mathematics

MDPI

Article

AIOL: An Improved Orthogonal Lattice Algorithm for the General Approximate Common Divisor Problem

Yinxia Ran [1,2,3], Yun Pan [1], Licheng Wang [4,*] and Zhenfu Cao [2,5,*]

[1] School of Computer and Cyberspace Secrity, Communication University of China (CUC), 1 Dingfuzhuang East Street, Beijing 100024, China
[2] Research Center for Basic Theories of Intelligent Computing, Research Institute of Basic Theories, Zhejiang Lab, Hangzhou 311121, China
[3] School of Mathematics and Information Technology, Longnan Teachers College (LNTC), 34 Longnan Road, Longnan 742500, China
[4] School of Cyberspace Science and Technology, Beijing Insititute of Technology (BIT), 5 Zhongguancun South Street, Beijing 100081, China
[5] Shanghai Key Laboratory of Trustworthy Computing, East China Normal University (ECNU), 3663 North Zhongshan Road, Shanghai 200062, China
* Correspondence: lcwang@bit.edu.cn (L.W.); zfcao@sei.ecnu.edu.cn (Z.C.)

Abstract: The security of several fully homomorphic encryption (FHE) schemes depends on the intractability assumption of the approximate common divisor (ACD) problem over integers. Subsequent efforts to solve the ACD problem as well as its variants were also developed during the past decade. In this paper, an improved orthogonal lattice (OL)-based algorithm, AIOL, is proposed to solve the general approximate common divisor (GACD) problem. The conditions for ensuring the feasibility of AIOL are also presented. Compared to the Ding–Tao OL algorithm, the well-known LLL reduction method is used only once in AIOL, and when the error vector **r** is recovered in AIOL, the possible difference between the restored and the true value of p is given. Experimental comparisons between the Ding-Tao algorithm and ours are also provided to validate our improvements.

Keywords: general approximate common divisors; fully homomorphic encryption; lattice attack; orthogonal lattice

MSC: 68W40

Citation: Ran, Y.; Pan, Y.; Wang, L.; Cao, Z. AIOL: An Improved Orthogonal Lattice Algorithm for the General Approximate Common Divisor Problem. *Mathematics* **2023**, *11*, 4989. https://doi.org/10.3390/math11244989

Academic Editor: Jonathan Blackledge

Received: 8 October 2023
Revised: 5 December 2023
Accepted: 13 December 2023
Published: 18 December 2023

1. Introduction

Background. The approximate common divisor (ACD) problem was first studied by Howgrave-Graham [1]. Further interest in this problem was inspired by the proposal of fully homomorphic encryption (FHE) by Van Dijk et al. [2], as well as the cryptographic constructions proposed subsequently [3–5]. The security of these cryptosystems depends on the hardness assumption of the ACD problem and its variants.

The ACD problem is usually formulated in two ways: the problem of the general approximate common divisor (GACD) and the problem of the partial approximate common divisor (PACD). Both of these formulations polynomially take as inputs many samples $x_i = pq_i + r_i$ with sufficiently small but non-zero r_i and aim to work out the *hidden common divisor* p, while the latter is given an additional *exact* sample $x_0 = pq_0$ (i.e., $r_0 = 0$). Intuitively, the PACD problem is easier than GACD, considering that one can work out p directly if he/she knows the factorization of the additional sample x_0, whereas the capability of integer factorization has no direct impact on the GACD problem. However, Van Dijk et al. pointed out that at present, there is no PACD algorithm that does not work for GACD [2]. And the usefulness of PACD has been demonstrated by a much more efficient construction of the FHE scheme [5], the security of which has been proven to rely on PACD rather than on GACD. The original papers [1,2] presented a few possible lattice

attacks on the GACD problem, including the orthogonal lattices (OL) method, simultaneous diophantine approximation (SDA) method, and multivariate polynomial equations (MP) method. During the past decade, several related improvements and cryptanalytic works were proposed [6–14]. Detailed comparisons of these methods are summarized in Table 1. Further explanations on these methods are given below.

- SDA methods. The basic idea of SDA methods is to note that if r_i is small, then the fraction q_i/q_0 is an instance of a simultaneous diophantine approximation to x_i/x_0 $(i = 1, 2, \cdots, t)$. Once q_0 is determined, r_0 can be computed from $r_0 \equiv x_0 \pmod{q_0}$. Hence, $p = (x_0 - r_0)/q_0$.
- OL methods. The common objective of OL methods is to find some short vectors that are orthogonal to certain *unknown referred* vector(s) $\mathbf{v_{ref}}$. The difference lies in the setting on $\mathbf{v_{ref}}$, as well as the methods for finding such short objective vectors. At EuroCrypt 2010, Van Dijk et al. [2] described two OL methods. The first is to set $\mathbf{v_{ref}} = (1, -r_1/R, \cdots, -r_t/R)$ with $R = 2^\rho$, and the second is to set $\mathbf{v_{ref}} = \mathbf{q} = (q_1, \cdots, q_t)$ and $\mathbf{v'_{ref}} = \mathbf{r} = (r_1, \cdots, r_t)$. About 4 years later, the second method was further improved by Ding and Tao [7] in the sense that they used only one referred vector $\mathbf{v_{ref}} = \mathbf{q}$. In this sequel, we mainly focus on this improved OL method. According to the shape of the basis of the working lattice $\mathcal{L}(\alpha)$, this kind of OL method can be further divided into two sub-categories: OL-$\wedge$, with a lower triangular matrix as the working lattice basis [8,9], and OL-$\vee$, with an upper triangular matrix as the working lattice basis [7,9,10].
- MP methods. The origin of MP methods can be traced back to Howgrave's work at CaLC 2001 [1], where the PACD problem was reduced to the problem of finding small roots of multivariate polynomial equations. This idea was further extended to suit the need to solve GACD [2,11,15,16]. The core idea of MP methods is to construct a t-element polynomial $Q(X_1, X_2, \cdots, X_t)$ of degree n in the variables $X_i (i = 1, \cdots, t)$ such that $Q(r_1, \cdots, r_t) \equiv \pmod{p^k}$ for a properly chosen n and k. Then, if $|Q(r_1, \cdots, r_t)| < p^k$, the equation $Q(r_1, \cdots, r_t) = 0$ holds over the integers. Eventually, to obtain $r_1, \cdots, r_t$, at least t algebraic independent target vectors were needed in order to be able to perform elimination to reduce $Q(r_1, \cdots, r_t) = 0$ to a univariate polynomial equation. After then, one can compute $p = \gcd(x_0, x_1 - r_1)$ easily. If we have very limited PACD samples, the MP method has advantages in computational cost. However, if sufficiently many PACD samples are available, the process of searching the required algebraic independent target vectors has a huge cost. In this case, Galbraith et al. [8] suggested the use of linear polynomials in the MP method, and this, in turn, is essentially equivalent to the orthogonal lattice method.

Among the above work, the OL algorithm by Ding and Tao [7] is ingenious due to its use of the well-known LLL algorithm twice to accurately recover the error vector $\mathbf{r}$. After mapping the given GACD instances into a lattice $\mathcal{L}$, the first calling of the LLL algorithm is to find suitable $t - z (z = 1, 2)$ short vectors $\mathbf{u}_i (i = 1, 2, \cdots, t - z)$ for establishing the equations

$$\mathbf{U} \cdot \mathbf{x} = \mathbf{U} \cdot \mathbf{r} \quad \text{for} \quad \mathbf{U} = [\mathbf{u}_1 | \cdots | \mathbf{u}_{t-z}],$$

where $\mathbf{u_i} = (u_{i1}, u_{i2}, \cdots, u_{it})$. Then, a new lattice $\mathcal{L}'$ is constructed using the base vectors of the solution space of the above equation, and the second calling of the LLL algorithm is to recover the error vector $\mathbf{r}$ accurately. Knowing $\mathbf{r}$, it is very easy to recover p, even for a primary school student, say, by using the extended Euclidean algorithm. According to Ding and Tao [7], it is an *amazing* thing that the first calling of the LLL algorithm over $\mathcal{L}$ should find multiple short vectors of appropriate length to construct the equations, and they claimed that a theoretical proof would be a very significant result. Another merit of the Ding–Tao method is that setting the related parameters is simple, and this makes the implementation of OL attacks against GACD-based cryptosystems very easy in practice. For example, the lower bound of the number of samples t depends only on γ, and the length of the short vector $\mathbf{v}$ depends only on t and γ.

Table 1. Comparisons of methods for GACD.

OL Attack \ Comparative Results		SDA [8,12,17]	MP [1,11,16,18]
OL-∧ [9]	$\alpha = 2^{\rho}$, [OL-∧]	SDA and OL-∧ with $\alpha = 2^{\rho}$ attack have similar performances.	MP is not better than OL-∧ with $\alpha = 2^{\rho}$ attack for practical cryptanalysis; both OL attacks have advantages over the MP approach.
OL-∨ [9]	α is in general, [OL-∧, OL-∨].	When $(\gamma - \rho)$ is very small, OL-∧ with a rounding technique is the fastest.	The cases with α in general and rounding techniques are more suitable for cases where ρ is no longer extremely smaller than η.

Motivation and Contributions. the italics should be retained With further experiments on the Ding–Tao algorithm, we find that the actual effect of the algorithm is *better* than they claimed. In particular, we realize that the conditions $\rho < \eta/2$ and $t \geq (4\gamma)^{1/3}$ could be relaxed and merged, and the second calling of the LLL algorithm could also be saved. Moreover, we find that even for failure executions of the Ding–Tao algorithm, there is a high probability that the recovered p differs from the actual value by only 1 or very small numbers. Therefore, our motivation in this work is to propose an improved OL algorithm to reduce both space and time costs for solving the GACD problem. Our main contributions are summarized as follows:

- First, we modify the range of parameters N, t and the length of the short target vector $\mathbf{v}$ in the Ding–Tao algorithm so that we need to build lattice and call the LLL algorithm only once. The success rate for recovering p reaches 100% under the merged condition

$$t \geq \max\left\{4, \frac{5}{3}\left\lceil \eta - \rho - \sqrt{(\eta-\rho)^2 - 1.2(\gamma+\rho)} \right\rceil\right\}.$$

  Note that this inequation also implies $\rho \leq \eta + 0.6 - \sqrt{(\eta+0.6)^2 + 1.2\gamma}$ no matter whether $\rho < \eta/2$ holds.
- Second, based on the above modification, we give a proof on why, in our algorithm AIOL, the method of only calling the LLL algorithm once gives us the desired short vectors. This can be viewed as a theoretical answer to Ding and Tao's *amazing* question.
- Third, we give the possible differences between the recovered p and the actual hidden common divisor when the error vector $\mathbf{r}$ is recovered. Knowing these differences is, in turn, helpful for recovering p and thus expanding the scope of OL attacks.

Roadmap. The remaining contents are organized as follows. In Section 2, the formal definitions of the problems of GACD and PACD are given, and the lattice concepts and the LLL algorithm are introduced briefly. In Section 3, the orthogonal lattice-based approaches, including our improvements, for GACD are explored and developed in detail. Experiments and comparisons, as well as related discussions, are presented in Section 4. Finally, concluding remarks are given in Section 5.

2. Preliminaries

Throughout this paper, we make the following agreement on notations: capital boldface letters denote matrices, e.g., $\mathbf{A}$, while lowercase bold letters denote vectors e.g., $\mathbf{a}$; let $(\cdot,\cdot)$ and $\|\cdot\|$ be the inner product and the l_2 Euclidean length, respectively, and $\mathbf{A}^{\mathbf{T}}$ denote the transpose of matrix $\mathbf{A}$; and the logarithmic notation log always takes 2 as the base, while $\lceil r \rceil$ denotes the smallest integer not less than r.

Definition 1 (ACD Distribution). *Given $\gamma, \eta, \rho \in \mathbb{N}$, let p be an η-bit odd integer. Then, the ACD distribution, $D_{\gamma,\rho}(p)$, is an efficiently sampleable distribution defined as follows:*

$$D_{\gamma,\rho}(p) = \{pq + r | q \leftarrow \mathbb{Z} \cap (0, 2^{\gamma}/p), r \leftarrow \mathbb{Z} \cap (-2^{\rho}, 2^{\rho})\}. \quad (1)$$

Definition 2 (GACD Problem). *Given access to an ACD distribution $D_{\gamma,\rho}(p)$ as an oracle, the objective of the general approximate common divisor (GACD) problem is to find p.*

Definition 3 (PACD Problem). *Given access to an ACD distribution $D_{\gamma,\rho}(p)$ as an oracle, with the restriction that the first output of $D_{\gamma,\rho}(p)$ is $x_0 = pq_0$ for some $q_0 \leftarrow \mathbb{Z} \cap (0, 2^{\gamma}/p)$, the objective of the partial approximate common divisor (PACD) problem is to find p.*

Remark 1. *Apparently, a PACD instance is a GACD by coincidence only with a probability that is negligible with respect to ρ.*

Definition 4 ($\delta-$LLL reduction basis). *Given a lattice basis $\mathbf{B} = (\mathbf{b_1}, \cdots, \mathbf{b_n})$, the corresponding Gram–Schmidt basis $\mathbf{B}^* = (\mathbf{b_1^*}, \cdots, \mathbf{b_n^*})$, $\mathbf{B}$ is a reduced basis if and only if the following two conditions are satisfied:*

(1) The size condition: $\mu_{i,j} = \dfrac{(\mathbf{b_i}, \mathbf{b_j^*})}{\|\mathbf{b_j^*}\|^2} \leq 1/2$, for all $1 \leq j < i \leq n$;

(2) The Lovász condition: $\|\mathbf{b_i^*}\|^2 \geq (\delta - \mu_{i,i-1}^2)\|\mathbf{b_{i-1}^*}\|^2$, for all $1 < i \leq n$, where $1/4 < \delta < 1$.

Definition 5 (Geometric Series Assumption [19]). *Given the Gram–Schmidt basis $(\mathbf{b_1^*}, \cdots, \mathbf{b_n^*})$,*

$$\frac{\|\mathbf{b_i^*}\|}{\|\mathbf{b_1}\|} = \theta^{i-1} \quad (2)$$

for $i = 1, 2, \cdots, n$, where $3/4 \leq \theta < 1$ is called GSA constant.

The geometric series assumption (GSA) means the length of the Gram–Schmidt basis $\|\mathbf{b_i^*}\|$ with LLL reduction decays geometrically with the quotient θ and indicates

$$\|\mathbf{b_i^*}\| \leq \|\mathbf{b_1}\| (i = 1, 2, \cdots, n). \quad (3)$$

Theorem 1 ([20]). *Given an LLL reduction lattice basis $\mathbf{B} = (\mathbf{b_1}, \cdots, \mathbf{b_n})$, $(\mathbf{b_1^*}, \cdots, \mathbf{b_n^*})$ is the corresponding Gram–Schmidt basis. The following results hold:*

(1) $\|\mathbf{b_1}\| \leq \alpha^{\frac{n-1}{4}} |\det(\mathbf{B})^{\frac{1}{n}}|$;

(2) $\|\mathbf{b_j^*}\| \leq \alpha^{\frac{(i-j)}{2}} \|\mathbf{b_i^*}\|$, for $1 \leq j < i \leq n$;

(3) $\|\mathbf{b_j}\| \leq \alpha^{\frac{(i-1)}{2}} \|\mathbf{b_i^*}\|$, for $1 \leq j < i \leq n$;

where $\alpha = \dfrac{1}{\delta - \frac{1}{4}}$, δ is the parameter in the Definition 4.

Theorem 2 ([21]). *The LLL basis reduction algorithm with the factor $\delta = \frac{3}{4}$ computes an LLL-reduced basis in polynomial time in the maximal bit-length of the coefficients of the input basis, the lattice rank n, and the space dimension m. Specifically, if $b_1, \cdots, b_n$ is an input lattice basis, $M = max\{\|b_1\|, \cdots, \|b_t\|\}$, then LLL runs in*

$$O\left(n^5 m \cdot (log_{\frac{4}{3}} M)^3\right)$$

bit operations under school multiplication.

3. Orthogonal Lattice (OL)-Based Approach

3.1. The Basic Idea of OL Algorithms

Nguyen and Stern [22] have demonstrated the usefulness of the orthogonal lattice in cryptanalysis, and this has been used in several ways to attack the ACD problem. The idea is to find the $\mathbf{u} = (u_1, u_2, \cdots, u_t) \in \mathcal{L}^{\perp}(\mathbf{q}, \mathbf{r})$ that is orthogonal to both $\mathbf{q} = (q_1, q_2, \cdots, q_t)$ and $\mathbf{r} = (r_1, r_2, \cdots, r_t)$. Since $x_i = pq_i + r_i$, $\mathbf{x} = (x_1, x_2, \cdots, x_t)$ is orthogonal to $\mathbf{u}$, the task is to find $t-1$ linearly independent vectors $\mathbf{u}$ shorter than any vector in $\mathcal{L}^{\perp}(\mathbf{x})$ to recover $\mathbf{q}$, $\mathbf{r}$ and therefore p.

Based on the idea of Nguyen and Stern, the current idea is to find $t - z(z = 1, 2)$ linearly independent vectors $\mathbf{u}$ that are only orthogonal to $\mathbf{q}$. The core steps of the current OL algorithm include the following two steps.

First, find $t - z(z = 1, 2)$ linearly independent vectors $\mathbf{u}$ orthogonal to $\mathbf{q}$, that is,

$$\sum_{i=1}^{t} u_i \cdot q_i = 0. \tag{4}$$

Then, establish and solve the indefinite equation $\mathbf{U} \cdot \mathbf{x} = \mathbf{U} \cdot \mathbf{r}$ for $\mathbf{U} = [\mathbf{u}_1 | \cdots | \mathbf{u}_{t-z}]$.

Second, find small positive-integer solutions to the above equations. At present, the common way to find the small solutions is to construct a lattice $\mathcal{L}'$ with a basis matrix

$$\mathbf{D} = \begin{pmatrix} \mathbf{d_0} \\ \mathbf{d_1} \\ \vdots \\ \mathbf{d_z} \end{pmatrix} \tag{5}$$

and then employ the LLL algorithm to reduce the basis matrix $\mathbf{D}$ with the hope that the first output is the vector $\mathbf{r}$. However, at present, only experimental conditions can meet this expectation, and there is still a lack of theory.

Let the general solution formula of the equations be

$$\mathbf{d} = \mathbf{d_0} + t_1\mathbf{d_1} + \cdots + t_z\mathbf{d_z} \tag{6}$$

where $\mathbf{d_0}$ is a special solution, $t_1, \cdots, t_z$ are integers, and $\mathbf{d_1}, \cdots, \mathbf{d_z}$ is a basis of the integer solution space for the corresponding homogeneous linear equations.

Let $\mathbf{d}' \in \mathcal{L}'$. Then,

$$\mathbf{d}' = k_0\mathbf{d_0} + k_1\mathbf{d_1} + \cdots + k_z\mathbf{d_z} \tag{7}$$

where $k_0, k_1, \cdots, k_z$ are integers. Obviously, when $k_0 = 1$, (7) = (6). Reduce the lattice $\mathbf{D}$ to $\mathbf{D}'$:

$$\mathbf{D}' = \begin{pmatrix} \mathbf{d}'_0 \\ \mathbf{d}'_1 \\ \vdots \\ \mathbf{d}'_z \end{pmatrix}. \tag{8}$$

To facilitate finding $\mathbf{r}$, consider the explicit vectors $\mathbf{d}'_0, \mathbf{d}'_1, \cdots, \mathbf{d}'_z$. It is easy to deduce that only one of them is the solution to the equations.

Let $\mathbf{d}'_i$ be the solution to the equations, and if $\mathbf{d}'_i = \mathbf{d}'_0$, then $\mathbf{d}'_0$ is probably equal to $\mathbf{r}$. With this in mind, Ding and Tao [7] found the conditions for which the algorithm can work well (theoretically not proved):

$$\rho < \frac{\eta}{2} \quad \text{and} \quad t \geq (4\gamma)^{1/3}. \tag{9}$$

In addition, if $\mathbf{d}'_i \neq \mathbf{d_0}$, we find an interesting occurrence, which is that the recovery value p' is only 1 or a very small number different from the true value p in many cases of

our experiment. And our experiments lead to the following general conclusions between p and p':

Let $\mathbf{d}'_{\mathbf{i}} = (u_{i1}, u_{i2}, \cdots, u_{it}) \neq \mathbf{d}'_{\mathbf{0}}, d_{ru} = \gcd(r_1 - u_{i1}, r_2 - u_{i2}, \cdots, r_t - u_{it})$. Then,

$$p' - p = d_{ru}, \tag{10}$$

where p' is the recovered value of p. Therefore, if $\mathbf{d}'_{\mathbf{i}} \neq \mathbf{d_0}$, using vector $\mathbf{d}'_{\mathbf{i}}$, p' can be restored. And since d_{ru} is bounded, p can be restored by p'.

In summary, one of the outputs $\mathbf{d_1}, \cdots, \mathbf{d_z}$ generated by the LLL algorithm can be used to recover $\mathbf{r}$ under the appropriate conditions.

3.2. Our Proposal

In this part, an improved OL algorithm (Algorithm 1) , AIOL, is described in detail.

Algorithm 1 (AIOL): An improved OL algorithm for GACD.

Input: The GACD parameters $\gamma, \eta, \rho \in \mathbb{N}$, and t ACD samples $\{x_1, \cdots, x_t\} \xleftarrow{\$} D_{\gamma,\rho}(p)$, with t satisfying

$$t \geq \max\left\{4, \left\lceil \frac{5}{3}\left(\eta - \rho - \sqrt{(\eta-\rho)^2 - 1.2(\gamma+\rho)}\right)\right\rceil\right\}. \tag{11}$$

Output: The approximate greatest common divisor p.

1. Randomly choose $N \in (2^{\gamma+\eta-1}, 2^{\gamma+\eta})$ and construct a lattice $\mathcal{L}$ with the basis

$$\mathbf{B} = \begin{pmatrix} 1 & & & x_1 \\ & 1 & & x_2 \\ & & \ddots & \vdots \\ & & 1 & x_t \\ & & & N \end{pmatrix}. \tag{12}$$

2. Reduce the lattice $\mathcal{L}$ by calling the LLL algorithm with $\delta = \frac{3}{4}$. Let the reduced basis be $\mathbf{V} = [\mathbf{v_1}|\cdots|\mathbf{v_{t+1}}]$, where $\mathbf{v_i} = (u_{i1}, \cdots, u_{it}, v_{i(t+1)}), (i = 1, 2, \cdots, t+1)$.

3. Collect short vectors from $\mathbf{V}$ so that $\|\mathbf{v_i}\| < 2^{\eta-\rho-2-\log\sqrt{t}}$, $(i = 1, 2, \cdots, t-z)$, where $z = 1, 2$. Then, solve the following Diophantine equations with t unknowns $r_1, \cdots, r_t$:

$$\sum_{j=1}^{t} u_{ij} \cdot r_i = \sum_{j=1}^{t} u_{ij} \cdot x_i (i = 1, \cdots, t-z). \tag{13}$$

4. Rewrite the integer solutions of (13) as follows:

$$\mathbf{d} = \mathbf{d_0} + t_1\mathbf{d_1} + \cdots + t_z\mathbf{d_z}, \tag{14}$$

where $\mathbf{d_0}$ is a special solution of the Diophantine equations, $t_1, \cdots, t_z$ are integers, and $\mathbf{d_1}, \cdots, \mathbf{d_z}$ is a basis of the integer solution space for the corresponding homogeneous linear equations.

5. Let $\mathbf{r} = \mathbf{d_0}$.

6. Compute $p = \gcd(x_1 - r_1, x_2 - r_2)$.

Through the proof in the next section, it can be seen that when the condition

$$(\eta - \rho)^2 \geq 1.2(\gamma + \rho) \tag{15}$$

or, equivalently,

$$\rho \leq \eta + 0.6 - \sqrt{(\eta + 0.6)^2 + 1.2\gamma} \tag{16}$$

holds, the AIOL algorithm will successfully recover p.

3.3. The Proof of the AIOL Algorithm

Lemma 1. *For* $\forall\, \mathbf{v} \in \mathcal{L}$, *if* $\|\mathbf{v}\| < 2^{\eta-\rho-2-log\sqrt{t}}$, *then Equation (13) holds.*

Proof. Let $\mathbf{v} = (u_1, u_2, \cdots, u_t, \sum_{i=1}^{t} u_i x_i + u_{t+1} N)$, $M = 2^{\eta-\rho-2-\log\sqrt{t}}$; then,

$$\|\mathbf{v}\| = \sqrt{\sum_{i=1}^{t} u_i^2 + \left(\sum_{i=1}^{t} u_i x_i + u_{t+1} N\right)^2} < M. \tag{17}$$

Thus,

$$|u_i| < M, \left|\sum_{i=1}^{t} u_i x_i + u_{t+1} N\right| < M (1 \le i \le t). \tag{18}$$

Since $2^{\gamma+\eta-1} \le N \le 2^{\gamma+\eta}$,

$$\begin{aligned} \left|\sum_{i=1}^{t} u_i x_i\right| &\le 2^{\gamma}\sqrt{t} \cdot \|\mathbf{u}\| \\ &\le 2^{\gamma}\sqrt{t} \cdot \|\mathbf{v}\| \\ &\le 2^{\gamma}\sqrt{t} \cdot 2^{\eta-\rho-2-\log\sqrt{t}} \\ &= 2^{\gamma+\eta-\rho-2} < N/2. \end{aligned} \tag{19}$$

Therefore, there is no modular N operation, and $u_{t+1} = 0$. Thus, $\mathbf{v} = (u_1, u_2, \cdots, u_t, \sum_{i=1}^{t} u_i x_i)$.

We also have

$$\left|\sum_{i=1}^{t} u_i r_i\right| \le 2^{\rho}\sqrt{t} \cdot \|\mathbf{v}\| \le 2^{\eta-2}. \tag{20}$$

To prove that $\left|\sum_{i=1}^{t} u_i q_i\right| = 0$ holds, suppose $\left|\sum_{i=1}^{t} u_i q_i\right| \neq 0$, so

$$p\left|\sum_{i=1}^{t} u_i q_i\right| \ge p \ge 2^{\eta-1} \tag{21}$$

$$\left|\sum_{i=1}^{t} u_i x_i\right| = \left|p\sum_{i=1}^{t} u_i q_i + \sum_{i=1}^{t} u_i r_i\right| \ge p\left|\sum_{i=1}^{t} u_i q_i\right| - \left|\sum_{i=1}^{t} u_i r_i\right| \ge 2^{\eta-1} - 2^{\eta-2} = 2^{\eta-2}, \tag{22}$$

but

$$\left|\sum_{i=1}^{t} u_i x_i + u_{t+1} N\right| = \left|\sum_{i=1}^{t} u_i x_i\right| < M = 2^{\eta-\rho-2-\log\sqrt{t}} < 2^{\eta-2}. \tag{23}$$

This is a contradiction. The Equations (4) and (13) hold. Then, Lemma 1 holds. □

Lemma 1 gives an upper bound on the length of the desired vectors in the lattice $\mathcal{L}$ that makes the the Equation (13) work.

Lemma 2. *If the number t of samples satisfies*

$$(4/3)^{(3t-2)/4} \cdot 2^{(\gamma+\eta)/(t+1)} \le 2^{\eta-\rho-2-\log\sqrt{t}}, \tag{24}$$

then LLL reduction basis vectors are valid for the construction of Equation (13).

Proof. According to Theorem 1, we consider the $(t-1)$-th LLL reduction basis vector $\mathbf{v_{t-1}}$, whose length can be estimated as below:

$$\begin{aligned}\|\mathbf{v_{t-1}}\| &\leq \alpha^{(t-1)/2}\|\mathbf{v_t^*}\|, ((3)\ of\ Theorem\ 1)\\ &= (4/3)^{(t-1)/2}\|\mathbf{v_t^*}\|,\ \ (\alpha = 4/3 \text{ as } \delta \to 1)\\ &\leq (4/3)^{(t-1)/2}\|\mathbf{v_1}\|,\ \ (GSA)\\ &\leq (4/3)^{(t-1)/2}\cdot(4/3)^{t/4}\cdot|\mathbf{B}|^{1/(t+1)}, ((1)\ of\ Theorem\ 1)\\ &\leq (4/3)^{(3t-2)/4}\cdot 2^{(\gamma+\eta)/(t+1)}, (|\mathbf{B}| \leq 2^{(\gamma+\eta)}).\end{aligned} \tag{25}$$

Therefore, by Lemma 1, Equation (13) holds when (24) is true. □

Lemma 2 estimates the length of the $(t-1)$-th vector output by the LLL algorithm and makes it fall within the range required by Lemma 1. Thus, the vectors found can be used to construct Equation (13).

Based on the above two lemmas, the following theorem can be obtained.

Theorem 3. *When GACD parameters satisfy*

$$(\eta-\rho)^2 \geq 1.2(\gamma+\rho)$$

or

$$\rho \leq \eta + 0.6 - \sqrt{(\eta+0.6)^2 + 1.2\gamma},$$

and the number of samples satisfy

$$t \geq \max\left\{4, \left\lceil \frac{5}{3}\left(\eta-\rho-\sqrt{(\eta-\rho)^2-1.2(\gamma+\rho)}\right)\right\rceil\right\},$$

then the equation

$$\sum_{j=1}^{t} u_{ij}\cdot r_i = \sum_{j=1}^{t} u_{ij}\cdot x_i (i = 1, \cdots, t-z)$$

holds.

Proof. From Condition (24), we can obtain that the length of the LLL reduction basis vectors satisfies Lemma 2. Thus, the LLL reduction basis vectors are valid for the construction of Equation (13). Combining the above two lemmas, we simplify the inequation and ignore some small terms to obtain the following bound of sample numbers t. The specific process is as follows. We take the logarithm base 2 on both sides of (24) to obtain:

$$\frac{3t-2}{4}\log\frac{4}{3} + \frac{\gamma+\eta}{t+1} \leq \eta-\rho-2-\log\sqrt{t}. \tag{26}$$

Removing some smaller items of (26), $\log\frac{4}{3} \approx 0.4$, we have

$$0.3t + \frac{\gamma+\eta}{t+1} \leq \eta-\rho, \tag{27}$$

By sorting out Formula (27), we obtain

$$0.3t^2 - (\eta-\rho-0.3)t + (\gamma+\eta) \leq 0, \tag{28}$$

Then,

$$0.3t^2 - (\eta-\rho)t + (\gamma+\eta) \leq 0, \tag{29}$$

By solving Inequality (29), when

$$\Delta = (\eta-\rho)^2 - 1.2(\gamma+\rho) \geq 0, \tag{30}$$

we can give a lower bound on t

$$t \geq \frac{5}{3}\left(\eta - \rho - \sqrt{(\eta-\rho)^2 - 1.2(\gamma+\rho)}\right). \tag{31}$$

Here, Conditions (15) and (16) are equivalent. By organizing Formula (16) into an inequality regarding ρ, we can obtain

$$\rho^2 - 2(\eta + 0.6)\rho - 1.2\gamma \geq 0. \tag{32}$$

Notice that $\Delta = 4(\eta+0.6)^2 + 4.8\gamma \geq 0$; thus, solving Inequality (32) gives an upper bound on ρ, $\rho \leq \eta + 0.6 - \sqrt{(\eta+0.6)^2 + 1.2\gamma}$.

In summary, when Conditions (16) and (31) hold, the algorithm can recover p successfully. Note that the condition $t \geq 4$ comes from the third step of AIOL, where we need to collect at least 2 short vectors to build the required Diophantine equations. Then, the condition

$$t \geq \max\left\{4, \left\lceil \frac{5}{3}\left(\eta - \rho - \sqrt{(\eta-\rho)^2 - 1.2(\gamma+\rho)}\right)\right\rceil\right\}.$$

is true. Hence, Theorem 3 holds. □

Theorem 3 gives a lower bound on the number of samples t. If the parameters γ, η, and ρ meet Conditions (15) or (16), the error term r_i can be established, and the secret number p can be determined.

3.4. The Complexity of the AIOL Algorithm

In the AIOL algorithm, the dominant computation is the LLL reduction of the lattice. Since only one appropriate set of N and t needs to be used at a time, the most complex calculations required of the AIOL algorithm are a single LLL lattice reduction. According to Theorem 2, the complexity of running the LLL lattice reduction algorithm is polynomial in γ, η, and t for $\delta = \frac{3}{4}$. More specifically, let $\mathcal{L}$ be a lattice of rank $t+1$ with the basis $b_1, \cdots, b_{t+1}$ and $\|b_i\| \leq 2^{\gamma+\eta+1}, (i = 1, \cdots, t+1)$. Then the number of bit operations needed by the LLL basis reduction in AIOL for $\delta = \frac{3}{4}$ is

$$O\Big((t+1)^6 \cdot (log_{\frac{4}{3}} 2^{\gamma+\eta+1})^3\Big), \tag{33}$$

or, equivalently,

$$O\Big((t+1)^6 \cdot (\gamma+\eta+1)^3\Big) \tag{34}$$

under school multiplication, where t satisfies Formula (11).

Remark 2. *The above analysis suggests that the asymptotical complexity of our AIOL algorithm is higher than that of the Ding–Tao algorithm, where the bit complexity is* $O\big((t+1)^6 \cdot (\gamma+1)^3\big)$ *under school multiplication (in [7], this complexity is given by* $O\big((t+1)^6 \cdot (2(\gamma+1))^3\big)$*. Here, we omit the constant 2 considering the effects of the leading symbol O). The reason is that in AIOL, the lattice parameter N is set to* $\gamma+\eta$ *bits, which is much bigger than in the Ding–Tao algorithm, where N is set to* γ *bits.*

4. Experiments and Comparisons

In this section, we conduct experiments on our algorithm AIOL, as well as the Ding–Tao algorithm. The experimental environment is specified as follows: an Intel Core i5-1235U CPU processor (1.30 GHz) with 16 GB of memory, Windows 10 OS, and Maple 2021 coding language.

The experiments are organized as follows. To test the effects of relaxation on conditions of the error length ρ and the required number of samples t, we adopted the following settings on the related parameters:

- We fixed $\eta = 160$, i.e., the bit-length of the hidden common divisor p;
- Let $\gamma = 300, 400, 500, 1000, 1500$, and 2000, respectively;
- For each case of setting on the bit-length of GACD samples γ, we ran the Ding–Tao algorithm and our AIOL algorithm 100 times for different ρ (resp., t) around the upper (resp., lower) bound of ρ (resp., t) given by the Ding–Tao Condition (9) and our Condition (11) and Condition (16), respectively.
- Then, for each case, we collected the success rate of recovering the hidden common divisor p, as well as the maximal ρ (resp., the minimal t) that enables the related algorithms work. That is, ρ_{max} and t_{min} represent the upper bound of ρ and the lower bound of t, respectively, when the corresponding algorithm can be used to recover p successfully.

The results of the first experiments are summarized in Table 2, where the symbol '–' indicates that in this case, the related algorithm failed to work out. We can see that:

- The overall success rate of our algorithm is 100%, which is observably higher than that of the Ding–Tao algorithm under the same settings of ρ, γ and a similar scale of t. Moreover, even for bigger settings of ρ in AIOL, the success rates are still higher than those obtained by the Ding–Tao algorithm for the smaller settings of ρ (intuitively, the bigger the value of ρ, the more errors are involved in the given ACD samples, and this, in turn, means more difficulty in solving the given GACD instances).
- The condition on ρ given by the Ding–Tao Condition (9) is *irrelevant* when considering that, for $\gamma = 300$ and $\gamma = 1000$, the maximal values of ρ to ensure the Ding–Tao algorithm has a high success rate are 103 and 30, respectively. These are, respectively, either observably bigger or smaller than the given bound $79 < \eta/2$.
- The condition on ρ given by AIOL is relaxed to the case of $\rho > \eta/2$. And this condition is *tight in the sense that* for all these cases, the maximal values of ρ to ensure the success of AIOL are almost same with the bound given by (16).
- The condition on t given by the Ding–Tao Condition (9) is rigorous *in the sense that* for even small values of t, our tests of the Ding–Tao algorithm failed, whereas the condition on t given by (11) in AIOL is *loose* since for even small t, our algorithm still works well. At present, we have no idea how to give a tight bound on choosing t for the AIOL algorithm.

Table 2. Experiments and comparisons: conditions on ρ, t and success rate ($\eta = 160$).

γ	Ding–Tao					AIOL				
	ρ (9)	$\rho_{\max}$	t (9)	t_{min}	succ %	ρ (16)	$\rho_{\max}$	t (11)	$t_{\min}$	succ %
300	79	103	11	11	82%	137	137	35	17	100%
400	79	91	12	12	87%	134	134	34	19	100%
500	79	80	13	13	90%	131	131	39	23	100%
1000	79	30	16	16	89%	122	122	54	33	100%
1500	79	–	19	–	–	115	115	60	40	100%
2000	79	–	21	–	–	109	109	72	46	100%

Remark 3. *Our experiments also indicate that the running time of AIOL becomes longer with an increase in the parameters γ, η, ρ. Under the same settings of γ and ρ, decreasing t will reduce the computational cost significantly. This is the reason why our AIOL algorithm runs much faster than the Ding–Tao algorithm for the above experimental GACD instances.*

5. Conclusions

Interest in the general approximate common divisor (GACD) problem has been excited by the possibility of building fully homomorphic encryptions over integers, though many

such kinds of cryptographic constructions have been broken. In fact, from even an abstract point of view, the GACD problem can be viewed as a *learning-with-error (LWE)* version of the greatest common divisor (GCD) problem over the 1-dimensional lattice $\mathbb{Z}$. Although we know that all lattice problems are easy to solve at low dimensions, more efforts are still needed to tackle the GACD problem. In this paper, an improved orthogonal lattice algorithm, AIOL, is proposed for solving the GACD problem. Compared with Ding and Tao's OL method, the parameter conditions applicable to AIOL are relaxed, and the experiments show that the success rate of AIOL is enhanced observably.

Author Contributions: Conceptualization, Y.R., L.W. and Z.C.; methodology, Y.R., L.W. and Z.C.; validation, Y.P. and L.W.; writing—original draft preparation, Y.R.; writing—review and editing, Y.R. and L.W.; code implementation, Y.R. and L.W.; supervision and project administration, Y.P. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported in part by the National Key Research and Development Program of China (Grant No. 2020YFA0712300), the National Defense Basic Scientific Research program of China (Grant No. JCKY2020602B008), and the National Natural Science Foundation of China (Grant No. 62272040, 62132005).

Data Availability Statement: Available on request from the corresponding author.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Howgrave-Graham, N. Approximate integer common divisors. In *Cryptography and Lattices*; Silverman, J., Ed.; Springer: Berlin/Heidelberg, Germany, 2001; Volume 2146, pp. 51–66.
2. Van Dijk, M.; Gentry, C.; Halevi, S.; Vaikuntanathan, V. Fully homomorphic encryption over the integers. In *Advances in Cryptology–EUROCRYPT 2010*; Gilbert, H., Ed.; Lecture Notes in Computer Sciences; Springer: Berlin/Heidelberg, Germany, 2010; Volume 6110, pp. 24–43.
3. Coron, J.S.; Naccache, D.; Tibouchi, M. Public Key Compression and Modulus Switching for Fully Homomorphic Encryption over the Integers. In *EUROCRYPT'12D*; Pointcheval, D., Johansson, T., Eds.; Springer: Berlin/Heidelberg, Germany, 2012; Volume 7237, pp. 446–464.
4. Cheon, J.H.; Stehlé, D. Fully Homomorphic Encryption over the Integers Revisited. In *EUROCRYPT'15*; Oswald, E., Fischlin, M., Eds.; Springer: Berlin/Heidelberg, Germany, 2015; Volume 9056, pp. 513–536.
5. Coron, J.S.; Mandal, D.; Tibouchi, N.M. Fully homomorphic encryption over the integers with shorter public keys. In *Advances in Cryptology-CRYPTO 2011*; Rogaway, P., Ed.; Lecture Notes in Computers Sciences; Springer: Berlin/Heidelberg, Germany, 2011; Volume 6841, pp. 487–504.
6. Chen, Y.; Nguyen, P.Q. Faster algorithms for approximate common divisors: Breaking fully homomorphic encryption challenges over the integers. In *Advances in Cryptology-EUROCRYPT 2012*; Springer: Berlin/Heidelberg, Germany, 2012; pp. 502–519.
7. Ding, J.; Tao, C. A New Algorithm for Solving the Approximate Common Divisor Problem and Cryptanalysis of the FHE based on GACD. *Iacr Cryptol. Eprint Arch.* 2014, *preprint*.
8. Galbraith, S.; Gebregiyorgis, S.; Murphy, S. Algorithms for the approximate common divisor problem. *LMS J. Comput. Math.* **2016**, *19*, 58–72. [CrossRef]
9. Xu, J.; Sarkar, S.; Hu, L. Revisiting orthogonal lattice attacks on approximate common divisor problems. *Theor. Comput. Sci.* **2022**, *911*, 55–69. [CrossRef]
10. Yu, X.; Wang, Y.; Xu, C.; Takagi, T. Studying the Bounds on Required Samples Numbers for Solving the General Approximate Common Divisors Problem. In Proceedings of the 2018 5th International Conference on Information Science and Control Engineering, Zhengzhou, China, 20–22 July 2018. [CrossRef]
11. Cohn, H.; Heninger, N. Approximate common divisors via lattices. In Proceedings of the ANTS X: Proceedings of the Tenth Algorithmic Number Theory Symposium, San Diego, CA, USA, 9–13 July 2012; Volume 1, pp. 271–293.
12. Gebregiyorgis, S. Algorithms for the Elliptic Curve Discrete Logarithm Problem and the Approximate Common Divisor Problem. PhD Thesis, The University of Auckland, Auckland, New Zealand, 2016.
13. Cheon, J.H.; Cho, W.; Hhan, M. Algorithms for CRT-variant of approximate greatest common divisor problem. *J. Math. Cryptol.* **2020**, *14*, 397–413. [CrossRef]
14. Cho, W.; Kim, J.; Lee, C. Extension of simultaneous Diophantine approximation algorithm for partial approximate common divisor variants. *IET Inf. Secur.* **2021**, *15*, 417–427. [CrossRef]
15. Takayasu, A.; Kunihiro, N. *Better Lattice Constructions for Solving Multivariate Linear Equations, Modulo Unknown Divisors*; Boyd, C., Simpson, L., Eds.; Springer: Berlin/Heidelberg, Germany, 2013; Volume 7959, pp. 118–135.
16. Takayasu, A.; Kunihiro, N. Better Lattice Constructions for Solving Multivariate Linear Equations Modulo Unknown Divisors. *IEICE Trans.* **2014**, *6*, 1259–1272. [CrossRef]

17. Lagarias, J.C. The computational complexity of simultaneous Diophantine approximation problems. *SIAM J. Comput.* **1985**, *14*, 196–209. [CrossRef]
18. Lepoint, T. Design and Implementation of Lattice-Based Cryptography. In *Cryptography and Security*; Ecole Normale Supérieure de Paris (ENS Paris): Paris, France, 2014.
19. Schnorr, C.-P. Lattice reduction by random sampling and birthday methods. In Proceedings of the STACS 2003, 20th Annual Symposium on Theoretical Aspects of Computer Science, Berlin, Germany, 27 February–1 March 2003; pp. 145–156.
20. Hoffstein, J.; Pipher, J.; Silverman, H.H. *An Introduction to Mathematical Cryptography*, 2nd ed.; Springer: Berlin/Heidelberg, Germany, 2014.
21. Nguyen, P.Q.; Valle, B. *The LLL Algorithm: Survey and Applications*; Springer: Berlin/Heidelberg, Germany, 2009.
22. Nguyen, P.Q.; Stern, J. The Two Faces of Lattices in Cryptology. In *Cryptography and Lattices*; Silverman, J., Ed.; Springer: Berlin/Heidelberg, Germany, 2001; pp. 146–180.