$$

Thus, combining them we have

$$\forall v \Big[f(v) \equiv 0 \pmod{N} \Rightarrow N^s g^{(s)}(v) \equiv 0 \pmod{N^{s+1}} \Big],$$

and the claim holds. □

Consider any s-canonical polynomial $g(x) = \sum_{i=0}^{s} q_i(x) N^{s-i} (F(x))^i$ where $q_i(x) \in \mathbb{Z}[x]$. Then, it is easy to see that $g^{(s)}(x) \equiv_N s! \cdot q_s(x) \cdot (F^{(1)}(x))^s + r(x) \cdot F(x)$ holds for $r(x) \in \mathbb{Z}_N[x]$. Thus, by using the above proposition, if (13) holds for $F(x)$ and $g(x)$, and we then have

$$\forall v \left[F(v) \equiv 0 \pmod{N} \Rightarrow q_s(v)(F^{(1)}(v))^s \equiv 0 \pmod{N} \right]. \tag{15}$$

5.4 Computing from a Non-Canonical Polynomial

We discuss what we are able to compute from a non-canonical initial polynomial for given $F(x)$, N, and m. We need to assume that $F(x)$ is *separable*, that is, $F(x)$ and its derivative are N-coprime to each other. Our result is given by the following theorem.

Theorem 3. *Assume that our target polynomial $F(x)$ is separable. For $F(x)$, N, and m, suppose that we have a non-canonical initial polynomial $g(x)$; that is, it satisfies both $\nu(g) \leq m-1$ and*

the condition (7). Then we can compute in polynomial-time w.r.t. $\log N$ *and* $\deg(g)$*, either a non-trivial factor of* N *or a polynomial* $G(x)$ *with* $\deg(G) \leq \deg(F) - 2$ *satisfying*

$$\forall v \ [\ F(v) \equiv 0 \pmod{N} \Rightarrow G(v) \equiv 0 \pmod{N}\ ]. \tag{16}$$

Moreover, $G(x)$ *is an* N*-divisor of* $F(x)$*, and hence, the separability of* $G(x)$ *is immediate from that of* $F(x)$*.*

Proof. Put $h(x) = N^{-r}g(x)$ where $r = \mathrm{ord}_N\big(g\big)$ and $s = \nu(h)$. (Recall that $\mathrm{ord}_N(g)$ is the largest integer r such that $g(x) \equiv_{N^r} 0$, and $\nu(h)$ is the $\langle F(x), N\rangle_{\mathbb{Z}[x]}$-adic order of $h(x)$.) Then, $s \leq m - r - 1$ and it holds that

$$\forall v \ \Big[\ F(v) \equiv 0 \pmod{N} \Rightarrow h(v) \equiv 0 \pmod{N^{s+1}}\ \Big]. \tag{17}$$

Express $h(x)$ as $h(x) = q_s(x)(F(x))^s + \cdots + N^s q_0(x)$ by using $q_s(x), \ldots, q_0(x) \in \mathbb{Z}_N[x]$. It can be assumed that $q_s(x) \not\equiv_N 0$ and each $q_i(x)$ satisfies $\deg(q_i) < \deg(F)$ without loss of generality. If the latter case fails, we reconstruct $h(x)$ by replacing current $q_i(x)$ by $\mathrm{rem}(q_i, F)$. Next suppose the former case fails to hold, i.e., $q_s(x) \equiv_N 0$. Removing such zero terms, we would have $h(x) = N^a q_{s-a}(x)(F(x))^{s-a} + \cdots$ and $q_{s-a}(x) \not\equiv_N 0$ for $a > 0$. Then, dividing $h(x)$ by N^a, we obtain t $(= s - a)$-canonical polynomial $\overline{h}(x)$ satisfying

$$\forall v \ \Big[\ F(v) \equiv 0 \pmod{N} \Rightarrow \overline{h}(v) \equiv 0 \pmod{N^{t+1}}\ \Big].$$

Thus, by renaming $\overline{h}(x)$ and t to $h(x)$ and s, we again have (17) with $q_s(x) \not\equiv_N 0$.

Now consider the s-th derivative of $h(x)$. Note that we have (15) for $F(x)$ and this $h(x) = q_s(x)(F(x))^s + \cdots$. Define monic polynomials $q(x)$ and $\bar{F}(x)$ from $q_s(x)$ and $F^{(1)}(x)$ by dividing their leading coefficients, and define $Q(x) = q(x) \cdot (\bar{F}(x))^s$. It clearly satisfies

$$\forall v \ \Big[\ F(v) \equiv 0 \pmod{N} \Rightarrow Q(v) \equiv 0 \pmod{N}\ \Big]. \tag{18}$$

Noting that for $s = 0$, it can directly take $Q(x)$ by $q(x)$ $(= q_0(x)/\mathrm{lc}(q_0).)$

Consider two cases: $F(x) \not|_N Q(x)$ and $F(x)|_N Q(x)$. For these cases, we show our claim.

The case $F(x)|_N Q(x)$: In this case, a non-trivial factor of N is always obtained efficiently from $F(x)$ and $\bar{F}(x)$.

Note first, that $s \geq 1$. Because otherwise, i.e., $s = 0$, we have $F(x)|_N q_0(x)$ which derives $q_0(x) \equiv_N 0$ by $\deg(q_0) < \deg(F)$, this contradicts the assumption that $q_s(x) \not\equiv_N 0$.

Then, write $Q(x) \equiv_N p(x)F(x)$ by using $p(x) \in \mathbb{M}_N[x]$. Apply the first half of Proposition 1 to $F(x)$ and $\bar{F}(x)$; if we obtain the desired factor, we have finished. Hence, suppose that we obtain a pair of polynomials $u(x), v(x) \in \mathbb{Z}_N[x]$ satisfying $1 \equiv_N u(x)F(x) + v(x)(\bar{F}(x))^s$. Multiply $q(x)$ to both sides, we have

$$\begin{aligned} q(x) &\equiv_N q(x)u(x)F(x) + v(x)Q(x) \equiv_N q(x)u(x)F(x) + v(x)p(x)F(x) \\ &\equiv_N (q(x)u(x) + v(x)p(x))F(x), \end{aligned}$$

which implies that $\deg(q) \geq \deg(F)$ since $q(x)$ and $F(x)$ are monic polynomial. This contradicts the assumption that $\deg(q) < \deg(F)$.

The case $F(x) \not|_N Q(x)$: Let $G(x)$ denote the greatest monic common N-divisor of $F(x)$ and $Q(x)$. Note that the divisor is computed by the standard Euclidean algorithm for polynomials under modulo N; from the problem-setting in the preliminary section, the polynomial division throughout the execution of the algorithm is always defined.

We show that $G(x)$ satisfies the sentence of this theorem. Since it can be written $G(x) \equiv_N u(x)F(x) + v(x)Q(x)$ by $u(x), v(x) \in \mathbb{Z}_N[x]$, (18) implies (16). Hence, it suffices to show that $\deg(G) \leq \deg(F) - 2$. It is clear that $G(x)|_N F(x)$ by definition, and write $F(x) \equiv_N w(x)G(x)$ for $w(x) \in \mathbb{M}_N[x]$. We show that $\deg(w) \geq 2$. Clearly $\deg(w) \geq 1$; thus, assume that $\deg(w) = 1$, that is, $w(x) = x - v_0$ for $v_0 \in \mathbb{Z}_N$. By (16), $F(v_0) \equiv G(v_0) \equiv 0 \pmod N$; thus, $(x - v_0)|_N G(x)$ holds by the factor theorem. Hence, we have $(x - v_0)^2|_N F(x)$, which implies that $(x - v_0)|_N F^{(1)}(x)$. This contradicts the separability of $F(x)$. Therefore, it needs that $\deg(w) \geq 2$ and $\deg(G) = \deg(F) - \deg(w) \leq \deg(F) - 2$. $\square$

Note that this theorem gives a polynomial-time algorithm that reduces a given target equation to a simpler one based on any non-canonical initial polynomial for the target polynomial. We expect that this reduction itself is impossible for various cases. Below, we show one example from the RSA cryptography.

5.5 Application for Coppersmith's Attack for RSA

We apply the above theorem for a Coppersmith's attack [9] for RSA. Consider an RSA ciphertext c that is encrypted from a plaintext p using a public key pair (e, N). Here, we use N for both the modulo of an RSA instance and that of the target equation. The situation considered in [9] is that the attacker has a public key pair, valid ciphertext, and a quantity of MSBs of the corresponding plaintext. Let C and P' denote integers respectively corresponding to the ciphertext c and the revealed part of p, and let k denote the length of the unrevealed part of p. Then the unrevealed part Q is the unique solution of $f_{\mathrm{RSA}}(x) \stackrel{\mathrm{def}}{=} (x + 2^k P')^e - C \equiv 0 \pmod N$, which we call the *RSA equation*. The attacker's task is to compute Q. Clearly, Q satisfies $0 \leq Q < 2^k$, and it can be easily shown that x is the unique solution of the RSA equation. Coppersmith showed that the equation is solved in polynomial-time w.r.t. $\log N$ when $0 < Q < N^{1/e}$, which corresponds to the situation in which the bit-length of the unknown part is smaller than $1/e$-times that of N.

From our result of Section 4, the length of the revealed part should be longer than $(1-1/e)\log_2 N$ bits in order to use the Coppersmith technique with the canonical initial polynomials. Now suppose that we have a non-canonical initial polynomial for the target equation $f_{\mathrm{RSA}}(x)$ (and an initial exponent m). From the separability of $f_{\mathrm{RSA}}(x)$ and by Theorem 2, we can compute either (i) a non-trivial factor of N, or (ii) a polynomial $G(x)$ with $\deg(G) \leq e - 2$ that has the same solutions with $f_{\mathrm{RSA}}(x)$. Clearly the attack succeeds in the former case. Consider the latter case. If $e = 3$, noting that G is a linear function, it is easy to see that the plaintext is computable from $G(x)$. In general, for any $e = \mathrm{poly}(\log N)$, if we can repeat this argument, (either a non-trivial factor of N is obtained, or) a trivial linear equation is derived to compute the plaintext. Note also that the length of the revealed part does not matter in this case. From this example, we can conclude that there is no general and efficient way to construct non-canonical initial polynomials.

6 Optimality of Small Inverse Problem Equation

We consider an application of our optimality argument to the simple bivariate equation

$$F(x,y) := -1 + x(y+M) = 0 \pmod{N} \tag{19}$$

of which our target is to find an integer solution (x,y) within the range $|x| < X$ and $|y| < Y$. The equation is called the "small inverse problem equation" and has been usually used for RSA cryptanalysis from Boneh-Durfee [4]. As the settings in the univariate case, suppose we have a composite integer N, whose factor p is unknown though we know $p \approx N^{\beta}$. We assume that the lattice construction does not use any information on factors of N. If one can use it, the range would be extended as in [1].

To argue the optimality of this equation under our framework, we follow Kunihiro's argument [22] that connects the optimality between the small inverse problem equation and May's equation. We introduce an outline of his argument.

He considered the family of equations $F_c(x,y) := -c + x(y+M) \equiv 0 \pmod{N}$ for integer constants c, and proved the if we have a good lattice construction for $c = 0$, then we can construct a lattice for May's equation $x + A \equiv 0 \pmod{N}$ that exceeds the original May's bound N^{β^2}. Thus, by contradiction, assuming that May's bound is optimal, we can prove the limitation of the equation for $c = 0$. On the other hand, what we actually need is the relation between the bivariate equation with $c = 1$ and May's bound. Although this was not theoretically proven, we probably expect there is a strong connection between the cases of $c = 1$ and of $c = 0$. In this section, we revisit the previous discussion to fit our framework, and give the explicit relationship between the equation (19) and May's equation.

We define the basic notions inspired from the univariate case. The standard expression of bivariate polynomial is $h(x,y) = \sum_{i,j} a_{i,j} x^i y^j$. The word *xy term* means that a monomial $a_{i,j}x^iy^j$ satisfies $i \cdot j \geq 1$. For integers X and Y, define the norm of polynomial $||h(x,y)||_{XY}^2 = \sum_{i,j} a_{i,j}^2 x^{2i} y^{2j}$.

6.1 Canonical Polynomial and its Representation in Bivariate Situation

We start our argument to define the canonical polynomial in bivariate situation. As the univariate case in Section 4, we say $h(x)$ is a canonical polynomial if it has the form

$$h(x,y) = \sum_{i=0}^{m} r_i(x,y) N^{m-i} (F(x,y))^i \tag{20}$$

where $r_i(x,y) \in \mathbb{Z}[x,y]$.

Because we can replace the xy terms in $r_i(x,y)$ to $F(x,y) - xM + 1$, we can assume that $r_i(x,y)$ consist of terms of the form $a_i x^i$ and $b_j y^j$. We fix this expression.

Lemma 5. *For any $h(x,y)$, the canonical representation (20) with respect to $F(x,y)$ and N is well-defined.*

Proof. Suppose there exists different expressions for the same $h(x,y)$ as

$$\sum_{i=0}^{m} r_i(x,y) N^{m-i} (F(x,y))^i = \sum_{i=0}^{m'} r'_i(x,y) N^{m-i} (F(x,y))^i \tag{21}$$

and both $r_i(x,y)$ and $r'_i(x,y)$ are not zero polynomials. We will prove if both sides are the same, $r_i(x,y)$ and $r'_i(x,y)$ are equivalent for all i by showing its contrapositive.

If $m > m'$, then both polynomials are different because the left-hand side have the term of $(xy)^m$ whereas the other polynomial does not have it. For the situation $m = m'$, we can assume $r_m(x,y) \neq r'_m(x,y)$ without loss of generality because if they are the same polynomial, we subtract $r_i(x,y)N^{m-i}(F(x,y))^i$ from the both side.

For the standard expression of $h(x,y)$, consider the terms divisible by $(xy)^m$ and define the sum

$$[h(x,y)]_m := \sum_{i \geq m \text{ and } j \geq m} a_{i,j} x^i y^j / (xy)^m.$$

The terms in the right-hand side is from the polynomial $r_m(x,y)(x(y+M))^m$ because for $i < m$, monomials in $r_i(x,y)N^{m-i}(F(x,y))^i$ are not divisible by $(xy)^m$, and for $r_m(x,y)(F(x,y))^m = r_m(x,y)\sum_{j=0}^{m}(-1)^{m-j}\binom{m}{j}(x(y+M))^j$, the monomials divisible by $(xy)^m$ must be contained in the term of $j = m$. The same argument holds for $r'_m(x,y)$. Thus, (21) implies that

$$[h(x,y)]_m = [r_m(x,y)(x(y+M))^m]_m = [r'_m(x,y)(x(y+M))^m]_m. \tag{22}$$

By the degree comparison in (22), it must be $r_m(x,y)$ and $r'_m(x,y)$ are univariate polynomials having the same variable and degree.

Suppose r_m and r'_m are the polynomial of x and let $d = \deg(r_i)$. Then, by comparison of coefficients c of $x^{m+d}y^m$ in (22), the leading coefficients of both polynomials must be the same. Substituting $cx^{m+d}y^m$ from both polynomial and repeating this argument, it must be $r_m = r'_m$. If the polynomials are of y, the same argument holds. Therefore, (22) implies $r_m = r'_m$, and the representation is unique. □

6.2 Proof of the Optimality

Fix integers $X = p = N^\delta$ and $Y = N^\alpha$; since we assumed p is an unknown factor of $N = pq$, this X is also unknown. Our main result in this section is the following theorem.

Theorem 4. *Suppose we have a canonical polynomial $h(x,y)$ with $||h||_{XY} < N^m$ for range parameter X and Y. Then, we can easily compute a canonical polynomial $\tilde{h}(y)$ for May's equation*

$$G(y) = y + N \equiv 0 \pmod{q}, \tag{23}$$

satisfying $||\tilde{h}(y)||_Y < q^m$.

Applying Theorem 2 to this $G(y)$, using that $q = N/p = N^{1-\delta}$, it must be the solvable range Y satisfies

$$Y \leq N^\alpha = Y \leq 16N^{(1-\delta)^2} = N^{(1-\delta)^2+\varepsilon}.$$

Using $Y = N^\alpha$, we have $\alpha < (1-\delta)^2 + \varepsilon$ and it is equivalent to

$$\delta \leq 1 - \sqrt{\alpha - \varepsilon}, \tag{24}$$

where $\varepsilon = \log_N(16)$, which is negligible for large numbers like an RSA moduli. In particular, considering $\alpha = 0.5$ as Boneh-Dufree's setting for RSA cryptanalysis, it must be $\delta < 1 - 1/\sqrt{2} \approx 0.292$.

Proof of the Theorem. We rewrite the canonical representation $h(x,y)$ by

$$h(x,y) = \sum_{i=0}^{m} \left(\sum_{j=0}^{i} a_{i,j} x^{i-j} F(x,y)^j N^{m-j} + \sum_{j=i+1}^{i_j} a_{i,j} y^{j-i} F(x,y)^i N^{m-i} \right).$$

and define its rounding by replacing $F(x,y)$ to $x(y+M)$, i.e.,

$$\begin{aligned} h^*(x,y) &= \sum_{i=0}^{m} \left(\sum_{j=0}^{i} a_{i,j} x^{i-j} (x(y+M))^j N^{m-j} + \sum_{j=i+1}^{i_j} a_{i,j} y^{j-i} (x(y+M))^i N^{m-i} \right) \\ &= \sum_{i=0}^{m} \left(\sum_{j=0}^{i} a_{i,j} (y+M)^j N^{m-j} + \sum_{j=i+1}^{i_j} a_{i,j} y^{j-i} (y+M)^i N^{m-i} \right) x^i. \end{aligned}$$

We further denote $h_i^*(x,y)$ be the polynomial corresponding to the index i. Thus, we have $h^*(x,y) = \sum_{i=0}^{m} h_i^*(x,y)$ and $h_i^*(x,y) = k_i^*(y)x^i$ holds for every i with using some univariate polynomial $k_i^*(y)$.

As the above, define $k_i(y)$ be the polynomial such that $h(x,y) = \sum_{i=0}^{m} k_i(y)x^i$. Then by construction, we have $k_i^*(y) = k_i(y)$ and

$$h(x,y) = k_m^*(y)x^m + \sum_{i=0}^{m-1} k_i(y)x^i$$

Hence,

$$||h||_{XY} > ||h_m^*||_{XY} \tag{25}$$

holds.

Next, for this $h^m(x,y)$, consider the univariate polynomial

$$h_m^*(p,y) = \left(\sum_{j=0}^{m} a_{m,j} (y+M)^j N^{m-j} + \sum_{j=m+1}^{j_m} a_{m,j} y^{j-m} (y+M)^m \right) p^m.$$

Then, defining $h\ (y) := p^{-m} h_m^*(p,y) \in \mathbb{Z}$, it is canonical for the equation (23) and the polynomial norm can be bounded as

$$h\ (y) = p^{-m} ||h_m^*(p,y)||_Y \leq p^{-m} ||h_m^*(x,y)||_{XY} < p^{-m} ||h(x,y)||_{XY} < q^m$$

where the first inequality is from the fact for $h^*(x,y) = \sum_{i,j} a_{i,j} x^i y^j$,

$$|| \sum_{i,j} a_{i,j} p^i y^j ||_Y^2 \leq \sum_{i,j} ||a_{i,j} p^i y^j||_Y^2 = \sum_{i,j} a_{i,j}^2 p^{2i} Y^{2j} = ||h^*(x,y)||_{XY}^2.$$

□

7 Concluding Remarks

We investigated the optimality of lattice constructions used in the Coppersmith technique for finding small roots of a univariate modular equation (1) and a small inverse problem equation (19).

For this purpose, we provide a framework of the technique and a sufficient and a failure conditions in which the technique works. Then we find that Coppersmith [9], May [26] and Boneh-Durfee [4] have given the best lattice construction under our framework and the reasonable assumption.

We also showed that a non-standard lattice construction would lead to quite a strong method for solving the original problem or factorizing a large number.

Following our results, we introduce two unsolved problem for future works to improve our mathematical techniques.

(i) Rigid proof of Assumption 1: The Rogers' theorem says for random lattices its shortest vector length is larger than $c \cdot GH(L)$ for $c < 1$ with overwhelming probability. In this paper, we assumed the Coppersmith type lattices, which is a small subset of random lattices, have the same property. Considering computer experiments in previous, works it is very likely correct, however, we have never obtain the theoretical proof.

(ii) Find an equation that the possibility result and limitation result are not matching: We think in such cases, the range of possible result exceeds our limitation because the gap of frameworks, and the problem of this type will investigate the research activity in this area. For example, using a factor of modulus can extend the solvable range as [1], which is not in our framework.

References

[1] Y. Aono, A new lattice construction for partial key exposure attack for RSA, in *Proc. of PKC 2009*, LNCS, vol. 5443, pp. 34-53, 2009. The full-version is available at `http://www.is.titech.ac.jp/research/research-report/C/C-257.pdf`.

[2] Y. Aono, Minkowski sum based lattice construction for multivariate simultaneous Coppersmith's technique and applications to RSA, in *Proc. of ACISP 2013*, LNCS vol. 7959, pp. 88-103, 2013.

[3] Y. Aono, M. Agrawal, T. Satoh and O. Watanabe, On the optimality of lattices for the Coppersmith technique, in *Proc. of ACISP 2012*, LNCS, vol. 7372, pp. 376-389, 2012.

[4] D. Boneh and G. Durfee, Cryptanalysis of RSA with private Key d Less Than $N^{0.292}$, in *Proc. of Eurocrypt 1999*, LNCS, vol. 1592, pp. 389-401, 1999.

[5] J. Blömer and A. May, New partial key exposure attacks on RSA, in *Proc. of CRTPTO 2003*, LNCS, vol. 2729, pp. 27-43, 2003.

[6] J. Blömer and A. May, A tool kit for finding small roots of bivariate polynomials over the integers, in *Proc. of Eurocrypt 2005*, LNCS, vol. 3494, pp. 351-367, 2005.

[7] D. Boneh, Finding smooth integers in short intervals using CRT decoding, in *Proc. of STOC 2000*, pp. 265-272.

[8] G. Castagnos, A. Joux, F. Laguillaumie, and P. Q. Nguyen, Factoring pq^2 with quadratic forms: Nice cryptanalyses, *Proc. of Asiacrypt 2009*, LNCS, vol. 5912, pp. 469-486, 2009.

[9] D. Coppersmith, Finding a small root of a univariate modular equation, *Proc. of Eurocrypt 1996*, LNCS, vol. 1070, pp. 155-165, 1996.

[10] D. Coppersmith, Finding small solutions to small degree polynomials, *Proc. of CaLC 2001*, LNCS, vol. 2146, pp. 20-31, 2001.

[11] G.E. Collins and A.G. Akritas, Polynomial real root isolation using Descartes' rule of signs, *Proc. of the ACM Symp. on Symbolic and Algebraic Computation 1976*, pp. 272-275.

[12] H. Cohn and N. Heninger, Ideal forms of Coppersmith's theorem and Guruswami-Sudan list decoding, *in Proc. of ICS 2011*, pp. 298-308.

[13] J.-S. Coron, A. Joux, I. Kizhvatov, D. Naccache, and P. Paillier, Fault attacks on RSA signatures with partially unknown messages, *Proc. of CHES 2009*, LNCS, vol. 5747, pp. 444-456, 2009.

[14] 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*, LNCS, vol. 3494, pp. 371-386, 2005.

[15] N. Gama and P. Q. Nguyen, Predicting lattice reduction, in *Proceedings of Eurocrypt 2008*, Lecture Notes in Computer Science, vol. 4965, pp. 31-51, 2008.

[16] P. Gianni and B. Trager, Square-free algorithms in positive characteristic, in *Applicable Algebra in Engineering, Communication and Computing*, Vol. 7, No. 1, pp.1-14, 1996.

[17] J. Håstad, Solving simultaneous modular equations of low degree, SIAM Journal on Computing, Vol. 17, No 2, pp. 336-341, 1988.

[18] N. Howgrave-Graham, Finding small roots of univariate modular equations revisited, *Proc. of Cryptography and Coding*, LNCS, vol. 1355, pp. 131-142, 1997.

[19] N. Howgrave-Graham, Approximate integer common divisors, *Proc. of CaLC 2001*, LNCS, vol. 2146, pp. 51-66, 2001.

[20] E. Jochemsz and A. May, A strategy for finding roots of multivariate polynomials with new applications in attacking RSA variants, in *Proc. of Asiacrypt 2006*, LNCS, vol. 4284, pp. 267-282, 2006.

[21] N. Kunihiro, Solving generalized small inverse problems, in *Proc. of ACISP 2010*, LNCS, vol. 6168, pp. 248-263, 2010.

[22] N. Kunihiro, On optimal bounds of small inverse problems and approximate GCD problems with higher degree, LNCS, vol. 7483, pp. 55-69, 2012.

[23] S. V. Konyagin and T. Steger, On polynomial congruences, *Mathematical Notes*, Vol. 55, No. 6, pp. 596-600, 1994.

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

[25] J. S. Milne, Étale cohomology, Princeton Math. Series 33, Princeton Univ. Press, 1980.

[26] A. May, New RSA vulnerabilities using lattice reduction methods, Ph.D thesis, University of Paderborn, 2003.

[27] P. Q. Nguyen and D. Stehlé, LLL on the average, in *Proc. of ANTS 2006*, LNCS, vol. 4076, pp. 238-256, 2006.

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

[29] T. Okamoto and A. Shiraishi, A fast signature scheme based on quadratic inequalities, in *Proc. of the Symposium on Security and Privacy*, IEEE, pp. 123-132, 1985.

[30] C. A. Rogers, "The number of lattice points in a set, Proc. of London Math. Soc., vol. 3, No. 6, pp. 305-320, 1956.

[31] A. P. Prudnikov, Y. A. Brychkov and O. I. Marichev, Integrals and Series, vol. 1, Elementary Functions, Gordon and Breach, New York (1986)

[32] G. Pólya and G. Szegő, Problems and Theorems in Analysis, Vol. II, Springer, 1976.

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

[34] V. Shoup, OAEP Reconsidered, in *Journal of Cryptology*, vol. 15, No. 4, pp. 223-249, 2002. online version is available at `http://shoup.net/papers/oaep.pdf`.

[35] C. P. Schnorr and M. Euchner, "Lattice basis reduction: improved practical algorithms and solving subset sum problems", Math. Program., vol. 66, no. 1-3, pp. 181-199, 1994.

[36] S. Maitra and S. Sarkar, A new class of weak encryption exponents in RSA, in *Proc. of Indocrypt 2008*, LNCS, vol. 5365, pp. 337-349, 2008.

[37] B. Vallee, M.Girault, and P. Toffin, How to Break Okamoto's Cryptosystems by Reducing Lattices Bases, in *Proc. of Eurocrypt 1988*, LNCS, vol. 330, pp. 281-291, 1988.