Almost NTRU: Revisiting Noncommutativity Against Lattice Attacks

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

[1] Department of Computer Science and Engineering,
Indian Institute of Technology Roorkee, Haridwar 247667, India
{ali_r,sugata.gangopadhyay}@cs.iitr.ac.in
[2] Department of Computer Engineering,
Gachon University, Seongnam 13307, South Korea
{vikaskumar250697,sohwang}@gachon.ac.kr

Abstract. NTRU is one of the most extensively studied lattice-based cryptographic schemes and is widely regarded as a strong candidate for post-quantum security. The most effective attacks on NTRU are lattice-based or lattice-related, which naturally guide the choice of parameters to achieve the desired security levels. In 1997, Hoffstein and Silverman proposed a variant of NTRU based on a noncommutative algebraic structure, claiming that it would mitigate lattice attacks. However, their scheme was later shown to be vulnerable to an algebraic attack by Coppersmith. Although several subsequent attempts have been made in the literature to develop noncommutative variants of NTRU, most of these designs have either been shown to be vulnerable to algebraic attacks or have failed to directly address lattice-based attacks.
In this work, we revisit the problem of constructing a noncommutative analog of NTRU that offers stronger resistance against direct lattice attacks. Firstly, we conceptualize the problem by introducing an almost unstructured variant, and then refine this idea towards a more compact instantiation by introducing a semi-structured variant. Our proposal may be viewed as a follow-up to the early noncommutative construction of Hoffstein and Silverman.
We further discuss the requirements towards having a fully structured construction in this paradigm.

Keywords: Post-Quantum · NTRU · Lattice attacks · Noncommutative

1 Introduction

NTRU is one of the earliest proposals for a post-quantum cryptographic scheme, conjectured to be secure based on hard mathematical problems related to lattices. The original construction of NTRU was introduced by Hoffstein, Pipher, and Silverman in 1997, based on a commutative ring of truncated polynomials. Although the original NTRU assumption was not initially grounded in formal security proofs, its credibility has been reinforced over nearly three decades of

cryptanalysis. This record of cryptanalysis has led to the development of versatile cryptographic primitives and advanced constructions whose security is based on the NTRU assumption. The flexibility in designing NTRU-like schemes has led to numerous variants defined over different algebraic rings, motivated by both security considerations and performance improvements. The original formulation of NTRU was introduced over the truncated polynomial ring, where the task of recovering the private key is reduced to finding a short vector (corresponding to the private key) in the lattice associated with the construction. Since then, the most effective attacks have consistently been based on lattice techniques, either in isolation or in combination with other methods. Naturally, lattice attacks have significantly influenced the parameter selection of NTRU-like constructions. With advances in cryptanalytic techniques and lattice reduction algorithms, parameters have been re-evaluated to require larger dimensions, which in turn lead to increased ciphertext sizes, larger public keys, and higher computational costs. Although the idea of employing noncommutative variants of NTRU to mitigate the impact of lattice attacks was already hinted at in the early cryptanalysis of Coppersmith and Shamir [9], to the best of our knowledge, only one concrete attempt—by Hoffstein and Silverman [22], which ultimately proved unsuccessful—has explored a noncommutative construction as a means of resistance. Other works considering noncommutative settings have merely instantiated NTRU over alternative rings, which do not fundamentally prevent lattice attacks and, in some cases, yield weaker instances. Hence, in our view, it is worthwhile to revisit the idea of designing a genuinely noncommutative NTRU construction—one that remains faithful to the spirit of NTRU while offering a promising path towards more compact schemes that either resist direct lattice attacks or mitigate their impact on parameter selection.

Many Variants. Following the original proposal of NTRU [21], numerous variants have been developed by instantiating the NTRU assumption over different algebraic structures. Most of the prominent designs have been defined over commutative algebraic rings. Among them, NTRU-HPS and NTRU-HRSS [27], submitted jointly under the name NTRU, advanced to the third round of the NIST standardization process as finalist candidates. In parallel, NTRU Prime [5] reached the same round as an alternate candidate. It is worth noting, however, that these constructions were ultimately outperformed by the selected standard Kyber [43], largely because the NTRU-based submissions were not designed over Number Theoretic Transform (NTT)-friendly rings, limiting their implementation efficiency. To address this limitation, NTTRU [36] was later introduced by Lyubashevsky and Seiler as an efficient Key Encapsulation Mechanism (KEM) based on an NTT-friendly cyclotomic ring, achieving better performance compared to the NTRU KEMs submitted to NIST. Building on this idea, NTRU+ was proposed and submitted to the Korean Post-Quantum Cryptography competition (KpqC). In 2025, NTRU+ [30] was selected for standardization by the KpqC as an efficient analog of NTRU.

On the other hand, the literature has explored a line of constructions based on noncommutative algebra, aiming to strengthen the security of NTRU against al-

gebraic attacks or to increase its overall resilience. The first such work, proposed by Hoffstein and Silverman [22], introduced the use of noncommutativity as a means to avoid lattice-based attacks. However, the design deviated significantly from the original NTRU structure in its encryption and decryption procedures, which eventually led to a successful cryptanalysis [8,46] that rendered the scheme insecure. Following this, several proposals [41,40,33,37,3,48] introduced NTRU over various noncommutative structures. Nevertheless, none of these constructions managed to fully resist direct lattice attacks, resulting in performance and security levels comparable to standard NTRU defined over commutative rings. Moreover, some proposals relied on highly structured rings, which in turn enabled even stronger lattice-based attacks, such as the one demonstrated by Raya et al. [39].

Context. The original NTRU cryptosystem was introduced as "a mixing system based on polynomial algebra and reduction modulo." During encryption, the message is mixed with a polynomial from the ring in such a way that only the party possessing the trapdoor associated with the public key of the cryptosystem can later *unmix* it. The public key equation of NTRU over the commutative truncated polynomial ring is given as

$$h = f^{-1} * g \pmod{q}, \tag{1}$$

where both f and g are chosen with small Euclidean norms and form the trapdoor associated with the public key.

On the other hand, the key equation of our construction, Almost NTRU (ANTRU), can be viewed as a rearrangement of the standard NTRU key equation, but formulated over a noncommutative ring rather than a commutative one, as in most conventional NTRU constructions.

In a commutative ring, decomposing f as $f = f_1 * f_2 \pmod{q}$ and rewriting the public key equation as

$$h = f_1^{-1} * g * f_2^{-1} \pmod{q}$$

has no effect, since multiplication commutes. However, in the noncommutative setting, this no longer holds:

$$h' = f_1^{-1} * g * f_2^{-1} \pmod{q} \neq h.$$

Consequently, the key equation of ANTRU is formulated in terms of h', where f_1 and f_2 are restricted to a commutative subring (to enable decryption), while g is sampled from the noncommutative ring (to resist direct lattice attacks). It is useful to view ANTRU as a two-layer NTRU problem: $f_1^{-1} * g' \pmod{q}$, where g' itself is defined as another NTRU instance, $g' = g * f_2^{-1} \pmod{q}$.

The idea of introducing a two-layer cryptographic hardness assumption has appeared previously, for example, in NTWE [18], where the authors modify the NTRU key generation equation $f^{-1} * g \pmod{q}$ by letting g be an LWE or Module-LWE sample. Interestingly, this resulted in more compact parameter sets than those of pure Module-LWE constructions. In contrast, our construction of ANTRU employs two layers of NTRU instances — a phenomenon that cannot arise in the commutative setting.

Our Work

- **Conceptualization of ANTRU:** Different from previous NTRU variants defined over noncommutative algebra, this work introduces a framework for constructing a noncommutative analog of NTRU in which direct lattice attacks alone appear ineffective for both key-recovery and message-recovery. We formulate the decision and search versions of the *Almost NTRU* problem (abbreviated as ANTRU) and describe the general attack landscape, supported by lower-bound security guarantees inherited from the NTRU assumption for the key generation. More specifically, we present two illustrative instantiations: ① Almost-unstructured, ② Semi-structured, and discuss the requirements of introducing a fully structured ANTRU instance.
- **Parameter Selection and Comparison:** We concretely select parameter sets for the almost unstructured variant and the semi-structured variant and compare them to those of matrix NTRU at equivalent levels of security.

1.1 Organization

The remainder of this paper is structured as follows. Section 2 establishes the necessary background on lattices, NTRU, and group rings. Section 3 introduces our almost unstructured NTRU (ANTRU). We then analyze the security of the proposed construction in Section 4 and discuss the parameter selection in Section 5. Finally, Section 6.1 concludes the paper with a proposal for a semi-structured variant and leaves an open question for fully structuring it for future research. Supplementary details are provided in the Appendices.

2 Preliminaries

2.1 Notations

We use '$*$' to denote multiplication between two elements in their underlying algebraic ring. The sets $\mathbb{Z}$ and $\mathbb{R}$ denote the integers and real numbers, respectively. Let $\mathbb{Z}_q$ denote the ring of integers modulo q, and let $\mathbb{Z}_q^\times$ be its group of units. The symbol $\mathcal{R}$ denotes an arbitrary ring or otherwise explicitly defined, and $M_n(\mathcal{R})$ denotes the set of all $n \times n$ matrices with entries from $\mathcal{R}$. Vectors are represented by bold lowercase letters, while matrices are denoted by bold uppercase letters. For a given matrix $\mathbf{X}$, we use the following notations: $\mathbf{X}^{(i,j)}$ denotes the element at row i and column j; $\mathbf{X}^{(i,:)}$ denotes the i-th row; $\mathbf{X}^{(:,j)}$ denotes the j-th column; $\mathbf{X}^{Tr}$ denotes the transpose of $\mathbf{X}$; and $\overrightarrow{\mathbf{X}}$ denotes the vectorization of $\mathbf{X}$ as a single column vector, obtained by stacking the rows of $\mathbf{X}$ from top to bottom. For a polynomial $v(x) = v_0 + v_1 x + \ldots + v_{n-1}x^{n-1}$ with coefficients $v_i \in \mathbb{Z}$, we define the vector $\mathbf{v} = (v_0, v_1, \ldots, v_{n-1}) \in \mathbb{Z}^n$ as the coefficient vector of the polynomial. The Euclidean norm of $\mathbf{v}$ is defined as $\|\mathbf{v}\| = \sqrt{\sum_{i=0}^{n-1} v_i^2}$, the supremum norm is defined as $\|v\|_\infty = \max_i |v_i|$, and the inner product between two vectors $\mathbf{u} = (u_0, u_1, \ldots, u_{n-1})$ and $\mathbf{v}$ is given by $\langle \mathbf{u}, \mathbf{v} \rangle = \sum_{i=0}^{n-1} u_i v_i$. For

$m \geq 1$, $\mathbf{I}_m$ and $\mathbf{0}_m$ denote identity and zero matrices of order m, respectively, and 0_m represents the zero vector of length m. Suppose D is a distribution on a set S, then $s \leftarrow D$ denotes that $s \in S$ is sampled according to the distribution D. The notation $U(S)$ represents a uniform distribution over a finite set S. For uniform distribution, we simply use $s \leftarrow_{\$} S$ to denote that s is sampled uniformly at random from S. $\mathcal{N}(\mu, \sigma^2)$ denotes the normal distribution with mean μ and variance σ^2. $O(\cdot), o(\cdot)$, $\Theta(\cdot)$, and $\Omega(n)$ are usual Big-O, small-o, Big-Theta, and Big-Omega notations, respectively.

2.2 Lattices

Definition 1 (Lattice). *Let* $\mathbf{B} = \left(\mathbf{b}_0 | \mathbf{b}_1 | \ldots | \mathbf{b}_{n-1}\right) \in \mathbb{Z}^{n \times d}$ *be a matrix whose rows* $\mathbf{b}_i \in \mathbb{Z}^d$ *are linearly independent. The lattice generated by* $\mathbf{B}$ *is defined as*

$$\mathcal{L}(\mathbf{B}) = \{\mathbf{x}\mathbf{B} \mid \mathbf{x} \in \mathbb{Z}^n\},$$

i.e., the set of all integer linear combinations of the rows of $\mathbf{B}$*.*

Here, n is the rank of the lattice, and d is its dimension. The lattice is called full-rank if $n = d$, in which case the volume of the lattice is given by $\mathsf{vol}(\mathcal{L}(\mathbf{B})) = |\det(\mathbf{B})|$. In this work, we consider only full-rank lattices.

Definition 2 (Shortest Vector Problem (SVP)). *Given a full-rank lattice* $\mathcal{L}(\mathbf{B})$ *generated by the basis* $\mathbf{B}$*, the* Shortest Vector Problem (SVP) *asks to find a non-zero vector* $\mathbf{v} \in \mathcal{L}(\mathbf{B}) \setminus \{\mathbf{0}\}$ *such that* $\|\mathbf{v}\| = \min_{\substack{\mathbf{w} \in \mathcal{L}(\mathbf{B}) \\ \mathbf{w} \neq \mathbf{0}}} \|\mathbf{w}\|$*. The minimum norm of a non-zero lattice vector is called the first successive minimum and denoted by* $\lambda_1(\mathcal{L}(\mathbf{B}))$*.*

Definition 3 (Closest Vector Problem (CVP)). *Given a full-rank lattice* $\mathcal{L}(\mathbf{B})$ *of dimension* n*, and a vector* $\mathbf{t} \in \mathbb{R}^n$ *(does not lie in the lattice), the* Closest Vector Problem (CVP) *asks to find* $\mathbf{v} \in \mathcal{L}(\mathbf{B})$ *which is the closest vector to* $\mathbf{t}$ *i.e.,* $\|\mathbf{v} - \mathbf{t}\| \leq \|\mathbf{w} - \mathbf{t}\|$ *for all* $\mathbf{w} \in \mathcal{L}(\mathbf{B})$*.*

The SVP and the CVP are computationally hard for certain types of bases, referred to as bad bases. This hardness forms the foundational assumption for the security of many cryptosystems in lattice-based cryptography. Furthermore, using *Kannan's embedding* [29], one can reduce a CVP instance in dimension n with respect to a target vector $\mathbf{t} \in \mathbb{R}^n$ to an SVP instance in dimension $n + 1$ by constructing the following lattice basis:

$$\mathbf{B}' = \begin{pmatrix} \mathbf{B} & \mathbf{0} \\ \mathbf{t} & u \end{pmatrix},$$

where u is the embedding factor, typically chosen as $u = 1$.

Definition 4 (Gaussian Heuristic). *The expected norm of the shortest non-zero vector* $\lambda_1(\mathcal{L}(\mathbf{B}))$ *in an* n*-dimensional lattice, according to the Gaussian heuristic, is denoted by* $\mathsf{gh}(\mathcal{L}(\mathbf{B}))$ *and is given by*

$$\mathsf{gh}(\mathcal{L}(\mathbf{B})) = \sqrt{\frac{n}{2\pi e}} \cdot \mathsf{vol}(\mathcal{L}(\mathbf{B}))^{1/n} \tag{2}$$

2.3 Group Ring

Definition 5 (Group Ring). *For a group $\mathcal{G} = \{g_0, g_1, \ldots, g_{n-1}\}$ of finite order n and a ring $\mathcal{R}$, the group ring $\mathcal{R}[\mathcal{G}]$ is defined as the set of all formal sums:*

$$\mathcal{R}[\mathcal{G}] = \left\{ a = \sum_{i=0}^{n-1} \alpha_{g_i} g_i \ : \ \alpha_{g_i} \in \mathcal{R} \right\}. \tag{3}$$

Let $x = \sum_{i=0}^{n-1} \alpha_{g_i} g_i,\ y = \sum_{i=0}^{n-1} \beta_{g_i} g_i$ be two elements in $\mathcal{R}[\mathcal{G}]$. Then, $\mathcal{R}[\mathcal{G}]$ forms a ring under the following operations:

- **Addition:** $x + y = \sum_{i=0}^{n-1} (\alpha_{g_i} + \beta_{g_i}) g_i$
- **Multiplication:** $x * y = \sum_{k=0}^{n-1} \gamma_{g_k} g_k,$ where $\gamma_{g_k} = \sum_{g_i g_j = g_k} \alpha_{g_i} \beta_{g_j}$.

If $\mathcal{R}$ has a multiplicative identity $1_{\mathcal{R}}$, then the group ring $\mathcal{R}[\mathcal{G}]$ is a ring with unity given by

- **Unity:** $1_{\mathcal{R}[\mathcal{G}]} = \sum_{g \in \mathcal{G}} a_g g, \quad$ where $a_e = 1_{\mathcal{R}}$ and $a_g = 0$ for $g \neq e$,

with e denoting the identity element of the group $\mathcal{G}$. Scalar multiplication can be defined for $\lambda \in \mathcal{R}$ and an element $x = \sum_{i=0}^{n-1} \alpha_{g_i} g_i \in \mathcal{R}[\mathcal{G}]$ as

- **Scalar multiplication:** $\lambda x = \lambda \left(\sum_{i=0}^{n-1} \alpha_{g_i} g_i \right) = \sum_{i=0}^{n-1} (\lambda \alpha_{g_i}) g_i$.

Additionally, if $\mathcal{R}$ is commutative, then $\mathcal{R}[\mathcal{G}]$ is naturally an $\mathcal{R}$-algebra. We can interchangeably use the vector form to represent an element $x \in \mathcal{R}[\mathcal{G}]$ as

- **Coefficient vector:** $\mathbf{x} = (\alpha_{g_0}, \alpha_{g_1}, \ldots, \alpha_{g_{n-1}}) \in \mathcal{R}^n$,

where the coefficient α_{g_i} corresponds to the group element $g_i \in \mathcal{G}$.

Matrix Representation: In [28], the author defines an isomorphism between the group ring $\mathcal{R}[\mathcal{G}]$ and a certain subring of $n \times n$ matrices over $\mathcal{R}$. The map τ sends an element $x \in \mathcal{R}[\mathcal{G}]$ to its corresponding matrix $\mathcal{M}_{\mathcal{R}[\mathcal{G}]}(x) \in \mathcal{M}_n(\mathcal{R})$, defined as follows:

$$\begin{aligned} \tau : x &\mapsto \mathcal{M}_{\mathcal{R}[\mathcal{G}]}(x) \\ \sum_{i=0}^{n-1} \alpha_{g_i} g_i &\mapsto \begin{pmatrix} \alpha_{g_0^{-1} g_0} & \alpha_{g_0^{-1} g_1} & \cdots\cdots & \alpha_{g_0^{-1} g_{n-1}} \\ \alpha_{g_1^{-1} g_0} & \alpha_{g_1^{-1} g_1} & \cdots\cdots & \alpha_{g_1^{-1} g_{n-1}} \\ \vdots & \vdots & \ddots & \vdots \\ \alpha_{g_{n-1}^{-1} g_0} & \alpha_{g_{n-1}^{-1} g_1} & \cdots\cdots & \alpha_{g_{n-1}^{-1} g_{n-1}} \end{pmatrix}. \end{aligned} \tag{4}$$

Let $\mathbf{x} = (\alpha_{g_0}, \alpha_{g_1}, \ldots, \alpha_{g_{n-1}})$ and $\mathbf{y} = (\beta_{g_0}, \beta_{g_1}, \ldots, \beta_{g_{n-1}})$ be the vectors corresponding to two elements $x, y \in \mathcal{R}[\mathcal{G}]$. Then, the following results hold:

1. Vector addition corresponds to component-wise addition:

$$\mathbf{x} + \mathbf{y} = (\alpha_{g_0} + \beta_{g_0}, \alpha_{g_1} + \beta_{g_1}, \ldots, \alpha_{g_{n-1}} + \beta_{g_{n-1}}).$$

2. Vector multiplication corresponds to the multiplication of $\mathbf{x}$ with the matrix representation of y:

$$\mathbf{x} \star \mathbf{y} = \mathbf{x}\mathcal{M}_{\mathcal{R}[\mathcal{G}]}(y) = (\alpha_{g_0}, \alpha_{g_1}, \ldots, \alpha_{g_{n-1}}) \begin{pmatrix} \beta_{g_0^{-1}g_0} & \beta_{g_0^{-1}g_1} & \cdots\cdots & \beta_{g_0^{-1}g_{n-1}} \\ \beta_{g_1^{-1}g_0} & \beta_{g_1^{-1}g_1} & \cdots\cdots & \beta_{g_1^{-1}g_{n-1}} \\ \vdots & \vdots & \ddots & \vdots \\ \beta_{g_{n-1}^{-1}g_0} & \beta_{g_{n-1}^{-1}g_1} & \cdots\cdots & \beta_{g_{n-1}^{-1}g_{n-1}} \end{pmatrix}.$$

3. The map τ is a bijective ring homomorphism [28, Theorem 1]. Specifically, $\tau(x+y) = \tau(x) + \tau(y)$ and $\tau(x * y) = \tau(x)\tau(y)$.
4. If $\mathcal{G}$ is abelian and $\mathcal{R}$ is a commutative ring, then $\tau(x)\tau(y) = \tau(y)\tau(x)$, and vector–matrix multiplication also commutes: $\mathbf{x} \star \mathbf{y} = \mathbf{x}\tau(y) = \mathbf{y}\tau(x)$.

Definition 6 (Twisted Group Ring). *The twisted group ring $\mathcal{R}^{\lambda}[\mathcal{G}]$ is defined similarly to the group ring in Definition 3, except that the multiplication is* twisted *according to a certain map λ:*

$$x * y = \sum_{k=0}^{n-1} \gamma_{g_k}^{\lambda} g_k, \quad \textit{where} \quad \gamma_{g_k}^{\lambda} = \sum_{g_i g_j = g_k} \lambda(g_i, g_j)\alpha_{g_i}\beta_{g_j}, \tag{5}$$

where $\lambda : \mathcal{G} \times \mathcal{G} \longrightarrow \mathbb{R}^{\times}$ is called a 2-cocycle that satisfies the following properties:

1. $\lambda(1,1) = 1$.
2. $\lambda(g_0 g_1, g_2)\,\lambda(g_0, g_1) = \lambda(g_0, g_1 g_2)\,\lambda(g_1, g_2)$ *for all* $g_0, g_1, g_2 \in \mathcal{G}$.

The matrix representation of $x \in \mathcal{R}^{\lambda}[\mathcal{G}]$ is given according to the map $\tau^{\lambda} : x \mapsto \mathcal{M}_{\mathcal{R}^{\lambda}[\mathcal{G}]}(x)$, where

$$\mathcal{M}_{\mathcal{R}^{\lambda}[\mathcal{G}]}(x) = \begin{pmatrix} \lambda(g_0, g_0^{-1}g_0)\alpha_{g_0^{-1}g_0} & \lambda(g_0, g_0^{-1}g_1)\alpha_{g_0^{-1}g_1} & \cdots & \lambda(g_0, g_0^{-1}g_{n-1})\alpha_{g_0^{-1}g_{n-1}} \\ \lambda(g_1, g_1^{-1}g_0)\alpha_{g_1^{-1}g_0} & \lambda(g_1, g_1^{-1}g_1)\alpha_{g_1^{-1}g_1} & \cdots & \lambda(g_1, g_1^{-1}g_{n-1})\alpha_{g_1^{-1}g_{n-1}} \\ \vdots & \vdots & \ddots & \vdots \\ \lambda(g_{n-1}, g_{n-1}^{-1}g_0)\alpha_{g_{n-1}^{-1}g_0} & \lambda(g_{n-1}, g_{n-1}^{-1}g_1)\alpha_{g_{n-1}^{-1}g_1} & \cdots & \lambda(g_{n-1}, g_{n-1}^{-1}g_{n-1})\alpha_{g_{n-1}^{-1}g_{n-1}} \end{pmatrix}. \tag{6}$$

Examples: The truncated polynomial ring $\mathcal{R}_{C_n} = \mathbb{Z}[x]/(x^n - 1)$ is isomorphic to the group ring $\mathbb{Z}C_n$, where $C_n = \langle x : x^n = 1 \rangle$ is the cyclic group of order n. Similarly, the ring $\mathcal{R}_{D_N} = \mathbb{Z}[x,y]/(x^N - 1,\ y^2 - 1,\ xy - yx^{N-1})$ is isomorphic to the group ring $\mathbb{Z}D_N$, where $D_N = \langle x, y \mid x^N = y^2 = 1,\ xy = yx^{N-1} \rangle$ is the dihedral group of order $2N$. In the twisted setting, the ring $\mathcal{R}_{C_n^+} = \mathbb{Z}[x]/(x^n + 1)$ is isomorphic to a twisted group ring $\mathbb{Z}^{\lambda_1}C_n$, and the ring $\mathcal{R}_{D_N^+} = \mathbb{Z}[x,y]/(x^N - 1,\ y^2 + 1,\ xy - yx^{N-1})$ is isomorphic to the twisted group ring $\mathbb{Z}^{\lambda_2}D_N$, where

$$\lambda_1(x^i, x^j) = \begin{cases} -1, & \text{if } i + j \geq n \\ 1, & \text{otherwise} \end{cases}, \quad \lambda_2(y^{j_1}x^{i_1}, y^{j_2}x^{i_2}) = \begin{cases} -1, & \text{for } i_1, i_2 \in \{0, 1, \ldots, n-1\} \\ & \text{and } j_1 = j_2 = 1 \\ 1, & \text{otherwise.} \end{cases}$$

Figure 1 illustrates the matrix representation $\tau^{\lambda}(a)$ for an element a from each of these rings, respectively.

$$\tau(a) = \begin{pmatrix} a_0 & a_1 & a_2 & \dots & a_{n-1} \\ a_{n-1} & a_0 & a_1 & \dots & a_{n-2} \\ a_{n-2} & a_{n-1} & a_0 & \dots & a_{n-3} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_1 & a_2 & a_3 & \dots & a_0 \end{pmatrix}$$

(a) $\tau(a)$ for $a \in \mathcal{R}_{C_n}$

$$\tau^{\lambda_1}(a) = \begin{pmatrix} a_0 & a_1 & a_2 & \dots & a_{n-1} \\ -a_{n-1} & a_0 & a_1 & \dots & a_{n-2} \\ -a_{n-2} & -a_{n-1} & a_0 & \dots & a_{n-3} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ -a_1 & -a_2 & -a_3 & \dots & a_0 \end{pmatrix}$$

(b) $\tau^{\lambda_1}(a)$ for $a \in \mathcal{R}_{C_n^+}$

$$\tau(a) = \left(\begin{array}{cccc|cccc} a_0 & a_1 & \dots & a_{N-1} & a_N & a_{N+1} & \dots & a_{2N-1} \\ a_{N-1} & a_0 & \dots & a_{N-2} & a_{N+1} & a_{N+2} & \dots & a_N \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ a_1 & a_2 & \dots & a_0 & a_{2N-1} & a_N & \dots & a_{2N-2} \\ \hline a_N & a_{N+1} & \dots & a_{2N-1} & a_0 & a_1 & \dots & a_{N-1} \\ a_{N+1} & a_{N+2} & \dots & a_N & a_{N-1} & a_0 & \dots & a_{N-2} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{2N-1} & a_N & \dots & a_{2N-2} & a_1 & a_2 & \dots & a_0 \end{array}\right)$$

(c) $\tau(a)$ for $a \in \mathcal{R}_{D_N}$

$$\tau^{\lambda_2}(a) = \left(\begin{array}{cccc|cccc} a_0 & a_1 & \dots & a_{N-1} & a_N & a_{N+1} & \dots & a_{2N-1} \\ a_{N-1} & a_0 & \dots & a_{N-2} & a_{N+1} & a_{N+2} & \dots & a_N \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ a_1 & a_2 & \dots & a_0 & a_{2N-1} & a_N & \dots & a_{2N-2} \\ \hline -a_N & -a_{N+1} & \dots & -a_{2N-1} & a_0 & a_1 & \dots & a_{N-1} \\ -a_{N+1} & -a_{N+2} & \dots & -a_N & a_{N-1} & a_0 & \dots & a_{N-2} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ -a_{2N-1} & -a_N & \dots & -a_{2N-2} & a_1 & a_2 & \dots & a_0 \end{array}\right)$$

(d) $\tau^{\lambda_2}(a)$ for $a \in \mathcal{R}_{D_N^+}$

Fig. 1: Matrix representations of a in (twisted) group rings over cyclic and dihedral group.

2.4 Public Key Encryption

A **P**ublic **K**ey **E**ncryption scheme or a PKE with the message space $\mathcal{M}$, ciphertext space $\mathcal{C}$, and the key space $\mathcal{K}$ is a tuple of three PPT (probabilistic polynomial time) algorithms $(\mathsf{PKE.KeyGen}, \mathsf{PKE.Encrypt}, \mathsf{PKE.Decrypt})$ where

- $\mathsf{PKE.KeyGen}(1^\lambda)$ is the key generation algorithm that outputs a secret and public key pair (sk, pk) according to the input parameter λ that corresponds to the security level, denoted as $(sk, pk) \leftarrow \mathsf{PKE.KeyGen}(1^\lambda)$.
- $\mathsf{PKE.Encrypt}(m, pk; r)$ outputs the ciphertext corresponding to the input message $m \in \mathcal{M}$, the public key pk, and a possible random string r, denoted as $c = \mathsf{PKE.Encrypt}(m, pk; r)$.
- $\mathsf{PKE.Decrypt}(c, sk)$ is the decryption algorithm that outputs a message $m' \in \mathcal{M}$ or a symbol $\perp \notin \mathcal{M}$ (when $c \notin \mathcal{C}$) on input a ciphertext $c \in \mathcal{C}$ and the secret key sk, denoted as $m' = \mathsf{PKE.Decrypt}(c, sk)$.

Correctness: For $0 \leq \delta < 1$, a PKE is δ-correct if for every key pair (sk, pk),

$$\Pr[\mathsf{PKE.Decrypt}(\mathsf{PKE.Encrypt}(m, pk), sk)] \neq m \leq \delta,$$

in the worst-case, where the probability is taken over all the randomness used in PKE.KeyGen and PKE.Encrypt. For $\delta = 0$, we say that the PKE is perfectly correct. We say that a PKE is γ-spread if

$$\min_{\substack{m \in \mathcal{M} \\ (sk, pk)}} \left(-\log \max_{c \in \mathcal{C}} \Pr[c = \mathsf{Encrypt}(m, pk, r)] \right) \geq \gamma.$$

Definition 7 (OW/IND-CPA PKE). *A* PKE *scheme* $\prod_{\mathsf{PKE}}$ *is* OW/IND-CPA *secure if for any PPT adversary* $\mathcal{A}$*, the advantages* $\mathsf{Adv}^{\mathsf{OW-CPA}}_{\prod_{\mathsf{PKE}}}(\mathcal{A}) := \Pr[m' = m]$ *and* $\mathsf{Adv}^{\mathsf{IND-CPA}}_{\prod_{\mathsf{PKE}}}(\mathcal{A}) := |\Pr[b' = b] - 1/2|$ *defined in the security games in Figure 2 are negligible.*

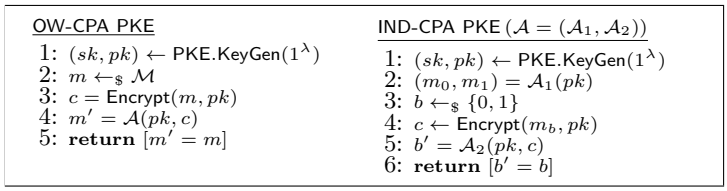


Fig. 2: OW/IND-CPA security game for the scheme $\prod_{\mathsf{PKE}}$.

2.5 Key Encapsulation Mechanism

A **K**ey **E**ncapsulation **M**echanism or a KEM is a tuple consisting of three PPT algorithms ($\mathsf{KEM.KeyGen}, \mathsf{KEM.Encap}, \mathsf{KEM.Decap}$) that facilitates the sharing of a secret key between two parties.

- $\mathsf{KEM.KeyGen}(1^\lambda)$ is the key generation algorithm that outputs a secret and public key pair (sk, pk) according to the input parameter λ that corresponds to the security level, denoted as $(sk, pk) \leftarrow \mathsf{KEM.KeyGen}(1^\lambda)$.
- $\mathsf{KEM.Encap}(pk)$ is the encapsulation algorithm[3] that outputs an encapsulated ciphertext $c \in \mathcal{C}$ and the associated session key $k \in \mathcal{K}$ on input a public key pk, denoted as $(c, k) = \mathsf{KEM.Encap}(pk)$.
- $\mathsf{KEM.Decap}(c, sk)$ is the decapsulation algorithm that outputs the session k on input an encapsulated ciphertext c and a secret key sk, denoted as $k = \mathsf{KEM.Decap}(c, sk)$.

Correctness: For $0 \leq \delta < 1$, a KEM is δ-correct if for every key pair (sk, pk),

$$\Pr[\mathsf{KEM.Decap}(c, sk) \neq k \mid \mathsf{KEM.Encap}(pk) = (c, k)] \leq \delta,$$

where the probability is taken over all the randomness used in $\mathsf{KEM.KeyGen}$ and $\mathsf{KEM.Encap}$. For $\delta = 0$, we say that the KEM is perfectly correct.

Definition 8 (IND-CCA KEM). *A* KEM *scheme* $\prod_{\mathsf{KEM}}$ *is called* IND-CCA *secure if the advantage* $\mathsf{Adv}^{\mathsf{IND-CCA}}_{\prod_{\mathsf{KEM}}}(\mathcal{A}) := \left|\Pr[b_i = b] - \frac{1}{2}\right|$ *of a PPT adversary* $\mathcal{A}$ *in the security game in Figure 3 is negligible.*

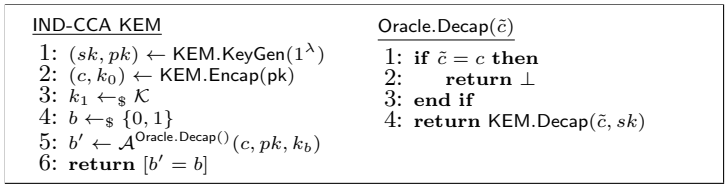


Fig. 3: IND-CCA security game for the scheme $\prod_{\mathsf{KEM}}$.

[3] KEM.Encap is a randomized algorithm that, everytime produces a fresh encapsulated key–ciphertext pair determined by its internally sampled randomness.

2.6 NTRU

We start with the conventional definition of NTRU as originally introduced in [21].

Definition 9 (NTRU). *Let n be a prime number and q be a positive integer. Let $f, g \in \mathbb{Z}[x]$ be polynomials of degree at most $n-1$, sampled from distributions χ_f and χ_g on $\mathcal{R}_{C_n} = \mathbb{Z}[x]/(x^n - 1)$, respectively, with the condition that f is invertible in the ring $\mathcal{R}^q_{C_n}$. Then, $h = f^{-1} * g \pmod q$ represents an NTRU instance. The decisional (n, q, χ_f, χ_g)-NTRU problem asks to distinguish h from a uniformly random element $u \leftarrow_\$ \mathcal{R}^q_{C_n}$. The search (n, q, χ_f, χ_g)-NTRU problem asks to recover (f, g) or a rotated pair $(x^i * f, x^i * g)$, given h.*

A matrix variant of the NTRU problem was introduced in [13,19] and is formulated as follows.

Definition 10 (Matrix NTRU). *Let n and q be positive integers. Let $\mathbf{F}, \mathbf{G} \in \mathbb{Z}^{n\times n}$ be matrices sampled from distributions $\chi_{\mathbf{F}}$ and $\chi_{\mathbf{G}}$ on $\mathbb{Z}^{n\times n}$, respectively, with the condition that $\mathbf{F}$ is invertible modulo q. Then, $\mathbf{H} = \mathbf{F}^{-1}\mathbf{G} \pmod q$ represents a matrix NTRU instance. The decisional $(n, q, \chi_{\mathbf{F}}, \chi_{\mathbf{G}})$-matrix NTRU problem asks to distinguish $\mathbf{H}$ from a uniformly random matrix $\mathbf{U} \leftarrow_\$ \mathbb{Z}_q^{n\times n}$. The search $(n, q, \chi_{\mathbf{F}}, \chi_{\mathbf{G}})$-matrix NTRU problem asks to recover $(\mathbf{F}, \mathbf{G})$, given $\mathbf{H}$.*

Another perspective on NTRU variants is provided by the group ring formulation, known as GR-NTRU [48]. Many variants of NTRU constructed over algebraic rings [21,40,3] can be seen as special cases of GR-NTRU or its twisted versions. For instance, the original NTRU scheme can be interpreted as a GR-NTRU instance where the ring is the ring of integers and the group is a finite cyclic group. This leads to a natural generalization of the NTRU problem over arbitrary algebraic structures, as formalized below.

Definition 11 (GR-NTRU). *Let $\mathcal{G}$ be a finite group of order n, let $\mathcal{R}$ be a ring, and let q be a positive integer. Let $f, g \in \mathcal{R}[\mathcal{G}]$ be two elements sampled from distributions χ_f and χ_g on $\mathcal{R}[\mathcal{G}]$, respectively, with the condition that f is invertible in the ring $(\mathcal{R}/\langle q\rangle)[\mathcal{G}]$. Then, $h = f^{-1} * g \pmod q$ represents a GR-NTRU instance. The decisional $(\mathcal{G}, \mathcal{R}, q, \chi_f, \chi_g)$-GR-NTRU problem asks to distinguish h from a uniformly random element $u \leftarrow_\$ (\mathcal{R}/\langle q\rangle)[\mathcal{G}]$. The search $(\mathcal{G}, \mathcal{R}, q, \chi_f, \chi_g)$-GR-NTRU problem asks to recover (f, g) or a rotation (with respect to the underlying group structure), given h.*

For any instance of group-ring NTRU, the element f is invertible over $(\mathcal{R}/\langle q\rangle)[\mathcal{G}]$ if and only if the corresponding matrix, defined by the map introduced in Equation (4), is invertible over $\mathcal{R}/\langle q\rangle$ [28, Theorem 2]. This result extends naturally to the twisted GR-NTRU setting, where the matrix defined in Equation (6) can similarly be used to verify the invertibility condition.

Additionally, the private elements f and g are typically sampled with coefficients drawn from a discrete Gaussian distribution with variance $\sigma^2 > 0$. In the original NTRU scheme [21], f and g are sampled as ternary polynomials

with approximately $n/3$ coefficients equal to -1, $n/3$ equal to 1, and $n/3$ equal to 0, which can be roughly interpreted as sampling from a discrete Gaussian distribution with $\sigma^2 \approx 2/3$.

Attacks Landscape

- **Search Attacks:** The most straightforward approach is to search for two elements $f', g' \in \mathbb{Z}_q^{\lambda}[\mathcal{G}]$ with small coefficients that satisfy the NTRU equation. Search attacks can be further enhanced using Meet-in-the-Middle techniques [23] and optimized combinatorial methods [38,14].
- **Lattice Attacks:** Instead of exhaustively searching for the private key, one can construct a lattice in which the key lies, and then apply lattice reduction techniques to recover a short vector that may serve as a valid decryption key. Given the public key h, the Coppersmith–Shamir lattice corresponding to the GR-NTRU instance is defined as

$$\mathcal{L}^{CS} = \{(\mathbf{g}, \mathbf{f}) \in \mathbb{Z}^{2n} \mid f * h = g \pmod{q}\}.$$

  Since the private key $(\mathbf{g}, \mathbf{f})$ typically has small coefficients, the corresponding vector is expected, with high probability, to be among the shortest in the lattice. Thus, an attacker may construct the lattice $\mathcal{L}^{CS}$ generated by the basis:

$$\mathcal{B}^{CS} = \begin{pmatrix} q\mathbf{I}_n & \mathbf{0}_n \\ \tau^{\lambda}(h) & \mathbf{I}_n \end{pmatrix}, \tag{7}$$

  and apply lattice reduction algorithms such as BKZ [42] to extract a short vector that could act as a decryption key. This attack approach is called the **Primal** attack. The cost of lattice reduction primarily depends on the volume of the lattice (q^n) and the ratio of the norm of the private key (g, f) to the expected shortest vector length given by the Gaussian heuristic (Definition 4). This ratio is referred to as the *lattice gap*. A larger lattice gap and smaller lattice volume make lattice reduction attacks more computationally demanding in practice. The primal attack can be combined with search-based attacks, resulting in a **Hybrid** attack. Hybrid attacks are considered the most powerful known methods against NTRU-like schemes employing ternary or sparse keys; consequently, the computational cost of these attacks is a critical factor in parameter selection. For a comprehensive description, we refer the reader to [26].

Lattice-based attacks, including combined strategies such as the hybrid attack, remain the most effective methods against NTRU-like constructions. Table 1 presents a comparison between the cost of the best-known combinatorial attacks and that of lattice-based attacks on various NTRU instances. The column (n, q, w) indicates the number of the coefficients n in the sampled polynomials, the modulus q, and the number of non-zero ternary coefficients w out of n. The column $\mathcal{C}$ indicates the cost of the combinatorial attack without Meet-in-the-Middle (MitM). Odlyzko refers to the MitM cost introduced in [23]. May [Free Mem] and May [Poly Mem] represent the combinatorial attack costs from [38] under free and polynomial memory access assumptions, respectively. Esser et al.

indicate the cost of a combinatorial attack under polynomial memory as proposed in [14]. Finally, the Lattice Core-SVP column estimates the lattice attack cost using the classical Core-SVP model $2^{0.292\beta}$, where β is the BKZ blocksize. The table shows that lattice-based techniques are significantly more efficient in practice. Therefore, the selection of parameters in these schemes is influenced by the cost of the lattice attacks, ensuring the desired levels of security.

Table 1: Comparison of the estimated costs of combinatorial and lattice-based attacks against selected NTRU-like schemes from the literature.

(n, q, w)	$\mathcal{C}$	Odlyzko [23]	May [Free Mem] [38]	May [Poly Mem] [38]	Esser et al. [14]	Lattice-Core-SVP
NTRU-HPS [6]						
(509,2048,254)	754	377	227	490	429	**105**
(677,2048,254)	891	445	273	581	492	**144**
(821,4096,510)	1286	643	378	852	750	**178**
NTRU Prime [5]						
(653,4621,288)	925	463	272	599	518	**129**
(761,4591,286)	1003	502	301	653	553	**153**
(857,5167,322)	1131	566	338	735	623	**175**
DiTRU$^+$ [41]						
(2*269,2048,254)	781	391	239	517	453	**108**
(2*347,2048,254)	902	451	280	595	504	**144**
(2*419,4096,510)	1309	654	385	869	766	**178**

3 Our Construction

This section introduces an almost unstructured variant of noncommutative NTRU, which we refer to as ANTRU.

3.1 Almost Unstructured Variant

We begin by introducing the definitions of the decisional and search problems for the Almost-NTRU (abbreviated as ANTRU) scheme.

Definition 12 (unstructured-ANTRU or u-ANTRU). *Let n be a prime number and q be a positive integer. Let $f \in \mathbb{Z}[x]$ be a polynomial of degree at most $n-1$, sampled from a distribution χ_f, and let $\mathbf{G} \in \mathbb{Z}^{n\times n}$ be sampled from a distribution $\chi_{\mathbf{G}}$. Let f_1 and f_2 be two invertible elements in the ring $\mathcal{R}^q_{C_n^+} = \mathbb{Z}_q[x]/(x^n+1)$, such that $f_1 \leftarrow_{\$} \mathcal{R}^q_{C_n^+}$ and $f_2 = f_1^{-1} * f \pmod q$. Let $\mathbf{F}_1$ and $\mathbf{F}_2$ denote the matrix representations of f_1 and f_2, respectively. Then, the matrix*

$$\mathbf{H} = \mathbf{F}_1^{-1}\mathbf{G}\mathbf{F}_2^{-1} \pmod q \tag{8}$$

is called a u-ANTRU instance.

Definition 13 (Decisional u-ANTRU Problem). *Given the parameters $(n, q, \chi_f, \chi_{\mathbf{G}})$, distinguish $\mathbf{H}$ from a uniformly random matrix $\mathbf{U} \leftarrow_{\$} \mathbb{Z}_q^{n\times n}$.*

Definition 14 (Search u-ANTRU Problem). *Given $\mathbf{H}$, recover a tuple $(f_1, f_2, \mathbf{G})$ such that f_1 and f_2 are invertible that satisfy $f_1 * f_2 = f \pmod q$ and the public key equation (8), for some f drawn from χ_f and invertible, under the same parameter set $(n, q, \chi_f, \chi_{\mathbf{G}})$.*

Consider the public key equation of NTRU (Definition 9): $h = f^{-1} * g \pmod q$. It is easy to observe that f can be written as a product $f = f_1 * f_2$ for some invertible polynomials f_1 and f_2. Substituting this into the public key equation yields $h = f_2^{-1} * f_1^{-1} * g \pmod q$, which can be rearranged as $h = f_1^{-1} * g * f_2^{-1} \pmod q$, since the underlying ring is commutative.

The public key equation of u-ANTRU is similar in form to that of NTRU, but with a crucial distinction: the order of multiplication matters, as the matrix $\mathbf{G}$ does not commute with the matrices $\mathbf{F}_1$ and $\mathbf{F}_2$. Therefore, while it holds that $\mathbf{H}' = \mathbf{F}_2^{-1}\mathbf{F}_1^{-1}\mathbf{G} = \mathbf{F}_1^{-1}\mathbf{F}_2^{-1}\mathbf{G} \pmod q$, this equivalence does not imply that $\mathbf{H} = \mathbf{H}' \pmod q$.

Nevertheless, one can show that the decisional and search versions of the u-ANTRU problem are at least as hard as the corresponding decisional and search versions of the NTRU problem.

Lemma 1. *Let $\mathcal{A}$ be a PPT (probabilistic polynomial time) algorithm that solves the decisional or search $(n, q, \chi_f, \chi_{\mathbf{G}})$-u-ANTRU problem with advantage ϵ. Then, there exists an algorithm $\mathcal{B}$ that, by invoking $\mathcal{A}$ once with a negligible amount of additional computations, can solve the decisional/search (n, q, χ_f, χ_g)-NTRU problem with the same advantage ϵ.*

Proof. Suppose the algorithm $\mathcal{A}$ can solve the decisional u-ANTRU problem with advantage ϵ. Then, trivially, an NTRU instance $h = f^{-1} * g \pmod q$ can be interpreted as an u-ANTRU instance with $f_1 = f$, $\mathbf{G}$ as the matrix representation of $g \in \mathcal{R}^q_{C_n^+}$, and $f_2 = 1$. Therefore, if the algorithm $\mathcal{A}$ can solve the decisional u-ANTRU problem with advantage ϵ, it should also be able to distinguish an NTRU instance with the same advantage. However, the oracle $\mathcal{A}$ may detect that the queried instance always falls under this special case. To address this, we randomize the input as follows.

Let h be an NTRU instance computed as $h = f^{-1} * g \pmod q$. Then, algorithm $\mathcal{B}$ proceeds as follows:

- Convert h into matrix form: compute $\mathbf{H} = \mathbf{F}^{-1}\mathbf{G} \pmod q$, as described in Figure 1.
- Generate an elementary matrix $\mathbf{I}_{e_n}$ by permuting the rows of the identity matrix $\mathbf{I}_n$, and compute $\mathbf{H}' = \mathbf{H}\mathbf{I}_{e_n} = \mathbf{F}^{-1}(\mathbf{G}\mathbf{I}_{e_n}) \pmod q$. Let $\mathbf{G}' = \mathbf{G}\mathbf{I}_{e_n}$.
- Sample a random invertible polynomial $f' \leftarrow_{\$} \mathcal{R}^q_{C_n^+}$, let $\mathbf{F}'$ be its matrix representation, and compute

$$\mathbf{H}'' = \mathbf{F}'\mathbf{H}'\mathbf{F}'^{-1} = (\mathbf{F}\mathbf{F}'^{-1})^{-1}\mathbf{G}'\mathbf{F}'^{-1} \pmod q.$$

- Query algorithm $\mathcal{A}$ once with the randomized instance $\mathbf{H}''$ representing an u-ANTRU sample.
- Algorithm $\mathcal{B}$ returns 1 if algorithm $\mathcal{A}$ outputs 1 (indicating that h is an NTRU instance), and returns 0 otherwise.

For the search problem, algorithm $\mathcal{B}$ follows the same steps to generate $\mathbf{H}''$ and provides it to algorithm $\mathcal{A}$, which solves the search problem of u-ANTRU.

If $\mathcal{A}$ returns a solution tuple $(f_1, f_2, \mathbf{G}') = (f * f'^{-1}, f', \mathbf{G}\mathbf{I}_{e_n})$, then algorithm $\mathcal{B}$ recovers $f = f_1 * f_2 \pmod q$ and $\mathbf{G} = \mathbf{G}'\mathbf{I}_{e_n}^{-1}$. Since g corresponds to the first row of $\mathbf{G}$, algorithm $\mathcal{B}$ finally returns (f, g) as the solution to the NTRU search problem. □

Encryption Scheme The encryption scheme based on u-ANTRU is similar to the general encryption scheme employed in NTRU-HPS, and is outlined in Figure 4. The notations used have the following meanings:

- $\mathcal{T}$ denotes the set of ternary elements in $\mathcal{R}_{C_n^+}$.
- $\mathcal{T}(a, b)$ denotes the set of elements in $\mathcal{R}_{C_n^+}$ with a coefficients equal to 1, b coefficients equal to -1, and the remaining coefficients set to 0.
- $\mathcal{T}_{\text{matrix}}(a, b)$ denotes the set of matrices in $\mathbb{Z}^{n\times n}$ such that each column has a entries equal to 1, b entries equal to -1, and the rest equal to 0.
- $\mathcal{L}_f = \mathcal{T}(d_f + 1, d_f), \mathcal{L}_{\mathbf{G}} = \mathcal{T}_{\text{matrix}}(d_g, d_g), \mathcal{L}_r = \mathcal{T}(d_r, d_r)$, and $\mathcal{L}_m = \mathcal{T}$.
- Sampler instantiated with a set and a seed deterministically returns an element from the underlying set associated with the seed.
- Centerlift$(\cdot, q)$ returns an element whose lie in the range $(-q/2, q/2]$.

KeyGen(1^λ)

1: initialize Sampler with $\mathcal{L}_f$ and seed_f
2: do $f \leftarrow$ Sampler until f is invertible in $\mathcal{R}^q_{C_n^+}$
3: initialize Sampler with $\mathcal{R}^q_{C_n^+}$ and seed_{f_1}
4: do $f_1 \leftarrow$ Sampler until f_1 is invertible in $\mathcal{R}^q_{C_n^+}$
5: let $f_2 = f_1^{-1} * f \pmod q$.
6: let f_p denotes the inverse of f in $\mathcal{R}^p_{C_n^+}$
7: let $\mathbf{F}_1$ and $\mathbf{F}_2$ denote the matrix representations of f_1 and f_2
8: initialize Sampler with $\mathcal{L}_{\mathbf{G}}$ and $\mathsf{seed}_{\mathbf{G}}$
9: $\mathbf{G} \leftarrow$ Sampler
10: $\mathbf{H} = \mathbf{F}_1^{-1}\mathbf{G}\mathbf{F}_2^{-1} \pmod q$
11: **return** $((f_1, f_2, f_p), \mathbf{H})$

Encrypt($\mathbf{H}, m, \mathsf{seed}_{r_1}, \mathsf{seed}_{r_2}$)

1: initialize Sampler with $\mathcal{L}_r$ and seed_{r_1}
2: $r_1 \leftarrow$ Sampler
3: initialize Sampler with $\mathcal{L}_r$ and seed_{r_2}
4: $r_2 \leftarrow$ Sampler
5: let $\mathbf{R}_1$ and $\mathbf{R}_2$ denote the matrix representations of r_1 and r_2
6: let $\mathbf{M}$ indicates the matrix representation of m
7: $\mathbf{C} = p\mathbf{R}_1\mathbf{H}\mathbf{R}_2 + \mathbf{M} \pmod q$
8: **return** $\mathbf{C}$

Decrypt($(f_1, f_2, f_p), \mathbf{C}$)

1: let $\mathbf{F}_1$ and $\mathbf{F}_2$ denote the matrix representations of f_1 and f_2
2: let $\mathbf{F}_p$ denotes the matrix representation of f_p
3: $\mathbf{T} = \mathbf{F}_1\mathbf{C}\mathbf{F}_2 \pmod q$
4: $\mathbf{A} =$ Centerlift$(\mathbf{T}, q)$
5: $\mathbf{M}' = \mathbf{F}_p\mathbf{A} \pmod p$
6: let m' equals the first row of $\mathbf{M}'$
7: $m \leftarrow$ Centerlift(m', p)
8: **return** m

Fig. 4: A sketched description of the encryption scheme of u-ANTRU.

3.2 Attack Strategy

Key Recovery Attack The combinatorial attack against the secret key involves first guessing a ternary polynomial $f \in \mathcal{L}_f$, followed by guessing the first factor $f_1 \in \mathcal{L}_f$. Once f_1 is determined, the second factor f_2 can be computed directly. For the correct guess, we obtain a matrix

$$\mathbf{G} = \mathbf{F}_1\mathbf{H}\mathbf{F}_2 \pmod q,$$

whose entries are small. The overall cost of such a combinatorial attack on the key is prohibitively large. For instance, when $\mathcal{L}_f = \mathcal{T}(d_f + 1, d_f)$, the search complexity is given by

$$q^n \binom{n}{d_f + 1} \binom{n - d_f - 1}{d_f}.$$

Lower Bound: Since the public key equation of u-ANTRU is inherently supported by the lower-bound security of the standard NTRU assumption, the best an attacker can hope for is to transform the u-ANTRU equation $\mathbf{H} = \mathbf{F}_1^{-1}\mathbf{G}\mathbf{F}_2^{-1} \pmod q$ into a commutative analog of the form $\mathbf{H}' = \mathbf{F}^{-1}\mathbf{G}' \pmod q$, where $\mathbf{F} = \mathbf{F}_1\mathbf{F}_2 \pmod q$ and $\mathbf{G}'$ is a matrix with small entries. In such a case, the attacker could attempt lattice reduction techniques on $\mathbf{H}'$ to recover $\mathbf{F}$, which might subsequently aid in message recovery. Therefore, when considering both the key and ciphertext equations, the attacker's task reduces to constructing a mapping that transforms the noncommutative equation into a commutative one. Let $\mathbf{RC}^{n\times n}$ denote the set of all right circulant matrices over $\mathbb{Z}^{n\times n}$. The attacker would then be seeking a map $\theta : \mathbb{Z}^{n\times n} \to \mathbf{RC}^{n\times n}$ satisfying the following properties:

- It is not the zero map.
- It is linear (a requirement for enabling decryption).
- It is both left- and right-linear with respect to elements belonging to the commutative subring (in this case, $\mathbf{RC}^{n\times n}$).
- It maps a matrix $\mathbf{G}$ with small entries to a matrix $\mathbf{G}'$ whose entries are bounded by $\alpha g_{\max}$, where α is a small integer and $g_{\max} = \max_{i,j} |\mathbf{G}^{(i,j)}|$, ensuring applicability of lattice reduction.
- It is easily (polynomial time) invertible on the images of elements representing the message space.

Upon identifying such a map θ, the attacker can recover $\mathbf{F}$ by applying lattice reduction to the lattice generated by $\begin{pmatrix} q\mathbf{I}_n & \mathbf{0}_n \\ \theta(\mathbf{H}) & \mathbf{I}_n \end{pmatrix}$. Subsequently, the attacker can compute

$$\begin{aligned} \mathbf{F}\theta(\mathbf{C}) &= p\mathbf{R}_1\mathbf{R}_2\mathbf{F}\theta(\mathbf{H}) + \mathbf{F}\theta(\mathbf{M}) \pmod q \\ &= p\mathbf{R}_1\mathbf{R}_2\theta(\mathbf{G}) + \mathbf{F}\theta(\mathbf{M}) \pmod q, \end{aligned}$$

which, when reduced modulo p, yields $\mathbf{F}\theta(\mathbf{M})$. Since $\mathbf{F}$ is invertible and by definition, θ is invertible on $\mathbf{M}$, the attacker can then reconstruct $\mathbf{M}$. In the case of u-ANTRU, a possible construction of such a map θ that satisfies the required properties is given by

$$\theta(\mathbf{X}) = \sum_{i=0}^{k} \mathbf{A}_i\mathbf{X}\mathbf{B}_i, \qquad \mathbf{A}_i, \mathbf{B}_i \in \mathbf{RC}^{n\times n}.$$

In Section 4, we give a concrete example about such map.

Message Recovery Attack The ciphertext equation is given by

$$\mathbf{C} = p\mathbf{R}_1\mathbf{H}\mathbf{R}_2 + \mathbf{M} \pmod q,$$

where

$$\mathbf{R}_1 = \begin{pmatrix} r_{10} & r_{11} & \dots & r_{1(n-1)} \\ -r_{1(n-1)} & r_{10} & \dots & r_{1(n-2)} \\ \vdots & \vdots & \ddots & \vdots \\ -r_{11} & -r_{12} & \dots & r_{10} \end{pmatrix}, \quad \mathbf{R}_2 = \begin{pmatrix} r_{20} & r_{21} & \dots & r_{2(n-1)} \\ -r_{2(n-1)} & r_{20} & \dots & r_{2(n-2)} \\ \vdots & \vdots & \ddots & \vdots \\ -r_{21} & -r_{22} & \dots & r_{20} \end{pmatrix}, \quad \mathbf{H} = \begin{pmatrix} h_0 & h_1 & \dots & h_{n-1} \\ h_n & h_{n+1} & \dots & h_{2n-1} \\ \vdots & \vdots & \ddots & \vdots \\ h_{n(n-1)} & h_{n(n-1)+1} & \dots & h_{n^2-1} \end{pmatrix}.$$

This matrix equation can be equivalently represented in polynomial form as

$$\overrightarrow{\mathbf{C}}^{\text{poly}} = \underbrace{\begin{pmatrix} h_0(x) & h_1(x) & \dots & h_{n-1}(x) \\ h_1(x) & h_2(x) & \dots & -h_0(x) \\ \vdots & \vdots & \ddots & \vdots \\ h_{n-1}(x) & -h_0(x) & \dots & -h_{n-2}(x) \end{pmatrix}}_{\mathbf{H}^{\text{poly}}} \begin{pmatrix} r_{10}\, r_2(x) \\ r_{11}\, r_2(x) \\ \vdots \\ r_{1(n-1)}\, r_2(x) \end{pmatrix} + \begin{pmatrix} m(x) \\ x * m(x) \\ \vdots \\ x^{n-1} * m(x) \end{pmatrix}, \quad (9)$$

where $h_i(x)$ denotes the polynomial corresponding to the coefficient vector of the ith column of the matrix $\mathbf{H}$, and $m(x)$ is the polynomial obtained from the coefficient vector of the first row of the matrix $\mathbf{M}$. Naively, the message can be recovered by solving the shortest vector problem (SVP) in a $(2n^2+1)$-dimensional lattice generated by $\begin{pmatrix} q\mathbf{I}_{n^2} & \mathbf{0}_{n^2} & 0_{n^2} \\ \boldsymbol{\mathcal{H}}^{\text{poly}} & \mathbf{I}_{n^2} & 0_{n^2} \\ \overrightarrow{\mathbf{C}}^{Tr} & \mathbf{0}_{n^2} & 1 \end{pmatrix}$, where $\boldsymbol{\mathcal{H}}^{\text{poly}}$ is the integer matrix representation corresponding to the polynomial matrix $\mathbf{H}^{\text{poly}}$, and $\overrightarrow{\mathbf{C}}$ denotes the vectorization of the ciphertext matrix $\mathbf{C}$. In the commutative case, the lattice dimension would reduce to $(2n+1)$. However, message recovery in the noncommutative setting is essentially equivalent to solving an over-structured Module-LWE instance. An attacker may therefore attempt to construct a homomorphism that reduces the lattice dimension from $2n^2+1$ to a smaller dimension, thereby extracting useful information about the message from lower-dimensional lattices. Nevertheless, any such homomorphism must map short vectors in the original lattice to short vectors in the reduced lattice.

Another observation is that the error polynomials are cyclic rotations of one another, a structure that may enable the construction of multiple noiseless equations, potentially leading to more efficient message-recovery attacks. In order to avoid such attacks, we modify the general design sketched in Figure 4 so that both f_1 and f_2 are sampled as invertible elements from $\mathcal{R}^q_{C_n^+}$, specifically from the space $\mathcal{L}_f = \mathcal{T}(d_f+1, d_f)$, and $\mathbf{M}$ is sampled as a ternary matrix similar to $\mathbf{G}$, rather than as a cyclic matrix. In this setting, the product $f_1 * f_2$ does not yield a ternary polynomial but instead a polynomial with larger coefficients. Consequently, during decryption, the value of q must be larger compared to the variant presented in Definition 12. Nevertheless, the u-ANTRU problem is still regarded as a two-level NTRU problem, and the u-ANTRU search problem is modified accordingly.

Definition 15 (Refined u-ANTRU instance). *Let n be a prime, q a positive integer, and $f_1, f_2 \in \mathbb{Z}[x]$ (with $\deg(f_i) \leq n-1$) drawn from χ_f, and let $\mathbf{G} \in \mathbb{Z}^{n\times n}$ be drawn from $\chi_{\mathbf{G}}$. Let $\mathbf{F}_1$ and $\mathbf{F}_2$ denote the matrix representations of f_1 and f_2, respectively; then $\mathbf{H} = \mathbf{F}_1^{-1}\mathbf{G}\mathbf{F}_2^{-1} \pmod q$ is called an u-ANTRU instance.*

Definition 16 (Refined Search u-ANTRU Problem). *Given $\mathbf{H}$, a modified u-ANTRU instance, recover $(f_1, f_2, \mathbf{G})$ such that f_1, f_2 are invertible and drawn from χ_f.*

For the modified construction of u-ANTRU, the attacker can attempt the following strategies:

(i) **Key Attack:** guess f_1. For the correct guess, $\mathbf{F}_1^{-1}\mathbf{H} \pmod q$ yields an NTRU instance, which can be recognized using the best available lattice reduction techniques.
(ii) **Message Attack:** guess r_1. For the correct guess, the pair $(\mathbf{R}_1\mathbf{H}, \mathbf{C})$ forms an LWE instance, which can likewise be identified via the best lattice reduction techniques.

Hence, the most effective strategy combines combinatorial guessing with lattice-based attacks. It is worth noting that improved search attacks, such as the meet-in-the-middle approach, cannot be applied directly since $\mathbf{F}_1\mathbf{H} \pmod q$ is not a matrix with small coefficients.

4 Security Analysis

4.1 Security Against Lattice Attacks

We discuss the almost-unstructured variant, while the semi-structured variant follows from essentially the same arguments.

The goal of an adversary as mentioned is Section 3.2 is to construct a map $\theta : M_n(\mathbb{Z}) \to \mathbf{R}\mathbf{C}^{n\times n}$ that reduces the equation $\mathbf{H} = \mathbf{F}_1^{-1}\mathbf{G}\mathbf{F}_2^{-1}$ into an NTRU-like equation of the form $\theta(\mathbf{H}) = \mathbf{F}'^{-1}\mathbf{G}' \pmod q$ such that $\mathbf{F}'$ and $\mathbf{G}'$ have relatively small coefficients, in order to perform lattice attacks. Given the fact that $\mathbf{F}_1$ and $\mathbf{F}_2$ are sampled from the ring $\mathcal{R}_{C_n^+}$ while $\mathbf{G}$ is a random matrix, the most plausible approach is to keep $\mathbf{F}_1$ and $\mathbf{F}_2$ intact and attempt to map $\mathbf{G}$ to an element $\theta(\mathbf{G})$ in the ring $\mathcal{R}_{C_n^+}$ that commutes with $\mathbf{F}_i$, thereby inducing an NTRU-like equation $\theta(\mathbf{H}) = (\mathbf{F}_1\mathbf{F}_2)^{-1}\theta(\mathbf{G}) \pmod q$, as also demonstrated in the maps discussed above. It should be noted that the cost of lattice reduction in this case is higher compared to the typical GR-NTRU, since

$$\|(\mathbf{F}_1\mathbf{F}_2, \theta(\mathbf{G}))\| > \|(\mathbf{F}, \mathbf{G})\|$$

for random ternary $\mathbf{F}$ and $\mathbf{G}$. Concretely, the adversary seeks $\mathbf{A}_i, \mathbf{B}_i$, for $i = 1, 2, \ldots, k$, such that

$$\theta(\mathbf{H}) = \sum_{i=1}^{k} \mathbf{A}_i\mathbf{H}\mathbf{B}_i = \sum_{i=1}^{k} \mathbf{A}_i(\mathbf{F}_1^{-1}\mathbf{G}\mathbf{F}_2^{-1})\mathbf{B}_i = \mathbf{F}_1^{-1}\theta(\mathbf{G})\mathbf{F}_2^{-1}$$

holds, where $\theta(\mathbf{G}) \in \mathcal{R}_{C_n^+}$ is small in Euclidean norm. For the second equality to hold, $\mathbf{A}_i$ and $\mathbf{B}_i$ must commute with $\mathbf{F}_1$ and $\mathbf{F}_2$. Since the $\mathbf{F}_i$ are sampled uniformly at random from $\mathcal{R}_{C_n^+}$ and remain secret, the only feasible choice is $\mathbf{A}_i, \mathbf{B}_i \in \mathcal{R}_{C_n+}$ in order to commute with $\mathbf{F}_i$ so that

$$\sum_{i=1}^{k} \mathbf{A}_i(\mathbf{F}_1^{-1}\mathbf{G}\mathbf{F}_2^{-1})\mathbf{B}_i = \mathbf{F}_1^{-1}\left(\sum_{i=1}^{k} \mathbf{A}_i\mathbf{G}\mathbf{B}_i\right)\mathbf{F}_2^{-1} = \mathbf{F}_1^{-1}\theta(\mathbf{G})\mathbf{F}_2^{-1}.$$

A natural way to build such matrix is as the following: We define the twisted shift matrix $\mathbf{X} \in \mathbf{RC}^{n\times n}$ as:

$$\mathbf{X} = \begin{pmatrix} 0 & 0 & \cdots & 0 & -1 \\ 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & \cdots & 0 & 0 \\ \vdots & & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & 0 \end{pmatrix}$$

and satisfying the following properties:

1. $\mathbf{X}^n = -\mathbf{I}_n$
2. $\mathbf{X}^{2n} = \mathbf{I}_n$, so $\mathbf{X}$ has order $2n$
3. Every $\mathbf{F} \in \mathbf{RC}^{n\times n}$ is a polynomial in $\mathbf{X}$, i.e.,$\mathbf{F} = \sum_{k=0}^{n-1} f_k\mathbf{X}^k$, for $f_k \in \mathbb{Z}$
4. $\mathbf{X}^j$ is a signed permutation matrix for all j, with entries in $\{0, +1, -1\}$

Hence, the map $\theta : M_n(\mathbb{Z}) \to \mathbf{RC}^{n\times n}$ can be defined as:

$$\theta(\mathbf{M}) = \sum_{j=0}^{2n-1} \mathbf{X}^j\mathbf{M}\mathbf{X}^{-j} \pmod q$$

This is an instance of the general linear map $\sum_i \mathbf{A}_i\mathbf{M}\mathbf{B}_i$ with:

$$\mathbf{A}_j = \mathbf{X}^j \in \mathbf{RC}^{n\times n}, \qquad \mathbf{B}_j = \mathbf{X}^{-j} \in \mathbf{RC}^{n\times n}, \qquad j = 0, 1, \ldots, 2n-1$$

Lemma 2. *For any $\mathbf{G} \in M_n(\mathbb{Z})$, define: $\mathbf{G}' = \sum_{j=0}^{2n-1} \mathbf{X}^j\mathbf{G}\mathbf{X}^{-j} \pmod q$, then $\mathbf{G}' \in \mathbf{RC}^{n\times n}$.*

Proof. It suffices to show that $\mathbf{G}'$ commutes with $\mathbf{X}$. Compute:

$$\mathbf{X}\mathbf{G}'\mathbf{X}^{-1} = \mathbf{X}\left(\sum_{j=0}^{2n-1} \mathbf{X}^j\mathbf{G}\mathbf{X}^{-j}\right)\mathbf{X}^{-1} = \sum_{j=0}^{2n-1} \mathbf{X}^{j+1}\mathbf{G}\mathbf{X}^{-(j+1)}$$

Re-index by $j \mapsto j-1$:

$$= \sum_{j=1}^{2n} \mathbf{X}^j\mathbf{G}\mathbf{X}^{-j} = \sum_{j=0}^{2n-1} \mathbf{X}^j\mathbf{G}\mathbf{X}^{-j} = \mathbf{G}'$$

where the last equality follows from $\mathbf{X}^{2n} = \mathbf{I}_n$ which identifies the $j = 2n$ term with the $j = 0$ term. Therefore $\mathbf{X}\mathbf{G}' = \mathbf{G}'\mathbf{X}$, and since $\mathbf{X}$ generates $\mathbf{RC}^{n\times n}$, we conclude $\mathbf{G}' \in \mathbf{RC}^{n\times n}$.

Therefore, applying the map θ to the public key $\mathbf{H}$ yields:

$$\theta(\mathbf{H}) = \sum_{j=0}^{2n-1} \mathbf{X}^j \mathbf{F}_1^{-1} \mathbf{G} \mathbf{F}_2^{-1} \mathbf{X}^{-j} \pmod{q}$$

Since $\mathbf{X}^j, \mathbf{F}_1^{-1} \in \mathbf{RC}^{n\times n}$ and the $\mathbf{RC}^{n\times n}$ algebra is commutative, $\mathbf{X}^j \mathbf{F}_1^{-1} = \mathbf{F}_1^{-1} \mathbf{X}^j$. Similarly $\mathbf{F}_2^{-1} \mathbf{X}^{-j} = \mathbf{X}^{-j} \mathbf{F}_2^{-1}$. Therefore:

$$\theta(\mathbf{H}) = \sum_{j=0}^{2n-1} \mathbf{F}_1^{-1} \mathbf{X}^j \mathbf{G} \mathbf{X}^{-j} \mathbf{F}_2^{-1} = \mathbf{F}_1^{-1} \left(\sum_{j=0}^{2n-1} \mathbf{X}^j \mathbf{G} \mathbf{X}^{-j} \right) \mathbf{F}_2^{-1} = \mathbf{F}_1^{-1} \mathbf{G}' \mathbf{F}_2^{-1} \pmod{q}$$

By Lemma 2, $\mathbf{G}' \in \mathbf{RC}^{n\times n}$, so $\mathbf{G}'$ commutes with both $\mathbf{F}_1^{-1}$ and $\mathbf{F}_2^{-1}$:

$$\theta(\mathbf{H}) = (\mathbf{F}_1 \mathbf{F}_2)^{-1} \mathbf{G}' \pmod{q}.$$

Alternatively, we can write $h' = f' * g' \pmod{q}$, where h', f', and g' are the polynomials from $\mathcal{R}_{C_n^+}$ corresponding to the first rows of $\theta(\mathbf{H})$, $(\mathbf{F}_1 \mathbf{F}_2)$, and $\mathbf{G}'$, respectively.

In order to estimate the hardness of the associated lattice problem, we need to check the distribution of f' and g' so that we can later estimate the hardness of retrieving the short vector (f', g') as per lattice reduction algorithms.

Lemma 3. *Let $f_1, f_2 \in \mathcal{R}_{C_n^+}$ be independently and uniformly sampled from $\mathcal{T}(d_f, d_f)$. Let*

$$f' = f_1 * f_2 \in \mathcal{R}_{C_n^+}$$

denote their product. Then, for sufficiently large n, each coefficient of f' is approximately distributed as $\mathcal{N}\left(0, \frac{4d_f^2}{n}\right)$.

Proof. Let W_k denote the k-th coefficient of f'. Then W_k is a sum of $2n$ terms of the form:

$$W_k = \sum_{i+j \equiv k \bmod n} \lambda_{i,j} \cdot (f_1)_i \cdot (f_2)_j$$

where $\lambda_{i,j} \in \{+1, -1\}$ are the twisting factors from the reduction rule $x^n \equiv -1$. For every pair (i, j):

$$\mathbb{E}[(f_1)_i (f_2)_j] = 0 \qquad \text{and} \qquad \mathrm{Var}[(f_1)_i (f_2)_j] = \frac{4d_f^2}{n^2}.$$

Consequently

$$\mathbb{E}[W_k] = 0 \qquad \text{and} \qquad \mathrm{Var}[W_k] = \frac{4d_f^2}{n}$$

Hence, by the Central Limit Theorem: $V_k \sim \mathcal{N}(0, \sigma_{f'}^2)$, where $\sigma_{f'}^2 = 4d_f^2/n$.

Lemma 4. *Let $\mathbf{G} \in M_n(\mathbb{Z})$ be a random ternary matrix whose columns are independently and uniformly sampled from $\mathcal{T}(d_g, d_g)$. Let*

$$\mathbf{G}' = \sum_{j=0}^{2n-1} \mathbf{X}^j \mathbf{G} \mathbf{X}^{-j},$$

and let $g' \in \mathcal{R}_{C_n^+}$ be the polynomial corresponding to the first row of $\mathbf{G}'$. Furthermore, let V_k denote the k-th coefficient of g'. Then, for sufficiently large n, each coefficient of g' is approximately distributed as $\mathcal{N}(0, 8d_g)$.

Proof. Since $\mathbf{X}^{j+n} = -\mathbf{X}^j$, we have

$$V_k = 2 \sum_{j=0}^{n-1} \epsilon_j \cdot \mathbf{G}_{j,\,(k+j) \bmod n}$$

where the row index j and column index $(k+j) \bmod n$ arise from the action of $\mathbf{X}^j$, and $\epsilon_j \in \{+1, -1\}$ are fixed signs from $\mathbf{X}^n = -\mathbf{I}$. Let $S_j = \epsilon_j \cdot G_{j,\,(k+j) \bmod n}$. Since ϵ_j is a fixed sign and G_{ab} takes values $+1$ and -1 each with probability d_g/n and 0 otherwise:

$$\mathbb{E}[S_j] = 0 \qquad \text{and} \qquad \mathbb{E}[S_j^2] = \epsilon_j^2 \cdot \mathbb{E}[G_{j,\,(k+j) \bmod n}^2] = 1 \cdot \frac{2d_g}{n} = \frac{2d_g}{n}$$

As j ranges over $0, \ldots, n-1$, the row index j cycles through all n distinct rows of G, so the terms S_j are independent. Consequently:

$$\mathbb{E}[V_k] = 0 \qquad \text{and} \qquad \mathrm{Var}[V_k] = 4 \sum_{j=0}^{n-1} \mathrm{Var}(S_j) = 4 \cdot n \cdot \frac{2d_g}{n} = 8d_g$$

Hence, by the Central Limit Theorem, for large n:

$$V_k \sim \mathcal{N}(0,\, \sigma_{g'}^2) \qquad \text{where} \qquad \sigma_{g'}^2 = 8d_g$$

As a result, the norm of the short vector f', g' grows significantly compared to the norm of the analogue commutative instantiation and hence requires more effort in lattice reduction attacks. Concrete evaluation is given in the Section 5 corresponding to the parameter selection.

4.2 Hardness Assumptions and Security Proof

The design modifications introduced by the negacyclic setting of $\mathcal{R}_{C_n^+} = \mathbb{Z}[x]/(x^n + 1)$ necessitate adapting the standard NTRU assumptions to this new framework. This subsection formally defines the *decisional u-ANTRU* and *search u-LWE* assumptions and presents a reduction showing that breaking the OW-CPA security of the proposed scheme is at least as hard as solving these problems.

Definition 17 (u-ANTRU distribution). *Let $\mathcal{R}_{C_n^+} = \mathbb{Z}[x]/(x^n+1)$, χ_f be a distribution on $\mathcal{R}_{C_n^+}$, and $\chi_{\mathbf{G}}$ be a distribution on $M_n(\mathbb{Z})$. The unstructured-ANTRU or u-ANTRU distribution is defined as*

$$\text{u-ANTRU}_{\mathcal{R}_{C_n^+},\chi_f,\chi_{\mathbf{G}}} = \left\{\mathbf{H} = \mathbf{F}_1^{-1}\mathbf{G}\mathbf{F}_2^{-1} \pmod q \;\middle|\; \mathbf{F}_1, \mathbf{F}_2 \leftarrow_{\$} \chi_f,\ \mathbf{G} \leftarrow_{\$} \chi_{\mathbf{G}}\right\}$$

where $\mathbf{F}_1, \mathbf{F}_2 \in \mathbf{RC}^{n\times n}$ are twisted right circulant matrices sampled from χ_f, and $\mathbf{G} \in M_n(\mathbb{Z})$ is a general random matrix sampled from $\chi_{\mathbf{G}}$.

Definition 18 (u-ANTRU problem). *Let $\mathcal{R}_{C_n^+} = \mathbb{Z}[x]/(x^n+1)$, χ_f be a distribution on $\mathcal{R}_{C_n^+}$, and $\chi_{\mathbf{G}}$ be a distribution on $M_n(\mathbb{Z})$.*

1. *The **decisional** unstructured-ANTRU or decisional u-ANTRU problem is to distinguish the $\text{u-ANTRU}_{\mathcal{R}_{C_n^+},\chi_f,\chi_{\mathbf{G}}}$ distribution from the uniform distribution on $M_n(\mathbb{Z}_q)$, given the same number of independent samples from both distributions.*
2. *The **search** unstructured-ANTRU or search u-ANTRU problem is to find $(\mathbf{F}_1, \mathbf{F}_2, \mathbf{G})$, given $\mathbf{H}$ sampled from the u-ANTRU distribution.*

Assumption 1 (Decisional u-ANTRU assumption) *For any PPT adversary $\mathcal{A}$, the advantage to successfully solve the decisional u-ANTRU problem is negligible, i.e.,*

$$\begin{aligned}\mathsf{Adv}^{\text{u-ANTRU}}_{\mathcal{R}_{C_n^+},\chi_f,\chi_g}(\mathcal{A}) = \big|\Pr[b=1 \mid \mathbf{H} \leftarrow \text{u-ANTRU}_{\mathcal{R}_{C_n^+},\chi_f,\chi_{\mathbf{G}}}\ ;\ b \leftarrow \mathcal{A}(\mathbf{H})] \\ - \Pr[b=1 \mid \mathbf{U} \leftarrow_{\$} M_n(\mathbb{Z}_q)\ ;\ b \leftarrow \mathcal{A}(\mathbf{U})]\big|\end{aligned}$$

is negligible.

Definition 19 (u-LWE distribution). *Let $\mathcal{R}_{C_n^+} = \mathbb{Z}[x]/(x^n+1)$, χ_r be a distribution on $\mathcal{R}_{C_n^+}$, and $\chi_{\mathbf{M}}$ be a distribution on $M_n(\mathbb{Z})$. The unstructured LWE or u-LWE distribution is defined as*

$$\begin{aligned}\text{u-LWE}_{\mathcal{R}_{C_n^+},\chi_r,\chi_{\mathbf{M}}} = \big\{ (\mathbf{A},\ \mathbf{R}_1\mathbf{A}\mathbf{R}_2 + \mathbf{M} \pmod q) \;\big|\; \mathbf{R}_1, \mathbf{R}_2 \leftarrow_{\$} \chi_r, \\ \mathbf{M} \leftarrow \chi_{\mathbf{M}},\ A \leftarrow_{\$} M_n(\mathbb{Z}_q) \big\}\end{aligned}$$

where $\mathbf{R}_1, \mathbf{R}_2 \in \mathbf{RC}^{n\times n}$ are twisted right circulant matrices and $\mathbf{M} \in M_n(\mathbb{Z}_q)$ is a general random matrix.

Definition 20 (search u-LWE problem). *Let $\mathcal{R}_{C_n^+} = \mathbb{Z}[x]/(x^n+1)$, χ_r be a distribution on $\mathcal{R}_{C_n^+}$, and $\chi_{\mathbf{M}}$ be a distribution on $M_n(\mathbb{Z})$. The search unstructured LWE or search u-LWE problem is to find $(\mathbf{R}_1, \mathbf{R}_2, \mathbf{M})$, given $(\mathbf{A},\ \mathbf{R}_1\mathbf{A}\mathbf{R}_2 + \mathbf{M} \pmod q) \leftarrow \text{u-LWE}_{\mathcal{R}_{C_n^+},\chi_r,\chi_{\mathbf{M}}}$.*

Assumption 2 (search u-LWE assumption) *For any PPT adversary $\mathcal{A}$, the advantage to successfully solve the search u-LWE problem is negligible, i.e.,*

$$\mathsf{Adv}^{\text{u-LWE}}_{\mathcal{R}_{C_n^+},\chi_r,\chi_{\mathbf{M}}}(\mathcal{A}) = \Pr_{\substack{\mathbf{R}_1,\mathbf{R}_2\leftarrow\chi_r,\ \mathbf{M}\leftarrow\chi_{\mathbf{M}} \\ \mathbf{A}\leftarrow_{\$} M_n(\mathbb{Z}_q)}}[(\mathbf{R}_1, \mathbf{R}_2, \mathbf{M}) \leftarrow \mathcal{A}(\mathbf{A},\ \mathbf{R}_1\mathbf{A}\mathbf{R}_2 + \mathbf{M} \pmod q)]$$

is negligible.

Theorem 1 (OW-CPA security). *Let $\mathcal{A}$ be a PPT adversary against the* OW-CPA *security of the scheme $\prod_{\text{u-ANTRU.PKE}}$. Then there exists an adversary $\mathcal{B}$ against the decisional u-ANTRU assumption and an adversary $\mathcal{C}$ against the search u-LWE assumption such that*

$$\mathsf{Adv}^{\text{OW-CPA}}_{\prod_{\text{u-ANTRU.PKE}}}(\mathcal{A}) \leq \mathsf{Adv}^{\text{u-ANTRU}}_{\mathcal{R}_{C_n^+},\chi_f,\chi_{\mathbf{G}}}(\mathcal{B}) + \mathsf{Adv}^{\text{u-LWE}}_{\mathcal{R}_{C_n^+},\chi_r,\chi_{\mathbf{M}}}(\mathcal{C}) \tag{10}$$

where both $\mathcal{B}$ and $\mathcal{C}$ have roughly the same running time as $\mathcal{A}$.

Proof. We consider the following sequence of games G_i to prove the OW-CPA security of our scheme $\prod_{\text{u-ANTRU.PKE}}$. Let $\mathsf{Adv}^{\text{OW-CPA}}_{G_i}$ denote the probability of winning game G_i.

Game G_0: This is the genuine OW-CPA game as shown in Figure 2. Then, by definition:

$$\mathsf{Adv}^{\text{OW-CPA}}_{G_0} = \mathsf{Adv}^{\text{OW-CPA}}_{\prod_{\text{u-ANTRU.PKE}}}(\mathcal{A})$$

Game G_1: This game is identical to game G_0 except that the public key $\mathbf{H} = \mathbf{F}_1^{-1}\mathbf{G}\mathbf{F}_2^{-1} \pmod q$ is replaced by a matrix $\mathbf{A}$ sampled uniformly at random from $M_n(\mathbb{Z}_q)$. Using the decisional u-ANTRU assumption, any PPT adversary $\mathcal{B}$ has negligible advantage in distinguishing $\mathbf{H}$ from a uniformly random matrix, hence:

$$\left|\mathsf{Adv}^{\text{OW-CPA}}_{G_0} - \mathsf{Adv}^{\text{OW-CPA}}_{G_1}\right| \leq \mathsf{Adv}^{\text{u-ANTRU}}_{\mathcal{R}_{C_n^+},\chi_f,\chi_{\mathbf{G}}}(\mathcal{B})$$

Game G_2: This game is identical to game G_1 except that the uniformly random matrix $\mathbf{A} \in M_n(\mathbb{Z}_q)$ in game $\mathbf{G}_1$ is replaced by $p^{-1}\mathbf{A} \pmod q$, where p is coprime to q. Since $\mathbf{A}$ is sampled uniformly at random from $M_n(\mathbb{Z}_q)$ and $\gcd(p, q) = 1$, the element $p \in \mathbb{Z}_q$ is invertible, hence left-multiplication by $p^{-1} \pmod q$ defines a bijection on $M_n(\mathbb{Z}_q)$. Therefore, the distribution of $\mathbf{A}' = p^{-1}\mathbf{A} \pmod q$ is also uniform over $M_n(\mathbb{Z}_q)$. Equivalently, if $\mathbf{A}$ is uniform in $M_n(\mathbb{Z}_q)$, then so is $p^{-1}\mathbf{A}$. As a result:

$$\mathsf{Adv}^{\text{OW-CPA}}_{G_1} = \mathsf{Adv}^{\text{OW-CPA}}_{G_2}$$

Moreover, under the search u-LWE assumption, any PPT adversary $\mathcal{C}$ has negligible advantage in recovering $(\mathbf{R}_1, \mathbf{R}_2, \mathbf{M})$ from $(\mathbf{A},\ \mathbf{R}_1\mathbf{A}\mathbf{R}_2 + \mathbf{M} \pmod q)$, hence:

$$\mathsf{Adv}^{\text{OW-CPA}}_{G_2} = \mathsf{Adv}^{\text{u-LWE}}_{\mathcal{R}_{C_n^+},\chi_r,\chi_{\mathbf{M}}}(\mathcal{C})$$

Combining the three games, we conclude:

$$\mathsf{Adv}^{\text{OW-CPA}}_{\prod_{\text{u-ANTRU.PKE}}}(\mathcal{A}) \leq \mathsf{Adv}^{\text{u-ANTRU}}_{\mathcal{R}_{C_n^+},\chi_f,\chi_{\mathbf{G}}}(\mathcal{B}) + \mathsf{Adv}^{\text{u-LWE}}_{\mathcal{R}_{C_n^+},\chi_r,\chi_{\mathbf{M}}}(\mathcal{C})$$

□

4.3 FO Transformation for IND-CCA KEM from OW-CPA PKE

Applying standard transformations like the Fujisaki–Okamoto (FO) transformation [16,17], an OW-CPA secure PKE leads to an IND-CCA secure KEM. We instantiate our KEM following the FO variant described in Fig. 5. The resultant KEM is an instance of $\mathsf{KEM}_m^{\perp}$, discussed in [24], whose security is supported by a tight reduction in the ROM (random-oracle model) and a non-tight reduction in the QROM (quantum random oracle model) to the OW-CPA security of the underlying PKE, as stated in the following theorem adapted from [24,12].

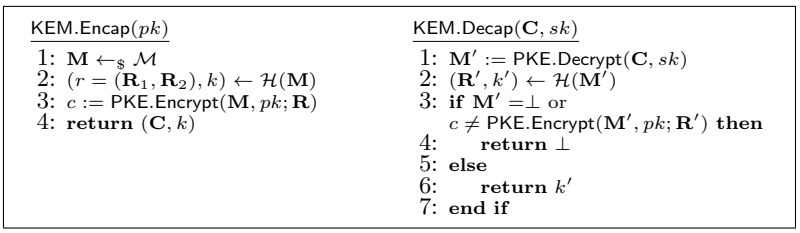


Fig. 5: General framework of Fujisaki-Okamoto transformation.

Theorem 2. *[24,12] Let* $\prod_{\mathsf{PKE}}$ *be a* δ*-correct and* γ *-spread public key encryption scheme. For any adversary* $\mathcal{A}$ *making at most* q_D *and* q_H *decapsulation and hash queries, respectively, against the* IND-CCA *security of the scheme* $\prod_{\mathsf{KEM}}$ *, there exists an adversary* $\mathcal{A}'$ *against the* OW-CPA *security of* $\prod_{\mathsf{PKE}}$ *such that*

- $\mathsf{Adv}^{\mathsf{IND-CCA}}_{\prod_{\mathsf{KEM}}}(\mathcal{A}) \leq 2q_H \cdot \mathsf{Adv}^{\mathsf{OW-CPA}}_{\prod_{\mathsf{PKE}}}(\mathcal{A}') + q_H 2^{-\gamma} + (2q_H + q_D) \cdot \delta$ *, with* $\mathsf{Time}(\mathcal{A}') \approx \mathsf{Time}(\mathcal{A})$ *, when the adversary has access to classical random oracles.*

- $\mathsf{Adv}^{\mathsf{IND-CCA}}_{\prod_{\mathsf{KEM}}}(\mathcal{A}) \leq 2q_T \sqrt{\mathsf{Adv}^{\mathsf{OW-CPA}}_{\prod_{\mathsf{PKE}}}(\mathcal{A}')} + 24q_T^2\sqrt{\delta} + 24\sqrt{q_T^3 q_D} \cdot 2^{-\gamma/4}$ *, with* $q_T = 2(q_D + q_H)$, $\mathsf{Time}(\mathcal{A}') \approx \mathsf{Time}(\mathcal{A}) + O(q_D \cdot q_H \mathsf{Time}(\mathsf{PKE.Encrypt}) + q_T^2)$ *, when* $\mathcal{A}$ *is a quantum adversary having access to quantum random oracles.*

4.4 Concrete Security Estimation

We estimate the cost of the concrete attacks and parametrize our cryptosystem accordingly. The security of the KEM is based on the OW-CPA of the underlying ANTRU problem, and hence a successful attack will require either retrieving a possible decryption key or the message. An attacker can opt for the following two approaches:

- **Search attack:** Brute force search for $f_1, f_2 \in \mathcal{L}_f$, and check if $f_1 * h * f_2 \in \mathcal{L}_g$, which costs
$$|\mathcal{L}_f \times \mathcal{L}_f| \approx \binom{n}{d_f}^2 \binom{n-d_f}{d_f}^2.$$
Although a straightforward meet-in-the-middle attack does not seem to be applicable, we conservatively estimate the search space size to be $\sqrt{|\mathcal{L}_f \times \mathcal{L}_f|}$.

– **Lattice attack:** As discussed earlier, the map θ can be applied to the public-key equation $\mathbf{H} = \mathbf{F}_1^{-1}\mathbf{G}\mathbf{F}_2^{-1} \pmod q$, resulting in an NTRU-like equation of the form $\theta(\mathbf{H}) = (\mathbf{F}_1\mathbf{F}_2)^{-1}\theta(\mathbf{G}) \pmod q$, which can be treated as an NTRU instance of the form $h' = f'^{-1} * g' \pmod q$, where f' is the polynomial corresponding to the first row of $\mathbf{F}_1\mathbf{F}_2$ and, as per Lemma 3, its coefficients follow approximately the distribution $\mathcal{N}(0, 4d_f^2/n)$. Similarly, g' is the polynomial corresponding to the first row of $\theta(\mathbf{G})$ and, as per Lemma 4, its coefficients follow approximately the distribution $\mathcal{N}(0, 8d_g)$. The norm of the target vector (f', g') is estimated to be $\sqrt{4d_f^2 + 8nd_g}$, which significantly increases the target vector norm compared to the case where the key is constructed from a similar equation over a commutative algebra. Hence, the cost of primal lattice attacks, as discussed later in Subsection 5.3, is higher. Additionally, hybrid attacks perform less efficiently since the target vector distribution is no longer ternary or sufficiently sparse to enable a successful trade-off between the search and lattice reduction costs. Even though recovering f' and g' does not provide a straightforward method for recovering f_1 and f_2 or the message $\mathbf{M}$, we conservatively regard the recovery of f' and g' as a successful attack when estimating the security level.
– **Combined approach**[4] In this approach, the attacker solves a *two-layer NTRU* problem. Specifically, the attacker first performs a brute-force search over $f_1 \in \mathcal{L}_f$ and then, using its matrix representation $\mathbf{F}_1$, computes

$$\mathbf{F}_1\mathbf{H} = \mathbf{G}\mathbf{F}_2^{-1} \pmod q,$$

which is in the form of a Matrix-NTRU problem. Therefore, the attacker needs to perform lattice reduction in a $2n$-dimensional lattice generated by the matrix

$$\begin{pmatrix} \mathbf{I}_n & \mathbf{F}_1\mathbf{H} \\ \mathbf{0}_n & q\mathbf{I}_n \end{pmatrix}.$$

Therefore, for the correct guess of $\mathbf{F}_1$ reducing the lattice enables retrieving the other parts of the secrete $\mathbf{F}_2$ and $\mathbf{G}$. However, we very conservatively (favouring the attacker) assume that guessing f_1 correctly is a valid attack. Thus, reducing the cost of attack to just $|\mathcal{L}_f|$. It is important to emphasize that a meet-in-the-middle (MitM) attack is not applicable in this context, since $f_1 * h$ constitutes another NTRU instance rather than a short vector as in conventional MitM attacks. Nevertheless, for a conservative security estimation, we still consider the cost of the search attack to be $\sqrt{|\mathcal{L}_f|}$.
Concrete security against lattice attacks is estimated based on the hardness of core-SVP [1] for BKZ-2.0 [7,2]. As mentioned earlier, the cost of BKZ with parameter β (known as blocksize) dominates the estimation. We used the NTRU fatigue estimator [13] to estimate the blocksize β, as it provides a reliable approximation for NTRU-like lattices. The estimator is based on the probabilistic simulation method [11], which is a revised version of the 2016

[4] We refer to it as the combined approach to avoid confusion with the term hybrid, which may be confused with hybrid lattice attacks.

estimation [1]. Consequently, the core-SVP cost of BKZ when combined with a sieving oracle is estimated to be $2^{0.292\beta+o(\beta)}$ (classical) [4] and $2^{0.265\beta+o(\beta)}$ (quantum) [35].

5 Parameter Selection

5.1 Analysis of Decryption Failure

The s-ANTRU encryption scheme requires the condition $4(pd_r^2 + d_f^2) < \frac{q}{2}$ for perfect correctness. However, by tolerating decryption failures within acceptable security thresholds, we can reduce the modulus q, thereby lowering bandwidth requirements. The following theorem gives the probability of decryption failure.

Theorem 3. *For the s-ANTRU encryption scheme with the secrets $f_i \in \mathcal{L}_f, g \in \mathcal{L}_g$, randomnesses $r_i \in \mathcal{L}_r$, and the message $m \in \mathcal{L}_m$, the probability of decryption failure is estimated as*

$$P_{dec_fail}^{N,q} = 4N \times \left(1 - \Phi\left(\frac{q}{2\tilde{\sigma}}\right)\right) < 2N \times \mathsf{erfc}\left(\frac{q}{2\sqrt{2}\tilde{\sigma}}\right), \tag{11}$$

where Φ is the cdf (cumulative distribution function) of the standard Normal distribution, erfc *is the complementary error function, and $\tilde{\sigma}^2 = \frac{4}{N}(p^2 d_r^2 d_g + d_f^2 d_m)$.*

The ciphertext equation is given by

$$\mathbf{C} = p(\mathbf{R}_1\mathbf{H}\mathbf{R}_2) + \mathbf{M} \pmod q. \tag{12}$$

For decryption, the receiver performs

$$\mathbf{F}_1\mathbf{C}\mathbf{F}_2 = p(\mathbf{R}_1\mathbf{G}\mathbf{R}_2) + \mathbf{F}_1\mathbf{M}\mathbf{F}_2 \pmod q. \tag{13}$$

Therefore, the correct decryption requires

$$\|p(\mathbf{R}_1\mathbf{G}\mathbf{R}_2) + \mathbf{F}_1\mathbf{M}\mathbf{F}_2\|_\infty < q/2.$$

where $\mathbf{R}_1, \mathbf{R}_2, \mathbf{F}_1$, and $\mathbf{F}_2$ are the matrix representation of r_1, r_2, f_1 and f_2 from $\mathcal{R}_{C_n^+}$, respectively. We have $r_1, r_2 \in \mathcal{L}_r$, $g \in \mathcal{L}_\mathbf{G}$ (each column has d_g ones and d_g minus ones and remaining coefficients are zeros), $f_1, f_2 \in \mathcal{T}(d_f, d_f)$ (instead of $\mathcal{L}_f$ for simplicity), and $\mathbf{M} \in \mathcal{L}_\mathbf{M}$(similar distribution of $\mathbf{G}$) Let us assume that $d_f \le d_r \le d_g \le d_m$. Then,

$$\|p(\mathbf{R}_1\mathbf{G}\mathbf{R}_2) + \mathbf{F}_1\mathbf{M}\mathbf{F}_2\|_\infty \le 4(pd_r^2 + d_f^2).$$

Consequently, perfect correctness can be achieved if

$$4(pd_r^2 + d_f^2) < q/2.$$

Therefore, even for very sparse f_i, r_i, allowing no decryption failure results in a significantly larger value for the modulus q, which consequently increases the

sizes of the public key and ciphertext. However, by permitting negligible decryption failures, in line with NIST guidelines, it is possible to choose smaller values for q.

We analyze the distribution of coefficients of $\mathbf{F}_1\mathbf{M}\mathbf{F}_2$, and similar results will hold for $\mathbf{R}_1\mathbf{G}\mathbf{R}_2$. Let U_k denotes the kth coefficient of the first row of $\mathbf{U} = \mathbf{F}_1\mathbf{M}$, then U_k is the sum of n terms of the form

$$U_k = \mathbf{F}_1^{(0,:)}\mathbf{M}^{(k,:)} \text{ (summing over } n \text{ pairs corresponding to } k).$$

Therefore,

$$E[\mathbf{F}_1^{(0,:)}\mathbf{M}^{(k,:)}] = 0 \text{ and } Var[\mathbf{F}_1^{(0,:)}\mathbf{M}^{(k,:)}] = \frac{4d_f d_m}{n^2}.$$

Hence,

$$E[U_k] = 0 \text{ and } Var[U_k] = \frac{4d_f d_m}{n}.$$

By the central limit theorem, for large N,

$$U_k \sim N(0, \sigma_1^2) \text{ where } \sigma_1^2 = \frac{4d_f d_m}{n}.$$

Let $\mathbf{V} = \mathbf{U}\mathbf{F}_2$, and the random variable V_k denotes the kth coefficient of the first row of $\mathbf{V}$. Then, again V_k is the sum of n terms of the form

$$V_k = \mathbf{U}^{(0,:)}\mathbf{F}_2^{(k,:)} \text{ (summing over } n \text{ pairs corresponding to } k).$$

Since f_2 takes exactly $2d_f$ nonzero values equal to ± 1, therefore, V_k is the sum of $2d_f$ random variables following $N(0, \sigma_1^2)$. As a result,

$$V_k \sim N(0, 2d_f\sigma_1^2). \tag{14}$$

Similarly, let the random variable W_k denotes the kth coefficient of the first row of $\mathbf{R}_1\mathbf{G}\mathbf{R}_2$ then

$$W_k \sim N(0, 2d_r\sigma_2^2) \tag{15}$$

where $\sigma_2^2 = 2d_r d_g/N$. For ease of calculations, let us assume that $\mathbf{G}$ and $\mathbf{M}$ follow same distribution, i.e., $d_g = d_m$, and r_i and f_i follow same distribution, i.e., $d_r = d_f$. This gives $\sigma_1^2 = \sigma_2^2 = \sigma^2$ (let), and $W_k \sim V_k$. Now, let us define a random variable Y_k as

$$Y_k = pW_k + V_k,$$

i.e., Y_k denotes the kth coefficient of the first row of $p(\mathbf{R}_1\mathbf{G}\mathbf{R}_2) + \mathbf{F}_1\mathbf{M}\mathbf{F}_2$. We have

$$Y_k \sim N(0, 2d_f(p^2+1)\sigma^2), \tag{16}$$

Other rows of the matrix $p(\mathbf{R}_1\mathbf{G}\mathbf{R}_2) + \mathbf{F}_1\mathbf{M}\mathbf{F}_2$ follows similar pattern and hence each entry of the matrix is following the distribution $N(0, 2d_f(p^2+1)\sigma^2)$. with

$$E[Y_k] = 0 \text{ and } Var[Y_k] = 2d_f\sigma^2(p^2+1) \ (= \tilde{\sigma}^2, \text{ let}). \tag{17}$$

For $q > 0$, consider

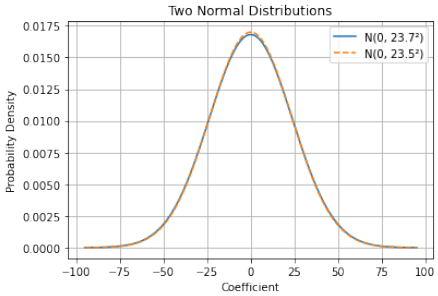
$$\begin{aligned}\Pr[|Y_k| > q/2] &= 2(1 - \Pr[Y_k \leq q/2]) \\ &= 2\left(1 - \Phi_{\tilde{\sigma}}\left(\frac{q}{2}\right)\right) \\ &= 2\left(1 - \Phi\left(\frac{q}{2\tilde{\sigma}}\right)\right) \end{aligned} \tag{18}$$

where $\Phi_{\tilde{\sigma}}(z)$ is the cdf of $N(0, \tilde{\sigma}^2)$, and $\Phi_{\tilde{\sigma}}(z) = \Phi(z/\tilde{\sigma})$, where Φ is the cdf of standard normal distribution $N(0, 1)$.

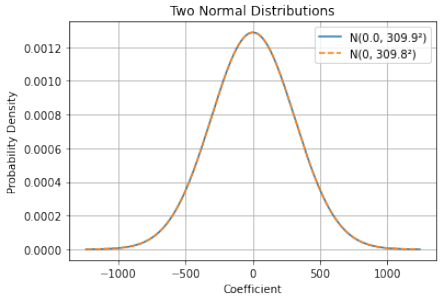
For a given $q > 0$, to compute the probability of decryption failure, we need to compute the probability of the event when any coefficient of $p(\mathbf{R}_1\mathbf{GR}2)+\mathbf{F}_1\mathbf{MF}_2$ exceeds $q/2$, which is equal to

$$\begin{aligned} P_{dec_fail}^{N,q} &= 2N \Pr[|Y_k| > q/2] \\ &= 4N\left(1 - \Phi\left(\frac{q}{2\tilde{\sigma}}\right)\right) \qquad (19) \\ &< 2N\mathsf{erfc}\left(\frac{q}{2\sqrt{2}\tilde{\sigma}}\right), \qquad (20) \end{aligned}$$

where erfc is the complementary error function.



(a) Distribution of V_k with $\sigma \approx 23.7$ (experimental in blue) vs 23.5 (theoretical in orange).



(b) Distribution of Y_k with $\sigma \approx 309.9$ (experimental in blue) vs 309.8 (theoretical in orange).

Fig. 6: Comparison of the experimental and theoretical normal distributions of the coefficients for $n = 512$, $q = 7681$, and $d_f = d_g = d_r = d_m = d = \lfloor n/6 \rfloor$. The figures demonstrate the near-perfect overlap between the experimental and theoretical distributions.

5.2 Non Overstretched Parameters

Ducas and Woerdan [13], refining the work in [31], demonstrated that the BKZ algorithm performs better against NTRU lattices with a very large modulus q than previously estimated by the 2016-estimator [1]. This improvement is attributed to a phenomenon known as Dense Sublattice Discovery (DSD). Essentially, BKZ discovers a basis for the dense sublattice formed by the rotations

of the short secret key prior to the actual Secret Key Recovery (SKR). This phenomenon is also observed in GR-NTRU lattices, which likewise contain a sublattice of half the dimension formed by the rotations of the secret key.

The term *Fatigue Point* is used for the modulus q value that separates the regular region and the overstretched region. Ducas and Woerdan established the following claim to provide an estimation of the Fatigue Point.

Theorem 4. *[13, Claim 3.5] The BKZ algorithm with blocksize $\beta = \mathcal{B}n$, applied to an NTRU instance with parameter $q = \Theta(n^{\mathcal{Q}})$ and secret $\|(f, g)\| = O(n^{\mathcal{P}})$, triggers the DSD event if*

$$\mathcal{B} = \frac{8\mathcal{P}}{\mathcal{Q}+1} + o(1). \tag{21}$$

The relative blocksizes given by the 2016-estimate are $\frac{2\mathcal{Q}}{(\mathcal{Q}+1-\mathcal{P})^2}$ for $\mathcal{P} < 1$ and $\frac{2}{\mathcal{Q}+2-2\mathcal{P}}$ for $\mathcal{P} \geq 1$ [13]. Comparing the relative blocksize of the Fatigue estimation and the 2016-estimation yields the following conditions for the Fatigue Point:

$$q = n^{2.484+o(1)} \text{ for } \mathcal{P} < 1 \text{ and } q = n^{3.732+o(1)} \text{ for } \mathcal{P} \geq 1. \tag{22}$$

Empirical results in [13] for the ternary secret vectors or secrets sampled from a discrete Gaussian distribution with a mean of zero and a variance of $\frac{2}{3}$, in general, the norm of the secret is $O(\sqrt{N})$ (i.e., $\mathcal{P} = \frac{1}{2}$) provides a concrete bound $q \leq 0.004 \times n^{2.484}$ for parameter sets to be non overstretched. However, in our case, the norm of the secret is

$$\|(f', g')\| \approx \|(f_1 * f_2, \theta(\mathbf{G}))\| = O(N), \text{ i.e., } \mathcal{P} = 1,$$

as per the discussion provided by Lemma 3 and 4. The experimental results indicate that our design parameters do not fall into the overstretched regime when $q \leq 0.00006 \times n^{3.732}$. It can be verified that the selected parameters do not lie in the overstretched regime.

5.3 Concrete Parametrization

Following the concrete security estimation, Tables 2 and 3 give two sets of parameters for u-ANTRU. Table 2 targets 128-, 192-, and 256-bit security, while Table 3 gives NTT-friendly parameter sets. We compare these to matrix NTRU at equivalent security levels and observe that lattice attacks against ANTRU are more expensive than those against matrix NTRU at similar dimensions, since applying $\theta(\mathbf{H})$ increases the norm of $\mathbf{G}'$ and consequently requires a larger blocksize to recover the private key (f', g') via lattice reduction.

5.4 Public Key Size

Both ANTRU and matrix NTRU represent the public key as an $n \times n$ matrix over $\mathbb{Z}_q$, so the raw public-key size is $n^2 \log_2 q$ bits in both cases, independent of the underlying group structure. Since u-ANTRU reaches a given security

Table 2: Parameter sets for ANTRU, targeting security levels of 128-, 192-, and 256-bits of security

Core-SVP	(n, d, q)	**GH**$(\mathcal{L})$	$\|(\mathbf{f}', \mathbf{g}')\|$	**Classical**		**Quantum**		**Combined Attack**	**Dec.** $\log_2(\delta)$
				Primal		**Primal**			
				β	Cost	β	Cost		
128	(263, 42, 2^{12})	335.17	308.91	484	**141**	484	128	157	< -129
192	(379, 66, 2^{13})	602.97	466.41	664	**194**	664	176	239	< -194
256	(503, 66, 2^{13})	694.64	531.99	885	**258**	885	235	270	< -259

Table 3: NTT-friendly parameter sets for ANTRU, targeting security levels of 128- and 256-bits of security

Core-SVP	(n, d, q)	**GH**$(\mathcal{L})$	$\|(\mathbf{f}', \mathbf{g}')\|$	**Classical**		**Quantum**		**Combined Attack**	**Dec.** $\log_2(\delta)$
				Primal		**Primal**			
				β	Cost	β	Cost		
128	(256, 63, 7681)	429.18	380.66	447	**131**	447	118	187	< -131
256	(512, 87, 12289)	1108.11	621.79	879	**257**	879	233	319	< -259

Table 4: Parameter sets for Matrix NTRU, targeting security levels of 128, 192, and 256 bits.

Core-SVP	$(\mathbf{n}, \mathbf{d}, \mathbf{q})$	**GH**$(\mathcal{L})$	$\|(\mathbf{f}', \mathbf{g}')\|$	**Classical**					**Quantum**				
				Primal		**Hybrid**			**Primal**		**Hybrid**		
				β	Cost	K	β	Cost	β	Cost	K	β	Cost
128	(587, 195, 2048)	265.3	27.95	456	133	166	438	**128**	456	121	156	454	**120**
192	(863, 159, 2048)	321	25.24	701	205	298	658	**192**	701	186	282	684	**181**
256	(1109, 369, 4096)	515.71	38.43	893	261	331	883	**258**	893	**237**	311	915	242

level at a substantially smaller n than matrix NTRU (Table 2 vs. Table 4), the resulting public keys are correspondingly smaller despite the identical matrix representation. Table 5 gives the public-key size at each security level for both constructions; u-ANTRU keys are 4.1–4.6× smaller, a reduction of roughly 76–78%.

Table 5: Public key size ($n^2 \log_2 q$ bits) for ANTRU vs. matrix NTRU at matched security levels.

Security	**ANTRU** (n, q)	**ANTRU size**	**Matrix NTRU** (n, q)	**Matrix NTRU size**	**Reduction**
128	(263, 2^{12})	101.3 KB	(587, 2048)	462.7 KB	78.1%
192	(379, 2^{13})	228.0 KB	(863, 2048)	1000.1 KB	77.2%
256	(503, 2^{13})	401.5 KB	(1109, 4096)	1801.6 KB	77.7%
128 (NTT)	(256, 7681)	103.3 KB	(587, 2048)	462.7 KB	77.7%
256 (NTT)	(512, 12289)	434.7 KB	(1109, 4096)	1801.6 KB	75.9%

5.5 Computational Cost

Multiplying two random $n \times n$ matrices is known to require $\Omega(n^2)$ operations, since every entry of the input must be read at least once, and classically requires $O(n^3)$ arithmetic operations. This can be improved via Strassen's method [45] to $O(n^{2.807})$, and theoretically further reduced to $O(n^{2.372})$ using the fastest known algorithms [10,47]. However, despite these theoretical gains, only Strassen's algorithm sees practical use, as algorithms achieving exponents closer to the theoretical bound carry impractically large hidden constants.

Inverting an $n \times n$ matrix over a field has the same asymptotic complexity as matrix multiplication, since inversion can be expressed entirely in terms of smaller inversions together with a constant number of matrix multiplications and additions of sub-blocks, which recursively inherit the same asymptotic cost as multiplication, as shown in [25].

ANTRU requires inverting two elements from $\mathcal{R}_{C_n^+}$ and two matrix multiplications during key generation, compared to inverting one matrix and performing one matrix multiplication in Matrix NTRU. Inverting an element over $\mathcal{R}_{C_n^+}$ can be performed efficiently in $O(n^2)$, where n denotes the ring dimension [44]. In contrast, inverting an $n \times n$ matrix over a field costs $O(n^3)$ arithmetic operations classically; using fast matrix multiplication this improves to $O(n^\omega)$, where $\omega \approx 2.372$ for the best known algorithms, though large constants and poor numerical stability keep $O(n^3)$ the practical standard.

For ANTRU, the two required matrix multiplications can each be computed in $O(n^2 \log_2 n)$ for NTT-friendly parameter sets, as established in Theorem 5, so their combined cost remains $O(n^2 \log_2 n)$. In contrast, the single matrix multiplication required for Matrix NTRU has a best-known theoretical complexity of $O(n^{2.372})$, and $O(n^3)$ in practice. Since $O(n^2 \log_2 n)$ is asymptotically smaller than both $O(n^{2.372})$ and $O(n^3)$, the two multiplications required by ANTRU are, in total, still more efficient than the single multiplication required by Matrix NTRU for comparable values of n. In practice, the value of n required for ANTRU is smaller than that of Matrix NTRU for equivalent security levels, yielding a further efficiency gain. The same comparison extends to the cost of encryption and decryption.

Theorem 5. *Let* $\mathbf{H} = \mathbf{F}_1^{-1}\mathbf{G}\mathbf{F}_2^{-1} \pmod q$ *be the public key of unstructured ANTRU, then the cost of computing the public key in* $O(n^2 \log_2(n))$ *for NTT-friendly parameters.*

Proof. Let $\mathbf{R}_0, \mathbf{H}$, and $\mathbf{R}_1 \in \mathbb{Z}^{n\times n}$. The product $\mathbf{R}_0\mathbf{H}\mathbf{R}_1$ is given by:

$$\mathbf{R}_0\mathbf{H}\mathbf{R}_1 = \begin{pmatrix} \langle (\mathbf{R}_0\mathbf{H})^{(0,:)}, \mathbf{R}_1^{(:,0)} \rangle & \langle (\mathbf{R}_0\mathbf{H})^{(0,:)}, \mathbf{R}_1^{(:,1)} \rangle & \dots & \langle (\mathbf{R}_0\mathbf{H})^{(0,:)}, \mathbf{R}_1^{(:,n-1)} \rangle \\ \langle (\mathbf{R}_0\mathbf{H})^{(1,:)}, \mathbf{R}_1^{(:,0)} \rangle & \langle (\mathbf{R}_0\mathbf{H})^{(1,:)}, \mathbf{R}_1^{(:,1)} \rangle & \dots & \langle (\mathbf{R}_0\mathbf{H})^{(1,:)}, \mathbf{R}_1^{(:,n-1)} \rangle \\ \vdots & \vdots & \ddots & \vdots \\ \langle (\mathbf{R}_0\mathbf{H})^{(n-1,:)}, \mathbf{R}_1^{(:,0)} \rangle & \langle (\mathbf{R}_0\mathbf{H})^{(n-1,:)}, \mathbf{R}_1^{(:,1)} \rangle & \dots & \langle (\mathbf{R}_0\mathbf{H})^{(n-1,:)}, \mathbf{R}_1^{(:,n-1)} \rangle \end{pmatrix}$$

Here, $(\mathbf{R}_0\mathbf{H})^{(i,:)}$ denotes the i-th row of the matrix product $\mathbf{R}_0\mathbf{H}$, and $\mathbf{R}_1^{(:,j)}$ denotes the j-th column of $\mathbf{R}_1$. Each entry of the resulting matrix is computed as the inner product of a row from $\mathbf{R}_0\mathbf{H}$ and a column from $\mathbf{R}_1$.

In the context of our constructions, we have:

$\mathbf{R}_0, \mathbf{R}_1 \in \mathcal{R}_{C_n^+}$, and $\mathbf{H} \in M_n(\mathbb{Z})$, i.e.,

$$\mathbf{R}_0 = \begin{pmatrix} r_{00} & r_{01} & \dots & r_{0(n-1)} \\ -r_{0(n-1)} & r_{00} & \dots & r_{0(n-2)} \\ \vdots & \vdots & \ddots & \vdots \\ -r_{01} & -r_{02} & \dots & r_{00} \end{pmatrix}, \ \mathbf{R}_1 = \begin{pmatrix} r_{10} & r_{11} & \dots & r_{1(n-1)} \\ -r_{1(n-1)} & r_{10} & \dots & r_{1(n-2)} \\ \vdots & \vdots & \ddots & \vdots \\ -r_{11} & -r_{12} & \dots & r_{10} \end{pmatrix}, \ \mathbf{H} = \begin{pmatrix} h_0 & h_1 & \dots & h_{n-1} \\ h_n & h_{n+1} & \dots & h_{2n-1} \\ \vdots & \vdots & \ddots & \vdots \\ h_{n(n-1)} & h_{n(n-1)+1} & \dots & h_{n^2-1} \end{pmatrix}$$

Therefore, the entry $\mathbf{R}_0\mathbf{H}\mathbf{R}_1^{(0,0)}$ has the following expanded form:

$$\begin{aligned}\langle(\mathbf{R}_0\mathbf{H})^{(0,:)},\, \mathbf{R}_1^{(:,0)}\rangle &= \sum_{i=0}^{n-1}(r_{0i}h_{in})\,r_{10} - \sum_{j=1}^{n-1}\sum_{i=0}^{n-1}(r_{0i}h_{in+j})\,r_{1(n-j) \bmod n} \\ &= (r_{00}h_0 + r_{01}h_n + \ldots + r_{0(n-1)}h_{n(n-1)})r_{10} \\ &\quad - (r_{00}h_1 + r_{01}h_{n+1} + \ldots + r_{0(n-1)}h_{n(n-1)+1})r_{1(n-1)} \\ &\quad - \ldots \\ &\quad - (r_{00}h_{n-1} + r_{01}h_{2n-1} + \ldots + r_{0(n-1)}h_{n^2-1})r_{11}.\end{aligned}$$

Similarly, other entries of the matrix $\mathbf{R}_0\mathbf{H}\mathbf{R}_1$ can be obtained. Therefore,

$$\overrightarrow{\mathbf{R}_0\mathbf{H}\mathbf{R}_1} = \left(\begin{array}{cccc:cccc:c:cccc:c}
h_0 & -h_{n-1} & \ldots & -h_1 & h_n & -h_{2n-1} & \ldots & -h_{n+1} & \ldots & h_{n(n-1)} & -h_{n^2-1} & \ldots & -h_{n(n-1)+1} & \begin{matrix}r_{00}r_{10}\\ r_{00}r_{11}\\ \vdots\\ r_{00}r_{1(n-1)}\end{matrix} \\
h_1 & h_0 & \ldots & -h_2 & h_{n+1} & h_n & \ldots & -h_{n+2} & \ldots & h_{n(n-1)+1} & h_{n(n-1)} & \ldots & -h_{n(n-1)+2} & \\
\vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \\
h_{n-1} & h_{n-2} & \ldots & h_0 & h_{2n-1} & h_{2n-2} & \ldots & h_n & \ldots & h_{n^2-1} & h_{n^2-2} & \ldots & h_{n(n-1)} & \\
\hdashline
h_n & -h_{2n-1} & \ldots & -h_{n+1} & h_{2n} & -h_{3n-1} & \ldots & -h_{2n+1} & \ldots & -h_0 & h_{n-1} & \ldots & h_1 & \begin{matrix}r_{01}r_{10}\\ r_{01}r_{11}\\ \vdots\\ r_{01}r_{1(n-1)}\end{matrix} \\
h_{n+1} & h_n & \ldots & -h_{n+2} & h_{2n+1} & h_{2n} & \ldots & -h_{2n+2} & \ldots & -h_1 & -h_0 & \ldots & h_2 & \\
\vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \\
h_{2n-1} & h_{2n-2} & \ldots & h_n & h_{3n-1} & h_{3n-2} & \ldots & h_{2n} & \ldots & -h_{n-1} & -h_{n-2} & \ldots & -h_0 & \\
\hdashline
\vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots \\
\hdashline
h_{n(n-1)} & -h_{n^2-1} & \ldots & -h_{n(n-1)+1} & -h_0 & h_{n-1} & \ldots & h_1 & \ldots & -h_{n(n-2)} & h_{n(n-1)-1} & \ldots & h_{n(n-2)+1} & r_{0(n-1)}r_{10} \\
h_{n(n-1)+1} & h_{n(n-1)} & \ldots & -h_{n(n-1)+2} & -h_1 & -h_0 & \ldots & h_2 & \ldots & -h_{n(n-2)+1} & -h_{n(n-2)} & \ldots & h_{n(n-2)+2} & r_{0(n-1)}r_{11} \\
\vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots \\
h_{n^2-1} & h_{n^2-2} & \ldots & h_{n(n-1)} & -h_{n-1} & -h_{n-2} & \ldots & -h_0 & \ldots & -h_{n(n-1)-1} & -h_{n(n-1)-2} & \ldots & -h_{n(n-2)} & r_{0(n-1)}r_{1(n-1)}
\end{array}\right)$$

This can be equivalently written using polynomial multiplication over the ring $\mathcal{R}_{C_n^+}$ as:

$$\overrightarrow{\mathbf{R}_0\mathbf{H}\mathbf{R}_1}^{\text{poly}} = \begin{pmatrix} h_0(x) & h_1(x) & \ldots & h_{n-1}(x) \\ h_1(x) & h_2(x) & \ldots & -h_0(x) \\ \vdots & \vdots & \ddots & \vdots \\ h_{n-1}(x) & -h_0(x) & \ldots & -h_{n-2}(x) \end{pmatrix} \begin{pmatrix} r_{00}r_1(x) \\ r_{01}r_1(x) \\ \vdots \\ r_{0(n-1)}r_1(x) \end{pmatrix} \tag{23}$$

where each polynomial $h_i(x)$ is constructed as:

$h_i(x) = h_{in} + h_{in+1}x + \ldots + h_{in+(n-1)}x^{n-1}$ and $r_1(x) = r_{10} + r_{11}x + \ldots + r_{1(n-1)}x^{n-1}$.

Reformulating the u-ANTRU multiplication as polynomial multiplication yields an efficient method that avoids an explicit matrix–matrix product involving the random matrix G. Since $r_1(x)$ is common to every row, each row-by-column product reduces to a single polynomial multiplication, computable in $O(n \log_2 n)$ time under NTT-friendly parameters. Summing this cost over the n rows gives a total multiplication complexity of $O(n^2 \log_2 n)$, and the coefficients of the resulting polynomials are then rearranged into an $n \times n$ matrix to recover the product in matrix form.

6 Towards Structured Instantiations

6.1 Semi-structured ANTRU or semi-ANTRU

The semi-structured construction balances the memory and time requirements, as well as the structural constraints, compared to the almost-unstructured variant introduced above in Subsection 3.1.

Rather than sampling $\mathbf{G}$ with a random structure, the semi-structured variant samples g (whose corresponding matrix form is $\mathbf{G}$) from the group ring constructed over the twisted dihedral group[5]:

$$\mathcal{R}^q_{D_N^+} \;=\; \mathbb{Z}_q D_N^+ \;\approx\; \frac{\mathbb{Z}_q[x,y]}{(x^N-1,\; y^2+1,\; xy-yx^{N-1})}. \tag{24}$$

Therefore, the key generation, encryption, and decryption of semi-ANTRU follow the same steps as sketched in Figure 4, with the note that $f_1, f_2, r_1, r_2 \in \mathcal{R}_{C_{2N}^+}$ and $g \in \mathcal{R}_{D_N^+}$. One can observe that the matrix representations of elements in $\mathcal{R}_{C_{2N}^+}$ and $\mathcal{R}_{D_N^+}$ share a common structure of the form $\begin{pmatrix} \mathbf{A} & \mathbf{B} \\ -\mathbf{B} & \mathbf{A} \end{pmatrix}$. In the case of $\mathcal{R}_{C_{2N}^+}$, the entire matrix is (twisted) right circulant. For $\mathcal{R}_{D_N^+}$, the submatrix $\mathbf{A}$ is right circulant, while $\mathbf{B}$ is left circulant (see Figure 1). Moreover, the product $\mathbf{H} = \mathbf{F}_1^{-1}\mathbf{G}\mathbf{F}_2^{-1}$ and as a result $\mathbf{C} = p\mathbf{R}_1\mathbf{H}\mathbf{R}_2 + \mathbf{M}$, where $\mathbf{M} = \begin{pmatrix} \mathbf{M}_0 & \mathbf{M}_1 \\ -\mathbf{M}_1 & \mathbf{M}_0 \end{pmatrix}$ is a random ternary matrix, exhibits a similar block pattern, although the resulting submatrices $\mathbf{A}$ and $\mathbf{B}$ may not retain any clearly identifiable structure. Consequently, the public key and the ciphertext are compressed for the semi-structured variant compared to the almost-unstructured construction, and one can further optimize the matrix multiplications due to their similar block structure.

6.2 Fully-structured ANTRU: open direction

To further reduce the public-key size, one may attempt to fully structurize the semi-structured variant by choosing f_1 and f_2 from a commutative subring of the twisted dihedral group ring, so that the resulting public key $h = f_1^{-1} * g * f_2^{-1}$ also belongs to $\mathcal{R}_{D_N^+}$. Consequently, the matrix representation of h retains the form $\begin{pmatrix} \mathbf{A} & \mathbf{B} \\ -\mathbf{B} & \mathbf{A} \end{pmatrix}$, which is uniquely determined by its first row, namely the coefficient vector of h. In contrast, the semi-structured variant requires storing at least the submatrices $\mathbf{A}$ and $\mathbf{B}$ to reconstruct the public key. This observation suggests a substantial reduction in public-key size.

Unfortunately, this optimization introduces a serious algebraic vulnerability. For decryption to remain correct, the secret elements f_1 and f_2 must be chosen from a commutative subring $\mathcal{S} \subseteq \mathcal{R}_{D_N^+}$ consisting of all elements that commute with y,

$$\mathcal{S} = \{f \in \mathcal{R}_{D_N^+} \mid fy = yf\}.$$

Equivalently, if $f = a_0(x) + ya_1(x) \in \mathcal{R}_{D_N^+}$, then $f \in \mathcal{S}$ if and only if $a_i(x) = a_i(x^{N-1})$ for $i = 0, 1$. Hence every element of $\mathcal{S}$ has the form

$$f = a_{00} + ya_{10} + \sum_{i=1}^{(N-1)/2} (a_{0i} + ya_{1i})(x^i + x^{N-i}),$$

[5] We use the rings $\mathcal{R}^q_{C_n^+}$ and $\mathcal{R}^q_{D_N^+}$ rather than $\mathcal{R}^q_{C_n}$ and $\mathcal{R}^q_{D_N}$, as the structure associated with these rings avoids unnecessary homomorphisms that could make solving the SVP problem easier in lattices of smaller dimensions. See Appendix B.

and such elements are referred to as symmetric elements.

Now suppose that $f_1, f_2 \in \mathcal{S}$ have small coefficients and that $g \in \mathcal{R}_{D_N^+}$ is also small. Then, using that fact that $x + yxy \in \mathcal{S}$ for all $x \in \mathcal{R}_{D_N^+}$, we get

$$\begin{aligned} h + yhy &= f_1^{-1} * g * f_2^{-1} + y(f_1^{-1} * g * f_2^{-1})y \\ &= f_1^{-1} * g * f_2^{-1} + f_1^{-1} * (ygy) * f_2^{-1} \qquad (\text{since } f_1, f_2 \in \mathcal{S}) \\ &= f_1^{-1} * (g + ygy) * f_2^{-1} \\ &= f_1^{-1} * f_2^{-1} * (g + ygy) \\ &= (f_1 * f_2)^{-1} * (g + ygy). \end{aligned}$$

Since f_1, f_2, and g have small coefficients, the products $f_1 * f_2$ and $g + ygy$ also remain small in Euclidean norm. Consequently, an adversary can transform the public equation into an NTRU-like equation simply by applying $h \mapsto h + yhy$. The remaining challenge is whether recovering the individual small factors f_1 and f_2 from their product $f_1 * f_2$ is computationally difficult.

In fact, Truman [46] showed that a similar attack extends beyond the twisted dihedral setting to general semidirect products of cyclic groups, of which the dihedral group D_N is a special case.

This naturally leads to the following open problem.

Open question. Does there exist a noncommutative algebra that supports a fully structured ANTRU construction while preventing the efficient transformation of expressions of the form $f_1^{-1} * g * f_2^{-1}$ into an NTRU-like equation?

References

1. 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), https://www.usenix.org/system/files/conference/usenixsecurity16/sec16_paper_alkim.pdf
2. 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
3. Bagheri, K., Sadeghi, M.R., Panario, D.: A Non-commutative Cryptosystem Based on Quaternion Algebras. Designs, Codes and Cryptography **86**, 2345–2377 (10 2018). https://doi.org/10.1007/s10623-017-0451-4
4. 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). https://doi.org/10.1137/1.9781611974331.ch2
5. Bernstein, D.J., Chuengsatiansup, C., Lange, T., van Vredendaal, C.: NTRU Prime: Reducing Attack Surface at Low Cost. In: Selected Areas in Cryptography – SAC 2017. pp. 235–260. Springer International Publishing (2018). https://doi.org/10.1007/978-3-319-72565-9_12

6. Chen, C., Danba, O., Hoffstein, J., H¨ulsing, A., Rijneveld, J., Schanck, J.M., Saito, T., Schwade, P.S., Whyte, W.W., Xagawa, K.X., Yamakawa, T., Zhang, Z.: PQC round-3 candidate: NTRU. technical report. Tech. rep., NTRU Cryptosystems Technical Report No.11, Version 2, March 2001. Report (2019), https://ntru.org/f/ntru-20190330.pdf
7. 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
8. 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
9. 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
10. Coppersmith, D., Winograd, S.: Matrix multiplication via arithmetic progressions. Journal of Symbolic Computation **9**(3), 251–280 (1990). https://doi.org/10.1016/S0747-7171(08)80013-2
11. Dachman-Soled, D., Ducas, L., Gong, H., Rossi, M.: LWE with Side Information: Attacks and Concrete Security Estimation. In: Advances in Cryptology – CRYPTO 2020. pp. 329–358. Springer International Publishing, Cham (2020). https://doi.org/10.1007/978-3-030-56880-1_12
12. Don, J., Fehr, S., Majenz, C., Schaffner, C.: Online-Extractability in the Quantum Random-Oracle Model. In: Dunkelman, O., Dziembowski, S. (eds.) Advances in Cryptology – EUROCRYPT 2022. pp. 677–706. Springer International Publishing, Cham (2022). https://doi.org/10.1007/978-3-031-07082-2_24
13. Ducas, L., van Woerden, W.: NTRU Fatigue: How Stretched is Overstretched? In: Advances in Cryptology – ASIACRYPT 2021. pp. 3–32. Springer International Publishing (2021). https://doi.org/10.1007/978-3-030-92068-5_1
14. Esser, A., Girme, R., Mukherjee, A., Sarkar, S.: Memory-efficient attacks on small lwe keys. In: Guo, J., Steinfeld, R. (eds.) Advances in Cryptology – ASIACRYPT 2023. pp. 72–105. Springer Nature Singapore, Singapore (2023). https://doi.org/10.1007/978-981-99-8730-6_3
15. 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., National Institute of Standards and Technology (NIST) (2018), https://www.di.ens.fr/~prest/Publications/falcon.pdf
16. Fujisaki, E., Okamoto, T.: Secure Integration of Asymmetric and Symmetric Encryption Schemes. In: Wiener, M. (ed.) Advances in Cryptology — CRYPTO' 99. pp. 537–554. Springer Berlin Heidelberg, Berlin, Heidelberg (1999). https://doi.org/10.1007/3-540-48405-1_34
17. Fujisaki, E., Okamoto, T.: Secure integration of asymmetric and symmetric encryption schemes. Journal of Cryptology **26**, 80–101 (2013), https://doi.org/10.1007/s00145-011-9114-1
18. Gärtner, J.: NTWE: A Natural Combination of NTRU and LWE. In: Johansson, T., Smith-Tone, D. (eds.) Post-Quantum Cryptography (PQCrypto 2023). pp. 321–353. Springer Nature Switzerland, Cham (2023), https://doi.org/10.1007/978-3-031-40003-2_12

19. Genise, N., Gentry, C., Halevi, S., Li, B., Micciancio, D.: Homomorphic encryption for finite automata. In: Galbraith, S.D., Moriai, S. (eds.) Advances in Cryptology – ASIACRYPT 2019. pp. 473–502. Springer International Publishing, Cham (2019). https://doi.org/10.1007/978-3-030-34621-8_17
20. 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
21. 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
22. Hoffstein, J., Silverman, J.H.: A non-commutative version of the NTRU public key cryptosystem. unpublished paper, February (1997)
23. 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), https://ntru.org/f/tr/tr004v2.pdf
24. Hofheinz, D., Hövelmanns, K., Kiltz, E.: A Modular Analysis of the Fujisaki-Okamoto Transformation. In: Kalai, Y., Reyzin, L. (eds.) Theory of Cryptography. pp. 341–371. Springer International Publishing, Cham (2017). https://doi.org/10.1007/978-3-319-70500-2_12
25. Hopcroft, J.E., Kerr, L.R.: On minimizing the number of multiplications necessary for matrix multiplication. SIAM Journal on Applied Mathematics **20**(1), 30–36 (1971). https://doi.org/10.1137/0120004
26. Howgrave-Graham, N.: A Hybrid Lattice-Reduction and Meet-in-the-Middle Attack Against NTRU. In: Advances in Cryptology - CRYPTO 2007. pp. 150–169. Springer Berlin Heidelberg (2007). https://doi.org/10.1007/978-3-540-74143-5_9
27. Hülsing, A., Rijneveld, J., Schanck, J.M., Schwabe, P.: NTRU-HRSS-KEM: Algorithm specifications and supporting documentation. NIST submission **30** (2017), https://csrc.nist.gov/Projects/post-quantum-cryptography/post-quantum-cryptography-standardization/round-1-submissions
28. 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
29. Kannan, R.: Improved algorithms for integer programming and related lattice problems. In: Proceedings of the fifteenth annual ACM symposium on Theory of computing. pp. 193–206 (1983), https://doi.org/10.1145/800061.808749
30. Kim, J., Park, J.H.: NTRU+: Compact Construction of NTRU Using Simple Encoding Method. IEEE Transactions on Information Forensics and Security **18**, 4760–4774 (2023). https://doi.org/10.1109/TIFS.2023.3299172, https://sites.google.com/view/ntruplus
31. Kirchner, P., Fouque, P.A.: Revisiting Lattice Attacks on Overstretched NTRU Parameters. In: Advances in Cryptology – EUROCRYPT 2017. pp. 3–26. Springer International Publishing (2017). https://doi.org/10.1007/978-3-319-56620-7_1
32. Kumar, V., Raya, A., Gangopadhyay, A.K., Gangopadhyay, S.: Dimension Reduction Attack on Noncommutative Group Ring NTRU Over the Dihedral Group. In: 2024 1st International Conference On Cryptography And Information Security (VCRIS). pp. 1–6 (2024). https://doi.org/10.1109/VCRIS63677.2024.10813443

33. Kumar, V., Raya, A., Gangopadhyay, A.K., Gangopadhyay, S., Hussain, M.T.: An Efficient Noncommutative NTRU from Semidirect Product. In: Mukhopadhyay, S., Stănică, P. (eds.) Progress in Cryptology – INDOCRYPT 2024. pp. 3–27. Springer Nature Switzerland, Cham (2025). https://doi.org/10.1007/978-3-031-80308-6_1
34. Kumar, V., Raya, A., Gangopadhyay, S., Gangopadhyay, A.K.: Cryptanalysis of Group Ring NTRU: The Case of the Dihedral Group. Security and Privacy **8**, 1–15 (2025), https://doi.org/10.1002/spy2.70020
35. Laarhoven, T.: Search problems in cryptography: from fingerprinting to lattice sieving. Phd thesis, Eindhoven University of Technology (2015), https://research.tue.nl/en/publications/search-problems-in-cryptography-from-fingerprinting-to-lattice-si
36. Lyubashevsky, V., Seiler, G.: NTTRU: Truly Fast NTRU Using NTT. IACR Transactions on Cryptographic Hardware and Embedded Systems **2019**(3), 180–201 (May 2019). https://doi.org/10.13154/tches.v2019.i3.180-201, https://tches.iacr.org/index.php/TCHES/article/view/8293
37. Malekian, E., Zakerolhosseini, A., Mashatan, A.: QTRU : a lattice attack resistant version of NTRU PKCS based on quaternion algebra. IACR Cryptology ePrint Archive **386** (2009), https://eprint.iacr.org/2009/386
38. 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
39. Raya, A., Kumar, V., Gangopadhyay, A.K., Gangopadhyay, S.: Giant Does NOT Mean Strong: Cryptanalysis of BQTRU. In: Niederhagen, R., Saarinen, M.J.O. (eds.) Post-Quantum Cryptography. pp. 312–348. Springer Nature Switzerland, Cham (2025). https://doi.org/10.1007/978-3-031-86599-2_11
40. Raya, A., Kumar, V., Gangopadhyay, S.: DiTRU: A Resurrection of NTRU over Dihedral Group. In: Vaudenay, S., Petit, C. (eds.) Progress in Cryptology - AFRICACRYPT 2024. pp. 349–375. Springer Nature Switzerland, Cham (2024), 10.1007/978-3-031-64381-1_16
41. Raya, A., Kumar, V., Gangopadhyay, S., Gangopadhyay, A.K.: Efficient Key Encapsulation Mechanisms from Noncommutative NTRU. Computer Networks **272**, 111704 (2025). https://doi.org/https://doi.org/10.1016/j.comnet.2025.111704
42. 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
43. Schwabe, P., Avanzi, R., Bos, J.B., Ducas, L.D., Kiltz, E., crede Lepoint, T., Lyubashevsky, V., M. Schanck, J., Seiler, G., Stehle, D.: Crystals-Kyber. Tech. rep., National Institute of Standards and Technology (NIST) (2017), https://csrc.nist.gov/projects/post-quantum-cryptography/round-1-submissions
44. Silverman, J.H.: Almost inverses and fast NTRU key creation. NTRU Cryptosystems Technical Report #14 (1999), https://ntru.org/f/tr/tr014v1.pdf
45. Strassen, V.: Gaussian elimination is not optimal. Numerische Mathematik **13**, 354–356 (1969). https://doi.org/10.1007/BF02165411
46. Truman, K.R.: Analysis and Extension of Non-Commutative NTRU. PhD dissertation, University of Maryland (2007), https://drum.lib.umd.edu/handle/1903/7344
47. Vassilevska Williams, V.: Multiplying matrices faster than coppersmith-winograd. In: Proceedings of the 44th Annual ACM Symposium on Theory of Computing (STOC). pp. 887–898. ACM (2012). https://doi.org/10.1145/2213977.2214056

48. 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

A Hoffstein and Silverman Construction

This section describes the first construction of noncommutative NTRU by Silverman and Hoffstein [22,46], intended to avoid lattice attacks.

Setup For a dihedral group of order $2N$ defined as $D_N = \langle x, y \mid x^N = y^2 = 1, xy = yx^{N-1} \rangle$, let $\mathcal{R}$ indicate the group ring of the dihedral group and the integer ring

$$\mathcal{R} = \mathbb{Z}D_N \approx \frac{\mathbb{Z}[x,y]}{(x^N - 1, y^2 - 1, xy - yx^{N-1})}. \tag{25}$$

Furthermore, the cryptosystem needs a commutative subring $\mathcal{S}$,

$$\mathcal{S} = \{f \in \mathcal{R} \mid fy = yf\}. \tag{26}$$

The subring contains all the elements in $\mathcal{R}$ that commutes with y, i.e., $\overline{f(x)} = f(x)$ for all $f(x)$ in $\mathcal{S}$. Furthermore, the cryptosystem involves the following moduli: $p, q, r, s, t \in \mathbb{N}$, where p, r, and t are considered small, while q and s are large. We assume that p and q are prime and are coprime with $2N$. The cryptosystem uses the following rings and subrings:

$$\begin{aligned} &\mathcal{R}_q = \mathbb{Z}_q D_N, && \mathcal{R}_s = \mathbb{Z}_s D_N, && \mathcal{R}_r = \mathbb{Z}_r D_N, \\ &\mathcal{S}_q = \{f \in \mathcal{R}_q \mid fy = yf\}, && \mathcal{S}_s = \{f \in \mathcal{R}_s \mid fy = yf\}, && \mathcal{S}_r = \{f \in \mathcal{R}_r \mid fy = yf\}. \end{aligned}$$

Further, let us consider the following sample spaces:

$$\mathcal{L}_f = \mathcal{S}_q,\ \mathcal{L}_m = \mathcal{R}_q,\ \mathcal{L}_\phi = \mathcal{S}_r,\ \mathcal{L}_\psi = \mathcal{S}_s, \text{ and } \mathcal{L}_\omega = \mathcal{R}_t.$$

$\underline{\mathsf{KeyGen}(1^\lambda)}$
1: initialize $\mathsf{Sampler}$ with $\mathcal{L}_f$ and seed_f
2: do $f \leftarrow \mathsf{Sampler}$ until f is invertible in $\mathcal{R}_q$
3: initialize $\mathsf{sampler}$ with $\mathcal{L}_\omega$ and seed_ω
4: do $\omega \leftarrow \mathsf{Sampler}$
5: $F = f^{-1} \pmod q$
6: $h = pF * \omega * f \pmod q$
7: $pk = h, sk = (f, F, \omega)$
8: **return** (sk, pk)

$\underline{\mathsf{Encrypt}(m, h, \mathsf{seed}_{\phi,\phi',\psi})}$
1: initialize $\mathsf{Sampler}$ with $\mathcal{L}_\phi$ and seed_ϕ
2: $\phi \leftarrow \mathsf{Sampler}$
3: initialize $\mathsf{Sampler}$ with $\mathcal{L}_\phi$ and $\mathsf{seed}_{\phi'}$
4: $\phi' \leftarrow \mathsf{Sampler}$
5: initialize $\mathsf{Sampler}$ with $\mathcal{L}_\psi$ and seed_ψ
6: $\psi \leftarrow \mathsf{Sampler}$
7: $\Psi = \psi \pmod p$
8: $e = \phi * h * \phi' + \psi \pmod q$
9: $E = \Psi * h + m \pmod q$
10: **return** (e, E)

$\underline{\mathsf{Decrypt}((e, E), (f, F, \omega))}$
1: $a = f * e * F = p\phi * \omega * \phi' + \psi \pmod q$
2: $a' = \mathsf{Centerlift}(a, q)$
3: $\Psi = a' \pmod p$
4: $m' = E - \Psi * h \pmod q$
5: **return** m'

Fig. 7: A sketched description of the noncommutative NTRU scheme by Hofffstein and Silverman.

Coppersmith's Attack An initial analysis of the security of the Hoffstein-Silverman design argued that it resists lattice-based attacks based on the following points:

(i) The public key $h = pF * \omega * f \pmod q$ is no longer a product of polynomials with small norms. As a result, attacking the public key cannot be directly reduced to solving the Shortest Vector Problem (SVP) in certain lattices.
(ii) In the first component of the ciphertext pair, $e = \phi * h * \phi' + \psi \pmod q$, the noise term ψ does not have small coefficients. Furthermore, due to the noncommutative nature of the multiplication, the unknowns ϕ and ϕ' cannot be combined into a single side of the expression. Consequently, formulating an attack as an instance of the Closest Vector Problem (CVP) is not feasible against e.
(iii) In the second component of the ciphertext pair, $E = \Psi * h + m \pmod q$, the message m does not consist of small coefficients. Therefore, lattice-based attacks via CVP are similarly ineffective against E.

However, Coppersmith [8] introduced an attack against this design and showed that a third party that does not know the private key can decrypt the message. The attack depends on the following observations:

(i) **Retrieving ω' with Small Coefficients:** Since the attacker knows h, they can construct the element $h + yhy \in \mathcal{S}$[6]

$$\begin{aligned} h + yhy &= pF(\omega + y\omega y) * f \pmod q \\ &= pF * f * (\omega + y\omega y) \pmod q \\ &= p(\omega + y\omega y) \pmod q \end{aligned} \tag{27}$$

The second line of Equation (27) follows from the fact that $\omega + y\omega y \in \mathcal{S}$, and therefore commutes with both f and F. Moreover, since $\omega \in \mathcal{L}_\omega = \mathcal{R}_t$ for small t, the coefficients of $\omega + y\omega y \pmod q$ remain small and do not wrap around modulo q. Hence, the attacker can construct $\omega' \in \mathcal{S}_w$ with small coefficients satisfying that

$$h + yhy = p(\omega' + y\omega' y) \pmod q. \tag{28}$$

Building such ω' is possible since:

$$\begin{aligned} \omega + y\omega y &= (\omega_0 + y\omega_1) + y(\omega_0 + y\omega_1)y \\ &= (\omega_0 + \overline{\omega_0}) + y(\omega_1 + \overline{\omega_1}) \\ &= \left(2w_{00} + \sum_{i=1}^{N-1} (\omega_{0i} + \omega_{0N-i})x^i\right) + y\left(2\omega_{10} + \sum_{i=1}^{N-1} (\omega_{1i} + \omega_{1N-i})x^i\right). \end{aligned} \tag{29}$$

The coefficients of ω_{0i}, ω_{1i} lie in $\mathbb{Z}_t$, and therefore the coefficients of $h' = h'_0 + yh'_1 = h + yhy \pmod q$ belong to the set $\{-p(t-1), -p(t-2), \ldots, p(t-2), p(t-1)\}$.

[6] **Caution:** For any element $h \in \mathbb{Z}D_N$, $h + yhy \in \mathcal{S}, h - yhy \in \mathcal{S}^-$. However, for an element in $h \in \mathbb{Z}D_N^+$ (twisted), $h + yhy \in \mathcal{S}^-, h - yhy \in \mathcal{S}$.

For each even coefficient of h'_0 written as $2kp$, one can assign $\omega'_{0i} = \omega'_{0N-i} = k$. For each odd coefficient written as $(2k+1)p$, one can assign $\omega'_{0i} = k+1$ and $\omega'_{0N-i} = k$. The same procedure is applied to the coefficients of ω'_1 using the corresponding coefficients from h'_1. Hence, we can construct ω' with small coefficients without knowledge of the secret key.

(ii) **Defining a Special-Purpose Map θ:** The legitimate decryption process relies on applying a specific map that enables the recovery of Ψ from the first ciphertext component e, given knowledge of both f and F (the secret key).
In contrast, the attack strategy introduces a map θ designed to extract Ψ from the ciphertext component e *without* requiring knowledge of the secret components (f, F).
Before defining θ, we introduce the subset $\mathcal{S}^- \subset \mathcal{R}$ as:

$$\mathcal{S}^- = \{f \in \mathcal{R} \mid fy = -yf\}.$$

The map θ is a linear transformation:

$$\theta : \mathcal{R}_q \mapsto \mathcal{R}_q,$$

with the following properties:
- acts as the identity on $\mathcal{S}$.
- maps any $f_1 \in \mathcal{S}^-$ to some $f_2 \in \mathcal{S}^-$.
- is left- and right-linear with respect to $\mathcal{S}$.
- satisfies $\theta(h) = p\omega'$, where $\omega' \pmod q$ has small coefficients (as discussed earlier).

To construct such a map θ, we first express h and $p\omega'$ as:

$$h = h_0 + yh_1 = (h_{0s} + h_{0s^-}) + y\,(h_{1s} + h_{1s^-}) = \underbrace{h_{0s} + yh_{1s}}_{h_s \in \mathcal{S}} + \underbrace{h_{0s^-} + yh_{1s^-}}_{h_{s^-} \in \mathcal{S}^-},$$

$$p\omega' = p\omega'_0 + yp\omega'_1 = \left(\omega'_{0s} + \omega'_{0s^-}\right) + y\left(\omega'_{1s} + \omega'_{1s^-}\right) = \underbrace{\omega'_{0s} + y\omega'_{1s}}_{w'_s \in \mathcal{S}} + \underbrace{\omega'_{0s^-} + y\omega'_{1s^-}}_{w'_{s^-} \in \mathcal{S}^-}.$$

Any element $f \in \mathcal{S}$ can be written as:

$$f = a_0 + yb_0 + \sum_{i=1}^{(N-1)/2} \left(c_i(x^i + x^{N-i}) + yd_i(x^i + x^{N-i})\right),$$

and any $g \in \mathcal{S}^-$ can be written as:

$$g = \sum_{i=1}^{(N-1)/2} \left(c'_i(x^i - x^{N-i}) + yd'_i(x^i - x^{N-i})\right).$$

We then set $\omega'_s = h_s$ and $\omega'_{s^-} = \theta(h_{s^-})$. Coppersmith showed that in the proposed attack, a single term $(x^i - x^{N-i})$ can generate $\mathcal{S}^-$ as an $\mathcal{S}$-module, i.e.,

$$(x^i - x^{N-i})(x^j + x^{N-j}) = t_i(x)$$

for some **specific** $i \in \{1, 2, \ldots, (N-1)/2\}$, $j = 1, 2, \ldots, (N-1)/2$, and $t_i(x) \in \mathcal{S}^-$. Thus, we can write:

$$(x^j - x^{N-j}) = (x^i - x^{N-i})d_j(x), \quad \text{for } d_j(x) \in \mathcal{S}. \tag{30}$$

As a result, both h_{s^-} and w'_{s^-} can be expressed as:

$$\begin{aligned} h_{s^-} &= (x^i - x^{N-i})t_0(x) + y(x^i - x^{N-i})t_1(x), \\ w'_{s^-} &= (x^i - x^{N-i})t'_0(x) + y(x^i - x^{N-i})t'_1(x), \end{aligned}$$

for some $t_0(x), t_1(x), t'_0(x), t'_1(x) \in \mathcal{S}$. Since $\theta(h_{s^-}) = \omega'_{s^-}$, we have:

$$\theta(x^i - x^{N-i}) * \underbrace{(t_0(x) + yt_1(x))}_{\in \mathcal{S}} = (x^i - x^{N-i}) * \underbrace{(t'_0(x) + yt'_1(x))}_{\in \mathcal{S}}. \tag{31}$$

Furthermore, since θ maps $\mathcal{S}^-$ to itself, we have:

$$\begin{aligned} \theta(x^i - x^{N-i}) &= \sum_{i=1}^{(N-1)/2} \left(c'_i(x^i - x^{N-i}) + yd'_i(x^i - x^{N-i}) \right) \\ &= (x^i - x^{N-i}) * \underbrace{(r_0(x) + yr_1(x))}_{\in \mathcal{S}} \quad \text{(from Equation (30))}. \end{aligned}$$

Substituting this into Equation (31) yields:

$$(x^i - x^{N-i}) * \underbrace{(r_0(x) + yr_1(x))}_{\text{unknown}} * \underbrace{(t_0(x) + yt_1(x))}_{\text{known}} = (x^i - x^{N-i}) * \underbrace{(t'_0(x) + yt'_1(x))}_{\text{known}}. \tag{32}$$

Consequently, one can recover $r(x) = r_0(x) + yr_1(x)$ and determine $\theta(x^i - x^{N-i})$. Then, using Equation (30), we can compute $\theta(x^j - x^{N-j})$ for all $j = 1, 2, \ldots, (N-1)/2$. The final step of the attack involves expressing the ciphertext component e as:

$$\theta(e) = e_0 + \theta(e_1), \qquad \text{for } e_0 \in \mathcal{S} \text{ and } e_1 \in \mathcal{S}^- \tag{33}$$

where reduction of $\theta(e)$ modulo p yields Ψ, which can subsequently be used to recover the message m from E.

Why does Coppersmith's Attack Work? In order for the attacker to succeed, they must find a map θ that cancels the effect of the coOuefficients introduced by the term $\phi * h * \phi'$, without disturbing the message component ψ. In the construction by Hoffstein and Silverman, although it is difficult to extract ω directly from the public key equation h, computing the expression $h + yhy$ yields $p(\omega + y\omega y)$, which facilitates the construction of an element ω' such that $\theta(h) = p\omega'$ (as discussed in point 2). Obtaining such an element with small coefficients is feasible because both f and F belong to $\mathcal{S}$ (i.e., the *commutative* subring), and are multiplicative inverses of each other. In contrast, if either $f \notin \mathcal{S}$ or $F \notin \mathcal{S}$, or if the product $f * F$ has large coefficients, it becomes difficult to recover a small ω' from the expression $h + yhy$.

B Selection of Twisted Dihedral Group Ring

The analysis of the underlying algebra in the (twisted) group ring NTRU framework and the structure of its associated lattice is a crucial aspect of secure cryptosystem design. In the following, we provide insights into the structure of the lattice and consequently our reason to work with the twisted dihedral group ring

$$\mathcal{R}^q_{D_N^+} = \mathbb{Z}_q^{\lambda_2} D_N \approx \frac{\mathbb{Z}_q[x,y]}{(x^N-1, y^2+1, xy-yx^{N-1})},$$

where $\lambda_2 : D_N \times D_N \to \mathbb{Z}_q^\times$ is a 2-cocyle defined as

$$\lambda_2(y^{j_1}x^{i_1}, y^{j_2}x^{i_2}) = \begin{cases} -1, & \text{for } i_1, i_2 \in \{0,1,\ldots,N-1\} \\ & \text{and } j_1 = j_2 = 1 \\ 1, & \text{otherwise.} \end{cases} \tag{34}$$

It should be noted that the matrices associated with the elements of $\mathcal{R}^q_{D_N^+}$ have the following pattern

$$\mathcal{B}^{CS}_{D_N^+} = \begin{pmatrix} \mathbf{I}_N & \mathbf{0}_N & \mathbf{H}_0 & \mathbf{H}_1 \\ \mathbf{0}_N & \mathbf{I}_N & -\mathbf{H}_1 & \mathbf{H}_0 \\ \mathbf{0}_N & \mathbf{0}_N & q\mathbf{I}_N & \mathbf{0}_N \\ \mathbf{0}_N & \mathbf{0}_N & \mathbf{0}_N & q\mathbf{I}_N \end{pmatrix}. \tag{35}$$

This structure is similar to the matrix representation of elements of the twisted cyclic group ring

$$\mathcal{R}^q_{C_n^+} = \mathbb{Z}_q^{\lambda_1} C_n \approx \frac{\mathbb{Z}_q[x]}{(x^n+1)},\ n \text{ is even (usually power of 2)},$$

where $\lambda_1 : C_n \times C_n \mapsto \mathbb{Z}_q^\times$ is a 2-cocyle defined as

$$\lambda_1(x^i, x^j) = \begin{cases} 1, & \text{for } i+j < n \\ -1, & \text{for } i+j \geq n \end{cases} \tag{36}$$

The main difference between the two matrix representations is that in the case of $\mathcal{R}^q_{C_n^+}$, the matrix $\begin{pmatrix} \mathbf{H}_0 & \mathbf{H}_1 \\ -\mathbf{H}_1 & \mathbf{H}_0 \end{pmatrix}$ is itself a (twisted) right circulant matrix, while in the case of $\mathcal{R}^q_{D_N^+}$, $\mathbf{H}_0$ is right circulant and $\mathbf{H}_1$ is a left circulant matrix.

The ring $\mathcal{R}^q_{C_n^+}$ has been used in various well-known lattice-based schemes, for example, Falcon [15] and Kyber [43], etc. The benefit of using the ring $\mathcal{R}^q_{C_n^+}$ lies in its algebraic structure that prevents certain attacks, like Gentry's attack [20], due to the fact that "the polynomial x^n+1 does not have any polynomial factor modulus q with small degree, e.g. all the factors of degree $n/2$ have at least Euclidean norm $\sqrt{q}$" [43, Page 35]. On the contrary, for n (a power of 2 or even composite), the polynomial x^n-1 splits into $x^{n/2}-1$ and $x^{n/2}+1$, two polynomials of half the degree and small norm $\sqrt{2}$. As a result,

$$\frac{\mathbb{Z}_q[x]}{(x^n-1)} \longrightarrow \frac{\mathbb{Z}_q[x]}{(x^{n/2}-1)} \times \frac{\mathbb{Z}_q[x]}{(x^{n/2}+1)} \quad \text{(Chinese Remainder Theorem)} \tag{37}$$

This allows mapping the secret key $f = f_0(x) + x^{n/2} f_1(x), g(x) = g_0(x) + x^{n/2} g_1(x)$ with coefficients from the set $\{-1, 0, 1\}$ to short polynomials

$$f_0(x)+f_1(x), g_0(x)+g_1(x) \in \frac{\mathbb{Z}_q[x]}{(x^{n/2}-1)},\ f_0(x)-f_1(x), g_0(x)-g_1(x) \in \frac{\mathbb{Z}_q[x]}{(x^{n/2}+1)} \quad (38)$$

with coefficients from the set $\{0, \pm 1, \pm 2\}$, as well as maintaining the relation

$$h_0(x) + h_1(x) = (f_0(x) + f_1(x))^{-1} * (g_0(x) + g_1(x)) \pmod q$$
$$h_0(x) - h_1(x) = (f_0(x) - f_1(x))^{-1} * (g_0(x) - g_1(x)) \pmod q.$$

Therefore, instead of attacking a $2n$-dimensional lattice $\mathcal{L}_{C_n}^{CS}$ generated by the matrix

$$\mathcal{B}_{C_n}^{CS} = \begin{pmatrix} \mathbf{I}_{n/2} & \mathbf{0}_{n/2} & \mathbf{H}_0 & \mathbf{H}_1 \\ \mathbf{0}_{n/2} & \mathbf{I}_{n/2} & \mathbf{H}_1 & \mathbf{H}_0 \\ \mathbf{0}_{n/2} & \mathbf{0}_{n/2} & q\mathbf{I}_{n/2} & \mathbf{0}_{n/2} \\ \mathbf{0}_{n/2} & \mathbf{0}_{n/2} & \mathbf{0}_{n/2} & q\mathbf{I}_{n/2} \end{pmatrix},$$

an attacker can perform lattice attacks on two n-dimensional lattices $\mathcal{L}_{C_n}^{CS,+}$ and $\mathcal{L}_{C_n}^{CS,-}$ generated by the matrices

$$\mathcal{B}_{C_n}^{CS,+} = \begin{pmatrix} \mathbf{I}_{n/2} & \mathbf{H}_0 + \mathbf{H}_1 \\ \mathbf{0}_{n/2} & q\mathbf{I}_{n/2} \end{pmatrix} \text{ and } \mathcal{B}_{C_n}^{CS,-} = \begin{pmatrix} \mathbf{I}_{n/2} & \mathbf{H}_0 - \mathbf{H}_1 \\ \mathbf{0}_{n/2} & q\mathbf{I}_{n/2} \end{pmatrix},$$

respectively, to recover the short vector $(\mathbf{f}_0+\mathbf{f}_1, \mathbf{g}_0+\mathbf{g}_1) \in \mathcal{L}_{C_n}^{CS,+}$ and $(\mathbf{f}_0-\mathbf{f}_1, \mathbf{g}_0-\mathbf{g}_1) \in \mathcal{L}_{C_n}^{CS,-}$, and lifting them back to recover $(\mathbf{f}, \mathbf{g}) \in \mathcal{L}_{C_n}^{CS}$. Since the cost of the lattice attack is directly proportional to the dimension of the lattice, halving the dimension reduces the attack cost drastically. Consequently, an NTRU-like scheme over the ring $\mathbb{Z}[x]/(x^n - 1)$ with n as a power of 2 or even a composite number is not considered secure, and it is recommended to select n to be a prime number.

Interestingly, the slight modification of replacing ring $\mathbb{Z}_q[x]/(x^n - 1)$ in the NTRU-framework to $\mathbb{Z}_q[x]/(x^n + 1)$ mitigates Gentry's attack and allows selecting n to be a power of 2 and an appropriate q to make the ring $\mathbb{Z}_q[x]/(x^n + 1)$ NTT-friendly for faster polynomial multiplications.

Raya et al. [39] presented another interpretation of Gentry's attack in terms of matrices and extended it to noncommutative algebraic structure used in [3]. We present a glimpse of their idea and associate it with the concerned twisted dihedral group ring.

For the ring $\mathbb{Z}_q[x]/(x^n - 1)$ with n as a power of 2, the matrix representation of elements of the ring has the form $\begin{pmatrix} \mathbf{H}_0 & \mathbf{H}_1 \\ \mathbf{H}_1 & \mathbf{H}_0 \end{pmatrix} \in \mathbb{Z}^{n \times n}$ which can be homomorphically mapped to $\mathbf{H}_0 + \mathbf{H}_1,\ \mathbf{H}_0 - \mathbf{H}_1 \in \mathbb{Z}^{n/2 \times n/2}$, matrices of half the dimensions. It is equivalent to applying the maps in Equations (37),(38) in terms of polynomials. This leads to exactly the same lattice attack discussed above. Since the matrix representation of elements of the dihedral group ring $\mathbb{Z}D_N$ is also $\begin{pmatrix} \mathbf{H}_0 & \mathbf{H}_1 \\ \mathbf{H}_1 & \mathbf{H}_0 \end{pmatrix} \in \mathbb{Z}^{2N \times 2N}$ with the difference that $\mathbf{H}_0$ and $\mathbf{H}_1$ are right and left circulant, respectively, the same homomorphisms can be used to reduce

the lattice dimension by half as shown in [34]. Moreover, if N is even, then the submatrices further possess special structures

$$\mathbf{H}_0 = \begin{pmatrix} \mathbf{H}_{00} & \mathbf{H}_{01} \\ \mathbf{H}_{01} & \mathbf{H}_{00} \end{pmatrix} \text{ and } \mathbf{H}_1 = \begin{pmatrix} \mathbf{H}_{10} & \mathbf{H}_{11} \\ \mathbf{H}_{11} & \mathbf{H}_{10} \end{pmatrix}, \tag{39}$$

that can be exploited using the same homomorphisms to further reduce the dimension by one-fourth as shown in [32]. Therefore, it was recommended in [32] to select N to be prime while designing a GR-NTRU cryptosystem over the dihedral group ring $\mathbb{Z}D_N$.

On the other hand, the only potential information-preserving homomorphisms for the matrix representation $\mathbf{H} = \begin{pmatrix} \mathbf{H}_0 & \mathbf{H}_1 \\ -\mathbf{H}_1 & \mathbf{H}_0 \end{pmatrix} \in \mathbb{Z}^{n\times n}$ of elements of the ring $\mathbb{Z}_q[x]/(x^n+1)$ are $\mathbf{H} \to \alpha\mathbf{H}_0 + \beta\mathbf{H}_1$ where $(\alpha, \beta) \in \{(1, i), (1, -i)\}$ and $i = \sqrt{-1}$. Therefore, an attacker can construct two n-dimensional lattices $\mathcal{L}^{CS,+}_{C_n^+}$ and $\mathcal{L}^{CS,-}_{C_n^+}$ generated by the matrices

$$\mathcal{B}^{CS,+}_{C_n^+} = \begin{pmatrix} \mathbf{I}_{n/2} & \mathbf{H}_0 + i\mathbf{H}_1 \\ \mathbf{0}_{n/2} & q\mathbf{I}_{n/2} \end{pmatrix} \text{ and } \mathcal{B}^{CS,-}_{C_n^+} = \begin{pmatrix} \mathbf{I}_{n/2} & \mathbf{H}_0 - i\mathbf{H}_1 \\ \mathbf{0}_{n/2} & q\mathbf{I}_{n/2} \end{pmatrix},$$

respectively, to recover the short vector $(\mathbf{f}_0 + i\mathbf{f}_1, \mathbf{g}_0 + i\mathbf{g}_1) \in \mathcal{L}^{CS,+}_{C_n^+}$ and $(\mathbf{f}_0 - i\mathbf{f}_1, \mathbf{g}_0 - i\mathbf{g}_1) \in \mathcal{L}^{CS,-}_{C_n^+}$. However, the existing lattice reduction algorithms are not designed for matrices with complex numbers, and hence, the attacker must map these complex matrices to ones with real entries to perform lattice attacks, thus effectively not gaining any advantage in terms of dimension reduction.

Overall, the structure of matrices of twisted dihedral group ring $\mathcal{R}^q_{D_N^+}$ and the twisted cyclic group ring $\mathcal{R}^q_{C_n^+}$ are similar.
Still, we base the lattice security of our design on the following argument. Any attack on the ring $\mathcal{R}_{D_N^+}$ that reduces the lattice dimension must have an equivalent representation/effect on the matrix $\mathbf{H}_{D_N^+} = \begin{pmatrix} \mathbf{H}_0 & \mathbf{H}_1 \\ -\mathbf{H}_1 & \mathbf{H}_0 \end{pmatrix}$. Due to the similar structure, the same attack must apply to the matrix $\mathbf{H}_{C_n^+} = \begin{pmatrix} \mathbf{H}_0 & \mathbf{H}_1 \\ -\mathbf{H}_1 & \mathbf{H}_0 \end{pmatrix}$ of the same overall pattern.