DiTRU: A Resurrection of NTRU over Dihedral Group

Ali Raya[1], Vikas Kumar[2], and Sugata Gangopadhyay[1]

[1] Department of Computer Science and Engineering
[2] Department of Mathematics
Indian Institute of Technology Roorkee, Roorkee 247667, India.
{ ali_r, sugata.gangopadhyay}@cs.iitr.ac.in, v_kumar@ma.iitr.ac.in

Abstract. NTRU-like cryptosystems are among the most studied lattice-based post-quantum candidates. While most NTRU proposals have been introduced over a commutative ring of quotient polynomials, other rings can be used. Noncommutative algebra has been endorsed as a direction to build new variants of NTRU a long time ago. The first attempt to construct a noncommutative variant was due to Hoffstein and Silverman motivated by more resistance to lattice attack. The scheme has been built over the group ring of a dihedral group. However, their design differed from standard NTRU and soon was found vulnerable to algebraic attacks. In this work, we revive the group ring NTRU over the dihedral group as an instance of the GR-NTRU framework.
Unlike many proposals of noncommutative variants in the literature, our work focuses on putting the scheme into practice. We clear all the aspects that make our scheme implementable by proposing an efficient inversion algorithm over the new setting of the noncommutative ring, describing the decryption failure model, and analyzing the lattice associated with our instantiation. Finally, we discuss the best-known attacks against our scheme and provide an implementation targeting 128-bit, 192-bit, and 256-bit levels of security as proof of its practicality. [‡]

Keywords: NTRU · noncommutative · post quantum · lattice-based

1 Introduction

The first NTRU cryptosystem was proposed early in 1996 by Hoffstein, Pipher, and Silverman [22]. Two decades of thorough cryptanalysis could not degrade the confidence in the hardness of NTRU assumption for well-chosen parameter sets. The hard problem of NTRU can be related to finding unusually short vectors in lattices of a particular structure(q-ary lattices), and in this regard, NTRU is classified as a lattice-based cryptosystem. Because of the efficiency and reasonable memory requirements of NTRU, different NTRU-like schemes have been proposed in the literature, resulting in a standard (IEEE-1363.1) [53] and

[‡]This paper is accepted for publication in AfricaCrypt 2024.

other competent candidates that progressed to the third round of NIST standardization process [8]. NTRU proposal inspired different NTRU-like designs that replace the underlying ring with other rings motivated by faster computations [30] or more resistance to some lattice attacks [40]. Noncommutativity has been endorsed as a promising direction for building NTRU-like schemes long ago [13]. Consequently, a few works in literature have introduced variants based on noncommutative algebra.

1.1 Related Work

Hoffstein and Silverman have introduced the first known noncommutative scheme [24] in literature based on the dihedral group, which was vulnerable to an attack by Coppersmith and Shamir [12]. The design of the key generation, encryption, and decryption procedures differs from the standard NTRU, and the attack by Coppersmith and Shamir exploits the fact that ciphertext is a pair of two elements from the ring. The attack applies a map on the first element and retrieves some information that helps recover the message from the second element. We refer the reader to the work by Truman [50] for a detailed analysis of this attack. Similarly, NNRU [52] and PairTRU [31] have been proposed as noncommutative analogs to NTRU operating over a matrix ring of $k \times k$ matrices of polynomials. The motivation behind these variants was to avoid lattice attacks; however, the schemes' design differs from NTRU, i.e., the public key in [31,52] is a pair of two elements from the underlying ring. Therefore, a thorough analysis of the hardness of the new assumption is required before establishing trust in the schemes.

Other works introduce noncommutative NTRU-like schemes from quaternion algebra where the lattice attacks are still applicable but harder to apply, according to the authors' claim. QTRU [40] is a noncommutative multi-dimensional NTRU-like scheme using quaternion algebra. The authors conclude that QTRU is four times slower than NTRU but more resistant to lattice attacks. BQTRU [6] is another example of a noncommutative scheme based on quaternion algebra with bivariate polynomials as the underlying ring. The design of BQTRU is inspired by QTRU, and the authors conclude that BQTRU can be faster than NTRU for equivalent levels of security if Genty's attack [18] is not applicable against the scheme. Further, Ling and Mendelsohn [38] introduced an interesting theoretical construction of an NTRU variant in quaternion algebras of bounded discriminant. The doubts related to these variants of NTRU arise from the poor analysis of the security of the associated lattices. For instance, none of these schemes analyze the behavior of the lattice reduction algorithms in practice. Moreover, the claim that Gentry's attack is not practically applicable to the associated lattices is not solid enough.

The Group ring NTRU or GR-NTRU [54] is an interesting proposal that generalizes NTRU to a group ring NTRU. In GR-NTRU, different schemes with an underlying hard assumption similar to that in the standard NTRU can be designed from the group ring $\mathbb{Z}G$, where $\mathbb{Z}$ refers to the integer ring and G is any abelian or nonabelian group.

1.2 Context

NTRU Hard Assumption: First NTRU scheme is defined over a truncated ring of polynomials $\mathcal{R}_q = \mathbb{Z}_q[x]/(x^N - 1)$ for prime N and modulo $q \in \mathbb{Z}$. The private key is a pair of polynomials $f, g \in \mathcal{R}_q$ with small coefficients (*ternary*) where f is an invertible polynomial. The public key is the polynomial calculated as $h := f^{-1} * g \in \mathcal{R}_q$. NTRU hard problem is formulated as: given h, find $f', g' \in \mathcal{R}_q$ (two polynomials with small coefficients) such that $f' * h = g' \pmod q$.

Knowing the public key h, there are mainly two paths to attack the problem: either by following an efficient search approach like Meet-in-the-Middle attack (MITM) [25,41,51] to find such pair (f', g'), or by mapping the problem of finding the private key into finding a short vector in a lattice built from the public key h using the lattice reduction algorithms [11, 45].

NTRU Learning Problem [44, 4.4.4]: is a relaxed variant of NTRU problem that assumes the attacker knows many samples of the public key h_i calculated using the same f but different g_i. The problem is formulated as: given $h_i := f^{-1} * g_i$ for $i \in \{1, 2, \ldots, m\}$, find f (or $x^k * f$ for some k). The NTRU learning problem has been studied to analyze a simplified NTRU problem [47]. It was believed to be hard and has been used to build some primitives like [1].

Recently, Kim and Lee [32] proposed a polynomial time attack that can break NTRU learning problem if the attacker knows N different h_i, where N refers to the extension degree of the ring $\mathcal{R}_q$. The attack exploits: ① The design of the NTRU variant that samples the private polynomial g with exactly d ones and d minus ones. ② The commutative algebra of the underlying ring $\mathcal{R}_q$.

The broad idea of the attack relies on the fact that $g_i * \bar{g}_i$, where $\bar{g}_i$ refers to the conjugate of g_i, always has a constant term c equal to the hamming weight of g_i. Therefore, for all the known h_i, and since the underlying ring is **commutative**, the attacker can build a system of linear equations of the form

$$constant(h_i * \bar{h_i} * f * \bar{f}) = c$$

This system has $f * \bar{f}$ as a root that can recover the private key in a polynomial time as described in [32]. Therefore, considering noncommutative algebras makes some algebraic attacks harder to apply, thus increasing the security of the cryptosystem constructed using them.

1.3 Our Work

Most of the noncommutative schemes in the literature have been proposed differently than the NTRU design, resulting in new schemes triggering doubts about the hardness of the new proposed assumptions. Other schemes have been introduced theoretically without clearing many aspects to make them implementable and practical.

In our work, we focus on clearing all the aspects of designing a noncommutative NTRU-like scheme based on the dihedral group. As an abbreviation, we

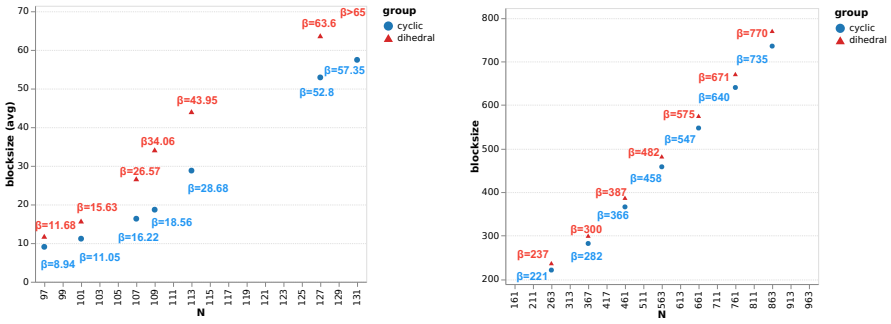
call it DiTRU. For DiTRU, we not only discuss the theoretical foundation of the cryptosystem but also provide experiments in support of our results. Our contribution can be summarized in the following:

- **DiTRU, a noncommutative analog of NTRU:** We instantiate the GR-NTRU framework [54] using the noncommutative dihedral group for our cryptosystem. The selection of the dihedral group is motivated by its closeness to the cyclic group, which enables the extension of many results and implementation constructions over the cyclic group(i.e., the underlying group of standard NTRU) to the new setting of the noncommutative ring.

- **Inversion algorithm:** We propose an inversion algorithm that can check the invertibility and find the inverse of elements over the group ring RD_N. We provide a necessary and sufficient condition to check/find inverses over RD_N by relating the inversion problem to the problem of checking/inverting elements over RC_N (Theorem 2). Particularly, for $R = \mathbb{Z}_q$ and q is a power of two, one can construct the inverse with complexity $O(N^2)$.

- **Analysis of DiTRU lattice:** We show that even if the DiTRU lattice is vulnerable to one layer of Gentry's attack, one need not exactly double the order of the dihedral group to match the same hardness of the SVP over the cyclic group of order N. For precise analysis, we describe the probability of decryption failure for DiTRU over D_N (of order $2N$) compared to NTRU over C_N(of order N). We show that the blocksize needed to retrieve the private key for DiTRU is larger than that for NTRU when a negligible decryption failure is allowed. This result follows as the lattice gap for DiTRU lattice is greater even if the SVP is being solved over lattices of the same dimension. For moderate lattice dimensions, we provide an experiment supporting our claim. In our experiment, we identify the smallest blocksize β required to retrieve a decryption key for DiTRU vs. NTRU lattices when the SVP is solved over the same dimension. Figure 1a summarizes the experimental results for moderate-size lattices, and Figure 1b compares the estimated blocksize to retrieve the private key according to *2016-estimator* in higher dimensions. Our experiment's implementation and detailed documentation can be accessed at `https://github.com/The-Isogeniest/DiTRU_blocksize_experiment`.

- **Full-fledged cryptosystem:** We discuss the cost of the search, primal, hybrid, and other related attacks against DiTRU, and based on that, we define three sets of parameters targeting 128-bit, 192-bit, and 256-bit security levels defined by NIST. We provide a C reference implementation of DiTRU and compare it with the parameter sets of NTRU that achieve the same security level according to the evaluation criteria followed for DiTRU. The package can be accessed at `https://github.com/The-Isogeniest/DiTRU`. To the best of our knowledge, this is the first noncommutative analog to NTRU, accompanied by a full-package implementation as proof of its practicality.



(a) Experimental results

(b) 2016-estimation

Fig. 1: Average blocksize needed to retrieve the key for DiTRU based on the dihedral group of order $2N$ after applying one layer of Gentry's attack vs. NTRU based on the cyclic group of order N. The experimental results are obtained for $q' = 512$ for NTRU lattice and q, achieving the same probability of decryption failure for DiTRU. While for 2016 estimation, the results are estimated for $(N, q') = (263, 1024), (367, 2048), (461, 2048), (563, 2048), (661, 2048), (761, 2048)$, and $(863, 2048)$ for NTRU and the equivalent parameters that achieve the same decryption failure for DiTRU. *2016-estimator* estimates that the gaps of the obtained blocksizes are $16, 18, 21, 24, 28, 31,$ and 35, respectively.

1.4 Organization

We introduce preliminaries, NTRU, GR-NTRU in Section 2. Section 3 introduces DiTRU along with the inversion algorithm and the analysis of the associated lattice. In Section 4, we provide cryptanalysis considering the well-known attacks against DiTRU. The selected parameters are presented in Section 5 followed by the adopted design rationale and implementation details in Section 6. Finally, we conclude our work in Section 7.

2 Preliminaries

2.1 Notations

- Symbol $*$, wherever it occurs, denotes the multiplication of elements with respect to the underlying structure.
- $\mathbb{Z}$ denote the set of integers and $\mathbb{Z}_q = \{a \pmod q \mid a \in \mathbb{Z}, -q/2 < a \leq q/2\}$ for a positive integer q.
- G denotes a finite group, R denotes a commutative ring with unity and R^* be the group of units of R.
- Let $\mathbb{R}^n$ be the Euclidean space of dimension n, given a vector $\mathbf{v} = (v_1, v_2, \ldots v_n) \in \mathbb{R}^n$:
  - The Euclidean norm ℓ_2 is denoted $\|.\|$ and calculated as $\|\mathbf{v}\| = \sqrt{\sum_{i=1}^{n} {v_i}^2}$.
  - The $\ell-$infinity norm ℓ_∞ is denoted $\|.\|_\infty$ and calculated as $\|\mathbf{v}\|_\infty = \max_{i=1}^{n} |v_i|$.

2.2 Definitions

Definition 1. *(Lattice). Let* $\mathbf{B} \in \mathbb{R}^{n \times m}$ *with independent rows* $\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_n \in \mathbb{R}^m$*. A lattice* $\mathcal{L}(\mathbf{B})$ *generated by the matrix* $\mathbf{B}$ *is the set of integer linear combination of rows of* $\mathbf{B}$*, i.e.,*

$$\mathcal{L}(\mathbf{B}) = \sum_{i=1}^{n} \mathbb{Z}\mathbf{b}_i = \left\{ \sum_{i=1}^{n} z_i \mathbf{b}_i, \text{ where } z_i \in \mathbb{Z} \right\}. \tag{1}$$

We call $\mathbf{B}$ a basis matrix of the lattice. If $\mathbf{b}_i \in \mathbb{Z}^n$, we call the lattice an integral lattice. This paper considers only full-rank integral lattices, i.e., $n = m$. We refer to the volume of the parallelepiped spanned by the basis $\mathbf{b}_i$'s as the volume of the lattice defined as $det(\mathcal{L}(\mathbf{B})) = \sqrt{|det(\mathbf{B}\mathbf{B}^T)|}$.

Definition 2. *(SVP). Given a lattice* $\mathcal{L}(\mathbf{B}) \subset \mathbb{R}^n$*, the Shortest Vector Problem (SVP) asks to find a non-zero vector* $\mathbf{v} \in \mathcal{L}(\mathbf{B})$ *such that* $\|\mathbf{v}\| \leq \|\mathbf{w}\|$ *for all non-zero vectors* $\mathbf{w} \in \mathcal{L}(\mathbf{B})$*; the length of the shortest vector in the lattice is denoted as* $\lambda_1(\mathcal{L}(\mathbf{B}))$*.*

A relaxed variant of the SVP called γ-SVP asks to find a short nonzero vector $\mathbf{x} \in \mathcal{L}(\mathbf{B})$ within an approximation factor $\gamma(n) \geq 1$ of the length of the shortest vector, i.e., $\|\mathbf{x}\| \leq \gamma(n)\lambda_1(\mathcal{L}(\mathbf{B}))$.

Definition 3. *(Gaussian heuristic). For a full rank lattice* $\mathcal{L}(\mathbf{B})$ *of dimension* d*, the estimation of the norm of the shortest vector according to the Gaussian heuristic is denoted as* $gh(\mathcal{L}(\mathbf{B}))$*, and calculated as*

$$gh(\mathcal{L}(\mathbf{B})) = \sqrt{d/2\pi e} \cdot \ det(\mathcal{L}(\mathbf{B}))^{1/d}. \tag{2}$$

Definition 4. *(Group ring). The group ring of a finite group* $G = \{g_i \mid i = 1, 2, \ldots, n\}$ *over* R *is the ring*

$$RG = \left\{ a = \sum_{i=1}^{n} \alpha_i g_i : \alpha_i \in R \text{ for } i = 1, 2, \ldots, n \right\} \tag{3}$$

with the following operations: Let $a = \sum_{i=1}^{n} \alpha_i g_i, b = \sum_{i=1}^{n} \beta_i g_i \in RG$*. Then*
1. Sum of a *and* b *is* $\sum_{i=1}^{n} \alpha_i g_i + \sum_{i=1}^{n} \beta_i g_i = \sum_{i=1}^{n} (\alpha_i + \beta_i) g_i$*.*
2. Product of a *and* b *is* $\sum_{i=1}^{n} \alpha_i g_i * \sum_{i=1}^{n} \beta_i g_i = \sum_{i=1}^{n} \left(\sum_{g_h g_k = g_i} \alpha_h \beta_k \right) g_i$*.*

Definition 5. *(Coefficient vector). The coefficient vector of* $a = \sum_{i=1}^{n} \alpha_i g_i \in RG$ *is* $\mathbf{a} = (\alpha_1, \alpha_2, \ldots, \alpha_n) \in R^n$*.*

We use a and $\mathbf{a}$ to denote the group ring elements interchangeably depending on the context.

Definition 6. *(Matrix representation). The RG-matrix of* $a = \sum_{i=1}^{n} \alpha_{g_i} g_i \in RG$ *is defined as*

$$\mathbf{M}_{RG}(a) = \begin{pmatrix} \alpha_{g_1^{-1}g_1} & \alpha_{g_1^{-1}g_2} & \ldots\ldots & \alpha_{g_1^{-1}g_n} \\ \alpha_{g_2^{-1}g_1} & \alpha_{g_2^{-1}g_2} & \ldots\ldots & \alpha_{g_2^{-1}g_n} \\ \vdots & \vdots & \ddots & \vdots \\ \alpha_{g_n^{-1}g_1} & \alpha_{g_n^{-1}g_2} & \ldots\ldots & \alpha_{g_n^{-1}g_n} \end{pmatrix}. \tag{4}$$

The matrix representation of the group ring elements is unique and satisfies

$$\mathbf{M}_{RG}(a+b) = \mathbf{M}_{RG}(a) + \mathbf{M}_{RG}(b),\ \mathbf{M}_{RG}(a*b) = \mathbf{M}_{RG}(a) * \mathbf{M}_{RG}(b) \tag{5}$$

for all $a, b \in RG$.

2.3 Lattice basis reduction

Given a publicly available *'bad'* basis with large and highly non-orthogonal vectors, a lattice reduction algorithm tries to find *'good'* basis consisting of reasonably short and orthogonal vectors that define the same lattice. LLL [37] is a famous example of a polynomial-time basis reduction algorithm that produces a good-reduced basis for low dimensions. Although LLL runs in polynomial time, the quality of the reduced basis degrades as the dimension of the lattice increases. BKZ [45], and its variants like BKZ2.0 [11] and Progressive BKZ [4] are generalizations of LLL that consider an additional parameter: the blocksize or β. The higher the value of β, the better the quality of the reduced basis and the higher the running time.

For a full rank lattice $\mathcal{L}(\mathbf{B})$ reduced with a blocksize β such that $\|\mathbf{b}_1\| \leq \|\mathbf{b}_2\| \leq \ldots \leq \|\mathbf{b}_d\|$, we give the following definitions:

Definition 7. *(Root Hermite factor). The root Hermite factor δ is defined via $\|\mathbf{b}_1\| = \delta^d det(\mathcal{L}(\mathbf{B}))^{1/d}$ and can be estimated for larger β [10] as*

$$\delta = \left(\frac{\beta}{2\pi e}(\pi\beta)^{\frac{1}{\beta}}\right)^{\frac{1}{2(\beta-1)}}. \tag{6}$$

Definition 8. *(Geometric Series Assumption). The Geometric Series Assumption(GSA) estimates that $\left\|b_i^*\right\| \approx \delta^{-2}\left\|b_{i-1}^*\right\|$, where δ is the root Hermite factor.*

The exact blocksize required to find a short vector with norm $\|\mathbf{v}\|$ lying in a lattice $\mathcal{L}(\mathbf{B})$ is still an active area of research. A few estimators like *2016-estimator* [3] have been introduced to estimate the blocksize β needed to retrieve a short vector $\mathbf{v} \in \mathcal{L}$. 2016-estimator briefly states that BKZ with blocksize β can retrieve the vector $\mathbf{v}$ given that:

$$\sqrt{\beta/d}\|\mathbf{v}\| < \delta^{2\beta-d-1} \cdot det(L)^{1/d}, \tag{7}$$

where δ indicates the root Hermite factor. The value of β and, therefore, the hardness of the problem increases with the increase of the lattice dimension d, and the lattice gap $\frac{\|\mathbf{v}\|}{gh(\mathcal{L}(\mathbf{B}))}$. A further discussion regarding the cost of the SVP problem concerning DiTRU lattice with respect to enumeration and sieving regimes is given in Subsection 4.2.

2.4 NTRU cryptosystem

There are many variants of NTRU in literature. We discuss the key generation, encryption, and decryption of NTRU cryptosystem as described in [21].

Let N, p, q be positive integers such that N, p are prime numbers, $p \ll q$, and q is a power of 2. Let $\mathcal{R}$, $\mathcal{R}_p$, and $\mathcal{R}_q$ be the truncated ring of polynomials of degree N defined as

$$\mathcal{R} = \frac{\mathbb{Z}[x]}{(x^N - 1)},\ \mathcal{R}_p = \frac{\mathbb{Z}_p[x]}{(x^N - 1)},\ \mathcal{R}_q = \frac{\mathbb{Z}_q[x]}{(x^N - 1)}. \tag{8}$$

Let $\mathcal{T}_N$ be the space of all N length ternary polynomials with coefficients $-1, 0, 1$ and $\mathcal{T}_N(d_1, d_2)$ be the space of ternary polynomials with d_1 coefficients equal to 1, d_2 coefficients equal to -1, and the remaining coefficients equal to 0.

- **Key generation:** The NTRU private key is pair $(f, g) \in \mathcal{T}_N(d+1, d) \times \mathcal{T}_N(d, d)$, where $d = \lfloor N/3 \rfloor$ and f is invertible in $\mathcal{R}_p$ with inverse f_p as well as in $\mathcal{R}_q$ with inverse f_q. NTRU public key is computed as $h = f_q * g \in \mathcal{R}_q$.
- **Encryption:** A message $m \in \mathcal{R}_p$ is encrypted as $c = pr * h + m \in \mathcal{R}_q$, where r is sampled randomly from $\mathcal{T}(d, d)$.
- **Decryption:** First, compute $a = f * c \in \mathcal{R}_q$, then the decrypted message is retrieved as $m' = f_p * a \in \mathcal{R}_p$.

NTRU lattice: The problem of finding the NTRU private key can be related to the SVP in a lattice of a certain form. Given the public information q, N and $h = f_q * g \pmod q$, construct the basis matrix for the lattice $\mathcal{L}(\mathbf{B_{cyclic}})$ as follows:

$$\mathbf{B_{cyclic}} = \begin{pmatrix} \mathbf{I}_N & \mathbf{H_{cyclic}} \\ \mathbf{0}_N & q\mathbf{I}_N \end{pmatrix}, \tag{9}$$

where $\mathbf{H_{cyclic}}$ is a right circulant matrix whose rows are the coefficient vectors of the polynomials $x^i * h$ for $i \in \{0, 1, \ldots, N-1\}$. The determinant of the lattice $\mathcal{L}(\mathbf{B_{cyclic}})$ is $det(\mathbf{B_{cyclic}}) = q^N$. Therefore, $gh(\mathcal{L}(\mathbf{B_{cyclic}})) = \sqrt{qN/\pi e}$.

While the norm of the private elements $(x^i * f, x^i * g)$ is approximately $\sqrt{4N/3}$ and $(x^i * f, x^i * g) \in \mathcal{L}(\mathbf{B_{cyclic}})$ since $(x^i * f) * h = x^i * g \pmod q$. Therefore, one expects (f, g) or its rotations to be the shortest vectors in the lattice $\mathcal{L}(\mathbf{B_{cyclic}})$ for large values of N.

2.5 Group ring NTRU/GR-NTRU

There are various attempts in the literature to generalize NTRU. We find the GR-NTRU by Yashuda et al. [54], the most reasonable description for designing NTRU-like cryptosystems. We first describe NTRU as a cryptosystem based on a group ring to lay the path for introduction to GR-NTRU.

One can think of the ring $\mathcal{R} = \mathbb{Z}[x]/\left(x^N - 1\right)$ as the group ring of cyclic group $C_N = \left\langle x \mid x^N = 1 \right\rangle$ of order N over the ring of integers $\mathbb{Z}$. In other words,

$\frac{\mathbb{Z}[x]}{(x^N-1)} \approx \mathbb{Z}C_N$. The matrix $\mathbf{H_{cyclic}}$ in equation 9 is $\mathbf{M}_{\mathbb{Z}C_N}(h)$, the $\mathbb{Z}C_N$-matrix of the public key h. The rest of the design criteria, as discussed in section 2.4, follow naturally over the group ring $\mathbb{Z}C_N$.

One can change the cyclic group C_N with other groups to construct new variants of NTRU. This motivates the definition of GR-NTRU introduced in [54] as follows:

Definition 9. *(GR-NTRU). The GR-NTRU over a finite group G is a cryptosystem built over the group ring $\mathbb{Z}G$ with key generation, encryption, and decryption the same as NTRU except that the operations are performed over the rings $\mathbb{Z}G, \mathbb{Z}_pG$, and $\mathbb{Z}_qG$, where $p \ll q$ are positive integers.*

In general, deciphering the private key of GR-NTRU is also related to the shortest vector problem in lattices of particular structures associated with the underlying group ring. This paper focuses on GR-NTRU based on the integral group ring of dihedral group, which we call DiTRU.

3 DiTRU (GR-NTRU over dihedral group)

Let $D_N = \langle x, y \mid x^N = y^2 = 1, xy = yx^{N-1} \rangle$ be dihedral group of order $2N$. DiTRU is a GR-NTRU over the group ring

$$\mathbb{Z}D_N \approx \frac{\mathbb{Z}[x,y]}{(x^N-1, y^2-1, xy-yx^{N-1})}. \tag{10}$$

Any element of the group ring $\mathbb{Z}D_N$ can be written in the form $f = f_0(x) + yf_1(x)$, where $f_0(x)$ and $f_1(x)$ are elements of the ring $\mathbb{Z}C_N \approx \mathbb{Z}[x]/(x^N-1)$.

Let $h = h_0(x) + yh_1(x)$ be the public key of DiTRU corresponding to the private key $(f,g) = (f_0(x) + yf_1(x), g_0(x) + yg_1(x))$, and

$$\mathbf{H_{dihedral}} = \mathbf{M}_{\mathbb{Z}D_N}(h) \tag{11}$$

be the matrix representation of h. Since $\mathbf{f} * \mathbf{H_{dihedral}} = \mathbf{g} \pmod q$, DiTRU can be associated with the lattice $\mathcal{L}(\mathbf{B_{dihedral}})$ that contains (f,g), generated by the basis matrix

$$\mathbf{B_{dihedral}} = \begin{pmatrix} \mathbf{I}_{2N} & \mathbf{H_{dihedral}} \\ \mathbf{0}_{2N} & q\mathbf{I}_{2N} \end{pmatrix}. \tag{12}$$

It is discussed in [35] that $\mathbf{H_{dihedral}} = \begin{pmatrix} \mathbf{H}_0 & \mathbf{H}_1 \\ \mathbf{H}_1 & \mathbf{H}_0 \end{pmatrix}$, where $\mathbf{H}_0, \mathbf{H}_1$ are right and left circulant matrices whose first rows are the coefficient vectors $\mathbf{h}_0$ of $h_0(x)$ and $\mathbf{h}_1$ of $h_1(x)$, respectively.

Theorem 1. *([35, Thereom 5]) For all $0 \le i \le N-1$, the 'rotations' of private key (f,g) given by*

$$(x^i * f_0, x^{-i} * f_1, x^i * g_0, x^{-i} * g_1) \text{ and } (x^i * f_1, x^{-i} * f_0, x^i * g_1, x^{-i} * g_0)$$

belong to the lattice $\mathcal{L}(\mathbf{B_{dihedral}})$.

3.1 Inversion algorithm

There are works on characterizing units in dihedral group rings [39,43]. However, all those classifications rely on group representation theory and are not easily implementable to construct units. Further, Hurley in [29] relates the invertibility of an element of the group ring RG with the invertibility of the associated RG-matrix over the ring R. However, this method of matrix inversion is inefficient for larger dimensions. Therefore, it becomes essential to look for alternative ways to check for units. In this section, we provide a time-effective algorithm to generate units in finite integral group rings of dihedral groups. Before discussing the main inversion algorithm, let us give the required definition and result.

Definition 10. *For $f(x) = f_0 + f_1x + \ldots + f_{N-1}x^{N-1} \in RC_N$, the conjugate of $f(x)$ is defined as $\overline{f(x)} = f(x^{N-1})$. In the vector form, conjugate of $\mathbf{f} = (f_0, f_1, \ldots, f_{N-1})$ is $\bar{\mathbf{f}} = (f_0, f_{N-1}, f_{N-2}, \ldots, f_1)$.*

One can check that $\overline{\overline{u(x)}} = u(x)$, $\overline{u(x) \pm v(x)} = \overline{u(x)} \pm \overline{v(x)}$ and $\overline{u(x) * v(x)} = \overline{u(x)} * \overline{v(x)}$ for all $u(x), v(x) \in RC_N$.

Multiplication in RD_N: The relation $xy = yx^{N-1}$ between the generators of D_N gives that the product between two elements $f = f_0(x) + yf_1(x)$ and $g = g_0(x) + yg_1(x)$ of the group ring RD_N is

$$f * g = f_0(x) * g_0(x) + \overline{f_1(x)} * g_1(x) + y(f_1(x) * g_0(x) + \overline{f_0(x)} * g_1(x)). \tag{13}$$

Theorem 2. *(Necessary and sufficient condition). Let $f = f_0(x) + yf_1(x) \in RD_N$. Then, f is a unit in RD_N if and only if $c(x) = f_1(x) * \overline{f_1(x)} - f_0(x) * \overline{f_0(x)}$ is a unit in RC_N. Moreover, if $i(x)$ denotes the inverse of $c(x)$, then the inverse of f is given by*

$$f^{-1} = -\overline{f_0(x)} * i(x) + yf_1(x) * i(x). \tag{14}$$

Proof. From Equation 13

$$\begin{aligned} f * f^{-1} &= (f_0(x) + yf_1(x)) * (-\overline{f_0(x)} * i(x) + yf_1(x) * i(x)) \\ &= (f_1(x) * \overline{f_1(x)} - f_0(x) * \overline{f_0(x)}) * i(x) = 1 \end{aligned}$$

Since $c(x) * i(x) = 1$ therefore $\overline{c(x) * i(x)} = \overline{c(x)} * \overline{i(x)} = 1$. Using the commutativity of RC_N, we get $\overline{c(x)} = c(x)$. Hence, by the uniqueness of the inverse, $i(x) = \overline{i(x)}$. Now, consider

$$\begin{aligned} f^{-1} * f &= (-\overline{f_0(x)} * i(x) + yf_1(x) * i(x)) * (f_0(x) + yf_1(x)) \\ &= (f_1(x) * \overline{f_1(x)} - f_0(x) * \overline{f_0(x)}) * \overline{i(x)} = 1 \end{aligned}$$

Conversely, suppose $f = f_0(x) + yf_1(x)$ is a unit in RD_N with inverse $f^{-1} = u(x) + yv(x)$. Then $f^{-1} * f = 1 + y0$ gives

$$f_0(x) * u(x) + \overline{f_1(x)} * v(x) = 1 \text{ and } f_1(x) * u(x) + \overline{f_0(x)} * v(x) = 0.$$

Equivalently $\begin{pmatrix} f_0(x) & \overline{f_1(x)} \\ f_1(x) & \overline{f_0(x)} \end{pmatrix} \begin{pmatrix} u(x) \\ v(x) \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$. The uniqueness of the inverse in a group ring guarantees that the matrix $\begin{pmatrix} f_0(x) & \overline{f_1(x)} \\ f_1(x) & \overline{f_0(x)} \end{pmatrix}$ is invertible; therefore, its determinant $f_0(x) * \overline{f_0(x)} - f_1(x) * \overline{f_1(x)}$ is a unit in RC_N. Further,

$$\begin{pmatrix} f_0(x) & \overline{f_1(x)} \\ f_1(x) & \overline{f_0(x)} \end{pmatrix}^{-1} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \frac{1}{f_0(x) * \overline{f_0(x)} - f_1(x) * \overline{f_1(x)}} \begin{pmatrix} \overline{f_0(x)} & -\overline{f_1(x)} \\ -f_1(x) & f_0(x) \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix}.$$

This gives that $u(x) = -\overline{f_0(x)} * i(x)$ and $v(x) = f_1(x) * i(x)$. □

Algorithm 1: Inversion in RD_N

Input: $f = f_0(x) + yf_1(x) \in RD_N$
Output: $f^{-1} = u(x) + yv(x) \in RD_N$ an inverse of f, or a failure

1 $mul_1 \leftarrow f_0(x) * \overline{f_0(x)}$ /* product in RC_N */
2 $mul_2 \leftarrow f_1(x) * \overline{f_1(x)}$ /* product in RC_N */
3 $c(x) \leftarrow mul_2 - mul_1$ /* Coefficient-wise subtraction in R */
4 $i(x), found \leftarrow$ find-inverse-in-RC_N(c(x))
5 **if** *not found* **then**
6 **return** *failure*
7 $u(x) \leftarrow -\overline{f_0(x)} * i(x)$ /* product in RC_N */
8 $v(x) \leftarrow f_1(x) * i(x)$ /* product in RC_N */
9 **return** $f^{-1} = u(x) + yv(x)$

Algorithm 1 relates the problem of finding the inverse of an element in RD_N to finding the inverse of an element in RC_N (line 4). Therefore, the cost of constructing the inverse in RD_N equals the cost of computing an inverse in RC_N plus $(4N^2 + N)$. In case $R = \mathbb{Z}_q$ for q prime or prime power, one can use an efficient algorithm to find the inverse for units in $\mathbb{Z}_q C_N$ as in [46], and therefore constructing units in $\mathbb{Z}_q D_N$ efficiently.

Corollary 1. *If $f = f_0(x) + yf_1(x)$ is a unit in RD_N then $f' = f_1(x) + yf_0(x)$ is also a unit in RD_N with inverse*

$$f'^{-1} = \overline{f_1(x)} * i(x) - yf_0(x) * i(x) \tag{15}$$

where $i(x)$ is the inverse of $f_1(x) * \overline{f_1(x)} - f_0(x) * \overline{f_0(x)}$ in RC_N.

Since $f \in \mathcal{T}(d+1, d)$ therefore $f_1(1) * \overline{f_1(1)} - f_0(1) * \overline{f_0(1)} \neq 0 \pmod 2$. Hence, for a prime N such that 2 is a primitive root modulo N, i.e., multiplicative order of 2 modulo N is $N-1$, the element $f_1(x) * \overline{f_1(x)} - f_0(x) * \overline{f_0(x)}$ is invertible in $\mathbb{Z}_q C_N$ with high probability and consequently f is invertible in $\mathbb{Z}_q D_N$ with high probability, where q is a power of 2 [20, Page 3].

3.2 Analysis of DiTRU lattice

One-layer of Gentry attack: The dihedral group D_N for prime N has a composite order $2N$; therefore, one needs to consider if an extension of Gentry's

attack [18] applies to the DiTRU lattice. Gentry's attack makes the problem of solving SVP easier by mapping the original lattice into smaller dimensional lattices through homomorphisms. In his original paper, Gentry elaborates his attack for a cyclic group of composite order 2^n for positive n. A similar one-layer attack can be extended to DiTRU lattice corresponding to D_N for prime N. If the vector corresponding to the private key $(\mathbf{f}_0, \mathbf{f}_1, \mathbf{g}_0, \mathbf{g}_1) \in \mathcal{L}(\mathbf{B_h})$, then according to the homomorphisms defined in Figure 2, $(\mathbf{f}_0 + \mathbf{f}_1, \mathbf{g}_0 + \mathbf{g}_1) \in \mathcal{L}(\mathbf{B_{h_0+h_1}})$ and $(\mathbf{f}_0 - \mathbf{f}_1, \mathbf{g}_0 - \mathbf{g}_1) \in \mathcal{L}(\mathbf{B_{h_0-h_1}})$. Therefore, it will be more beneficial for the attacker to find these images and build them back to get the original vector corresponding to the private key. Refer to [35] for a detailed discussion. One can

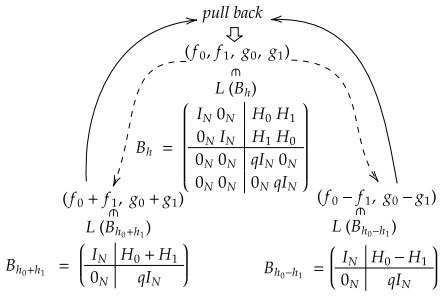


Fig. 2: One-layer of Gentry attack against DiTRU lattice

think that the hardness of solving the SVP for the DiTRU lattice of order $2N$ is equivalent to that for a lattice built for a cyclic group of order N. However, for accurate hardness analysis, the lattice gap of the images of the short vector in the dimension-reduced lattices should be analyzed.

For a better understanding, we compare the hardness of the SVP for NTRU lattices based on C_N(order N) to that based on D_N(order $2N$) when an equal negligible decryption failure is allowed. Hence, we first introduce the estimation of the decryption failure for DiTRU before providing our analysis.

Decryption failure: The probability of decryption failure can be estimated similarly to the discussion in [19]. For DiTRU based on a dihedral group of order $2N$ and designed according to the steps mentioned in section 2.4, a decryption failure occurs if the absolute value of any coefficients in $a = pr*g+m*f$ is greater than $t = q/2$. Therefore, if $g \in \mathcal{T}(d_g, d_g)$ and $d'_g \approx d_g/2$ (similar assumptions are considered for f, r, and m), the probability of decryption failure is defined as

$$p_{dec}(t) = \texttt{Prob}(\, \|a\|_\infty \geq t), \tag{16}$$

and calculated as

$$p_{dec} = 2N * \texttt{erfc}(t/\sigma\sqrt{2}), \tag{17}$$

where $\sigma^2 = 8\left(\frac{p^2(d'_r d'_g)+d'_f d'_m}{N}\right)$ and $\texttt{erfc}$ refers to the complementary error function. To prove the correctness of 17, we make the following valid assumptions:

Assumption 1 *Let $g = g_0(x) + yg_1(x)$ be elements of $\mathbb{Z}D_N$ such that $g \in \mathcal{T}(d_g, d_g)$, then for large N, we approximately expect that $g_0(x), g_1(x) \in \mathcal{T}(d'_g, d'_g)$ (similar assumptions are considered for f, r, and m).*

Assumption 2 *The coefficients of $g_0(x)$ are independent random variables taking the values 1 and -1 with probability d'_g/N, and 0 with probability $(N - 2d'_g)/N$. Assuming the same for $g_1(x)$, $f_0(x)$, $f_1(x)$, $m_0(x)$, and $m_1(x)$.*

We know that a can be written as $a = a_0(x) + ya_1(x)$ for:

$$a_0(x) = p\left(r_0(x) * g_0(x) + \overline{r_1(x)} * g_1(x)\right) + m_0(x) * f_0(x) + \overline{m_1(x)} * f_1(x)$$
$$a_1(x) = p\left(\overline{r_0(x)} * g_1(x) + r_1(x) * g_0(x)\right) + \overline{m_0(x)} * f_1(x) + m_1(x) * f_0(x).$$

We give the discussion for $a_0(x)$; the same discussion can be translated to $a_1(x)$. Let X_j denote a coefficient in $a_0(x)$, then X_j is the sum of N terms as

$$X_j = \sum_{i=1}^{N} \big(p(z_{0_i} + z_{1_i}) + (w_{0_i} + w_{1_i})\big),$$

where z_{0_i}, z_{1_i}, w_{0_i}, and w_{1_i} denote the coefficient of $r_0(x)*g_0(x)$, $\overline{r_1(x)}*g_1(x)$, $m_0(x)*f_0(x)$, and $\overline{m_1(x)} * f_1(x)$, respectively. As a result, the variance

$$\sigma^2 = E(X_j^2) = \sum_{i=1}^{N} \left(p^2(E(z_{0_i}^2) + E(z_{1_i}^2)) + E(w_{0_i}^2) + E(w_{1_i}^2)\right)$$
$$= 2\left(p^2 \frac{4d'_r d'_g}{N} + \frac{4d'_f d'_m}{N}\right) = 8\left(\frac{p^2(d'_r d'_g) + d'_f d'_m}{N}\right). \tag{18}$$

For large N, we can apply the central limit theorem twice to estimate the probability, consequently

$$\texttt{Prob}(\,|X_j| \geq t) < \frac{2}{\sqrt{2\pi}} \int_{t/\sigma}^{\infty} e^{-x^2/2} dx \Rightarrow \texttt{Prob}(\,|X_j| \geq t) < \texttt{erfc}(t/\sigma\sqrt{2}).$$

Therefore, the probability of decryption failure can be conservatively estimated to have one coefficient or more, either in $a_0(x)$ or $a_1(x)$ with a value greater than t. Hence

$$p_{dec} = 2N * \texttt{erfc}(t/\sigma\sqrt{2}), \quad \text{for} \quad \sigma^2 = 8\left(\frac{p^2(d'_r d'_g) + d'_f d'_m}{N}\right). \tag{19}$$

Experimental results: As subsection 3.2 mentions, the DiTRU lattice is vulnerable to one layer of Gentry's attack. To understand the hardness of reducing DiTRU lattices built over a dihedral group D_N of order $2N$, we experiment to figure out the minimum blocksize needed to retrieve a decryption key for moderate lattice dimensions; then, the results can be extended using simulators to estimate the blocksize for higher dimensions (Figure. 1b). For reference comparison, we compare the obtained blocksizes to those needed to reduce NTRU lattices over a cyclic group C_N of order N. For a negligible decryption failure, we set our experiment as the following:

- identify q', the modulo for cyclic NTRU, as the minimum power of 2 that satisfies a decryption failure probability p'_{dec} (as computed in [19]) equal to or smaller than the targeted.
- identify q, the modulo for DiTRU over D_N that gives an equal decryption failure p_{dec} compared to NTRU over C_N.
- for each parameter set for cyclic and dihedral, generate 100 random private keys, then build and publish the public key [§].
- for NTRU over the cyclic group, find the minimum blocksize β_{cyclic} needed to retrieve a decryption key (non-ternary and ternary).
- for DiTRU lattice:
  - apply one layer of Gentry's attack and build the lattices $\mathcal{L}(\mathbf{B}_{h_0+h_1})$ and $\mathcal{L}(\mathbf{B}_{h_0-h_1})$.
  - identify the smaller blocksize β_1 to get a non-ternary decryption key and β_2 the minimum blocksize to retrieve the ternary key.

We highlight that the experiment uses *progressive* BKZ with increasing blocksizes up to 65 with eight tours per blocksize and enumeration as an SVP-oracle. We ran the experiment depending on FPyLLL [49] as a Python wrapper to FPLLL [48] on a system Linux (Ubuntu 22.04.2 LTS) with Intel(R) Xeon(R) CPU E3-1246 v3 @ 3.50GHz and 32 GB installed RAM. Furthermore, all the tested parameter sets are not in the overstretched regime of NTRU lattices, and the non-ternary decryption key is accepted if its norm is at most four times the original key. Table 1 shows the experimental results

Table 1: Average blocksizes needed to retrieve a decryption key checked experimentally (NTRU over cyclic group vs. DiTRU after one-layer of Gentry's attack.)

N	**71**	**73**	**79**	**83**	**89**	**97**	**101**	**107**	**109**	**113**	**127**	**131**
β_{cyclic}	2.28	2.48	3.02	3.64	5.22	8.94	11.05	16.22	18.56	28.68	52.8	57.35
β_1	2.62	2.95	3.54	4.94	7.06	11.62	15.59	25.47	32.75	43.1	63	_
β_2	2.87	3	3.88	5.06	7.18	11.68	15.63	26.57	34.06	43.95	63.6	_

tested for the corresponding N and $q' = 512$ for the NTRU over the cyclic group, and $q = 2 * \texttt{erfc}^{-1}(p'_{dec}/2N) * \sigma\sqrt{2}$ for DiTRU parameter sets where p'_{dec} is the probability of decryption failure in the case of the cyclic parameters. While the difference between the blocksizes may seem small, the gap in the running time is significantly large. For instance, for $N = 127$, the running time took an average of 626.6 core hours to retrieve the shortest vector in the case of DiTRU, while for NTRU, it took, on average, only 114.9 core hours.

4 Best known attacks

4.1 Search attack

A DiTRU private key (f, g) is a ternary vector where $f = (f_0, f_1) \in \mathcal{T}_{2N}(d+1, d)$ and $g = (g_0, g_1) \in \mathcal{T}_{2N}(d, d)$ with $d \leq \lfloor 2N/3 \rfloor$. For convenience, let us denote $\mathcal{T}_{2N}(d+1, d)$ simply by $\mathcal{T}$. An attacker can brute force search for an $f' \in \mathcal{T}$ such that $f' * h \pmod q$

[§]Starting from $N = 113$, the results are averaged over at least 20 trials (only) since the time taken by one trial becomes extensively high. For $N = 127$ with DiTRU lattice, we recorded the trials that found the key with $\beta \leq 65$.

is short, possibly ternary. Therefore, the cost of a combinatorial search on DiTRU is given by

$$\frac{|\mathcal{T}|}{2N} = \frac{1}{2N}\binom{2N}{d}\binom{2N-d}{d+1}, \tag{20}$$

where we have divided by $2N$ to account for all the $2N$ rotations of f'. In fact, combinatorial meet-in-the-middle (MITM) [¶] attacks by Odlyzko [25] and Howgrave et al. [27] with complexity $\left(|\mathcal{T}|/2N\right)^{0.5}$ (classically) and $\left(|\mathcal{T}|/2N\right)^{0.25}$ (quantumly) can be mounted by decomposing search space $\mathcal{T}$ into $\mathcal{T}' \oplus \mathcal{T}'$ such that $|\mathcal{T}'| = \sqrt{|\mathcal{T}|}$.

Knowing the fact that partial information about the secret key is veiled in smaller dimensional lattices $\mathcal{L}(\mathbf{B}_{h_0+h_1})$ and $\mathcal{L}(\mathbf{B}_{h_0-h_1})$ in the form of (f_0+f_1, g_0+g_1) and (f_0-f_1, g_0-g_1), the attacker can hope to search in possible smaller spaces. Let

$$\mathcal{F}_N(d_1,d_2,d_3,d_4) = \left\{ f \in \mathbb{Z}^N \middle| \begin{array}{l} f \text{ has } d_1 \text{ coefficients equal to } 1 \\ f \text{ has } d_2 \text{ coefficients equal to } -1 \\ f \text{ has } d_3 \text{ coefficients equal to } 2 \\ f \text{ has } d_4 \text{ coefficients equal to } -2 \\ \text{and other coefficients are } 0 \end{array} \right\}, \tag{21}$$

and $\mathcal{F}_N \subset \mathbb{Z}^N$ be the space of all N length sequences with coefficients from the set $\{0, \pm 1, \pm 2\}$.

According to assumptions 1, 2, for any $f = (f_0, f_1) \in \mathcal{T}$, the attacker can expect with high probability that $f_0 + f_1, f_0 - f_1 \in \mathcal{S} = \mathcal{F}_N(d_1, d_2, d_3, d_4)$ with

$$d_1 = d_2 = \frac{d_f(N-d_f)}{N},\ d_3 = d_4 = \frac{{d_f}^2}{4N}.$$

Therefore, a brute force search with cost $O(|\mathcal{S}|)$ can be performed over the search space $\mathcal{S}$ to find private vectors in the lattices $\mathcal{L}(\mathbf{B}_{h_0+h_1})$ and $\mathcal{L}(\mathbf{B}_{h_0-h_1})$, where

$$|\mathcal{S}| = \binom{N}{d_1}\binom{N-d_1}{d_1}\binom{N-2d_1}{d_3}\binom{N-2d_1-d_3}{d_3}. \tag{22}$$

Further, MITM attacks cost $\left(\frac{|\mathcal{S}|}{2N}\right)^{0.5}$ (classically) and $\left(\frac{|\mathcal{S}|}{2N}\right)^{0.25}$ (quantumly).

4.2 Cost of SVP algorithms

Before discussing Primal and Hybrid attacks, we briefly introduce the cost of lattice reduction by an algorithm like BKZ. BKZ with blocksize β produces a $BKZ-\beta$ reduced basis by calling the SVP oracle in smaller lattices of dimension β. Enumeration and sieving are the most studied and used SVP oracles in the literature. The called oracle heavily affects the memory and time requirements for running the BKZ algorithm. Enumeration algorithms [16, 42] solve the SVP with polynomial memory requirements and super-exponential time requirements while sieving algorithms [7, 36]

[¶] May [41] proposed an MITM attack on NTRU-type cryptosystems with a complexity $O(|\mathcal{T}|^{0.3})$(classic). However, it cannot be combined with hybrid attacks; therefore, we do not use it in our cost estimations.

have exponential time and memory requirements. Kirshanova et al. [34] found experimentally that sieving starts outperforming enumeration from dimension 65 onwards. The best records for solving the SVP are over the sieving regime; however, the memory consumption is extensive for these algorithms. To give a conservative parameter selection, we consider the model described in [3] that assumes that the sieving algorithm works in the RAM model, i.e., the attacker can access any amount of the memory for free. The classical asymptotic estimation of the number of the operations according to this model of sieving is $\sqrt{3/2}^{\beta+o(\beta)} \approx 2^{0.292\beta+o(\beta)}$ classically [7] that can be brought down to $\approx 2^{0.265\beta+o(\beta)}$ by employing Grover's search [36]. However, a thorough analysis of the quantum asymptotic estimation in [2] shows that the result may be far from practicality. It is clear that the reduction cost increases with β, and therefore, from the attacker's perspective, it is beneficial to perform the reduction in the two lattices $\mathcal{L}(\mathbf{B}_{h_0+h_1})$ and $\mathcal{L}(\mathbf{B}_{h_0-h_1})$. Considering assumptions 1, 2, we expect

$$\big\|(f_0+f_1, g_0+g_1)\big\| \approx \big\|(f_0-f_1, g_0-g_1)\big\| \approx \sqrt{2}\sqrt{d_f+d_g}. \tag{23}$$

Gaussian heuristic estimates the expected length of the shortest vector in lattices $\mathcal{L}(\mathbf{B}_{h_0+h_1})$ and $\mathcal{L}(\mathbf{B}_{h_0-h_1})$ to be

$$gh(\mathcal{L}(\mathbf{B}_{h_0+h_1})) = gh(\mathcal{L}(\mathbf{B}_{h_0-h_1})) = \sqrt{\frac{qN}{\pi e}}. \tag{24}$$

Since $d_f, d_g \leq 2N/3$, and $q = O(N)$, the ratios

$$\frac{\big\|(f_0+f_1, g_0+g_1)\big\|}{gh(\mathcal{L}(\mathbf{B}_{h_0+h_1})} \approx \frac{\big\|(f_0-f_1, g_0-g_1)\big\|}{gh(\mathcal{L}(\mathbf{B}_{h_0-h_1})} \approx O\left(\frac{1}{\sqrt{N}}\right). \tag{25}$$

Therefore, the vectors (f_0+f_1, g_0+g_1) and (f_0-f_1, g_0-g_1) and all their rotations are shortest vectors in the lattices $\mathcal{L}(\mathbf{B}_{h_0+h_1})$ and $\mathcal{L}(\mathbf{B}_{h_0-h_1})$, respectively, with a very high probability.

4.3 Primal attack

We follow the methodology of **Core-SVP and GSA** to parameterize the proposed cryptosystem. The Core-SVP [‖] is a conservative methodology of estimating the security that considers one call of the SVP oracle to be enough to solve the SVP. To estimate β according to this methodology, we model the behavior of BKZ according to the geometric series assumption (GSA, Definition 8), and depending on *2016-estimator* (Equation 7), we find the required blocksize β that is the input for the sieving or enumeration model. In our case, we consider the sieving regime for security estimation.

4.4 Hybrid attack

The MITM attack can be combined with the lattice reduction attack called the hybrid attack, introduced by Howgrave [26]. The basic idea is to reduce a $(r_2 - r_1)$ sized block

[‖] In NTRUPrime, the authors conclude that according to the submission to NIST standardization process, a cryptosystem achieves levels of security corresponding to AES-128, AES-192, and AES-256, if the classical (pre-quantum) Core-SVP model assign at least $2^{125}, 2^{181}$, and 2^{254}, respectively to the selected parameter sets.

$\mathbf{L}'$ of the matrix

$$\mathbf{B}_{\mathbf{H}'} = \left(\begin{array}{c|c} q\mathbf{I}_N & 0_N \\ \hline \mathbf{H}' & \mathbf{I}_N \end{array}\right) = \left(\begin{array}{c|c|c} q\mathbf{I}_{r_1} & 0 & 0 \\ \hline * & \mathbf{L}' & 0 \\ \hline * & * & \mathbf{I}_{2N-r_2} \end{array}\right). \tag{26}$$

In our case, $\mathbf{H}' = \mathbf{H}_0 + \mathbf{H}_1$ and $\mathbf{H}' = \mathbf{H}_0 - \mathbf{H}_1$. Let $\mathbf{U}'$ be an unimodular matrix such that $\mathbf{U}'\mathbf{L}'$ is reduced, and $\mathbf{Y}'$ be an orthogonal transformation such that $\mathbf{T}' = \mathbf{U}'\mathbf{L}'\mathbf{Y}'$ is a lower triangular matrix. Then, the lattice generated by the matrix

$$\mathbf{T} = \mathbf{U}\mathbf{B}_{\mathbf{H}'}\mathbf{Y} = \left(\begin{array}{c|c|c} \mathbf{I}_{r_1} & 0 & 0 \\ \hline 0 & \mathbf{U}' & 0 \\ \hline 0 & 0 & \mathbf{I}_{2N-r_2} \end{array}\right) \left(\begin{array}{c|c|c} q\mathbf{I}_{r_1} & 0 & 0 \\ \hline * & \mathbf{L}' & 0 \\ \hline * & * & \mathbf{I}_{2N-r_2} \end{array}\right) \left(\begin{array}{c|c|c} \mathbf{I}_{r_1} & 0 & 0 \\ \hline 0 & \mathbf{Y}' & 0 \\ \hline 0 & 0 & \mathbf{I}_{2N-r_2} \end{array}\right) \tag{27}$$

is isomorphic to the original lattice $\mathcal{L}(\mathbf{B}_{\mathbf{H}'})$. Therefore, $(g', f')\mathbf{Y}$ is a short vector in the resulting lattice, where $(g', f') = (g_0 + g_1, f_0 + f_1)$ for $\mathbf{H}' = \mathbf{H}_0 + \mathbf{H}_1$ and $(g', f') = (g_0 - g_1, f_0 - f_1)$ (or) for $\mathbf{H}' = \mathbf{H}_0 - \mathbf{H}_1$.

The diagonal entries of $\mathbf{T}$ are $\{q^{\alpha_0}, q^{\alpha_1}, \ldots, q^{\alpha_{2N-1}}\}$, where $\sum_{i=0}^{2N-1} \alpha_i = N$, $\alpha_i = 1$ for $i < r_1$, and $\alpha_i = 0$ for $i > r_2$. The matrix $\mathbf{L}'$ roughly obeys the geometric series assumption (GSA), and the rate of decrease of α_i can be estimated based on the Hermite root factor δ achieved by the lattice reduction algorithm. As calculated in [20]

$$\alpha_{r_1} = \frac{N - r_1}{r_2 - r_1} + (r_2 - r_1)\log_q(\delta),\ \alpha_{r_2} = \frac{N - r_1}{r_2 - r_1} - (r_2 - r_1)\log_q(\delta) \tag{28}$$

and α_i for $i \in [r_1, r_2]$ decrease almost linearly.

Let $K = 2N - r_2$; an attacker strives to balance the cost of combinatorial search over the K coordinates against the cost of the lattice reduction. An MITM search can be performed over the last K entries, and we assume that all collisions occur to have a conservative security estimation.

Let $\pi : \mathbb{Z}^N \to \mathbb{Z}^K$ be the projection onto the last K coordinates and

$$\mathcal{F}_\pi = \{\pi(v) : v \in \mathcal{S}\} \subset \mathbb{Z}^K. \tag{29}$$

Since $f' \in \mathcal{S}$, therefore, the projected component of f' appears in $\mathcal{F}_\pi$, and the search can be performed with $O(\sqrt{|\mathcal{F}_\pi|})$ time and memory consumption. However, estimating $|\mathcal{F}_\pi|$ is not straightforward. Let

$$\mathcal{F}_\pi(a_1, a_2, a_3, a_4) = \left\{ v \in \mathcal{S} \;\middle|\; \begin{array}{l} \pi(v) \text{ has } a_1 \text{ coefficients equal to } 1 \\ \pi(v) \text{ has } a_2 \text{ coefficients equal to } -1 \\ \pi(v) \text{ has } a_3 \text{ coefficients equal to } 2 \\ \pi(v) \text{ has } a_4 \text{ coefficients equal to } -2 \\ \text{and other coefficients are } 0 \end{array} \right\}, \tag{30}$$

and $P : \mathcal{F}_K \to \mathbb{R}$ be the probability mass function for the distribution induced on $\mathcal{F}_K$ by uniform and random sampling on $\mathcal{S}$ and projecting onto the last K coordinates. Then, the size of the search space $\mathcal{F}_\pi$ can be estimated as $2^{H(P)}$, where $H(P)$ is the Shannon entropy of P.

For every tuple (a_1, a_2, a_3, a_4), let us fix a representative $v_{(a_1,a_2,a_3,a_4)}$ of the set $\mathcal{F}_\pi(a_1, a_2, a_3, a_4)$. Since the space $\mathcal{S}$ is symmetric under coordinate permutations therefore $P(\pi(v)) = P(\pi(v_{(a_1,a_2,a_3,a_4)}))$ for all $v \in \mathcal{F}_\pi(a_1, a_2, a_3, a_4)$. The probability of

every representative is given by

$$P\left(\pi(v_{(a_1,a_2,a_3,a_4)}\right) = \frac{1}{K_{(a_1,a_2,a_3,a_4)}} \frac{|\mathcal{F}_\pi(a_1,a_2,a_3,a_4)|}{|\mathcal{S}|}, \tag{31}$$

where

$$K_{(a_1,a_2,a_3,a_4)} = \binom{K}{a_1}\binom{K-a_1}{a_2}\binom{K-a_1-a_2}{a_3}\binom{K-a_1-a_2-a_3}{a_4},$$

$$|\mathcal{F}_\pi(a_1,a_2,a_3,a_4)| = K_{(a_1,a_2,a_3,a_4)} \times \binom{N-K}{d_1-a_1}\binom{N-K-d_1+a_1}{d_1-a_2} \times \binom{N-K-2d_1+a_1+a_2}{d_3-a_3}$$
$$\binom{N-K-2d_1-d_3+a_1+a_2+a_3}{d_3-a_4}.$$

Thus, the entropy of P is

$$H(P) = -\sum_{v\in\mathcal{F}_K} P(v)\log_2 P(v) \; = -\sum_{v\in\mathcal{S}} P(\pi(v))\log_2 P(\pi(v))$$
$$= -\sum_{\substack{0\le a_1,a_2\le d_1\\ 0\le a_3,a_4\le d_3}} K_{(a_1,a_2,a_3,a_4)} P\left(\pi(v_{(a_1,a_2,a_3,a_4)})\right)\log_2 P\left(\pi(v_{(a_1,a_2,a_3,a_4)})\right).$$

Considering the rotations, the search space size can be further decreased by a factor of $2N$, and the log base 2 complexity of the hybrid MITM search is $\xi(H(P) - \log_2(2N))$ where $\xi = 0.5(0.25)$ classically (quantumly).

In order to resist the hybrid attack and achieve a security level equal to λ, for each fixed K we must have

$$\log_2(\text{hybrid attack cost}) \text{ or } \log_2(\text{lattice reduction cost}) \ge \lambda \tag{32}$$

where the root Hermite factor δ satisfies $\alpha_{r_2} \ge \log_q(4)$. Equivalently,

$$\log_2(\delta) \le \frac{N-r_1}{(2N-K-r_1)^2}\log_2 q - \frac{2}{2N-K-r_1}. \tag{33}$$

For hybrid attack cost estimation, we find an optimal K that balances and minimizes the maximum of both costs in 32.

4.5 Subfield attack

As mentioned earlier, the standard NTRU lattice does not include only the vector corresponding to the secret key (f, g) but also all the vectors corresponding to the rotations $(x^i * f, x^i * g)$ for $0 \le i \le n$ where n is the group order. All these rotations form a dense sublattice(i.e., lattice with many exceptionally short vectors). Finding a basis for this dense sublattice is called Dense Lattice Discovery (DSD). For large values of q, Kirchner and Fouque [33] observed that DSD happens before the event of finding a short vector in the lattice, and therefore, the SVP becomes easier to solve. Under this condition, the NTRU cryptosystem is called *overstretched*. A refined analysis by Ducas and Woerden [15] shows that an NTRU-like cryptosystem becomes overstretched when the value of q approximately exceeds $0.004 * n^{2.484}$ for $n > 100$. Similarly, for DiTRU, the associated lattice does not include only the

vector corresponding to the key (f_0, f_1, g_0, g_1), but also all the rotations of the form $(x^i * f_0, x^{-i} * f_1, x^i * g_0, x^{-i} * g_1)$ and $(x^i * f_1, x^{-i} * f_0, x^i * g_1, x^{-i} * g_0)$ for $0 \leq i \leq N$ where $n = 2N$ is the group order. Even after applying one layer of Gentry's attack, the two lattices $\mathcal{L}(\mathbf{B}_{\mathbf{h}_0+\mathbf{h}_1}), \mathcal{L}(\mathbf{B}_{\mathbf{h}_0-\mathbf{h}_1})$ contain dense sublattices corresponding to the images of the private key and its rotations according to the homomorphism defined in Figure 2. Consequently, the estimation [15] remains valid for N that defines the dihedral group D_N. Obviously, one can notice that the selected parameter sets in Table 3 are not in the *overstretched* regime. Therefore, the subfield attack is not applicable to our parameters.

4.6 Coppersmith and Shamir attack

We discuss the attack by Coppersmith and Shamir [12] on the first noncommutative version of NTRU over the dihedral group ring by Hoffstein and Silverman [23], and its inapplicability to DiTRU. The old cyptosystem is built over the subring $R_0 = \{\alpha \in \mathbb{Z}D_N : \alpha y = y\alpha\}$. The private key is (f, ω) where $f \in R_0$ with coefficients from the interval $\left(-\frac{q-1}{2}, \frac{q-1}{2}\right]$ and $\omega \in \mathbb{Z}D_N$ is an element with ternary coefficients. The public key is constructed as $h = pf * \omega * F \pmod q$, where $f * F = 1 \pmod q$. The ciphertext of any ternary message $m \in \mathbb{Z}D_N$ is a pair (e, E) computed as $e = \phi * h * \phi' + \psi (\mathrm{mod} q)$ and $E = \Psi * h + m (\mathrm{mod} q)$, where $\phi, \phi' \in R_0$, and $\psi \in \mathbb{Z}D_N$ are ternary elements with $\psi = \Psi \pmod p$. First, one can observe that any attack on this cryptosystem to recover the secret key from the public key or the message from the ciphertext is not applicable in the case of DiTRU or, in general, GR-NTRU, as the design of Hoffstein and Silverman's scheme, i.e., key generation and encryption-decryption, is entirely different from DiTRU.

Coppersmith and Shamir broke this cryptosystem using a subset $\{\alpha \in \mathbb{Z}D_N : \alpha y = -y\alpha\}$ and a linear map $\theta : \mathbb{Z}_q D_N \to \mathbb{Z}_q D_N$ that is identity on R_0 and maps R_1 to itself. An attacker tries to find an alternative ω' with small coefficients such that $\theta(h) = p\omega'$. Then, applying θ to e helps recover ψ and consequently, the message. An elaborate discussion on the Coppersmith attack regarding the construction of such a θ and finding ω' is provided in [50]. However, in the case of GR-NTRU, the ciphertext is given by $e = pr * h + m (\mathrm{mod} q)$, and any map θ that recovers r breaks the standard NTRU. Therefore, DiTRU, by design, is not vulnerable to this attack.

5 Parameter selection

According to NIST's definition of the level of security, we propose three parameter sets for levels 1,3 and 5. The parameters for DiTRU in Table 2 are selected according to the cost of the previous attacks with $\xi = 0.5(0.25)$ for the classical (quantum) cost of the meet-in-the-middle search and considering the maximum depth of the quantum circuit to be 2^{96} when one is performing the quantum search. For the sake of accurate comparison, we describe the parameter sets of NTRU in Table 3 that achieve the same level of security according to the same evaluation criteria followed for DiTRU.

6 Design rationale

We follow a design rationale similar to the one used in the NTRUEncrypt submission that designs the encryption scheme as a partially correct probabilistic public key scheme

Table 2: Core-SVP cost against DiTRU parameter sets

security level	(N, d, q)	Classical					Quantum				
		primal attack		hybrid attack			primal attack		hybrid attack		
		β	cost	K	β	cost	β	cost	K	β	cost
128	(541, 234, 2048)	445	**130**	164	524	153	445	**118**	155	545	144
192	(797, 530, 4096)	660	**193**	217	800	234	660	**175**	203	832	220
256	(1039, 478, 4096)	882	**258**	318	1057	309	882	**234**	300	1099	291

Table 3: Core-SVP cost against NTRU parameter sets

security level	(N, d, q)	Classical					Quantum				
		primal attack		hybrid attack			primal attack		hybrid attack		
		β	cost	K	β	cost	β	cost	K	β	cost
128	(587, 195, 2048)	456	133	166	438	**128**	456	121	156	454	**120**
192	(863, 159, 2048)	701	205	298	658	**192**	701	186	282	684	**181**
256	(1109, 369, 4096)	893	261	331	883	**258**	893	**237**	311	915	242

(PPKE). One can notice that the design of the PPKE in Figure 3 is identical to that used in standard NTRU, while the only difference is changing the underlying ring to the noncommutative group ring of the dihedral group. Therefore, similar to standard NTRU, the CPA security of the PPKE is based on the hardness of the NTRU assumption. We provide a CCA-2 secure implementation of the proposed PPKE scheme for DiTRU using the NAEP transformation [28] **. It can be converted into KEM following similar steps as given in [9, Algorithm 9,10].

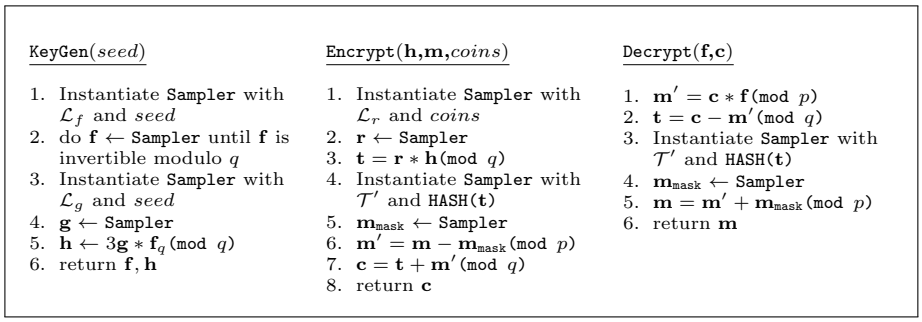


Fig. 3: A PPKE scheme for DiTRU

$*$: product over the group ring $\mathbb{Z}D_N$ modulo q and p.
Sampler : randomly samples an element unique to the seed from the input space.

$\mathcal{L}_f := \{1 + 3\mathbf{F} : \mathbf{F} \in \mathcal{T}_{2N}(d_f + 1, d_f)\},\ \mathcal{L}_g := \mathcal{T}_{2N}(d_g, d_g),\ \mathcal{L}_r := \mathcal{T}_{2N}(d_r, d_r),\ \mathcal{T}' := \mathcal{T}_{2N},$

where $d_g = \lfloor 2N/3 \rfloor$ and $d_r = d_f$. The decryption failure probability, according to the considered design criteria, is given by equation 17 with $\sigma^2 = 2\left(\frac{d_r d_g + d_f d_m}{N}\right)$ and $t = \frac{q-2}{2p}$.

**Our implementation is based on NTRU submissions to the first and third round of NIST competition with the required modifications to the dihedral group setup.

Table 4 records the memory requirements and the average cycle counts for the recommended parameter sets of DiTRU, while Table 5 compares the implementation costs for DiTRU vs. NTRU while encrypting/decrypting messages of the same length(for every level of security, the length of the polynomial corresponding to the message is the order of the dihedral group multiplied by the order of the cyclic group). The results are measured on a device with the same specification mentioned in subsection 3.2 on a single core, TurboBoost, and hyper-threading disabled. We compiled the code using GCC version 4:11.2.0-1ubuntu1 with **no** optimization flags enabled.

Table 4: Memory requirements and implementation cost for DiTRU parameters

DiTRU2048_541				**DiTRU4096_797**				**DiTRU4096_1039**			
size (in bytes)		**cpu cycles (ref)**		**size (in bytes)**		**cpu cycles (ref)**		**size (in bytes)**		**cpu cycles (ref)**	
sk:	217	**gen:**	83063049	**sk:**	319	**gen:**	178993352	**sk:**	416	**gen:**	301007142
pk:	1488	**enc:**	12653184	**pk:**	2391	**enc:**	26350252	**pk:**	3117	**enc:**	43440107
ct:	1488	**dec:**	23848004	**ct:**	2391	**dec:**	50365994	**ct:**	3117	**dec:**	83684000

Table 5: DiTRU vs. NTRU implementation cost (average CPU cycles)

Security level	**Key Generation**			**Encryption**			**Decryption**		
	DiTRU	**NTRU**	**ratio**	**DiTRU**	**NTRU**	**ratio**	**DiTRU**	**NTRU**	**ratio**
128	83063049	67573991	1.23	7477322227	4509252772	1.65	14088417108	8267862833	1.70
192	178993352	144771971	1.24	23400674192	13256355697	1.77	44737178607	24807462541	1.80
256	301007142	237913501	1.26	48268245887	27633654748	1.75	92916097739	52110865556	1.78

One can notice from Table 5, the efficiency of our inversion algorithm for DiTRU. Further, the cost of encryption/decryption is less than two times that of NTRU for equivalent levels of security, even though the underlying algebra is noncommutative for DiTRU.

7 Final remarks

This paper introduces DiTRU, a noncommutative analog of NTRU as GR-NTRU instantiated over the dihedral group. Our work focuses on clearing all the aspects that make the scheme practical. As a result, we provide a full-package cryptosystem accompanied by a detailed cryptanalysis. The security evaluation considers the one layer of Gentry's attack due to the algebraic structure of the dihedral group ring. Avoiding the one-layer of Gentry's attack means that lower values of N can achieve the same level of security, thereby improving the time and memory requirements of DiTRU. One way to achieve this could be twisting the multiplication of the group ring, resulting in a generalized form of group rings called twisted group rings. We briefly discuss the concept of twisted group rings.

Definition 11. *(2-cocycle) A map $\lambda : G \times G \to R^*$ is called a 2-cocycle if satisfies $\lambda(1,1) = 1$ and $\lambda(g_1g_2, g_3)\lambda(g_1, g_2) = \lambda(g_1, g_2g_3)\lambda(g_2, g_3)$ for all $g_1, g_2, g_3 \in G$.*

Definition 12. *(Twisted group ring) A twisted group ring $R^\lambda G$ of group G over the ring R corresponding to the 2-cocycle λ is same as the group ring RG as in definition 4 but with a twisted multiplication given by*

$$\sum_{i=1}^{n} \alpha_i g_i * \sum_{i=1}^{n} \beta_i g_i = \sum_{i=1}^{n} \left(\sum_{g_h g_k = g_i} \alpha_h \beta_k \lambda(g_h, g_k) \right) g_i. \tag{34}$$

If we twist the dihedral group ring $\mathbb{Z}_q D_N$ with the 2-cocycle $\lambda : D_N \times D_N \to \mathbb{Z}_q^*$ defined as

$$\lambda(y^k x^i, y^l x^j) = \begin{cases} -1, & \text{for } \ i,j \in \{0,1,\ldots,N-1\} \text{ and } k = l = 1 \\ 1, & \text{otherwise} \end{cases}$$

then, the elements in the twisted group ring $\mathbb{Z}_q^\lambda D_N$ have the matrix representation of the form

$$\mathbf{H} = \begin{pmatrix} \mathbf{H}_0 & \mathbf{H}_1 \\ -\mathbf{H}_1 & \mathbf{H}_0 \end{pmatrix}. \tag{35}$$

To our understanding, it is not possible to reduce $\mathbf{H}$ into integral matrices of smaller dimensions such that the corresponding lattices contain short vectors carrying partial information about the secret key. One homomorphism that we can think of is $\mathbf{H} \to \mathbf{H}_0 \pm i\mathbf{H}_1$ where $i = \sqrt{-1}$, but then the smaller matrices have complex entries, and to apply lattice reduction algorithms, one again needs to map these complex matrices to larger dimensional real matrices. Moreover, the matrix representation of elements in the ring $\mathbb{Z}_q[x]/(x^N + 1)$ is also the same as for $\mathbf{H}$. Therefore, if one can reduce $\mathbf{H}$ into the desired form, then possibly the same reduction can be applied in the case of $\mathbb{Z}_q[x]/(x^N+1)$. However, it is known that the polynomial x^N+1 does not factor over $\mathbb{Z}_q$ into smaller degree polynomials with small norm [5]. This is why the ring $\mathbb{Z}_q[x]/(x^N+1)$ is used in some cryptographic designs like [14, 17]. Although we have selected our parameters considering one layer of Gentry's attack, twisting the underlying algebra can prevent the dimension reduction. This idea seems promising but needs a rigorous analysis and thus is left as future work.

References

1. Agrawal, S., Pellet-Mary, A.: Indistinguishability obfuscation without maps: Attacks and fixes for noisy linear FE. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 110–140. Springer (2020). https://doi.org/10.1007/978-3-030-45721-1_5
2. Albrecht, M.R., Gheorghiu, V., Postlethwaite, E.W., Schanck, J.M.: Estimating quantum speedups for lattice sieves. In: Moriai, S., Wang, H. (eds.) Advances in Cryptology – ASIACRYPT 2020. pp. 583–613. Springer International Publishing, Cham (2020). https://doi.org/10.1007/978-3-030-64834-3_20
3. Alkim, E., Ducas, L., Pöppelmann, T., Schwabe, P.: Post-quantum key {Exchange—A} new hope. In: 25th USENIX Security Symposium (USENIX Security 16). pp. 327–343 (2016)
4. Aono, Y., Wang, Y., Hayashi, T., Takagi, T.: Improved progressive BKZ algorithms and their precise cost estimation by sharp simulator. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 789–819. Springer (2016). https://doi.org/10.1007/978-3-662-49890-3_30
5. Avanzi, R., Bos, J., Ducas, L., Kiltz, E., Lepoint, T., Lyubashevsky, V., Schanck, J.M., Schwabe, P., Seiler, G., Stehlé, D.: CRYSTALS-Kyber algorithm specifications and supporting documentation. NIST PQC Round (2020), `https://csrc.nist.gov/Projects/post-quantum-cryptography/post-quantum-cryptography-standardization/round-3-submissions`
6. Bagheri, K., Sadeghi, M.R., Panario, D.: A Non-commutative Cryptosystem Based on Quaternion Algebras. Designs, Codes and Cryptography **86** (10 2018). https://doi.org/10.1007/s10623-017-0451-4

7. Becker, A., Ducas, L., Gama, N., Laarhoven, T.: New directions in nearest neighbor searching with applications to lattice sieving. In: Proceedings of the twenty-seventh annual ACM-SIAM symposium on Discrete algorithms. pp. 10–24. SIAM (2016)
8. Chen, C., Danba, O., Hoffstein, J., Hülsing, A., Rijneveld, J., Schanck, J.M., Schwabe, P., Whyte, W., Zhang, Z.: NTRU: Algorithm specifications and supporting documentation. NIST (2020), https://csrc.nist.gov/Projects/post-quantum-cryptography/post-quantum-cryptography-standardization/round-3-submissions
9. Chen, C., Hoffstein, J., Whyte, W., Zhang, Z.: NIST PQ Submission: NTRUEncrypt A lattice based encryption algorithm. NIST (2017)
10. Chen, Y.: Réduction de réseau et sécurité concrete du chiffrement completement homomorphe (Ph. D. thesis) (2013)
11. Chen, Y., Nguyen, P.Q.: BKZ 2.0: Better lattice security estimates. In: International Conference on the Theory and Application of Cryptology and Information Security. pp. 1–20. Springer (2011). https://doi.org/10.1007/978-3-642-25385-0_1
12. Coppersmith, D.: Attacking non-commutative NTRU. Tech. rep., Technical report, IBM research report, April 1997. Report (2006), https://dominoweb.draco.res.ibm.com/d102d0885e971b558525659300727a26.html
13. Coppersmith, D., Shamir, A.: Lattice Attacks on NTRU. In: Advances in Cryptology — EUROCRYPT '97. pp. 52–61. Springer Berlin Heidelberg, Berlin, Heidelberg (1997). https://doi.org/10.1007/3-540-69053-0_5
14. Ducas, L., Durmus, A., Lepoint, T., Lyubashevsky, V.: Lattice Signatures and Bimodal Gaussians. In: Advances in Cryptology – CRYPTO 2013. pp. 40–56. Springer Berlin Heidelberg, Berlin, Heidelberg (2013). https://doi.org/10.1007/978-3-642-40041-4_3
15. Ducas, L., van Woerden, W.: NTRU fatigue: how stretched is overstretched? In: Advances in Cryptology–ASIACRYPT 2021: 27th International Conference on the Theory and Application of Cryptology and Information Security, Singapore, December 6–10, 2021, Proceedings, Part IV 27. pp. 3–32. Springer (2021). https://doi.org/10.1007/978-3-030-92068-5_1
16. Fincke, U., Pohst, M.: Improved methods for calculating vectors of short length in a lattice, including a complexity analysis. Mathematics of computation **44**(170), 463–471 (1985)
17. Fouque, P.A., Hoffstein, J., Kirchner, P., Lyubashevsky, V., Pornin, T., Prest, T., Ricosset, T., Seiler, G., Whyte, W., Zhan, Z.: FALCON: Fast-fourier lattice-based compact signatures over NTRU. Tech. rep. (2018), https://www.di.ens.fr/~prest/Publications/falcon.pdf
18. Gentry, C.: Key recovery and message attacks on NTRU-composite. In: Pfitzmann, B. (ed.) Advances in Cryptology — EUROCRYPT 2001. pp. 182–194. Springer Berlin Heidelberg, Berlin, Heidelberg (2001), https://doi.org/10.1007/3-540-44987-6_12
19. Hirschhorn, P.S., Hoffstein, J., Howgrave-Graham, N., Whyte, W.: Choosing NTRUEncrypt parameters in light of combined lattice reduction and MITM approaches. In: Applied Cryptography and Network Security: 7th International Conference, ACNS 2009, Paris-Rocquencourt, France, June 2-5, 2009. Proceedings 7. pp. 437–455. Springer (2009)
20. Hoffstein, J., Pipher, J., Schanck, J.M., Silverman, J.H., Whyte, W., Zhang, Z.: Choosing Parameters for NTRUEncrypt. In: Topics in Cryptology – CT-RSA 2017. pp. 3–18. Springer International Publishing, Cham (2017)
21. Hoffstein, J., Pipher, J., Silverman, J.: An Introduction to Mathematical Cryptography. Springer Publishing Company, Incorporated, NY, 1 edn. (2008)

22. Hoffstein, J., Pipher, J., Silverman, J.H.: NTRU: A ring-based public key cryptosystem. In: International algorithmic number theory symposium. pp. 267–288. Springer, Berlin, Heidelberg (1998). https://doi.org/10.1007/BFb0054868
23. Hoffstein, J., Silverman, J.: A non-commutative version of the NTRU public key cryptosystem. It was for a while available at (1997), http://www.tiac.net/users/ntru/NTRUFTP.html
24. Hoffstein, J., Silverman, J.H.: A non-commutative version of the NTRU public key cryptosystem. unpublished paper, February (1997)
25. Hoffstein, J., Silverman, J.H., Whyte, W.: Meet-in-the-middle attack on an NTRU private key. Tech. rep., Technical report, NTRU Cryptosystems, July 2006. Report (2006)
26. Howgrave-Graham, N.: A hybrid lattice-reduction and meet-in-the-middle attack against NTRU. In: Menezes, A. (ed.) Advances in Cryptology - CRYPTO 2007. pp. 150–169. Springer Berlin Heidelberg, Berlin, Heidelberg (2007). https://doi.org/10.1007/978-3-540-74143-5_9
27. Howgrave-Graham, N., Silverman, J.H., Whyte, W.: A meet-in-the-middle attack on an NTRU private key. NTRU cryptosystem technical report #004. (2003), https://www.securityinnovation.com/uploads/Crypto/NTRUTech004v2.pdf
28. Howgrave-Graham, N., Silverman, J.H., Whyte, W.: Choosing parameter sets for ntruencrypt with naep and sves-3. In: Menezes, A. (ed.) Topics in Cryptology – CT-RSA 2005. pp. 118–135 (2005). https://doi.org/10.1007/978-3-540-30574-3_10
29. Hurley, T.: Group rings and rings of matrices. International Journal of Pure and Applied Mathematics **31**, 319–335 (01 2006), https://www.researchgate.net/publication/228928727_Group_rings_and_rings_of_matrices
30. Jarvis, K., Nevins, M.: ETRU: NTRU over the eisenstein integers. Designs, Codes and Cryptography **74**(1), 219–242 (2015). https://doi.org/10.1007/s10623-013-9850-3
31. Karbasi, A.H., Atani, S.E., Atani, R.E.: PairTRU: Pairwise Non-commutative Extension of the NTRU public key cryptosystem. International Journal of Information Security Science **8**, 1–10 (03 2018)
32. Kim, J., Lee, C.: A polynomial time algorithm for breaking NTRU encryption with multiple keys. Designs, Codes and Cryptography pp. 1–11 (2023). https://doi.org/10.1007/s10623-023-01233-5
33. Kirchner, P., Fouque, P.A.: Revisiting lattice attacks on overstretched NTRU parameters. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 3–26. Springer (2017). https://doi.org/10.1007/978-3-319-56620-7_1
34. Kirshanova, E., May, A., Nowakowski, J.: New NTRU records with improved lattice bases. In: Johansson, T., Smith-Tone, D. (eds.) Post-Quantum Cryptography. pp. 167–195. Springer Nature Switzerland, Cham (2023)
35. Kumar, V., Raya, A., Gangopadhyay, S., Gangopadhyay, A.K.: Lattice attack on group ring NTRU: The case of the dihedral group (2023), https://doi.org/10.48550/arXiv.2309.08304
36. Laarhoven, T.: Search problems in cryptography: from fingerprinting to lattice sieving. Phd thesis, Eindhoven University of Technology (2015), available at https://research.tue.nl/en/publications/search-problems-in-cryptography-from-fingerprinting-to-lattice-si
37. Lenstra, A.K., Lenstra, H.W., Lovász, L.: Factoring polynomials with rational coefficients. Mathematische annalen **261**(ARTICLE), 515–534 (1982). https://doi.org/10.1007/BF01457454

38. Ling, C., Mendelsohn, A.: Ntru in quaternion algebras of bounded discriminant. In: Johansson, T., Smith-Tone, D. (eds.) Post-Quantum Cryptography. pp. 256–290. Springer Nature Switzerland, Cham (2023). https://doi.org/10.1007/978-3-031-40003-2_10
39. Makhijani, N., Sharma, R., Srivastava, J.: Units in finite dihedral and quaternion group algebras. Journal of the Egyptian Mathematical Society **24**(1), 5–7 (2016). https://doi.org/https://doi.org/10.1016/j.joems.2014.08.001
40. Malekian, E., Zakerolhosseini, A., Mashatan, A.: QTRU : a lattice attack resistant version of NTRU PKCS based on quaternion algebra. IACR Cryptology ePrint Archive **2009** (2009), https://eprint.iacr.org/2009/386
41. May, A.: How to meet ternary LWE keys. In: Malkin, T., Peikert, C. (eds.) Advances in Cryptology – CRYPTO 2021. pp. 701–731. Springer International Publishing, Cham (2021). https://doi.org/10.1007/978-3-030-84245-1_24
42. Micciancio, D., Walter, M.: Fast lattice point enumeration with minimal overhead. In: Proceedings of the twenty-sixth annual ACM-SIAM symposium on Discrete algorithms. pp. 276–294. SIAM (2014)
43. Miyata, T.: On the units of the integral group ring of a dihedral group. Journal of the Mathematical Society Japan **32**(4) (1980)
44. Peikert, C.: A decade of lattice cryptography. Foundations and trends® in theoretical computer science **10**(4), 283–424 (2016)
45. Schnorr, C.P.: A hierarchy of polynomial time lattice basis reduction algorithms. Theoretical computer science **53**(2-3), 201–224 (1987). https://doi.org/10.1016/0304-3975(87)90064-8
46. Silverman, J.H.: Almost Inverses and Fast NTRU Key Creation. NTRU Cryptosystems Technical Report #14 (1999)
47. Singh, S., Padhye, S.: Cryptanalysis of NTRU with n public keys. In: 2017 ISEA Asia Security and Privacy (ISEASP). pp. 1–6 (2017). https://doi.org/10.1109/ISEASP.2017.7976980
48. Development team, T.F.: fplll, a lattice reduction library, Version: 5.4.4 (2023), available at https://github.com/fplll/fplll
49. Development team, T.F.: fpylll, a Python wraper for the fplll lattice reduction library, Version: 0.5.9 (2023), available at https://github.com/fplll/fpylll
50. Truman, K.R.: Analysis and Extension of Non-Commutative NTRU. PhD dissertation, University of Maryland (2007), https://drum.lib.umd.edu/handle/1903/7344
51. Van Hoof, I., Kirshanova, E., May, A.: Quantum key search for ternary lwe. In: Cheon, J.H., Tillich, J.P. (eds.) Post-Quantum Cryptography. pp. 117–132. Springer International Publishing, Cham (2021). https://doi.org/10.1007/978-3-030-81293-5_7
52. Vats, N.: NNRU, a noncommutative analogue of NTRU (2009), https://arxiv.org/abs/0902.1891
53. Working Group of the C/MM Committee and others: IEEE P1363.1 Standard Specification for Public-Key Cryptographic Techniques Based on Hard Problems over Lattices (2009)
54. Yasuda, T., Dahan, X., Sakurai, K.: Characterizing NTRU-variants using group ring and evaluating their lattice security. IACR Cryptol. ePrint Arch. p. 1170 (2015), http://eprint.iacr.org/2015/1170