

Public Key Compression and Fast Polynomial Multiplication for NTRU using the Corrected Hybridized NTT-Karatsuba Method

Rohon Kundu¹^a, Alessandro De Piccoli²^b and Andrea Visconti²^c

¹*Department of Electrical and Information Technology, Lund University, Box 117, 221 00 Lund, Sweden*

²*Department of Computer Science “Giovanni Degli Antoni”, Università degli Studi di Milano, via Celoria 18, 20133 Milano MI, Italy*

rohon.kundu@eit.lth.se, {alessandro.deplicoli, andrea.visconti}@unimi.it

Keywords: Post-Quantum Cryptography, Lattice-Based Cryptography, Ring-Learning With Errors Problem, NTRU Algorithm, Number Theoretic Transformation, Hybridized NTT-Karatsuba Algorithm, Key size


Abstract: NTRU is a lattice-based public-key cryptosystem that has been selected as one of the Round III finalists at the NIST Post-Quantum Cryptography Standardization. Compressing the key sizes to increase efficiency has been a long-standing open question for lattice-based cryptosystems. In this paper we provide a solution to three seemingly opposite demands for NTRU cryptosystem: compress the key size, increase the security level, optimize performance by implementing fast polynomial multiplications. We consider a specific variant of NTRU known as NTRU-NTT. To perform polynomial optimization, we make use of the Number-Theoretic Transformation (NTT) and hybridize it with the Karatsuba Algorithm. Previous work done in providing 2-part Hybridized NTT-Karatsuba Algorithm contained some operational errors in the product expression, which have been detected in this paper. Further, we conjectured the corrected expression and gave a detailed mathematical proof of correctness. In this paper, for the first time, we optimize NTRU-NTT using the corrected Hybridized NTT-Karatsuba Algorithm. The significance of compressing the value of the prime modulus q lies with decreasing the key sizes. We achieve a 128-bit post-quantum security level for a modulus value of 83,969 which is smaller than the previously known modulus value of 1,061,093,377, while keeping n constant at 2048.


1 INTRODUCTION


The abstract algebraic structure of a lattice plays a vital role in developing post-quantum cryptographic schemes. Lattice-based protocols are considered to be one of the most suitable candidates against quantum threats. In December 2016, the US National Institute of Standards and Technology (NIST) initiated the PQC project intending to develop, evaluate and standardize public-key encryption schemes for the quantum age. Among the NIST Round II candidates (Alagic et al., 2019), five submissions are based on lattice-based cryptography. Among the Round III finalist announced on July 22, 2020, are NTRU (Chen et al., 2019), CRYSTAL-KYBER (Avanzi et al., 2017) and, SABER (Karmakar et al., 2018) all of which are lattice-based public-key encryption schemes.

In this paper, we focus on the NTRU, one of the well known public-key cryptosystems. It was first introduced by Hoffstein, Pipher, and Silverman (Hoffstein et al., 1998). The time complexity of the NTRU algorithm depends on how fast we can multiply two input polynomials. Both the encryption and the decryption process rely on polynomial multiplications. In reality, we deal with polynomials with a substantially large degree like 1024, 2048, and 4096. To ensure a higher security level the input polynomial has to be of a higher degree which in turn increases the computational complexity, eventually resulting in decreasing efficiency of the algorithm.

Various optimization techniques like Karatsuba Algorithm and Fast Fourier Transform (FFT) have been proposed to improve the polynomial multiplication. In the case of FFT, the roots of unity belong to the field of complex numbers \mathbb{C}^n . The R-LWE ring structure is denoted by the finite ring quotient $\mathbf{R}_q = \mathbb{Z}_q[x]/\langle x^n + 1 \rangle$, where n is a power of 2 and q is the prime modulus. In this case, the n -th root of

^a  <https://orcid.org/0000-0002-4306-2637>

^b  <https://orcid.org/0000-0002-6399-3164>

^c  <https://orcid.org/0000-0001-5689-8575>

unity ω belongs to the finite Galois Field $\text{GF}(2^m)$ also denoted by \mathbb{F}_{2^m} , $\forall m \in \mathbb{N}$. As a result, the analogous concept of Number Theoretic Transformation (NTT) is used to perform polynomial optimization over the R-LWE ring. In the papers (Zhu et al., 2019; Zhou et al., 2018) the concept of Hybridizing NTT with Karatsuba has been proposed to optimize polynomial multiplication over R-LWE ring. The generalized ring structure of \mathbf{R}_q is given by $\mathbb{Z}_q[x]/\langle\Phi_m(x)\rangle$. Where $\Phi_m(x)$ is a cyclotomic polynomial of degree n having exactly m -th root of unity in \mathbb{Z}_q .

Depending on the adjoint cyclotomic polynomial NTRU can be categorized into three main types (Bernstein and Lange, 2017) : a) NTRU-Classic b) NTRU-NTT and c) NTRU-Prime. The ring structure of NTRU considered in the NIST Round III finalist is that of NTRU-Classic, i.e., where $\Phi_m(x) = x^n - 1$ and n is prime. In this paper we only focus on NTRU-NTT, i.e., where $\Phi_m(x) = x^n + 1$ and n is a power of 2. There has been much work on optimizing NTRU-Classic (Hülsing et al., 2017), but little attention has been given to NTRU-NTT. Still, all variants are assumed to be post-quantum secure. We propose for the first time how to optimize the polynomial multiplication for NTRU-NTT using the Hybridized NTT-Karatsuba technique (Zhu et al., 2019). We identify an error in the product expression mentioned in the paper (Zhu et al., 2019) for the 2-part Hybridized NTT-Karatsuba. Further, we discuss the consequences of the error and provide a new correctness expression. A detailed mathematical proof of the conjectured product formula is also provided. With the corrected expression, we can calculate the appropriate time complexity and also use it to decrease the value of the prime modulus q .

Next, we focus on the relevance of the parameter q in the case of NTRU-NTT. The parameter q defines the key size, but it also influences the efficiency of the algorithm. Thus, a shorter key size would result in a more efficient algorithm. However, the security parameter of NTRU-NTT is given by n , and an important goal is to reduce q while keeping n large, i.e., 2048 or 4096 bits.

Previous research shows that when we try to keep the value of the security parameter n high (like 2048, 4096) the value of the prime modulus q increases significantly (Chen et al., 2014), (Akleylek et al., 2015). As a result, practical implementations were not feasible. Our calculation shows a substantial decrease in the value of the prime modulus q by using the 2^α -part separation method.

2 PRELIMINARIES

2.1 R-LWE Problem

The Ring – LWE Problem is parameterized by

- n be a power of two i.e $n = 2^m, \forall m \in \mathbb{Z}^+$
- q be a prime modulus satisfying $q \equiv 1 \pmod{2n}$
- $\mathbf{R}_q = \mathbb{Z}_q[x]/\langle x^n + 1 \rangle$ as the ring containing all polynomials over the field \mathbb{Z}_q in which x^n is identified with -1 .

In Ring-LWE we are given samples of the form $(\mathbf{a}, \mathbf{b} = \mathbf{a} \cdot \mathbf{s} + \boldsymbol{\varepsilon}) \in \mathbf{R}_q \times \mathbf{R}_q$ where $\mathbf{s} \in \mathbf{R}_q$ is a fixed secret, $\mathbf{a} \in \mathbf{R}_q$ is chosen uniformly, and $\boldsymbol{\varepsilon}$ is an error term chosen independently from some error distribution over \mathbf{R}_q .

The goal is to recover the secret key \mathbf{s} from these samples (for all \mathbf{s} , with high probability). The above concept can be extended to somewhat more general cyclotomic polynomial $\Phi_m(x)$ of degree n , but in our paper we consider $\Phi_m(x) = x^n + 1$.

2.2 Number Theoretic Transformation (NTT)

Number Theoretic Transform is a special case of Fast Fourier Transform over finite fields, as defined by Pollard in this paper (Pollard, 1971). In practice constructing algorithm based on FFT over finite field has been a hard problem. For our case we consider the the FFT over the finite Galois Field $\text{GF}(2^m)$ also denoted by \mathbb{F}_{2^m} , $\forall m \in \mathbb{N}$ (Pollard, 1971; Fedorenko and Trifonov, 2002).

Before giving the definition of NTT of a vector, we set the notation for the vector operations.

Definition 1 (Notation). Let $\mathbf{a} = (a_0, a_1, \dots, a_{n-1})$ and $\mathbf{b} = (b_0, b_1, \dots, b_{n-1})$ be two elements of \mathbb{Z}_q^n , i.e. two n -dimensional vectors. We indicate with $+_q$ and \circ_q the component-wise operations between vectors, namely:

- $\mathbf{a} +_q \mathbf{b} = (a_0 + b_0, a_1 + b_1, \dots, a_{n-1} + b_{n-1}) \pmod{q}$;
- $\mathbf{a} \circ_q \mathbf{b} = (a_0 \cdot b_0, a_1 \cdot b_1, \dots, a_{n-1} \cdot b_{n-1}) \pmod{q}$.

Moreover, throughout the paper we will use the two dots notation for integer intervals. For instance, $[1..n]$ means $\{1, 2, 3, \dots, n\}$.

Definition 2 (NTT). Let $\mathbf{R}_q = \mathbb{Z}_q[x]/\langle x^n + 1 \rangle$ be the truncated polynomial ring and x a root of $x^n + 1$. Here n is a non trivial power of 2 i.e. $n = 2^m$, $m \geq 1$ and $q \equiv 1 \pmod{2n}$. Let $f \in \mathbf{R}_q$, explicitly given as

$$f = a_0 + a_1x + \dots + a_{n-1}x^{n-1}$$

and define the n -dimensional vector $\mathcal{F} = [a_0, a_1, \dots, a_{n-1}]$. Define ω as the n -th primitive root of unity in \mathbb{Z}_q , such that $\omega^n \equiv 1 \pmod{q}$ and $\omega^k \not\equiv 1 \pmod{q}$, $k \in [1..n-1]$. Then, the NTT of \mathcal{F} is a vector whose components are

$$\text{NTT}(\mathcal{F})_i = \hat{\mathcal{F}}_i = \sum_{j=0}^{n-1} a_j \omega^{ij} \pmod{q}, \quad i \in [0..n-1].$$

Definition 3 (NTT^{-1}). The i -th component of the inverse transformation $\mathcal{F} = \text{NTT}^{-1}(\hat{\mathcal{F}})$ is given by

$$\mathcal{F}_i = n^{-1} \sum_{j=0}^{n-1} \hat{\mathcal{F}}_j \cdot \omega^{-ij} \pmod{q}$$

where n^{-1} and ω^{-1} are the inverse in \mathbb{Z}_q .

In (Pollard, 1971), it has been shown that the product $h = f \cdot g$ is given by

$$h = f \cdot g = \text{NTT}^{-1}(\hat{\mathcal{F}} \circ_q \hat{\mathcal{G}}) \pmod{x^n + 1}$$

where \circ_q is the component-wise product \pmod{q} .

Again, it is easy to prove (see lemma 1) that

$$\text{NTT}(\mathcal{F} +_q \mathcal{G}) = \text{NTT}(\mathcal{F}) +_q \text{NTT}(\mathcal{G})$$

and show (see the next example 1) that

$$\text{NTT}(\mathcal{F} \circ_q \mathcal{G}) \neq \text{NTT}(\mathcal{F}) \circ_q \text{NTT}(\mathcal{G}).$$

Example 1. Consider the polynomial ring $\mathbf{R}_q = \mathbb{Z}_q[x]/\langle x^n + 1 \rangle$, where $n = 8$ and $q = 17$. We have

$$f = 1 + x + x^2 + x^3 \quad \text{and} \quad g = 1 + x^3$$

So

$$\mathcal{F} = [1, 1, 1, 1, 0, 0, 0, 0] \quad \text{and} \quad \mathcal{G} = [1, 0, 0, 1, 0, 0, 0, 0]$$

therefore

$$\mathcal{F} +_q \mathcal{G} = [2, 1, 1, 2, 0, 0, 0, 0] \quad \text{and}$$

$$\mathcal{F} \circ_q \mathcal{G} = [1, 0, 0, 1, 0, 0, 0, 0].$$

Also, we choose the value of $\omega = 2$ as the 8-th root of unity in \mathbb{Z}_{17} . Using the definition, we calculate the NTT of the above vectors as

$$\text{NTT}(\mathcal{F}) = [4, 15, 0, 7, 0, 12, 0, 4] \quad \text{and}$$

$$\text{NTT}(\mathcal{G}) = [2, 9, 14, 3, 0, 10, 5, 16]$$

so

$$\text{NTT}(\mathcal{F}) +_q \text{NTT}(\mathcal{G}) = [6, 7, 14, 10, 0, 5, 5, 3]$$

$$\text{NTT}(\mathcal{F}) \circ_q \text{NTT}(\mathcal{G}) = [8, 16, 0, 4, 0, 1, 0, 13]$$

but note that

$$\text{NTT}(\mathcal{F} +_q \mathcal{G}) = [6, 7, 14, 10, 0, 5, 5, 3]$$

$$\text{NTT}(\mathcal{F} \circ_q \mathcal{G}) = [2, 9, 14, 3, 0, 10, 5, 16].$$

Therefore $\text{NTT}(\mathcal{F} +_q \mathcal{G}) = \text{NTT}(\mathcal{F}) +_q \text{NTT}(\mathcal{G})$ is satisfied and it is clear that $\text{NTT}(\mathcal{F} \circ_q \mathcal{G}) \neq \text{NTT}(\mathcal{F}) \circ_q \text{NTT}(\mathcal{G})$.

2.3 Description of NTRU-NTT

As mentioned in the papers (Ducas et al., 2013; Bayer-Fluckiger and Suarez, 2006) the arithmetic of NTRU-NTT depends on two integer parameters (n, q) . Let $\mathbb{Z}_q = \mathbb{Z}/q\mathbb{Z}$ denote the ring of integers modulo q . The operations of NTRU-NTT took place in the ring of truncated polynomials $\mathbf{R}_q = \mathbb{Z}_q[x]/\langle x^n + 1 \rangle$. Where n is a power of 2 and q is a sufficiently large prime such that $q \in 1 + 2n\mathbb{Z}$.

2.4 Key Generation, Encryption and Decryption Process

1. Key Generation

- **Parameters:** n is a power of 2. $f(x) = x^n + 1$. We define the polynomial ring \mathbf{R} as $\mathbf{R} = \mathbb{Z}[x]/\langle f(x) \rangle$ and for sufficiently large prime q we have $\mathbf{R}_q = \mathbf{R}/q\mathbf{R}$.
- **Private Key:** $s, g \in \mathbf{R}$ short polynomial, (i.e. with small coefficients) such that s is invertible \pmod{q} and $\pmod{2}$.
- **Public Key:** $h = 2g \times s^{-1} \in \mathbf{R}_q$ with $g \in \mathbf{R}$ short polynomial.

2. Encryption

- Choose a short vector $e \in \mathbf{R}$ such that $e \pmod{2}$ encodes the desired bit, choose $r \in \mathbf{R}_q$ random and compute the ciphertext $c = h \times r + e \in \mathbf{R}_q$.

3. Decryption

- Multiply the ciphertext and the secret key to get $c \times s = (2g \times r) + (e \times s) \in \mathbf{R}_q$, lift it in \mathbf{R} as $(2g \times r) + (e \times s) \in \mathbf{R}_q$ possible if g, r, e, s are short enough compared to q and reduce it mod 2 obtaining $e \times s \pmod{2}$ and therefore the initial bits.

2.5 Karatsuba Algorithm for 2^α -part Separation

Let $f, g \in \mathbf{R}_q$, we want to split the given large polynomial into 2^α -parts. Here we have to impose one more condition i.e. $\frac{n}{2^{\alpha-1}} \mid q-1$. We can write the n -bit polynomial in the following way:

$$f = \sum_{i=0}^{2^\alpha-1} \left(x^{\frac{in}{2^\alpha}} \cdot f_i \right)$$

and

$$g = \sum_{j=0}^{2^\alpha-1} \left(x^{\frac{jn}{2^\alpha}} \cdot g_j \right)$$

where f_i and $g_j \forall i, j = 0, \dots, 2^{\alpha-1}$ are the primary polynomials same as that of f_0, f_1, g_0, g_1 for the case of $\alpha = 1$.

Then we have the polynomial multiplication as:

$$\begin{aligned} h &= f \cdot g \\ &= \sum_{i=0}^{2^{\alpha}-1} \left(x^{\frac{in}{2^{\alpha}}} f_i \right) \cdot \sum_{j=0}^{2^{\alpha}-1} \left(x^{\frac{jn}{2^{\alpha}}} g_j \right) \\ &= \sum_{i=0}^{2^{\alpha}-1} \sum_{j=0}^{2^{\alpha}-1} \left(x^{\frac{(i+j)n}{2^{\alpha}}} f_i \cdot g_j \right) \end{aligned}$$

When $\alpha = 1$, we have the Karatsuba algorithm for 2-part separation as follows:

$$f = f_0 + x^{\frac{n}{2}} f_1, \quad g = g_0 + x^{\frac{n}{2}} g_1$$

where f_0, f_1, g_0, g_1 are the polynomials of lower degree, called the primary polynomials. Then the product of the two polynomials are given by:

$$\begin{aligned} h &= f_0 \cdot g_0 - f_1 \cdot g_1 + x^{\frac{n}{2}} ((f_0 + f_1) \cdot (g_0 + g_1) \\ &\quad - f_0 \cdot g_0 - f_1 \cdot g_1) \end{aligned}$$

2.6 Limitation of Karatsuba Algorithm

When it comes to polynomial optimization in NTRU using Karatsuba, we face certain parametric limitations. Karatsuba Algorithm that we have discussed so far can only be applied on the NTRU Cryptosystem for $n \leq 768$. For further details one can refer to (Dai et al., 2018, Section 4.2.5). This can be a major setback, as the security standard for the lattice based cryptosystems like NTRU depends on the higher values of n i.e. the higher dimension lattices. In order to overcome the parametric limitations we propose to use the Hybridized NTT-Karatsuba Algorithm, to be discussed in the next section.

3 Hybridized NTT-Karatsuba Multiplication

The idea of combining both Number Theoretic Transformation and Karatsuba Algorithm has been mentioned in the paper by (Zhu et al., 2019). Still now the application of this approach is not available. Here we propose to apply the Hybridized NTT-Karatsuba Algorithm for optimizing NTRU-NTT Cryptosystem. Also we will be providing various technical improvement and practical example in order to implement the Hybridized Algorithm in practice.

3.1 Why Hybridization is Necessary?

- When it comes to optimizing NTRU polynomial multiplication using Karatsuba there are some limitations based on parameters. This algorithm handles polynomial multiplications of degree less than 768 as mentioned in the work (Dai et al., 2018, Section 4.2.5). This limitation over the parameter n can be overcome by using the hybridized technique.
- While multiplying two polynomials using NTT we know that the multiplication is given by $h = f \cdot g = \text{NTT}^{-1}(\text{NTT}(\mathcal{F}) \circ_q \text{NTT}(\mathcal{G}))$. By hybridizing with Karatsuba we only need to find the NTT^{-1} of NTT for the multiplication of primary polynomials f_0, f_1, g_0, g_1 . This could reduce the time complexity of the algorithm. As we have seen that Karatsuba algorithm breaks large degree polynomials into combination of smaller degree polynomial, this attribute to acceleration of component wise multiplication of NTT once the Hybridized technique is applied.

3.2 Hybridized NTT-Karatsuba Algorithm for 2-part Separation Corresponding to $\alpha = 1$

Let $f, g \in \mathbf{R}_q$ be any two of degree n , where n is a power of 2 and $n \mid q - 1$. We can split the higher degree polynomials into primary polynomials as follows:

$$f = f_0 + x^{\frac{n}{2}} f_1, \quad g = g_0 + x^{\frac{n}{2}} g_1$$

and we get the product as

$$\begin{aligned} h &= f_0 \cdot g_0 - f_1 \cdot g_1 + x^{\frac{n}{2}} ((f_0 + f_1) \cdot (g_0 + g_1) \\ &\quad - f_0 \cdot g_0 - f_1 \cdot g_1) \end{aligned}$$

By the definition of NTT in subsection we know that h is given by

$$h = \text{NTT}^{-1}(\hat{\mathcal{F}} \circ_q \hat{\mathcal{G}}) \pmod{(x^n + 1)}$$

where \circ_q is the component-wise product, where $\hat{\mathcal{F}}, \hat{\mathcal{G}}$ is the NTT of the n -dimensional vectors $\mathcal{F}, \mathcal{G} \in \mathbb{Z}_q^n$. Here also we apply same concept, but over each com-

ponent. Hence we have

$$\begin{aligned}
h &= f \cdot g \\
&= \left(f_0 + x^{\frac{n}{2}} f_1 \right) \cdot \left(g_0 + x^{\frac{n}{2}} g_1 \right) \\
&= f_0 \cdot g_0 - f_1 \cdot g_1 + \\
&\quad x^{\frac{n}{2}} \left((f_0 + f_1) \cdot (g_0 + g_1) - f_0 \cdot g_0 - f_1 \cdot g_1 \right) \\
&= \text{NTT}^{-1} \left(\text{NTT}(f_0 \cdot g_0 - f_1 \cdot g_1 + \right. \\
&\quad \left. x^{\frac{n}{2}} \left((f_0 + f_1) \cdot (g_0 + g_1) - f_0 \cdot g_0 - f_1 \cdot g_1 \right) \right) \\
&= \text{NTT}^{-1} \left(\text{NTT}(f_0 \cdot g_0) - \text{NTT}(f_1 \cdot g_1) \right. \\
&\quad \left. + \text{NTT}(x^{\frac{n}{2}}) \text{NTT} \left((f_0 + f_1) \cdot (g_0 + g_1) \right) \right. \\
&\quad \left. - (f_0 \cdot g_0) - (f_1 \cdot g_1) \right) \\
&= \text{NTT}^{-1} \left(\hat{\mathcal{F}}_0 \circ \hat{\mathcal{G}}_0 - \hat{\mathcal{F}}_1 \circ \hat{\mathcal{G}}_1 + \right. \\
&\quad \left. + x^{\frac{n}{2}} \circ (\hat{\mathcal{F}}_0 + \hat{\mathcal{F}}_1) \circ (\hat{\mathcal{G}}_0 + \hat{\mathcal{G}}_1) - \hat{\mathcal{F}}_0 \circ \hat{\mathcal{G}}_0 - \hat{\mathcal{F}}_1 \circ \hat{\mathcal{G}}_1 \right)
\end{aligned}$$

But the above reasoning claimed in (Zhu et al., 2019, Section 3.1) is wrong and the counter example in section 4.3 show us that the expression

$$\begin{aligned}
h &= \text{NTT}^{-1} \left(\hat{\mathcal{F}}_0 \circ \hat{\mathcal{G}}_0 - \hat{\mathcal{F}}_1 \circ \hat{\mathcal{G}}_1 + \right. \\
&\quad \left. x^{\frac{n}{2}} \circ (\hat{\mathcal{F}}_0 + \hat{\mathcal{F}}_1) \circ (\hat{\mathcal{G}}_0 + \hat{\mathcal{G}}_1) - \hat{\mathcal{F}}_0 \circ \hat{\mathcal{G}}_0 - \hat{\mathcal{F}}_1 \circ \hat{\mathcal{G}}_1 \right) \quad (1)
\end{aligned}$$

is not the correct formula as mentioned in (Zhu et al., 2019) of section 3.1. We claim that the correct formula is:

$$\begin{aligned}
h &= \text{NTT}^{-1} \left(\hat{\mathcal{F}}_0 \circ \hat{\mathcal{G}}_0 - \hat{\mathcal{F}}_1 \circ \hat{\mathcal{G}}_1 \right) \\
&\quad + x^{\frac{n}{2}} \cdot \text{NTT}^{-1} \left(\left((\hat{\mathcal{F}}_0 + \hat{\mathcal{F}}_1) \circ (\hat{\mathcal{G}}_0 + \hat{\mathcal{G}}_1) + \right. \right. \\
&\quad \left. \left. - \hat{\mathcal{F}}_0 \circ \hat{\mathcal{G}}_0 - \hat{\mathcal{F}}_1 \circ \hat{\mathcal{G}}_1 \right) \right) \quad (2)
\end{aligned}$$

As will be shown in Section 4.1, this correct expression will allow us to reduce the value of the parameter q , which in turn gives us much more efficient encryption and decryption for NTRU-NTT.

3.3 Proof of Correctness

We first need a preliminary result.

Lemma 1. NTT is a $(\mathbb{Z}_q^n; +_q)$ group automorphism.

Proof. It is a simple check using definition 2.

- $\text{NTT}([0, 0, \dots, 0]) = [0, 0, \dots, 0]$
- Let $\mathbf{a} = (a_0, a_1, \dots, a_{n-1}) \in \mathbb{Z}_q^n$. Its inverse is $-\mathbf{a} = (-a_0, -a_1, \dots, -a_{n-1})$. For $i = 0, \dots, n-1$, it follows that

$$\text{NTT}(-\mathbf{a})_i = \sum_{j=0}^{n-1} -a_j \omega^{ij} \pmod{q}$$

$$= - \sum_{j=0}^{n-1} a_j \omega^{ij} \pmod{q} = -\text{NTT}(\mathbf{a})_i$$

therefore $\text{NTT}(-\mathbf{a}) = -\text{NTT}(\mathbf{a})$.

- Let $\mathbf{a} = (a_0, a_1, \dots, a_{n-1})$ and $\mathbf{b} = (b_0, b_1, \dots, b_{n-1}) \in \mathbb{Z}_q^n$. For $i = 0, \dots, n-1$, it follows that

$$\begin{aligned}
\text{NTT}(\mathbf{a} + \mathbf{b})_i &= \sum_{j=0}^{n-1} (a_j + b_j) \omega^{ij} \pmod{q} \\
&= \sum_{j=0}^{n-1} a_j \omega^{ij} \pmod{q} + \\
&\quad \sum_{j=0}^{n-1} b_j \omega^{ij} \pmod{q} \\
&= \text{NTT}(\mathbf{a})_i + \text{NTT}(\mathbf{b})_i
\end{aligned}$$

therefore $\text{NTT}(\mathbf{a} + \mathbf{b}) = \text{NTT}(\mathbf{a}) + \text{NTT}(\mathbf{b})$.

This completes the proof for Lemma 1. \square

Lemma 2. NTT^{-1} is a $(\mathbb{Z}_q^n; +_q)$ group automorphism.

Proof. The proof follows from Lemma 1 and definition 3.

- $\text{NTT}^{-1}([0, 0, \dots, 0]) = [0, 0, \dots, 0]$
- Let $\mathbf{a} = (a_0, a_1, \dots, a_{n-1}) \in \mathbb{Z}_q^n$. Its inverse is $-\mathbf{a} = (-a_0, -a_1, \dots, -a_{n-1})$. For $i = 0, \dots, n-1$, it follows that

$$\begin{aligned}
\text{NTT}^{-1}(-\mathbf{a})_i &= n^{-1} \sum_{j=0}^{n-1} -a_j \omega^{-ij} \pmod{q} \\
&= -n^{-1} \sum_{j=0}^{n-1} a_j \omega^{-ij} \pmod{q} \\
&= -\text{NTT}^{-1}(\mathbf{a})_i
\end{aligned}$$

therefore $\text{NTT}^{-1}(-\mathbf{a}) = -\text{NTT}^{-1}(\mathbf{a})$.

- Let $\mathbf{a} = (a_0, a_1, \dots, a_{n-1})$ and $\mathbf{b} = (b_0, b_1, \dots, b_{n-1}) \in \mathbb{Z}_q^n$. For $i = 0, \dots, n-1$, it follows that

$$\begin{aligned}
\text{NTT}^{-1}(\mathbf{a} + \mathbf{b})_i &= n^{-1} \sum_{j=0}^{n-1} (a_j + b_j) \omega^{-ij} \pmod{q} \\
&= n^{-1} \sum_{j=0}^{n-1} a_j \omega^{-ij} \pmod{q} \\
&\quad + n^{-1} \sum_{j=0}^{n-1} b_j \omega^{-ij} \pmod{q} \\
&= \text{NTT}^{-1}(\mathbf{a})_i + \text{NTT}^{-1}(\mathbf{b})_i
\end{aligned}$$

therefore $\text{NTT}^{-1}(\mathbf{a} + \mathbf{b}) = \text{NTT}^{-1}(\mathbf{a}) + \text{NTT}^{-1}(\mathbf{b})$.

This completes the proof for Lemma 2. \square

Proposition 1. Formula (2) correctly recovers the product between two polynomials $f, g \in \mathbf{R}_q$

Proof. We simply need to prove that

$$\text{NTT}^{-1} \left(\hat{\mathcal{F}}_0 \circ \hat{\mathcal{G}}_0 - \hat{\mathcal{F}}_1 \circ \hat{\mathcal{G}}_1 \right) = f_0 g_0 - f_1 g_1 \quad (3)$$

and

$$\begin{aligned} \text{NTT}^{-1} \left(\left(\hat{\mathcal{F}}_0 + \hat{\mathcal{F}}_1 \right) \circ \left(\hat{\mathcal{G}}_0 + \hat{\mathcal{G}}_1 \right) - \hat{\mathcal{F}}_0 \circ \hat{\mathcal{G}}_0 - \hat{\mathcal{F}}_1 \circ \hat{\mathcal{G}}_1 \right) \\ = (f_0 + f_1) \cdot (g_0 + g_1) - f_0 \cdot g_0 - f_1 \cdot g_1 \quad (4) \end{aligned}$$

We start by proving (3). Let

$$\mathcal{H}_0 = \text{NTT}^{-1} \left(\hat{\mathcal{F}}_0 \circ \hat{\mathcal{G}}_0 - \hat{\mathcal{F}}_1 \circ \hat{\mathcal{G}}_1 \right)$$

we have

$$\begin{aligned} \mathcal{H}_0 &= \text{NTT}^{-1} \left(\hat{\mathcal{F}}_0 \circ \hat{\mathcal{G}}_0 \right) - \text{NTT}^{-1} \left(\hat{\mathcal{F}}_1 \circ \hat{\mathcal{G}}_1 \right) \\ &= f_0 g_0 - f_1 g_1, \end{aligned}$$

where the first equality follows from Lemma 2 and the second equality follows from (Pollard, 1971). Next we need to show (4). Let

$$\begin{aligned} \mathcal{H}_1 &= \text{NTT}^{-1} \left(\left(\hat{\mathcal{F}}_0 + \hat{\mathcal{F}}_1 \right) \circ \left(\hat{\mathcal{G}}_0 + \hat{\mathcal{G}}_1 \right) \right. \\ &\quad \left. - \hat{\mathcal{F}}_0 \circ \hat{\mathcal{G}}_0 - \hat{\mathcal{F}}_1 \circ \hat{\mathcal{G}}_1 \right) \end{aligned}$$

we have

$$\begin{aligned} \mathcal{H}_1 &= \text{NTT}^{-1} \left(\left(\hat{\mathcal{F}}_0 + \hat{\mathcal{F}}_1 \right) \circ \left(\hat{\mathcal{G}}_0 + \hat{\mathcal{G}}_1 \right) \right) \\ &\quad - \text{NTT}^{-1} \left(\hat{\mathcal{F}}_0 \circ \hat{\mathcal{G}}_0 \right) - \text{NTT}^{-1} \left(\hat{\mathcal{F}}_1 \circ \hat{\mathcal{G}}_1 \right) \\ &= \text{NTT}^{-1} \left(\text{NTT}(\mathcal{F}_0 + \mathcal{F}_1) \circ \text{NTT}(\mathcal{G}_0 + \mathcal{G}_1) \right) \\ &\quad - \text{NTT}^{-1} \left(\hat{\mathcal{F}}_0 \circ \hat{\mathcal{G}}_0 \right) - \text{NTT}^{-1} \left(\hat{\mathcal{F}}_1 \circ \hat{\mathcal{G}}_1 \right) \\ &= (f_0 + f_1) \cdot (g_0 + g_1) - f_0 g_0 - f_1 g_1 \end{aligned}$$

where, again, the first equality follows from Lemma 2 and the third equality follows from (Pollard, 1971). This completes the proof of proposition 1 and hence the proof of correctness. \square

4 Compression of Public Key (Parameter q) using Hybridized Technique

In this section we will discuss about the significance of the parameters n and q . Here n corresponds to the

security parameter which is the dimension of the lattice under consideration and prime number q decide how large the ring \mathbf{R}_q will be. If the value of the parameter q is large then the key size of the underlying cipher text will also be large. This could result in increasing bandwidth which in turn decreases the efficiency of the algorithm (Akleyek et al., 2015; Chen et al., 2014). Our aim of this section is to clearly explain the calculation of the parameter q to the reader and illustrate how we can optimize the value of the parameter q through specific examples. Here we use the 2^α -part separation technique introduced in (Zhu et al., 2019) and calculate the value of q by varying the value of α for a given value of n . We showed that by using the 2^α -part separation technique we could decrease the value of q by a substantial amount in comparison to the previous results (Akleyek et al., 2015; Chen et al., 2014). We could conclude that these optimized value of the parameter q for large value of n have significant positive effect in efficiency if implemented correctly.

4.1 Calculation of q

Till now we have directly stated the case respective values of q , required for the particular example. But we have not stated the method of calculating the parameter q . As we already know that q is a sufficiently large prime modulus and this parameter defines how large the parent ring structure will be. In the cryptographic language, the key size of the cipher depends of the value of q . Larger the value of q the key size will be more. But in order to develop a more efficient post-quantum algorithm we need to decrease the size of the ciphertext.

Now we give the condition for finding the value of q for the following n -degree input polynomials:

$$\begin{aligned} f &= a_0 + a_1 x + \dots + a_{n-1} x^{n-1} \quad \text{and} \\ g &= b_0 + b_1 x + \dots + b_{n-1} x^{n-1}. \end{aligned}$$

Let $\max\{a_i, b_j\} \leq d, \forall i, j = 0, 1, \dots, n-1$. We define the Maximum Modulus $M = d^2 n$, subsequently we also define another parameter $Q = M + 1$. Then the sufficiently large prime modulus should be $q \geq Q$. With this condition we have to keep in mind the original condition on q as $n \mid q - 1$.

In order to keep the value of q to be comparatively small in our illustrated example 1 we have chosen the coefficients $a_i, b_j \in \mathbb{Z}_2$. But this is not always the case, we can certainly have input polynomials with larger coefficients.

Consider the following example. Let the value of $d = 9$ and $n = 512$. Calculate the prime modulus q

for $\alpha = 1$ and $\alpha = 3$.

For $\alpha = 1$:

We have 2-part hybridized NTT-Karatsuba algorithm with the precondition that $n \mid q - 1$. Therefore we have $512 \mid q - 1 \implies q = 512k + 1, \forall k \in \mathbb{N}$. On the other hand $Q = 512 \cdot (9)^2 + 1 = 41473 \implies q \geq 41473$. We need to find a least positive k s.t. $q = 512k + 1$ and $q \geq 41473$. Such a suitable value of k is 89. The value the parameter $q = 45569$, which is a prime.

For $\alpha = 3$:

We have, 2^3 -part hybridized NTT-Karatsuba algorithm with the precondition that $\frac{n}{2^{\alpha-1}} \mid q - 1$ i.e. $\frac{512}{2^{3-1}} \mid q - 1$. Therefore we have $128 \mid q - 1 \implies q = 128k + 1, \forall k \in \mathbb{N}$. On the other hand $Q = 512 \cdot (9)^2 + 1 = 41473 \implies q \geq 41473$. We need to find a least positive k s.t. $q = 128k + 1$ and $q \geq 41473$. Such a suitable value of k is 326. The value the parameter $q = 41729$, which is a prime.

We noticed that with the same input parameters, but by increasing the value of α from 1 to $\alpha = 3$, the value of the parameter q decreases from $q = 44569$ to $q = 41479$. Therefore, the hybridized 2^α -part separation method enhances the efficiency of the NTRU-NTT algorithm by sufficiently reducing the key size of the cipher text.

4.2 New Parametric Values for NTRU-NTT

Till now there has been no NTRU-NTT algorithm for $n = 2048$. As we have mentioned in the beginning that our aim for implementing the hybridized NTT-Karatsuba algorithm is to work on higher dimensional lattices. In order to achieve higher bit security of the improved NTRU-NTT (Lyubashevsky and Seiler, 2019) we need to increase the value of n . But one of the main difficulty that the cryptographer may face while working over such higher dimension lattices is the substantial increase in the value of the prime modulus q , which results in the increase in the running time of the algorithm. If the parameter q becomes too large the key size of the ciphertext will be large too, which will result in the decrease in the efficiency of the algorithm.

So keeping in mind the security standard as well as the computational complexity, we propose to use the hybridized 2^α -part separation method in order to keep the value q considerably smaller than that of the values mentioned in the papers (Akleyek et al., 2015; Chen et al., 2014). More precisely,

- in (Chen et al., 2014, Section III) partial results related to Homomorphic Encryption Scheme were obtained: the value of the prime modulus q for $n = 1024$ is 1061093377 and for $n = 2048$ is $2^{57} + 25 \cdot 2^{13} + 1$, which is significantly larger than the improved prime modulus suggested earlier.
- in (Akleyek et al., 2015, Section 4) is mentioned another result related to the value of the parameter q : the value of the prime modulus q for $n = 1024$ is 8383489.

Our suggestions for q values are

i) **NTRU-NTT for $n = 1024$**

Let the value of the parameter d be 9 i.e. the maximum value of the coefficients of the input polynomial is 9, therefore $q \geq 9^2 \cdot 1024 + 1 \implies q \geq 82945$. We know that the precondition must hold $\frac{n}{2^{\alpha-1}} \mid q - 1$.

$$\alpha = 2 \implies \frac{1024}{2^{2-1}} \mid q - 1 \implies q = 512k + 1, k \in \mathbb{N}$$

The suitable value of a least positive k satisfying both the condition is 164. Therefore the value of the prime modulus q is 83969. Our value of q for $\alpha = 2$ is sufficiently smaller than the previous results i.e. 1061093377 and 8383489. By using this approach we can sufficiently reduce the key size of the cipher.

ii) **NTRU-NTT for $n=2048$**

Let the value of the parameter d be 9 i.e. the maximum value of the coefficients of the input polynomial is 9, therefore $q \geq 9^2 \cdot 2048 + 1 \implies q \geq 165889$. We know that the precondition must hold $\frac{n}{2^{\alpha-1}} \mid q - 1$.

$$\alpha = 2 \implies \frac{2048}{2^{2-1}} \mid q - 1 \implies q = 1024k + 1, k \in \mathbb{N}$$

The suitable value of a least positive k satisfying both the condition is 172. Therefore the value of the prime modulus q is 176129. Again our value of q for $\alpha = 2$ is sufficiently smaller than the previous result i.e. $2^{57} + 25 \cdot 2^{13} + 1$.

By using this approach we can sufficiently reduce the key size of the cipher. Also note that by using our result the key size of the cipher for $n = 2048$ is smaller than the key size of the cipher for $n = 1024$ used in previous papers. This clearly shows that our approach could be beneficial in order to compress the public key even if we are working on such higher dimensional lattices like $n = 2048$. Further we can compress the prime modulus q for $n = 2048$ by increasing the value of α , resulting in some interesting parametric values. As an example,

$$\alpha = 3 \implies \frac{2048}{2^{3-1}} \mid q - 1 \implies q = 512k + 1, k \in \mathbb{N}$$

The suitable value of a least positive k satisfying both the conditions is 329. Therefore the improved value of the prime modulus q is 168449. As another example,

$$\alpha = 4 \implies \frac{2048}{2^{4-1}} \mid q-1 \implies q = 256k + 1, k \in \mathbb{N}$$

The suitable value of a least positive k satisfying both the condition is 651. Therefore another improved value of the prime modulus q is 166657.

4.3 Hybridized Karatsuba-NTT: A Complete Example

In this section, we give a practical example, showing how the computations of the incorrect (1) and correct (2) formulas are performed. In order to do that, we choose the following two polynomials $f, g \in \mathbf{R}_{17} = \mathbb{Z}_{17}[x]/\langle x^8 + 1 \rangle$:

$$\begin{aligned} f &= 1 + x + x^2 + x^3 + x^4 + x^5 + x^6 + x^7 \\ g &= 1 + x^3 + x^6 + x^7 \end{aligned}$$

therefore with parameters $n = 8$ and $q = 17$. Moreover, we choose $\omega \equiv 2 \pmod{17}$ as a primitive 8-th root of unity along with $\omega^{-1} \equiv 9 \pmod{17}$. We can split the polynomials f and g into 2 parts as follows:

$$\begin{aligned} f &= (1 + x + x^2 + x^3) + x^4(1 + x + x^2 + x^3) \\ g &= (1 + 0 \cdot x + 0 \cdot x^2 + 1 \cdot x^3) + \\ &\quad x^4(0 + 0 \cdot x + 1 \cdot x^2 + 1 \cdot x^3) \end{aligned}$$

therefore we get

$$\begin{aligned} f_0 &= 1 + x + x^2 + x^3 \implies \mathcal{F}_0 = [1, 1, 1, 1, 0, 0, 0, 0] \\ f_1 &= 1 + x + x^2 + x^3 \implies \mathcal{F}_1 = [1, 1, 1, 1, 0, 0, 0, 0] \\ g_0 &= 1 + x^3 \implies \mathcal{G}_0 = [1, 0, 0, 1, 0, 0, 0, 0] \\ g_1 &= x^2 + x^3 \implies \mathcal{G}_1 = [0, 0, 1, 1, 0, 0, 0, 0] \end{aligned}$$

From definition 2, we have $\hat{\mathcal{F}}_0 = \text{NTT}(\mathcal{F}_0) = \sum_{j=0}^7 a_j \omega^{ij} \pmod{17}$, $\forall i = 0, \dots, 7$. We are going to explicitly show how $\text{NTT}(\mathcal{F}_0)$ is calculated, which will help the reader to understand the calculation of NTT for the other vectors. In particular we have

$$(\hat{\mathcal{F}}_0)_0 = a_0 \omega^{0 \cdot 0} + a_1 \omega^{0 \cdot 1} + \dots + a_7 \omega^{0 \cdot 7} \pmod{17} = 4$$

and, analogously,

$$\begin{aligned} (\hat{\mathcal{F}}_0)_1 &= 15 \pmod{17} & (\hat{\mathcal{F}}_0)_2 &= 0 \pmod{17} \\ (\hat{\mathcal{F}}_0)_3 &= 7 \pmod{17} & (\hat{\mathcal{F}}_0)_4 &= 7 \pmod{17} \\ (\hat{\mathcal{F}}_0)_5 &= 12 \pmod{17} & (\hat{\mathcal{F}}_0)_6 &= 0 \pmod{17} \\ (\hat{\mathcal{F}}_0)_7 &= 4 \pmod{17} \end{aligned}$$

Therefore we have

$$\hat{\mathcal{F}}_0 = \hat{\mathcal{F}}_1 = [4, 15, 0, 7, 0, 12, 0, 4]$$

and, similarly, we calculate

$$\begin{aligned} \hat{\mathcal{G}}_0 &= \text{NTT}(\mathcal{G}_0) = [2, 9, 14, 3, 0, 10, 5, 16] \\ \hat{\mathcal{G}}_1 &= \text{NTT}(\mathcal{G}_1) = [2, 12, 12, 15, 0, 13, 3, 11] \end{aligned}$$

Let us now calculate some components using notation in definition 1 and useful for formulas (1) and (2):

1. $\hat{\mathcal{F}}_0 \circ_q \hat{\mathcal{G}}_0 = [8, 16, 0, 4, 0, 1, 0, 13]$
2. $\hat{\mathcal{F}}_1 \circ_q \hat{\mathcal{G}}_1 = [8, 10, 0, 3, 0, 3, 0, 10]$
3. $\hat{\mathcal{F}}_0 +_q \hat{\mathcal{F}}_1 = [8, 13, 0, 14, 0, 7, 0, 8]$
4. $\hat{\mathcal{G}}_0 +_q \hat{\mathcal{G}}_1 = [4, 4, 9, 1, 0, 6, 8, 10]$
5. $(\hat{\mathcal{F}}_0 +_q \hat{\mathcal{F}}_1) \circ_q (\hat{\mathcal{G}}_0 +_q \hat{\mathcal{G}}_1) = [15, 1, 0, 14, 0, 8, 0, 12]$
6. x^4 can be seen as the vector $[0, 0, 0, 0, 1, 0, 0, 0]$, so $\text{NTT}([0, 0, 0, 0, 1, 0, 0, 0]) = [1, 16, 1, 16, 1, 16, 1, 16]$ (being $n = 8$)
7. $\hat{\mathcal{F}}_0 \circ_q \hat{\mathcal{G}}_0 - \hat{\mathcal{F}}_1 \circ_q \hat{\mathcal{G}}_1 = [0, 6, 0, 1, 0, 15, 0, 3]$
8. $(\hat{\mathcal{F}}_0 +_q \hat{\mathcal{F}}_1) \circ_q (\hat{\mathcal{G}}_0 +_q \hat{\mathcal{G}}_1) - \hat{\mathcal{F}}_0 \circ_q \hat{\mathcal{G}}_0 - \hat{\mathcal{F}}_1 \circ_q \hat{\mathcal{G}}_1 = [16, 9, 0, 7, 0, 4, 0, 6]$

It is now a straightforward check that formula (1) gives

$$\begin{aligned} h &= \text{NTT}^{-1}([0, 6, 0, 1, 0, 15, 0, 3] + \\ &\quad [16, 8, 0, 10, 0, 13, 0, 11]) \\ &= \text{NTT}^{-1}([16, 14, 0, 11, 0, 11, 0, 14]) \end{aligned}$$

From definition 3, we have $h = \text{NTT}^{-1}(\mathcal{H}) = n^{-1} \sum_{j=0}^7 a_j \omega^{-ij} \pmod{17}$, $\forall i = 0, \dots, 7$. In particular we have

$$\begin{aligned} h_0 &= 15 [a_0 \omega^{-0 \cdot 0} + a_1 \omega^{-0 \cdot 1} + \dots + a_7 \omega^{-0 \cdot 7}] \\ &= 4 \pmod{17} \end{aligned}$$

and, analogously,

$$\begin{aligned} h_1 &= 4 \pmod{17} & h_2 &= 2 \pmod{17} \\ h_3 &= 0 \pmod{17} & h_4 &= 0 \pmod{17} \\ h_5 &= 0 \pmod{17} & h_6 &= 2 \pmod{17} \\ h_7 &= 4 \pmod{17} \end{aligned}$$

Therefore we have

$$h = [4, 4, 2, 0, 0, 0, 2, 4] \rightarrow 4 + 4x + 2x^2 + 2x^6 + 4x^7$$

The formula (2) gives

$$\begin{aligned} h &= \text{NTT}^{-1}([0, 6, 0, 1, 0, 15, 0, 3]) \\ &\quad + x^4 \cdot \text{NTT}^{-1}([16, 9, 0, 7, 0, 4, 0, 6]) \\ &= [1, 1, 0, 0, 16, 16, 0, 0] + x^4 \cdot [1, 1, 2, 4, 3, 3, 2, 0] \\ &= [15, 15, 15, 0, 0, 0, 2, 4] \\ &\rightarrow 15 + 15x + 15x^2 + 2x^6 + 4x^7 \end{aligned}$$

The latter is the correct result and can be checked with the well known algorithm of the polynomial product.

5 Conclusion and Future Work

In this paper we have provided an improved polynomial optimization technique for the NTRU-NTT cryptosystem. The corrected hybridized product formula could provide optimized result for the existing NTRU algorithm when implemented. The application of the 2^α -part separation method in decreasing the value of the prime modulus q while keeping the value of the security parameter n considerably high has been introduced in the paper for the first time. We have successfully shown that for $n = 1024$ the value of the parameter q has been decreased from 1061093377 to 83969 and for $n = 2048$ the value of q has been decreased from $2^{57} + 25 \cdot 2^{13} + 1$ to 166657. This could be considered a substantial improvement in terms of decreasing the key sizes. As a part of future work, it would be interesting to generalize the concept and provide a similar mathematical proof for higher values of α i.e. for any 2^α -part separation. The theoretical compression in the value of the prime modulus q corresponding to some specific values of n has been shown in the paper. It would also be very interesting to implement these parametric values and check the difference in the time complexity for the NTRU cryptosystem.

ACKNOWLEDGEMENTS

This work was in part financially supported by the Swedish Foundation for Strategic Research, grant RIT17-0035. We would like to sincerely thank Prof. Martin Hell and Prof. Elena Pagnin from the Department of Electrical and Information Technology, Lund University for their valuable insights and discussions in order to successfully complete the work.

REFERENCES

- Akleyek, S., Dağdelen, Ö., and Tok, Z. Y. (2015). On the efficiency of polynomial multiplication for lattice-based cryptography on GPUs using CUDA. In *International Conference on Cryptography and Information Security in the Balkans*, pages 155–168. Springer.
- Alagic, G., Alagic, G., Alperin-Sheriff, J., Apon, D., Cooper, D., Dang, Q., Liu, Y.-K., Miller, C., Moody, D., Peralta, R., et al. (2019). *Status report on the first round of the NIST post-quantum cryptography standardization process*. US Department of Commerce, National Institute of Standards and Technology.
- Avanzi, R., Bos, J., Ducas, L., Kiltz, E., Lepoint, T., Lyubashevsky, V., Schanck, J. M., Schwabe, P., Seiler, G., and Stehlé, D. (2017). CRYSTALS-KYBER algorithm specifications and supporting documentation. *NIST PQC Round, 2:4*.
- Bayer-Fluckiger, E. and Suarez, I. (2006). Ideal lattices over totally real number fields and euclidean minima. *Archiv der Mathematik*, 86(3):217–225.
- Bernstein, D. J. and Lange, T. (2017). Post-quantum cryptography. *Nature*, 549(7671):188–194.
- Chen, C., Danba, O., Hoffstein, J., Hülsing, A., Rijneveld, J., Schanck, J. M., Schwabe, P., Whyte, W., and Zhang, Z. (2019). Algorithm specifications and supporting documentation. *Brown University and On-board security company, Wilmington USA*.
- Chen, D. D., Mentens, N., Vercauteren, F., Roy, S. S., Cheung, R. C., Pao, D., and Verbauwhede, I. (2014). High-speed polynomial multiplication architecture for ring-lwe and she cryptosystems. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 62(1):157–166.
- Dai, W., Whyte, W., and Zhang, Z. (2018). Optimizing polynomial convolution for NTRUEncrypt. *IEEE Transactions on Computers*, 67(11):1572–1583.
- Ducas, L., Durmus, A., Lepoint, T., and Lyubashevsky, V. (2013). Lattice signatures and bimodal Gaussians. In *Annual Cryptology Conference*, pages 40–56. Springer.
- Fedorenko, S. and Trifonov, P. (2002). On computing the fast Fourier transform over finite fields. In *Proc. 8th Int. Workshop on Algebraic and Combinatorial Coding Theory, Tsarskoe Selo, Russia*, pages 108–111.
- Hoffstein, J., Pipher, J., and Silverman, J. H. (1998). NTRU: A ring-based public key cryptosystem. In *International Algorithmic Number Theory Symposium*, pages 267–288. Springer.
- Hülsing, A., Rijneveld, J., Schanck, J., and Schwabe, P. (2017). High-speed key encapsulation from ntru. In *International Conference on Cryptographic Hardware and Embedded Systems*, pages 232–252. Springer.
- Karmakar, A., Mera, J. M. B., Roy, S. S., and Verbauwhede, I. (2018). SABER on arm CCA-secure module lattice-based key encapsulation on arm. *Cryptology ePrint Archive*.
- Lyubashevsky, V. and Seiler, G. (2019). NTTRU: truly fast NTRU using NTT. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, pages 180–201.
- Pollard, J. M. (1971). The fast Fourier transform in a finite field. *Mathematics of computation*, 25(114):365–374.
- Zhou, S., Xue, H., Zhang, D., Wang, K., Lu, X., Li, B., and He, J. (2018). Preprocess-then-NTT technique and its applications to KYBER and NEW HOPE. In *International Conference on Information Security and Cryptology*, pages 117–137. Springer.
- Zhu, Y., Liu, Z., and Pan, Y. (2019). When NTT meets Karatsuba: Preprocess-then-NTT technique revisited. *IACR Cryptol. ePrint Arch.*, 2019:1079.