An Introduction to Number Theoretic Transform in Post-Quantum Lattice Cryptography

EE5163 Advanced Digital Signal Processing: Final Report

Lo-Chun, Chou
R13922136 [1]

[1]Department of Computer Science, National Taiwan University

June 11, 2025

Abstract

In the domain of post-quantum cryptography (PQC), lattice-based cryptography stands out as one of the most promising approaches due to its balanced performance between security and efficiency. A key computational task in these schemes is polynomial multiplication, where Number Theoretic Transform (NTT) is particularly effective for this task. NTT achieves quasilinear time complexity $O(n \log n)$, similar to the Fast Fourier Transform (FFT), but operates entirely over integers, thus avoiding rounding errors from floating-point arithmetic. In this report, we explore the foundations of NTT, drawing from some recent research papers, and provide an overview of its principles, implementation strategies, and applications in lattice-based cryptographic systems.

1 Introduction

1.1 Motivation

While both Number Theoretic Transform (NTT) and Fast Fourier Transform (FFT) are used in polynomial multiplication, NTT is a relative new approach, which is often not covered in the textbooks. This lack of accessible resources is also noted in the reference work *A Beginner's Guide to Number Theoretic Transforms (NTTs)*, where the authors mentioned that the lack of guidance and tutorial available in one place is the main obstacle for learning NTT[9], which is also the reason why they wrote the material.

While the beginner's guide offers a solid introduction to the basic ideas of NTT, it occasionally assumes familiarity with advanced mathematical concepts. Therefore, we expand upon its foundation by including additional mathematical background, clarifying essential concepts such as rings, primitive roots, and modular arithmetic, so all mathematical backgrounds that are needed but unfamiliar to a non-math-major would be covered.

Our aim is to provide a basic overview of NTT, so that after reading this report, readers would have a basic understanding of how NTT works, the jargon used in this field, its application and limitations.

1.2 Report Structure

In this report, we would first introduce the mathematical foundations of NTT, which would cover the concepts required for understanding the definitions and basic ideas.

After we're equipped with the necessary knowledge, we could move on to check the definitions for both forward and inverse NTT, some advantages of NTT are then shown after we're familiar with the definitions. Additionally, we would also compare NTT with FFT rigorously to see how they are similar in their structures.

Finally, we would briefly discuss the applications of NTT in lattice-based cryptography, followed with its limitations.

In the last part, we would conclude with a brief summary of the report, and pointed out some future directions for further research.

2 Preliminaries: Mathematical Foundations of the Number Theoretic Transform

2.1 Rings and quotient rings

To properly define primitive roots and illustrate the structure of the Number Theoretic Transform (NTT), it is necessary to first introduce the concepts of rings, ideals, and quotient rings.

These algebraic structures are needed because they provide the foundation for polynomial multiplication in NTT, since in order to do polynomial multiplication for polynomials $G(x)$ and $H(x)$, both of them must belong to the same quotient ring. Due to the fact that quotient rings are constructed on the notions of rings and ideals, we begin by briefly reviewing these foundational concepts.

Definition 2.1 (Ring, [8], pp. 85–86)**.** *A set R is called a **ring** if it has two binary operations, written as addition and multiplication, satisfying the following axioms for all $a, b, c \in R$:*

1. $a + b = b + a$.
2. $a + (b + c) = (a + b) + c$.
3. *An element* 0 *in* R *exists such that* $0 + a = a$ *for all* a.
4. *For each* $a \in R$ *an element* $-a \in R$ *exists such that* $a + (-a) = 0$.

5. $a(bc) = (ab)c$.

6. *An element* 1 *in* R *exists such that* $1 \cdot a = a = a \cdot 1$ *for all* a.

7. $a(b + c) = ab + ac$ *and* $(b + c)a = ba + ca$.

Definition 2.2 (Ideal, [5], p. 1)**.** *A subset* A *of a ring* R *(commutative, with* 1*) is said to be an* ***ideal*** *if*

1. $0 \in A$ *and* $a, b \in A \Rightarrow a + b, -a \in A$ *(so* A *is an additive subgroup of* R*).*

2. $(a \in A, r \in R)$ $ra \in A$.

Intuitively, an ideal is a special subset of a ring, which includes the zero element and is closed under addition, also, it would satisfy that every multiple of an element in A is also in A.

Definition 2.3 (Quotient Ring, [5], p. 3)**.** *Let* A *be an ideal in the ring* R*. The* ***quotient ring*** R/A *is defined as follows:*

$$\begin{aligned} \textit{Set} &\coloneqq \{r + A \mid r \in R\} \qquad \textit{(additive cosets)} \\ 0 &\coloneqq A \\ 1 &\coloneqq 1 + A \\ (r + A) + (s + A) &\coloneqq (r + s) + A \\ (r + A)(s + A) &\coloneqq (rs) + A \end{aligned}$$

where $r, s \in R$.

To be simple, a quotient ring R/A is formed by grouping elements of the ring R into disjoint sets called cosets, where each coset has the form $r + A = \{r + a \mid r \in R\}$. These cosets partition the ring in a way that reflects the structure of the ideal A.

One may wonder why quotient rings are introduced in the literature of NTT. The idea is that, we treat every element in the same coset as "equivalent", just like in modular arithmetic.

An important example would be $\mathbb{Z}/n\mathbb{Z}$, which represents the ring of integers modulo n, where all numbers differed by a multiple of n are considered the same. In the same way, a quotient ring allows us to "mod out" an ideal A and treat every element in A as zero, simplifying the structure of the ring and allowing new algebraic manipulations.

To illustrate what this means, we can consider the following example:

If given $R = \mathbb{Z}$ and $A = 2\mathbb{Z}$, which means that we have the ring of integers and ideal of even numbers, respectively. Then $R/A = \mathbb{Z}/2\mathbb{Z}$ would be defined as having the set:

$$\{r + 2\mathbb{Z} \mid r \in \mathbb{Z}\}$$

Some of its cosets would be:

$$\begin{aligned}\{1+2\mathbb{Z}\} &= \{1,3,5,7,\ldots\}\\ \{2+2\mathbb{Z}\} &= \{2,4,6,8,\ldots\}\\ \{3+2\mathbb{Z}\} &= \{3,5,7,9,\ldots\}\\ &\vdots\end{aligned}$$

However, we can see that $\{1+2\mathbb{Z}\}$ and $\{3+2\mathbb{Z}\}$ are the same, thus there would actually be only two cosets:

$$\begin{aligned}\{0+2\mathbb{Z}\} &= \{0,2,4,6,\ldots\}\\ \{1+2\mathbb{Z}\} &= \{1,3,5,7,\ldots\}\end{aligned}$$

Thus, if we have two elements r_1 and r_2 in the same coset, say $0+2\mathbb{Z}$, then we can write:

$$\begin{aligned}r_1 &= 0+2z_1 \quad \text{for some } z_1 \in \mathbb{Z}\\ r_2 &= 0+2z_2 \quad \text{for some } z_2 \in \mathbb{Z}\end{aligned}$$

and we can see that $r_1 \equiv r_2 \pmod{2\mathbb{Z}}$ since $r_1 - r_2 = 2(z_1 - z_2) \in 2\mathbb{Z}$. This can also be written as $r_1 - r_2 \equiv 0 \pmod{2\mathbb{Z}}$, and $r_1 - r_2 \in 2\mathbb{Z}$. What this means is that, for any element in the ideal $2\mathbb{Z}$, it would be treated as equivalent to 0 in the quotient ring $\mathbb{Z}/2\mathbb{Z}$.

2.2 Modular Arithmetic and Roots of Unity

In this subsection, we first introduce the concept of residue class modulo n, in order to be used in defining $\mathbb{Z}_n$, then we further define the order of an integer under modulo n, so that primitive root can be defined. The reason why primitive root is needed is that it is used to construct the NTT matrix, which we would talk about the details in the next subsection.

Definition 2.4 (Residue Class Modulo n, [8], p. 30)**.** *If a is an integer, its equivalent class $[a]$ with respect to congruence modulo n is called its* ***residue class modulo*** *n, and we write $\overline{a} = [a]$ for convenience:*

$$\overline{a} = [a] = \{x \in \mathbb{Z} \mid x \equiv a \pmod{n}\}.$$

Intuitively saying, after we selected an integer n to be the modulus, we can divide all the integers into n classes, each class is a set of integers that having the same remainder when divided by n.

Therefore, for any integer n, there are n residue classes modulo n, which are $\overline{0}, \overline{1}, \overline{2}, \ldots, \overline{n-1}$, if we collect all of these residue classes as a set, then we get the set of integers modulo n, which is defined as follows:

Definition 2.5 (Integers Modulo n, [8], p. 30)**.** *The set of all residue classes modulo n is denoted*

$$\mathbb{Z}_n = \{\overline{0}, \overline{1}, \overline{2}, \ldots, \overline{n-1}\}.$$

and is called the set of ***integers modulo*** *n.*

Before introducing the definition of primitive root, we first define the order of an integer modulo n as follows:

Definition 2.6 (Order of an Integer Modulo n , [3], p. 147)**.** *Let $n > 1$ and $\gcd(a, n) = 1$. The* ***order of*** *a* ***modulo*** *n is the smallest positive integer k such that*

$$a^k \equiv 1 \pmod{n}.$$

Using the definition of order, we can then define the primitive root as follows:

Definition 2.7 (Primitive Root, [3], p. 150)**.** *If $\gcd(a, n) = 1$ and a is of order $\varphi(n)$ modulo n, then a is called a* ***primitive root*** *of the integer n. Equivalently, a is a primitive root modulo n if*

$$a^{\varphi(n)} \equiv 1 \pmod{n} \quad \textit{but} \quad a^k \not\equiv 1 \pmod{n} \quad \forall\ 1 \leq k < \varphi(n).$$

Note that the primitive root may not be unique, and since we use the primitive root to construct the NTT matrix, the NTT of a polynomial may not be unique and would depend on the choice of the primitive root. [9, pp. 5–6]

In the context of NTT, we choose the modulus n to be a prime number, let it be denoted as q, and under the assumption that our polynomials $G(x)$ and $H(x)$ are of degree $n-1$, we try to find primitive roots ω that would satisfy:

$$\omega^n \equiv 1 \pmod{q} \quad \text{and} \quad \omega^k \not\equiv 1 \pmod{q} \quad \forall\ 1 \leq k < n.$$

Which means that if we're trying to do convolution of polynomials of degree 3, then we should find a primitive root that would be congruent to 1 when raised to the power of 4, under some choice of modulus q.

And such ω is called the **primitive n-th root of unity** in the ring $\mathbb{Z}_q$. [9, p. 4]

2.3 Cyclotomic Polynomials

From the previous subsections, we have already defined the quotient ring and the primitive root. We then introduce the concept of cyclotomic polynomials, which is chosen to construct the ideal of the quotient ring. [7, p. 3]

Definition 2.8 (Cyclotomic Polynomial, [2], p. 11)**.** *The n-th* ***cyclotomic polynomial*** $\Phi_n(x)$ *is defined by*

$$\Phi_n(x) = \prod_{\omega \in P(n)} (x - \omega)$$

where $P(n)$ denotes the set of all primitive n-th roots of unity.

But how do we connect the n-th cyclotomic polynomial to construct the ideal of the quotient ring? To answer this question, we first introduce our focus, the two widely used rings in lattice-based schemes are:

$$\mathbb{Z}_q[x]/(x^n - 1) \quad \text{and} \quad \mathbb{Z}_q[x]/(x^n + 1)$$

[7, p. 3]

Here we could see that the ideals of the quotient rings are $(x^n - 1)$ and $(x^n + 1)$, respectively. To construct these two ideals, we need the following proposition:

Proposition 2.1 (Fundamental Relation, [2], p. 12)**.** *For any positive integer n,*

$$x^n - 1 = \prod_{d|n} \Phi_d(x)$$

The above proposition shows that, if we are to choose the ideal with the form $(x^n - 1)$, then we could construct it by finding all the factors d that could divide n, then multiply all the d-th cyclotomic polynomials together.

To see how this works, we can consider the example of having $n = 4$, with the aid of the following table:

n	$\Phi_n(x)$	n	$\Phi_n(x)$
1	$x - 1$	6	$x^2 - x + 1$
2	$x + 1$	7	$x^6 + x^5 + x^4 + x^3 + x^2 + x + 1$
3	$x^2 + x + 1$	8	$x^4 + 1$
4	$x^2 + 1$	9	$x^6 + x^3 + 1$
5	$x^4 + x^3 + x^2 + x + 1$	10	$x^4 - x^3 + x^2 - x + 1$

Table 1.1: The first ten cyclotomic polynomials.

Figure 1: Cyclotomic Table. Source: [2, p. 11]

We can see that, for $n = 4$, the factors of n are $1, 2, 4$, and the corresponding cyclotomic polynomials are:

$$\Phi_1(x) = x - 1$$
$$\Phi_2(x) = x + 1$$
$$\Phi_4(x) = x^2 + 1$$

According to the fundamental relation, we have:

$$\begin{aligned} x^4 - 1 &= \prod_{d|4} \Phi_d(x) \\ &= \Phi_1(x)\Phi_2(x)\Phi_4(x) \\ &= (x-1)(x+1)(x^2+1) \\ &= (x^2-1)(x^2+1) \\ &= x^4 - 1 \end{aligned}$$

which matches the result in the table.

This could easily be generalized to the case when we choose the ideal with the form $(x^n + 1)$. Observe that:

$$x^{2n} + 1 = (x^n + 1)(x^n - 1)$$

So we have:

$$x^n + 1 = \frac{x^{2n} + 1}{x^n - 1}$$

Using the proposition of the fundamental relation, we have:

$$x^{2n} - 1 = \prod_{d|2n} \Phi_d(x)$$
$$x^n - 1 = \prod_{d|n} \Phi_d(x)$$

Therefore, we can rewrite the above equation* as:

$$x^n + 1 = \frac{x^{2n} + 1}{x^n - 1} = \frac{\prod_{d|2n} \Phi_d(x)}{\prod_{d|n} \Phi_d(x)} = \prod_{d\nmid n,\ d|2n} \Phi_d(x)$$

This means that, if we are to choose the ideal with the form $(x^n + 1)$, then we could construct it by finding all the factors d that could divide $2n$ but not n, then multiply all the d-th cyclotomic polynomials together.

In particular, if n is of the special form:

$$n = 2^k \qquad \text{for some } k \in \mathbb{N}$$

then we have:

$$x^{2^k} + 1 = \prod_{d \nmid 2^k,\ d \mid 2^{k+1}} \Phi_d(x) = \Phi_{2^{k+1}}(x) = \Phi_{2n}(x)$$

Therefore, we arrived at the conclusion that, if we are to choose the ideal with the form $(x^n + 1)$ with $n = 2^k, \quad k \in \mathbb{N}$, then we could construct it by finding the $2n$-th cyclotomic polynomial $\Phi_{2n}(x)$.

This property is quite useful, so in the context of PQC, the chosen ring is mostly $\mathbb{Z}_q[x]/(x^n + 1)$ instead of $\mathbb{Z}_q[x]/(x^n - 1)$. [9, p. 8] And we further denote the $2n$-th root of unity as ψ, with the detailed definition shown below:

Definition 2.9 ($2n$-th Root of Unity, [9], p. 8)**.** *Let $\mathbb{Z}_q$ be an integer ring modulo q, and $n-1$ is the polynomial degree of $G(x)$ and $H(x)$ and ω is the primitive n-th root of unity. Define ψ as the* ***primitive*** *$2n$-**th root of unity***

$$\iff \psi^2 \equiv \omega \pmod{q} \quad \textit{and} \quad \psi^n \equiv -1 \pmod{q}$$

With all of these defined, we can then move on to the next section, where the definition of the Number Theoretic Transform (NTT) would be introduced based on the above definitions.

2.4 Notations

Some of the notations vary across the referenced literature, and are modified to maintin the consistency of the notations.

In this report, we assume that the two given polynomials $G(x)$ and $H(x)$ to be multiplied / convolved, are of degree $n - 1$ if not specified, which can be written as:

$$G(x) = \sum_{i=0}^{n-1} G[i]x^i \quad \text{and} \quad H(x) = \sum_{i=0}^{n-1} H[i]x^i$$

This can also be represented in the form of a vector:

$$G = (G[0], G[1], \ldots, G[n-1]) \quad \text{and} \quad H = (H[0], H[1], \ldots, H[n-1])$$

Other notations follow the same convention as in the preliminaries above.

3 The Number Theoretic Transform (NTT)

3.1 Definition of Forward and Inverse NTT

In the previous subsection, we introduced the formulation of the two widely used quotient rings in lattice-based schemes, if we focus on the case that the ideal is chosen to be of the form $(x^n + 1)$, then the convolution we're dealing with is the **negative-wrapped convolution**, or **negacyclic convolution**. The reason why we're not focusing on the positive-wrapped convolution, even though its more intuitive, is because in order to compute:

$$G(x) * H(x) \pmod{x^n + 1}$$

The positive-wrapped convolution requires the length of the polynomials to be extended to $2n$, which doubles the input length. [1, p. 5]

To continue with negative-wrapped convolution, we then denote the quotient rings by R and R_q, respectively:

$$R = \mathbb{Z}[x]/(x^n + 1) \quad \text{and} \quad R_q = \mathbb{Z}_q[x]/(x^n + 1)$$

Assume the special case where $n = 2^k$, and consider ω to be a primitive n-th root of unity in $\mathbb{Z}_q$ which

$$\omega^n \equiv 1 \pmod{q} \quad \text{where} \quad q \equiv 1 \pmod{2n},\ q : \text{prime},\ n : 2^k$$

we could define the NTT of a polynomial $G(x) \in R_q$ as:

$$\hat{G} = \mathbf{NTT}(G(x)) = \begin{bmatrix} \hat{G}[0] \\ \hat{G}[1] \\ \vdots \\ \hat{G}[n-1] \end{bmatrix}$$

$$\hat{G}[i] = \sum_{j=0}^{n-1} G[j]\psi^j\omega^{ij} \pmod{q} \quad \text{for } i = 0, 1, \ldots, n-1$$

[9, p. 8]

Representing in the matrix multiplication form, we have:

$$\hat{G} = \mathbf{NTT}(G(x)) = \begin{bmatrix} \psi^0\omega^0 & \psi^1\omega^0 & \cdots & \psi^{n-1}\omega^0 \\ \psi^0\omega^0 & \psi^1\omega^1 & \cdots & \psi^{n-1}\omega^{n-1} \\ \vdots & \vdots & \ddots & \vdots \\ \psi^0\omega^0 & \psi^1\omega^{n-1} & \cdots & \psi^{n-1}\omega^{(n-1)(n-1)} \end{bmatrix} \begin{bmatrix} G[0] \\ G[1] \\ \vdots \\ G[n-1] \end{bmatrix}$$

Note that we previously defined the primitive $2n$-th root of unity ψ, which satisfies $\psi^2 \equiv \omega \pmod{q}$, hence, by plugging ψ into the equation and the matrix above, we have:

$$\begin{aligned}\hat{G} &= \mathbf{NTT}(G(x)) \\ \hat{G}[i] &= \sum_{j=0}^{n-1} G[j]\psi^{2ij+j} \pmod{q} \quad \text{for } i = 0, 1, \ldots, n-1\end{aligned}$$

and the matrix form:

$$\hat{G} = \mathbf{NTT}(G(x)) = \begin{bmatrix} \psi^0 & \psi^1 & \cdots & \psi^{n-1} \\ \psi^0 & \psi^3 & \cdots & \psi^{3(n-1)} \\ \vdots & \vdots & \ddots & \vdots \\ \psi^0 & \psi^{(2n-1)} & \cdots & \psi^{(2n-1)(n-1)} \end{bmatrix} \begin{bmatrix} G[0] \\ G[1] \\ \vdots \\ G[n-1] \end{bmatrix}$$

And the inverse NTT of $\hat{G}$ is denoted as $\mathbf{NTT^{-1}}(\hat{G})$, and defined as:

$$\begin{aligned}G &= \mathbf{NTT^{-1}}(\hat{G}) \\ G[i] &= \sum_{j=0}^{n-1} n^{-1}\hat{G}[j]\psi^{-j}\omega^{-ij} \pmod{q} \quad \text{for } i = 0, 1, \ldots, n-1\end{aligned}$$

Similarly, if we plug $\psi^2 \equiv \omega \pmod{q}$ in, then we have:

$$\begin{aligned}G[i] &= \sum_{j=0}^{n-1} n^{-1}\hat{G}[j]\psi^{-j}(\psi^2)^{-ij} \pmod{q} \quad \text{for } i = 0, 1, \ldots, n-1 \\ &= \sum_{j=0}^{n-1} n^{-1}\hat{G}[j]\psi^{-(2ij+j)} \pmod{q} \quad \text{for } i = 0, 1, \ldots, n-1\end{aligned}$$

3.2 Advantages of NTT

Albeit we focused on the negative-wrapped convolution in the previous subsection, and assumed the special case where $n = 2^k$, which is used in the general case, for all kinds of NTT and INTT, they possess the following advantages:

- Both NTT and INTT are linear transformations.
- NTT preserves the randomness, which means that given a random polynomial $G(x)$, $\hat{G}(x) = \mathbf{NTT}(G(x))$ is also a random polynomial.

- Both NTT and INTT preserve the dimension and bit length of the input polynomial, meaning that $\hat{G}(x)$ and $G(x)$ have the same dimension and bit length. This allows the result of NTT $\hat{G}(x)$ to be stored in the same memory space as $G(x)$.
- If an input polynomial $G(x)$ requires to be involved in multiple multiplications, we could compute the NTT of $G(x)$ once, and store $\hat{G}(x)$ to use in subsequent multiplications.

[7, p. 5]

3.3 Comparison with Fast Fourier Transform (FFT)

From the Cooley-Tukey algorithm [4, p. 370], we recall that the DFT of a sequence $x[n]$ of length $N = 2^k$ is given by:

$$\begin{aligned} X[m] &= \sum_{n=0}^{N-1} x[n] e^{-j\frac{2\pi mn}{N}} \\ &= \sum_{n=0}^{N/2-1} x[2n] e^{-j\frac{2\pi m(2n)}{N}} + \sum_{n=0}^{N/2-1} x[2n+1] e^{-j\frac{2\pi m(2n+1)}{N}} \\ &= \sum_{n=0}^{N/2-1} x[2n] e^{-j\frac{2\pi mn}{N/2}} + \underbrace{e^{-j\frac{2\pi m}{N}}}_{\text{twiddle factors}} \sum_{n=0}^{N/2-1} x[2n+1] e^{-j\frac{2\pi mn}{N/2}} \end{aligned}$$

where the second equality is by splitting the summation into two parts, one for the even-indexed terms and the other for the odd-indexed terms, then they could further be derived into two $\frac{N}{2}$-point DFTs, with twiddle factor $e^{-j\frac{2\pi m}{N}}$.

Since NTT is DFT in another ring, we could also apply similar techniques as in the Cooley-Tukey algorithm to optimize NTT, making the original complexity of NTT from $O(n^2)$ to $O(n \log n)$. This is achieved by utilizing the two properties of the primitive $2n$-th root of unity ψ [9, p. 11]:

$$\begin{aligned} &\text{periodicity:} \quad \psi^{k+2n} = \psi^k \\ &\text{symmetry:} \quad \psi^{k+n} = -\psi^k \qquad \text{where } k \in \mathbb{Z}^+ \end{aligned}$$

By these properties, we could divide the calculation of n-point NTT and INTT into two $\frac{n}{2}$-point NTT and INTT, similar to what we have done in the DFT case:

$$\begin{aligned} \hat{G} &= \mathbf{NTT}(G(x)) \\ \hat{G}[i] &= \sum_{j=0}^{n-1} G[j] \psi^{2ij+j} \pmod{q} \quad \text{for } i = 0, 1, \ldots, n-1 \end{aligned}$$

Split the summation into even and odd indices:

$$\begin{aligned}
\hat{G}[i] &= \sum_{j=0}^{n-1} G[j]\psi^{2ij+j} \pmod q \\
&= \sum_{j=0}^{n/2-1} \psi^{4ij+2j} G[2j] + \sum_{j=0}^{n/2-1} \psi^{4ij+2i+2j+1} G[2j+1] \pmod q \\
&= \sum_{j=0}^{n/2-1} \psi^{4ij+2j} G[2j] + \underbrace{\psi^{2i+1}}_{\text{twiddle factor}} \sum_{j=0}^{n/2-1} \psi^{4ij+2j} G[2j+1] \pmod q
\end{aligned}$$

Define:

$$A_i = \sum_{j=0}^{n/2-1} \psi^{4ij+2j} G[2j], \quad B_i = \sum_{j=0}^{n/2-1} \psi^{4ij+2j} G[2j+1]$$

We could then express the above result as:

$$\hat{G}[i] = A_i + \psi^{2i+1} B_i \pmod q$$

Next, using the symmetry property $\psi^{k+n} \equiv -\psi^k \pmod q$, we can derive:

$$\hat{G}[i+n/2] = A_i - \psi^{2i+1} B_i \pmod q$$

Thus, we obtain the fast-NTT update rules, which could be applied to each butterfly module:

$$\begin{aligned}
\hat{G}[i] &= A_i + \psi^{2i+1} B_i \pmod q \\
\hat{G}[i+n/2] &= A_i - \psi^{2i+1} B_i \pmod q
\end{aligned}$$

Since we have already assumed that $n = 2^k$, the process can be repeated for all coefficients, and this is structurally identical to the butterfly unit in FFT, replacing the complex twiddle factor $e^{-j\frac{2\pi}{N}m}$ with integer twiddle factors ψ^{2i+1} in $\mathbb{Z}_q$. [9, p. 12]

An example of the butterfly structure is shown in the following figure:

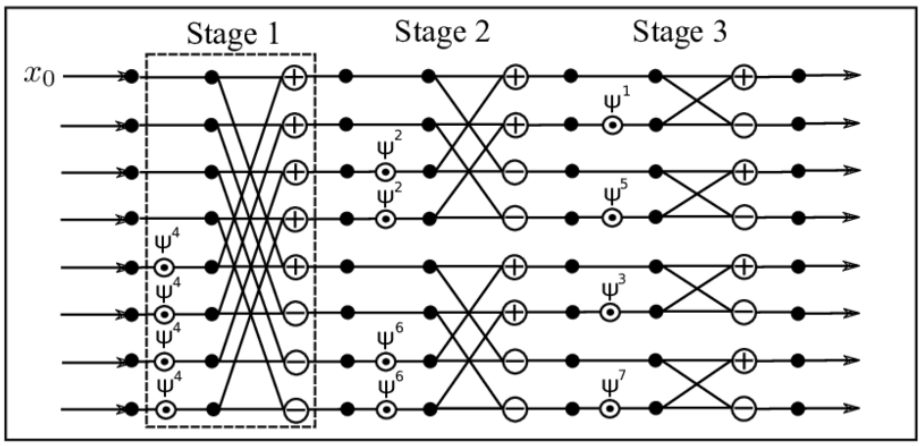

Figure 2: CT-butterfly based NTT for size $n = 8$. Source: `https://shorturl.at/YE629`

4 Applications of NTT in Lattice-Based Cryptography

Lattice-based cryptography has emerged as a leading candidate in the post-quantum cryptography landscape, due to its strong security guarantees and efficient implementation potential.

Among various techniques, the Number Theoretic Transform (NTT) plays a crucial role, particularly in accelerating polynomial multiplication within lattice-based schemes.

Unlike the Discrete Fourier Transform (DFT), which operates over the complex field, NTT works over finite fields, allowing all operations to be conducted on integers, and stay within modular arithmetic. This avoids the errors caused by floating-point rounding, and enables faster, more reliable computations.

The National Institute of Standards and Technology (NIST) has emphasized the importance of NTT in its post-quantum cryptography standardization process, particularly in schemes like CRYSTALS-Kyber and CRYSTALS-Dilithium, which both rely on polynomial multiplication modulo $x^n + 1$ over $\mathbb{Z}_q$. [1, p. 2]

Thanks to the quasi-linear time complexity $O(n \log n)$, NTT significantly outperforms traditional multiplication methods and forms the computational backbone of many signature, encryption, and key encapsulation mechanisms in standardized lattice-based cryptosystems.

By ensuring both speed and security, NTT enables practical deployment of quantum-resistant cryptographic protocols across real-world systems.

5 Limitations of NTT

Despite the significant advantages and its usage in lattice-based cryptography, NTT is not without limitations.

The efficiency of NTT relies heavily on careful parameter selection—particularly the modulus q and the size n. Additionally, unlike DFT which supports a broader range of input types and applications, NTT is restricted to integer arithmetic in finite fields, making it less versatile outside cryptographic domains.

These constraints also introduce implementation challenges, especially in resource-constrained environments or when generalizing to new cryptographic constructions.

In the following subsections, we would discuss the limitations mentioned above in detail.

5.1 Parameter Restrictions

The traditional form of NTT has two major problems in its applications:

1. Requires $2n \mid q - 1$
2. $n = 2^k,$ for some $k \in \mathbb{N}$

To relax the first restriction, some approaches are proposed, such as **preprocess-then-NTT** (Pt-NTT) and **truncated-NTT** (T-NTT), where the former is combined with the **Karatsuba technique** to reduce the number of multiplications at the cost of additional additions. [6, p. 416]

Viewing from the whole picture, research aiming to weaken the above restrictions over recent years can be categorized into three groups by their methods:

1. Method based on incomplete FFT trick.
2. Method based on splitting polynomial ring.
3. Method based on large modules.

[7, p. 11]

For the first two methods, the value range of modulus q could be expanded, however, the limitations on q still cannot be ignored since q must still be a NTT-friendly prime such that $\mathbb{Z}_q$ is a finite field and $x^n \pm 1$ can be split into polynomials of small degree over $\mathbb{Z}_q$. Moreover, the second approach leads to more modular implementation. [7, pp. 14–15]

For the last method, which is based on large modules, could be applied to any original modulus q, and can completely remove the restrictions on q. However, it has various drawbacks:

- Greater storage (for coefficients) and computing-resource consumption.
- Need to compute NTT more than once, which is more time and resource consuming, since the original full NWC-based NTT, the previous two categories to remove the restrictions, all require only one NTT computation.

[7, p. 16]

5.2 Hardware Bottlenecks and Communication Overheads

In hardware implementations of the Number Theoretic Transform (NTT), one major source of inefficiency arises from the intricate data dependencies among different NTT stages and the required division operations. [10]

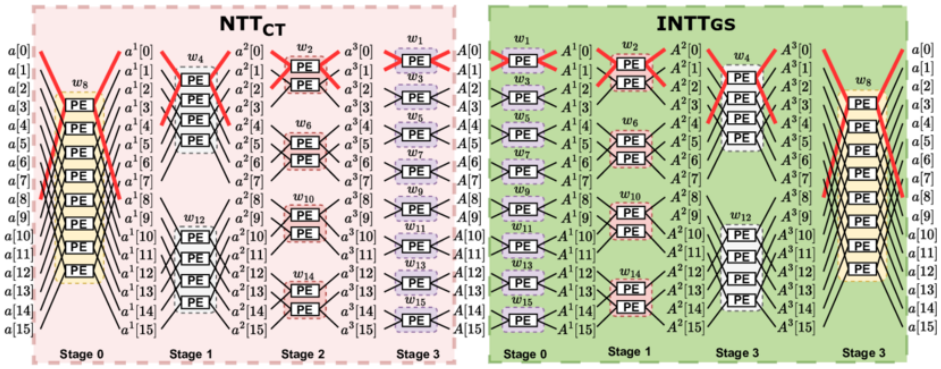

Figure 3: Butterfly Diagram of NTT and INTT using CT and GS algorithms. Source: `https://rb.gy/vdxjrn`

As visualized in the left part of the above diagram, for the Cooley-Tukey NTT algorithm, multiple stages are required to process even small-length inputs, and each stage introduces data dependencies that result in extensive data movement.

To solve the problem, ASIC/FPGA-based accelerators are proposed, however, while the performance is indeed enhanced, they still suffer from frequent data movements between processing components and memory, limitating the perfomance growth. [10]

To eliminate this data movement bottleneck, some existing in-memory computing techniques are proposed, but they would cause the following problems at the same time:

1. Security vulnerabilities, since the computing base are expanded to off-chip memories.
2. High are overhead incurred due to introducing complex pheripheral circuits.
3. Lossing generality and flexibility since the techniques are specialized for NTT.

[10]

6 Conclusion

In this report, we provided a comprehensive overview of the Number Theoretic Transform (NTT), covering its mathematical foundations, algorithmic structures, and practical relevance.

We explored the relationship between NTT and the Discrete Fourier Transform (DFT), and examined the crucial role NTT plays in enabling efficient polynomial arithmetic in lattice-based cryptographic schemes.

Despite its advantages, we also highlighted several limitations, particularly in terms of hardware constraints and algorithmic flexibility —which suggest further research and development for broader adoption.

Through this exploration, we aim to offer both an accessible introduction, and a forward-looking perspective on the continued evolution of NTT.

References

[1] Kasra Ahmadi, Saeed Aghapour, Mehran Mozaffari Kermani, and Reza Azarderakhsh. Efficient algorithm level error detection for number-theoretic transform used for kyber assessed on fpgas and arm, 2024.

[2] James Belk. Fields and cyclotomic polynomials. Lecture notes, Cornell University, n.d. Last accessed 10 June 2025.

[3] D.M. Burton. *Elementary Number Theory*. Tata McGraw-Hill Publishing Company Limited, 2006.

[4] JianJiun Ding. Ix. basic implementation techniques and fast algorithm. Lecture notes, National Taiwan University, 2025. Last accessed 10 June 2025.

[5] E.V. Flynn. Rings & arithmetic 3: Ideals and quotient rings. Lecture notes, University of Oxford, 2005. Lectures for Part A of Oxford FHS in Mathematics and Joint Schools.

[6] Zhichuang Liang, Shiyu Shen, Yuantao Shi, Dongni Sun, Chongxuan Zhang, Guoyun Zhang, Yunlei Zhao, and Zhixiang Zhao. Number theoretic transform: Generalization, optimization, concrete analysis and applications. In Yongdong Wu and Moti Yung, editors, *Information Security and Cryptology*, pages 415–432, Cham, 2021. Springer International Publishing.

[7] Zhichuang Liang and Yunlei Zhao. Number theoretic transform and its applications in lattice-based cryptosystems: A survey, 2022.

[8] W. Keith Nicholson. *Introduction to Abstract Algebra*. Wiley Publishing, 4th edition, 2012.

[9] Ardianto Satriawan, Rella Mareta, and Hanho Lee. A complete beginner guide to the number theoretic transform (ntt), 2024. Last accessed 9 June 2025.

[10] Jingyao Zhang, Mohsen Imani, and Elaheh Sadredini. Bp-ntt: Fast and compact in-sram number theoretic transform with bit-parallel modular multiplication, 2023.