A Complete Beginner Guide to the Number Theoretic Transform (NTT)

Ardianto Satriawan[1], Rella Mareta[1], Hanho Lee[1]

[1]Dept. of Electrical and Computer Engineering, Inha University, Incheon, South Korea.

Contributing authors: asatriawan@inha.ac.kr; rmareta@inha.edu; hhlee@inha.ac.kr;

Abstract

The Number Theoretic Transform (NTT) is a powerful mathematical tool that has become increasingly important in developing Post Quantum Cryptography (PQC) and Homomorphic Encryption (HE). Its ability to efficiently calculate polynomial multiplication using the convolution theorem with a quasi-linear complexity $\boldsymbol{O(n \log n)}$ instead of $\boldsymbol{O(n^2)}$ when implemented with Fast Fourier Transform-style algorithms has made it a key component in modern cryptography. FFT-style NTT algorithm or fast-NTT is particularly useful in lattice-based cryptography. In this short note, we briefly introduce the basic concepts of linear, cyclic, and negacyclic convolutions via traditional schoolbook algorithms, traditional NTT, its inverse (INTT), and FFT-like versions of NTT/INTT. We then provide consistent toy examples through different concepts and algorithms to understand the basics of NTT concepts.

Keywords: Number Theoretic Transform, Post Quantum Cryptography, Homomorphic Encryption

1 Introduction

This note provides a beginner guide for learning the concepts of the Number Theoretic Transform. One problem with learning NTT/INTT concepts was that there was no guidance or tutorial available in one place. All the resources were scattered around mathematics, engineering, computer science, and cryptography papers. From our personal experiences, those factors make it hard to learn. Also, many similar and unfamiliar terms were found when learning, making it more confusing. For instance, we did not know the difference between a positively-wrapped convolution and a negatively-wrapped one.

Motivated by the steep curve of NTT/INTT learning concepts, we wrote this note, mostly taken from my review paper about the hardware implementation of NTT [1]. In this note, we provide:

1. Introduction about the schoolbooks multiplication, convolutions, and long divisions, which are the concepts NTT/INTT optimize. This will be explained in Section 2.
2. The concepts of the NTT/INTT transformation as the Discrete Fourier Transform in the polynomial ring are provided in Section 3.
3. The optimization of the NTT/INTT transformation by using the Fast-Fourier Transformation (FFT)-style calculation: the Cooley-Tukey (CT) and the Gentleman-Sande (GS) butterflies algorithm. Those techniques are explained in Section 4.

2 Preliminaries: Schoolbook Convolutions

This section briefly explains the definition of linear, cyclic, and negacyclic convolutions between polynomials with integer coefficients to show their basic concepts and differences. We also provide simple and consistent toy examples throughout the section to clarify how different concepts work. In this section, we assume the modulus, q, is large enough so that the arithmetic calculations do not cause integer overflows.

2.1 Polynomial Multiplication and Linear Convolution

Definition 2.1. *Suppose that $G(x)$ and $H(x)$ are polynomials of degree $n-1$ in the ring $\mathbb{Z}_q[x]$ where $q \in \mathbb{Z}$ and x is the polynomial variable, a* ***polynomial multiplication*** *of $G(x)$ and $H(x)$ is defined as:*

$$Y(x) = G(x) \cdot H(x) = \sum_{k=0}^{2(n-1)} y_k x^k \tag{1}$$

where $y_k = \sum_{i=0}^{k} g_i h_{k-i} \mod q$, $\boldsymbol{g}$ and $\boldsymbol{h}$ are the polynomial coefficients of $G(x)$ and $H(x)$ respectively.

Polynomial multiplication is equivalent to a discrete **linear convolution** between the coefficients' vectors $\boldsymbol{g}$ and $\boldsymbol{h}$ [2].

$$\boldsymbol{y}[k] = (\boldsymbol{g} * \boldsymbol{h})[k] = \sum_{i=0}^{k} \boldsymbol{g}[i]\boldsymbol{h}[k-i] \tag{2}$$

Example 2.1. *Let*

$G(x) = 1 + 2x + 3x^2 + 4x^3$, *and*
$H(x) = 5 + 6x + 7x^2 + 8x^3$

or in vector notation:
$\boldsymbol{g} = [1, 2, 3, 4]$, *and*
$\boldsymbol{h} = [5, 6, 7, 8]$.

Figure 1 shows the schoolbook method of how a typical polynomial multiplication or linear convolution is done. This traditional multiplication algorithm has a $O(n^2)$ complexity.

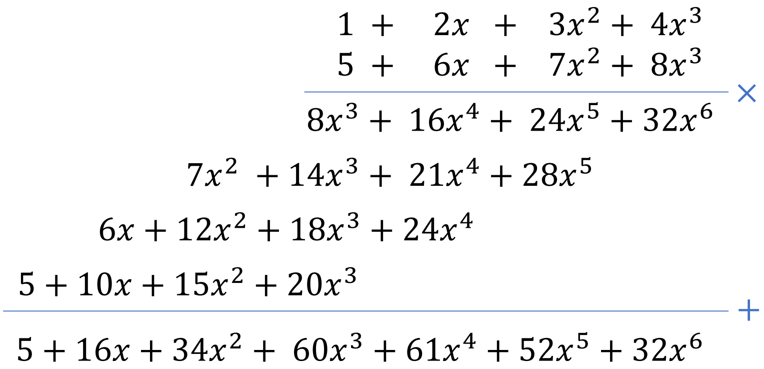

1 + 2x + 3x^2 + 4x^3
5 + 6x + 7x^2 + 8x^3
×
8x^3 + 16x^4 + 24x^5 + 32x^6
7x^2 + 14x^3 + 21x^4 + 28x^5
6x + 12x^2 + 18x^3 + 24x^4
5 + 10x + 15x^2 + 20x^3
+
5 + 16x + 34x^2 + 60x^3 + 61x^4 + 52x^5 + 32x^6

Fig. 1: Schoolbook method for polynomial multiplication or linear convolution.

Thus, the linear convolution result is $Y(x) = 5 + 16x + 34x^2 + 60x^3 + 61x^4 + 52x^5 + 32x^6$, or in vector notation, $\boldsymbol{y} = [5, 16, 34, 60, 61, 52, 32]$.

The algorithm has been implemented in many mathematical programming libraries, such as MATLAB's `conv` [3] and Numpy's `convolve` [4] with integer array inputs combined with modular arithmetic operations.

2.2 Cyclic Convolution

Definition 2.2. *Suppose that $G(x)$ and $H(x)$ are polynomials of degree $n-1$ in the quotient ring $\mathbb{Z}_q[x]/(x^n - 1)$ where $q \in \mathbb{Z}$. A* ***cyclic convolution*** *or* ***positive wrapped convolution****, $PWC(x)$ is defined as:*

$$PWC(x) = \sum_{k=0}^{n-1} c_k x^k \tag{3}$$

where $c_k = \sum_{i=0}^{k} g_i h_{k-i} + \sum_{i=k+1}^{n-1} g_i h_{k+n-i} \mod q$. *If* $Y(x)$ *is the result of their linear convolution in the ring* $\mathbb{Z}_q[x]$, *it also can be defined as*

$$\boxed{PWC(x) = Y(x) \bmod (x^n - 1)} \tag{4}$$

Traditional or schoolbook methods to calculate a cyclic convolution through a polynomial multiplication are shown in Example 2.1, followed by a long division. The method has $O(n^2)$ complexity.

Example 2.2. *Let*

$G(x) = 1 + 2x + 3x^2 + 4x^3$, *and*
$H(x) = 5 + 6x + 7x^2 + 8x^3$

or in vector notation:
$\boldsymbol{g} = [1, 2, 3, 4]$, *and*
$\boldsymbol{h} = [5, 6, 7, 8]$.

We have calculated $Y(x)$ *in Example 2.1, thus we only need to do a long division by* $x^n - 1$
Figure 2 shows how schoolbook long division is used to calculate a cyclic convolution with the dividend as the linear convolution result of $G(x)$ *and* $H(x)$. *The remainder of the long division algorithm is the cyclic convolution result.*

$$
\begin{array}{rll}
 & 32x^2 + 52x + 61 & \\ \cline{2-2}
x^4 - 1 & \big/ \; 32x^6 + 52x^5 + 61x^4 + 60x^3 + 34x^2 + 16x + 5 & \\
 & 32x^6 + 0x^5 + 0x^4 + 0x^3 - 32x^2 & \\ \cline{2-2}
 & 52x^5 + 61x^4 + 60x^3 + 66x^2 + 16x + 5 & - \\
 & 52x^5 + 0x^4 + 0x^3 + 0x^2 - 52x & \\ \cline{2-2}
 & 61x^4 + 60x^3 + 66x^2 + 68x + 5 & - \\
 & 61x^4 + 0x^3 + 0x^2 + 0x - 61 & \\ \cline{2-2}
 & 60x^3 + 66x^2 + 68x + 66 & -
\end{array}
$$

Fig. 2: Schoolbook method for positively wrapped modular polynomial multiplication or cyclic convolution.

Thus, the result of the cyclic convolution is $PWC(x) = 66 + 68x + 66x^2 + 60x^3$ *or* $[66, 68, 66, 60]$. *Notice that we present the result sorted in increasing power.*

The MATLAB function `cconv` [5] can calculate a cyclic convolution using integer array inputs and modular arithmetic operations. Notice that the result of cyclic convolution, unlike linear convolution, has a length of n instead of $2n - 1$.

2.3 Negacyclic Convolution

Definition 2.3. *Suppose that* $G(x)$ *and* $H(x)$ *are polynomials of degree* $n-1$ *in the quotient ring* $\mathbb{Z}[x]/(x^n + 1)$ *where* $q \in \mathbb{Z}$. *A* ***negacyclic convolution*** *or* ***negative wrapped convolution***, $NWC(x)$ *is defined as:*

$$NWC(x) = \sum_{k=0}^{n-1} c_k x^k \tag{5}$$

where $c_k = \sum_{i=0}^{k} g_i h_{k-i} - \sum_{i=k+1}^{n-1} g_i h_{k+n-i} \mod q$. *If* $Y(x)$ *is the result of their linear convolution in the ring* $\mathbb{Z}[x]$, *it also can be defined as*

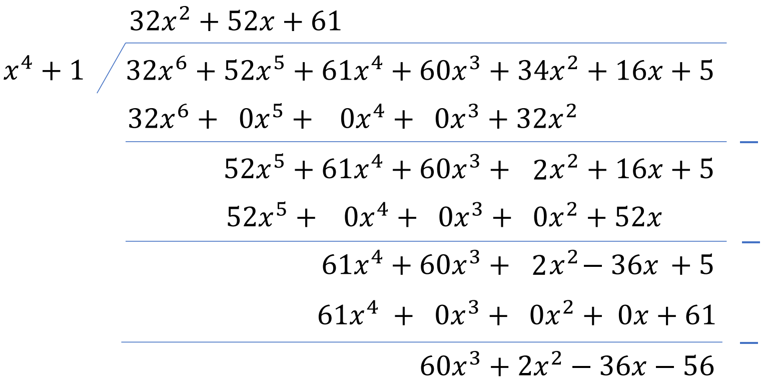
$$\boxed{NWC(x) = Y(x) \bmod (x^n + 1)} \tag{6}$$

Example 2.3. *Let*

$G(x) = 1 + 2x + 3x^2 + 4x^3$, *and*
$H(x) = 5 + 6x + 7x^2 + 8x^3$

or in vector notation:
$\boldsymbol{g} = [1, 2, 3, 4]$, *and*
$\boldsymbol{h} = [5, 6, 7, 8]$.

We have calculated $Y(x)$ in Example 2.1, thus we only need to do a long division by $x^n + 1$. Figure 3 shows how schoolbook long division calculates a negacyclic convolution, the remainder of the division.



$$
\begin{array}{r|l}
 & 32x^2 + 52x + 61 \\
x^4 + 1 & 32x^6 + 52x^5 + 61x^4 + 60x^3 + 34x^2 + 16x + 5 \\
 & 32x^6 + 0x^5 + 0x^4 + 0x^3 + 32x^2 \quad - \\
 & 52x^5 + 61x^4 + 60x^3 + 2x^2 + 16x + 5 \\
 & 52x^5 + 0x^4 + 0x^3 + 0x^2 + 52x \quad - \\
 & 61x^4 + 60x^3 + 2x^2 - 36x + 5 \\
 & 61x^4 + 0x^3 + 0x^2 + 0x + 61 \quad - \\
 & 60x^3 + 2x^2 - 36x - 56
\end{array}
$$

Fig. 3: Schoolbook method for negatively wrapped modular polynomial multiplication or negacyclic convolution.

Thus, the result of the negacyclic convolution is $NWC(x) = -56 - 36x + 2x^2 + 60x^3$ or $[-56, -36, 2, 60]$. Again, notice that we present the result sorted in increasing power.

Note that the only difference between cyclic and negacyclic convolution is the divisor. The cyclic convolution uses $x^n - 1$ while the negacyclic convolution uses $x^n + 1$. Those schoolbook algorithms have $O(n^2)$ complexity. Many efforts have been made to reduce their complexities by dividing the multiplier and multiplicand into several parts [6–9] or by parallelizing the algorithm on the implementation side [10]. However, those efforts are not scalable as the polynomial degree grows higher.

3 NTT-Based Convolutions

In this section, we present the basics of NTT-based convolutions. Many researchers do not differentiate the term NTT and FFT-based algorithms to calculate NTT, which creates confusion when understanding the topic. This report refers to the transformation itself as **NTT** and the FFT-like algorithms as **fast-NTT**, which are explained in Section 4. The classical NTT has a quadratic complexity of $O(n^2)$ when computed directly, while fast-NTT algorithms have a more efficient quasi-linear complexity $O(n \log n)$.

3.1 Primitive n-th Root of Unity

Definition 3.1. *Let $\mathbb{Z}_q$ be an integer ring modulo q, and $n - 1$ is the polynomial degree of $G(x)$ and $H(x)$. Such rings have a multiplicative identity (unity) of 1. Define ω as* ***primitive n-th root of unity** in $\mathbb{Z}_q$* *if and only if:*

$$\omega^n \equiv 1 \mod q \tag{7}$$

and

$$\omega^k \not\equiv 1 \mod q \tag{8}$$

for $k < n$.

One thing to note is that the primitive $n-th$ root of unity in a ring $\mathbb{Z}_q$ might not be unique. We show the following example for $q = 7681$, used in Kyber in Rounds 1 and 2 of the NIST-PQC Competition [11, 12], however, in our toy example we show for $n = 4$ instead of $n = 256$.

Example 3.1. *In a ring $\mathbb{Z}_{7681}$ and $n = 4$, the 4-th root of unity which satisfy the condition $\omega^4 \equiv 1 \mod 7681$ are $\{3383, 4298, 7680\}$. Out of three roots, 7680 is **not a primitive** n-th root of unity, as there exist $k = 2 < n$ that satisfy $\omega^2 \equiv 1 \mod 7681$. Therefore $\omega = 3383$ or $\omega = 4298$ are the primitive 4-th root of unity in $\mathbb{Z}_{7681}$.*

n-th root: n 次方 mod q 同餘 1
"primitive" n-th root: n 次方 mod q 同餘 1 且不存在比它小的 k 次方 mod q 也同餘 1

The value of ω will be important in calculating NTT and positive-wrapped convolution. Calculating the ω of a ring with a large number modulus q is tricky and tedious. One alternative library that provides a function to calculate ω is Sympy via the function `nthroot_mod` [13].

3.2 NTT-Based Positive-Wrapped Convolution

This section explains the definition of Number Theoretic Transform (NTT) and its inverse (INTT) based on n-th root of unity, ω. The NTT of a polynomial does not have any physical meaning, unlike Discrete Fourier Transform (DFT) which represents a signal in the frequency domain. However, NTT preserves one of the important properties of DFT: the convolution theorem, which is valuable in calculating polynomial multiplication.

3.2.1 Number Theoretic Transform Based on ω

Definition 3.2. *The Number Theoretic Transform (NTT) of a vector of polynomial coefficients $\boldsymbol{a}$ is defined as $\hat{\boldsymbol{a}} = NTT(\boldsymbol{a})$, where:*

$$\hat{\boldsymbol{a}}_j = \sum_{i=0}^{n-1} \omega^{ij} a_i \mod q \tag{9}$$

and $j = 0, 1, 2, \ldots, n-1$

Example 3.2. *Let $G(x) = 1 + 2x + 3x^2 + 4x^3$ or in vector notation $\boldsymbol{g} = [1, 2, 3, 4]$. We can infer that $n = 4$. Suppose we work in the ring $\mathbb{Z}_{7681}$ and ω is its primitive n-th root of unity. The NTT of $\boldsymbol{g}$, $\hat{\boldsymbol{g}}$, can be calculated by the following matrix multiplication:*

$$\hat{\boldsymbol{g}} = \begin{bmatrix} \omega^{0\times0} & \omega^{0\times1} & \omega^{0\times2} & \omega^{0\times3} \\ \omega^{1\times0} & \omega^{1\times1} & \omega^{1\times2} & \omega^{1\times3} \\ \omega^{2\times0} & \omega^{2\times1} & \omega^{2\times2} & \omega^{2\times3} \\ \omega^{3\times0} & \omega^{3\times1} & \omega^{3\times2} & \omega^{3\times3} \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \end{bmatrix}$$

Notice that the power of ω is the multiplication between the row and column numbers. As ω is the n-root of unity, $\boxed{\omega^k = \omega^{(k \bmod n)} \text{ for } k > n}$ Thus:

$$\hat{\boldsymbol{g}} = \begin{bmatrix} \omega^0 & \omega^0 & \omega^0 & \omega^0 \\ \omega^0 & \omega^1 & \omega^2 & \omega^3 \\ \omega^0 & \omega^2 & \omega^{\boxed{4}} & \omega^6 \\ \omega^0 & \omega^3 & \omega^6 & \omega^9 \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \end{bmatrix}$$

次方一樣取 mod n (此： mod 4)
所以 omega^4 = omega^{4 (mod 4)} = omega^0

$$\hat{\boldsymbol{g}} = \begin{bmatrix} \omega^0 & \omega^0 & \omega^0 & \omega^0 \\ \omega^0 & \omega^1 & \omega^2 & \omega^3 \\ \omega^0 & \omega^2 & \omega^{\boxed{0}} & \omega^2 \\ \omega^0 & \omega^3 & \omega^2 & \omega^1 \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \end{bmatrix}$$

From Example 3.1 we obtained one of the n-th roots of unity in $\mathbb{Z}_{7681}$ is $\omega = 3383$. Substituting into the equation:

$$\hat{\boldsymbol{g}} = \begin{bmatrix} 3383^0 & 3383^0 & 3383^0 & 3383^0 \\ 3383^0 & 3383^1 & 3383^2 & 3383^3 \\ 3383^0 & 3383^2 & 3383^0 & 3383^2 \\ 3383^0 & 3383^3 & 3383^2 & 3383^1 \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \end{bmatrix}$$

$$\hat{\boldsymbol{g}} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 3383 & 7680 & 4298 \\ 1 & 7680 & 1 & 7680 \\ 1 & 4298 & 7680 & 3383 \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \end{bmatrix}$$

$$\hat{\boldsymbol{g}} = \begin{bmatrix} 10 \\ 913 \\ 7679 \\ 6764 \end{bmatrix}$$

Therefore, the NTT$(\boldsymbol{g}) = [10, 913, 7679, 6764]$ *in* $\mathbb{Z}_{7681}$.

Example 3.3. *Let* $H(x) = 5 + 6x + 7x^2 + 8x^3$ *or in vector notation* $\boldsymbol{g} = [5, 6, 7, 8]$ *in the ring* $\mathbb{Z}_{7681}$ *and* $\omega = 3383$*. Using the same principle as Example 3.2, the NTT of* $\boldsymbol{h}$ *is:*

$$\hat{\boldsymbol{h}} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 3383 & 7680 & 4298 \\ 1 & 7680 & 1 & 7680 \\ 1 & 4298 & 7680 & 3383 \end{bmatrix} \begin{bmatrix} 5 \\ 6 \\ 7 \\ 8 \end{bmatrix} = \begin{bmatrix} 26 \\ 913 \\ 7679 \\ 6764 \end{bmatrix}$$

Therefore, the NTT$(\boldsymbol{h}) = [26, 913, 7679, 6764]$ *in* $\mathbb{Z}_{7681}$.

Note that the NTT of a particular polynomial is not always unique. It depends on the choice of ω. The NTT result of Example 3.2 and 3.3 will differ if one uses $\omega = 4298$ instead of $\omega = 3383$.

3.2.2 Inverse Number Theoretic Transform Based on ω

Definition 3.3. *The Inverse of Number Theoretic Transform (INTT) of an NTT vector* $\hat{\boldsymbol{a}}$ *is defined as* $\boldsymbol{a} = INTT(\hat{\boldsymbol{a}})$*, where:*

$$\boldsymbol{a}_i = n^{-1} \sum_{j=0}^{n-1} \omega^{-ij} \hat{a}_j \mod q \tag{10}$$

and $j = 0, 1, 2, \ldots, n-1$

Note that the INTT has a very similar formula to NTT. The only differences are ω replaced by its inverse in $\mathbb{Z}_q$ and a n^{-1} scaling factor. It always holds that $\boldsymbol{a} = \text{INTT}(\text{NTT}(\boldsymbol{a}))$.

Example 3.4. *Given NTT*$(\boldsymbol{g}) = \hat{\boldsymbol{g}} = [10, 913, 7679, 6764]$ *in* $\mathbb{Z}_{7681}$ *and* $\omega = 3383$*. We can calculate the inverse of* ω*, which is* $\omega^{-1} = 4298$ *and the scaling factor* $n^{-1} = 5761$*. One can calculate the INTT(NTT(*$\hat{\boldsymbol{g}}$*)) by the following matrix multiplication:*

$$g = n^{-1} \begin{bmatrix} \omega^{-0\times0} & \omega^{-0\times1} & \omega^{-0\times2} & \omega^{-0\times3} \\ \omega^{-1\times0} & \omega^{-1\times1} & \omega^{-1\times2} & \omega^{-1\times3} \\ \omega^{-2\times0} & \omega^{-2\times1} & \omega^{-2\times2} & \omega^{-2\times3} \\ \omega^{-3\times0} & \omega^{-3\times1} & \omega^{-3\times2} & \omega^{-3\times3} \end{bmatrix} \begin{bmatrix} 10 \\ 913 \\ 7679 \\ 6764 \end{bmatrix}$$

$$g = n^{-1}\begin{bmatrix}\omega^0 & \omega^0 & \omega^0 & \omega^0\\ \omega^0 & \omega^{-1} & \omega^{-2} & \omega^{-3}\\ \omega^0 & \omega^{-2} & \omega^{-4} & \omega^{-6}\\ \omega^0 & \omega^{-3} & \omega^{-6} & \omega^{-9}\end{bmatrix}\begin{bmatrix}10\\913\\7679\\6764\end{bmatrix}$$

$$g = n^{-1}\begin{bmatrix}\omega^0 & \omega^0 & \omega^0 & \omega^0\\ \omega^0 & \omega^{-1} & \omega^{-2} & \omega^{-3}\\ \omega^0 & \omega^{-2} & \omega^{-0} & \omega^{-2}\\ \omega^0 & \omega^{-3} & \omega^{-2} & \omega^{-1}\end{bmatrix}\begin{bmatrix}10\\913\\7679\\6764\end{bmatrix}$$

$$g = 5761\begin{bmatrix}4298^0 & 4298^0 & 4298^0 & 4298^0\\ 4298^0 & 4298^1 & 4298^2 & 4298^3\\ 4298^0 & 4298^2 & 4298^0 & 4298^2\\ 4298^0 & 4298^3 & 4298^2 & 4298^1\end{bmatrix}\begin{bmatrix}10\\913\\7679\\6764\end{bmatrix}$$

$$g = 5761\begin{bmatrix}1 & 1 & 1 & 1\\ 1 & 4298 & 7680 & 3383\\ 1 & 7680 & 1 & 7680\\ 1 & 3383 & 7680 & 4298\end{bmatrix}\begin{bmatrix}10\\913\\7679\\6764\end{bmatrix} = \begin{bmatrix}1\\2\\3\\4\end{bmatrix}$$

Therefore, the $\boldsymbol{g} = [1, 2, 3, 4]$*, which is the initial polynomial coefficients given in Example 3.2*

Example 3.5. *Given* $NTT(\boldsymbol{g}) = \hat{\boldsymbol{h}} = [26, 913, 7679, 6764]$ *in* $\mathbb{Z}_{7681}$ *and* $\omega = 3383$*. We can similarly calculate the INTT to the previous example:*

$$h = 5761\begin{bmatrix}1 & 1 & 1 & 1\\ 1 & 4298 & 7680 & 3383\\ 1 & 7680 & 1 & 7680\\ 1 & 3383 & 7680 & 4298\end{bmatrix}\begin{bmatrix}26\\913\\7679\\6764\end{bmatrix} = \begin{bmatrix}5\\6\\7\\8\end{bmatrix}$$

Therefore, the $\boldsymbol{h} = [5, 6, 7, 8]$*, which is the initial polynomial coefficients given in Example 3.3*

3.2.3 Using NTT to Calculate Positive-Wrapped Convolutions

Because NTT is a variant of DFT in the polynomial ring. One can apply DFT's convolution theorem to calculate positive-wrapped convolution [14, 15]:

Proposition 3.1. *Let* $\boldsymbol{a}$ *and* $\boldsymbol{b}$ *are the multiplicands' vectors of polynomial coefficients. The positive-wrapped convolution of* $\boldsymbol{a}$ *and* $\boldsymbol{b}$*,* $\boldsymbol{c}$ *can be calculated by:*

$$\boldsymbol{c} = INTT(NTT(\boldsymbol{a}) \circ NTT(\boldsymbol{b})) \tag{11}$$

where $\circ$ *is an element-wise vector multiplication in* $\mathbb{Z}_q$*.*

Example 3.6. *Let* $\boldsymbol{g} = [1, 2, 3, 4]$ *and* $\boldsymbol{h} = [5, 6, 7, 8]$*. From Example 3.2 and 3.3, we know that the NTT of them in in* $\mathbb{Z}_{7681}$ *are* $\hat{\boldsymbol{g}} = [10, 913, 7679, 6764]$ *and* $\hat{\boldsymbol{h}} = [10, 913, 7679, 6764]$ *when* $\omega = 3383$*. We can calculate their positive-wrapped convolution by:*

$$INTT(\begin{bmatrix}10\\913\\7679\\6764\end{bmatrix} \circ \begin{bmatrix}26\\913\\7679\\6764\end{bmatrix}) = INTT(\begin{bmatrix}260\\4021\\4\\3660\end{bmatrix}) = 5761\begin{bmatrix}1 & 1 & 1 & 1\\ 1 & 4298 & 7680 & 3383\\ 1 & 7680 & 1 & 7680\\ 1 & 3383 & 7680 & 4298\end{bmatrix}\begin{bmatrix}260\\4021\\4\\3660\end{bmatrix} = \begin{bmatrix}66\\68\\66\\60\end{bmatrix} \tag{12}$$

Therefore, their positive-wrapped convolution is $[66, 68, 66, 60]$*, the same result as calculated by schoolbook multiplication and long division in Example 2.2.*

While positive-wrapped convolution, commonly known as cyclic convolution, is useful, its implementation is primarily outside the cryptography domain. One such example is the implementation of Schönhage–Strassen algorithm [16] for large integer multiplication.

However, in the context of PQC and HE, the chosen ring is mostly $\mathbb{Z}_q[n]/(x^n+1)$ instead of $\mathbb{Z}_q[n]/(x^n-1)$. One must calculate the polynomial multiplications via the negative-wrapped convolution in such rings.

3.3 NTT-Based Negative-Wrapped Convolution

This section explains the definition of Number Theoretic Transform (NTT) and its inverse (INTT) based on $2n$-th root of unity, ψ, and how to utilize them to calculate negative-wrapped or negacyclic convolution.

3.3.1 Primitive $2n$-th Root of Unity

To calculate negative-wrapped convolution, one needs the primitive $2n$-th root of unity, ψ.

Definition 3.4. *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 its primitive n-th root of unity. Define ψ as the primitive $2n$-th root of unity if and only if:*

$$\psi^2 \equiv \omega \mod q \tag{13}$$

and

$$\psi^n \equiv -1 \mod q \tag{14}$$

Example 3.7. *In a ring $\mathbb{Z}_{7681}$ and $n = 4$, when $\omega = 3383$, the value of ψ can be 1925 or 5756 as* $1925^2 = 5756^2 \equiv 3383 \mod 7681$ *and* $1925^4 = 5756^4 = 7680 \equiv -1 \mod 7681$. *Therefore, one can choose the value of $\psi = 1925$ or $\psi = 5756$.*

3.3.2 Number Theoretic Transform Based on ψ

Definition 3.5. *The Negative-Wrapped Number Theoretic Transform (NTT^{ψ}) of a vector of polynomial coefficients $\boldsymbol{a}$ is defined as $\hat{\boldsymbol{a}} = NTT^{\psi}(\boldsymbol{a})$, where:*

$$\hat{\boldsymbol{a}}_j = \sum_{i=0}^{n-1} \psi^i \omega^{ij} a_i \mod q \tag{15}$$

and $j = 0, 1, 2, \ldots, n-1$. As $\psi^2 \equiv \omega \mod q$, we can substitute $\omega = \psi^2$ to equation (15):

$$\hat{\boldsymbol{a}}_j = \sum_{i=0}^{n-1} \psi^{2ij+i} a_i \mod q \tag{16}$$

Example 3.8. *Let $\boldsymbol{g} = [1, 2, 3, 4]$, $n = 4$ and $\psi = 1925$ in the ring $\mathbb{Z}_{7681}$. The $NTT^{\psi}(\boldsymbol{g}) = \hat{\boldsymbol{g}}$, can be calculated by the following matrix multiplication:*

$$\hat{\boldsymbol{g}} = \begin{bmatrix} \psi^{2(0\times0)+0} & \psi^{2(0\times1)+1} & \psi^{2(0\times2)+2} & \psi^{2(0\times3)+3} \\ \psi^{2(1\times0)+0} & \psi^{2(1\times1)+1} & \psi^{2(1\times2)+2} & \psi^{2(1\times3)+3} \\ \psi^{2(2\times0)+0} & \psi^{2(2\times1)+1} & \psi^{2(2\times2)+2} & \psi^{2(2\times3)+3} \\ \psi^{2(3\times0)+0} & \psi^{2(3\times1)+1} & \psi^{2(3\times2)+2} & \psi^{2(3\times3)+3} \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \end{bmatrix}$$

$$\hat{\boldsymbol{g}} = \begin{bmatrix} \psi^0 & \psi^1 & \psi^2 & \psi^3 \\ \psi^0 & \psi^3 & \psi^6 & \psi^9 \\ \psi^0 & \psi^5 & \psi^{10} & \psi^{15} \\ \psi^0 & \psi^7 & \psi^{14} & \psi^{21} \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \end{bmatrix} = \begin{bmatrix} \psi^0 & \psi^1 & \psi^2 & \psi^3 \\ \psi^0 & \psi^3 & \psi^6 & \psi^1 \\ \psi^0 & \psi^5 & \psi^2 & \psi^7 \\ \psi^0 & \psi^7 & \psi^6 & \psi^5 \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \end{bmatrix}$$

$$\hat{\boldsymbol{g}} = \begin{bmatrix} 1925^0 & 1925^1 & 1925^2 & 1925^3 \\ 1925^0 & 1925^3 & 1925^6 & 1925^1 \\ 1925^0 & 1925^5 & 1925^2 & 1925^7 \\ 1925^0 & 1925^7 & 1925^6 & 1925^5 \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \end{bmatrix}$$

$$\hat{\boldsymbol{g}} = \begin{bmatrix} 1 & 1925 & 3383 & 6468 \\ 1 & 6468 & 4298 & 1925 \\ 1 & 5756 & 3383 & 1213 \\ 1 & 1213 & 4298 & 5756 \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \end{bmatrix} = \begin{bmatrix} 1467 \\ 2807 \\ 3471 \\ 7621 \end{bmatrix}$$

Therefore, the $NTT^{\psi}(\boldsymbol{g}) = [1467, 2807, 3471, 7621]$ *when* $\psi = 1925$ *in* $\mathbb{Z}_{7681}$.

Example 3.9. *Let* $\boldsymbol{h} = [5, 6, 7, 8]$, $n = 4$ *and* $\psi = 1925$ *in the ring* $\mathbb{Z}_{7681}$. *The* $NTT^{\psi}(\boldsymbol{h}) = \hat{\boldsymbol{h}}$, *can be calculated similarly by the following matrix multiplication:*

$$\hat{\boldsymbol{h}} = \begin{bmatrix} 1 & 1925 & 3383 & 6468 \\ 1 & 6468 & 4298 & 1925 \\ 1 & 5756 & 3383 & 1213 \\ 1 & 1213 & 4298 & 5756 \end{bmatrix} \begin{bmatrix} 5 \\ 6 \\ 7 \\ 8 \end{bmatrix} = \begin{bmatrix} 2489 \\ 7489 \\ 6478 \\ 6607 \end{bmatrix}$$

Therefore, the $NTT^{\psi}(\boldsymbol{h}) = [2489, 7489, 6478, 6607]$.

3.3.3 Inverse Number Theoretic Transform Based on ψ

Definition 3.6. *The Negative-Wrapped Inverse of Number Theoretic Transform (INTT) of an NTT vector* $\hat{\boldsymbol{a}}$ *is defined as* $\boldsymbol{a} = INTT^{\psi^{-1}}(\hat{\boldsymbol{a}})$, *where:*

$$\boldsymbol{a}_i = n^{-1} \sum_{j=0}^{n-1} \psi^{-j} \omega^{-ij} \hat{a}_j \mod q \tag{17}$$

and $i = 0, 1, 2, \ldots, n-1$. *Substituting* $\omega = \psi^2$ *yields:*

$$\boldsymbol{a}_i = n^{-1} \sum_{j=0}^{n-1} \psi^{-(2ij+j)} \hat{a}_j \mod q \tag{18}$$

Note that the differences between NTT$^{\psi}$ and INTT$^{\psi}$ are the scaling factor n^{-1}, the replacement of ψ by ψ^{-1}, and the transpose of the exponents of ψ matrix.

Example 3.10. *Let* $NTT^{\psi}(\boldsymbol{g}) = \hat{\boldsymbol{g}} = [1467, 2807, 3471, 7621]$ *and* $\psi = 1925$ *in the ring* $\mathbb{Z}_{7681}$. *Note that* $\psi^{-1} = 1213$ *and* $n^{-1} = 5761$. *The vector* $\boldsymbol{g}$ *can be calculated by the following matrix multiplication:*

$$\boldsymbol{g} = n^{-1} \begin{bmatrix} \psi^{-0} & \psi^{-0} & \psi^{-0} & \psi^{-0} \\ \psi^{-1} & \psi^{-3} & \psi^{-5} & \psi^{-7} \\ \psi^{-2} & \psi^{-6} & \psi^{-10} & \psi^{-14} \\ \psi^{-3} & \psi^{-9} & \psi^{-15} & \psi^{-21} \end{bmatrix} \begin{bmatrix} 1467 \\ 2807 \\ 3471 \\ 7621 \end{bmatrix}$$

$$\boldsymbol{g} = n^{-1} \begin{bmatrix} \psi^{-0} & \psi^{-0} & \psi^{-0} & \psi^{-0} \\ \psi^{-1} & \psi^{-3} & \psi^{-5} & \psi^{-7} \\ \psi^{-2} & \psi^{-6} & \psi^{-2} & \psi^{-6} \\ \psi^{-3} & \psi^{-1} & \psi^{-7} & \psi^{-5} \end{bmatrix} \begin{bmatrix} 1467 \\ 2807 \\ 3471 \\ 7621 \end{bmatrix}$$

$$\boldsymbol{g} = 5761 \begin{bmatrix} 1213^0 & 1213^0 & 1213^0 & 1213^0 \\ 1213^1 & 1213^3 & 1213^5 & 1213^7 \\ 1213^2 & 1213^6 & 1213^2 & 1213^6 \\ 1213^3 & 1213^1 & 1213^7 & 1213^5 \end{bmatrix} \begin{bmatrix} 1467 \\ 2807 \\ 3471 \\ 7621 \end{bmatrix}$$

$$\boldsymbol{g} = 5761 \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1213 & 5756 & 6468 & 1925 \\ 4298 & 3383 & 4298 & 3383 \\ 5756 & 1213 & 1925 & 6468 \end{bmatrix} \begin{bmatrix} 1467 \\ 2807 \\ 3471 \\ 7621 \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \end{bmatrix}$$

Therefore $\boldsymbol{g} = [1, 2, 3, 4]$.

Example 3.11. *Let* $NTT^{\psi}(\boldsymbol{h}) = \hat{\boldsymbol{h}} = [2489, 7489, 6478, 6607]$ *and* $\psi = 1925$ *in the ring* $\mathbb{Z}_{7681}$*. The vector* $\boldsymbol{h}$ *can be calculated by the following matrix multiplication:*

$$\boldsymbol{h} = 5761 \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1213 & 5756 & 6468 & 1925 \\ 4298 & 3383 & 4298 & 3383 \\ 5756 & 1213 & 1925 & 6468 \end{bmatrix} \begin{bmatrix} 2489 \\ 7489 \\ 6478 \\ 6607 \end{bmatrix} = \begin{bmatrix} 5 \\ 6 \\ 7 \\ 8 \end{bmatrix}$$

Therefore, the $\boldsymbol{h} = [5, 6, 7, 8]$.

3.3.4 Using NTT$^{\psi}$ to Calculate Negative-Wrapped Convolutions

Like its positive-wrapped version, the negative-wrapped NTT can evaluate the negative-wrapped convolutions, commonly referred to as negacyclic convolutions.

Proposition 3.2. *Let* $\boldsymbol{a}$ *and* $\boldsymbol{b}$ *are the multiplicands' vectors of polynomial coefficients. The negative-wrapped convolution of* $\boldsymbol{a}$ *and* $\boldsymbol{b}$*,* $\boldsymbol{c}$ *can be calculated by:*

$$\boldsymbol{c} = INTT^{\psi^{-1}}(NTT^{\psi}(\boldsymbol{a}) \circ NTT^{\psi}(\boldsymbol{b})) \tag{19}$$

where $\circ$ *is an element-wise vector multiplication in* $\mathbb{Z}_q$*.*

Example 3.12. *Let* $\boldsymbol{g} = [1, 2, 3, 4]$ *and* $\boldsymbol{h} = [5, 6, 7, 8]$*. From Example 3.8 and 3.9, we know that the* NTT^{ψ} *of them in in* $\mathbb{Z}_{7681}$ *are* $\hat{\boldsymbol{g}} = [1467, 2807, 3471, 7621]$ *and* $\hat{\boldsymbol{h}} = [2489, 7489, 6478, 6607]$ *when* $\psi = 1925$*. We can calculate their negative-wrapped convolution by:*

$$INTT(\begin{bmatrix} 1467 \\ 2807 \\ 3471 \\ 7621 \end{bmatrix} \circ \begin{bmatrix} 2489 \\ 7489 \\ 6478 \\ 6607 \end{bmatrix}) = INTT(\begin{bmatrix} 2888 \\ 6407 \\ 2851 \\ 2992 \end{bmatrix})$$

$$= 5761 \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1213 & 5756 & 6468 & 1925 \\ 4298 & 3383 & 4298 & 3383 \\ 5756 & 1213 & 1925 & 6468 \end{bmatrix} \begin{bmatrix} 2888 \\ 6407 \\ 2851 \\ 2992 \end{bmatrix} = \begin{bmatrix} 7625 \\ 7645 \\ 2 \\ 60 \end{bmatrix}$$

Therefore, $[7625, 7645, 2, 60]$ *– or when written with negative numbers* $[-56, -36, 2, 60]$ *is their negacyclic convolution, the same result as calculated by schoolbook multiplication and long division in Example 2.3*

3.4 The Choice of Modulus

To make NTT transformation available, the modulus q has to satisfy the following requirements:

1. The n-th root of unity ω exists in ring $\mathbb{Z}_q$. The existence of ω enables one to utilize NTT to perform positive-wrapped convolutions.
2. Furthermore, the $2n$-th root of unity ψ exists in ring $\mathbb{Z}_q$ to make negative-wrapped convolutions work.

The modulus q has to satisfy the following theorem to guarantee that ω exists [14, 17, 18]:

Theorem 3.1. *If* q *is prime, then* n *must divide* $q - 1$*. If* q *is composite such that:*

$$q = {q_1}^{m_1} \cdot {q_2}^{m_2} \cdot {q_3}^{m_3} \dots {q_k}^{m_k}$$

then n must divide the greatest common divisor (GCD) of $(q_1 - 1, q_2 - 1, q_3 - 1, \ldots, q_k - 1)$.

However, while Theorem 3.1 guarantees the existence of ω does not guarantee the existence of ψ. To guarantee the existence of ψ in $\mathbb{Z}_q$:

Theorem 3.2. *If q is prime, then $2n$ must divide $q - 1$. If q is composite such that:*

$$q = {q_1}^{m_1} \cdot {q_2}^{m_2} \cdot {q_3}^{m_3} \ldots {q_k}^{m_k}$$

then $2n$ must divide the greatest common divisor (GCD) of $(q_1 - 1, q_2 - 1, q_3 - 1, \ldots, q_k - 1)$.

Many researchers proposed various moduli that might satisfy the requirements, such as Mersenne [14] and Fermat [19] prime numbers. Here we define *NTT-friendly modulus* based on its abilities to perform the type of convolutions:

Definition 3.7. *A **PWC-NTT friendly** modulus q is defined if and only if an n-th root of unity, ω, exists in $\mathbb{Z}_q$.*

Definition 3.8. *An **NWC-NTT friendly** modulus q is defined if and only if n-th root of unity, ω, and $2n$-th root of unity, ψ,exists in $\mathbb{Z}_q$.*

In the schemes proposed for the NIST-PQC competition, the values of n and q are standardized. Table 1 summarizes the schemes and their NTT-friendliness.

Table 1: Standardized values of n and q NIST-PQC scheme.

Scheme	n	q	PWC-NTT Friendly	NWC-NTT Friendly
Dilithium [20]	256	8380417	✔	✔
Falcon [21]	512	12289	✔	✔
	1024	12289	✔	✔
Kyber (Version 1 and 2) [11]	256	7681	✔	✔
Kyber (version 3) [12, 22]	256	3329	✔	✘

In the context of Post-Quantum Cryptography and Homomorphic Encryption, the terms ”Number Theoretic Transform” and ”Convolutions” usually refer to their negacyclic or negative-wrapped versions. Therefore, for the rest of the report, we refer to all the terms ”NTT,” ”INTT,” and ”convolutions” for their negative-wrapped versions.

4 Fast NTT: An Adaptation of Fast-Fourier Transform to the Number Theoretic Transform

In the previous sections, the presented NTT and INTT transformation pairs have $O(n^2)$ complexity, thus making no difference from the traditional method of negacyclic convolution. However, the NTT is the Discrete Fourier Transform in another ring. Therefore, the DFT optimization techniques can also be applied to NTT. The well-known technique of DFT optimization is called the Fast-Fourier Transform (FFT), proposed independently by Cooley-Tukey [23] and Gentleman-Sande [24]. Both using similar butterflies divide-and-conquer technique to reduce the complexity to $O(n \log n)$

To reduce the complexity and fasten the process of the matrix multiplication needed for the NTT transformation, one can use ”divide and conquer” techniques by utilizing the periodicity and symmetry property of ψ:

$$\boxed{\begin{aligned} \text{periodicity} : \psi^{k+2n} &= \psi^k \\ \text{symmetry} : \psi^{k+n} &= -\psi^k \end{aligned}} \qquad \begin{aligned} (20) \\ (21) \end{aligned}$$

Where k is a non-negative integer. The calculation of n point NTT and INTT can be divided into two $n/2$ points. However, the dividing techniques for NTT and INTT are slightly different.

4.1 Cooley-Tukey (CT) Algorithm for Fast-NTT

From equation (16), one can separate the summation into two parts based on the summation index parity:

$$\begin{aligned}\hat{\boldsymbol{a}}_j &= \sum_{i=0}^{n-1} \psi^{2ij+i} a_i \mod q \\ &= \sum_{i=0}^{n/2-1} \psi^{4ij+2i} a_{2i} + \sum_{i=0}^{n/2-1} \psi^{4ij+2j+2i+1} a_{2i+1} \mod q \\ &= \sum_{i=0}^{n/2-1} \psi^{4ij+2i} a_{2i} + \psi^{2j+1} \sum_{i=0}^{n/2-1} \psi^{4ij+2i} a_{2i+1} \mod q \end{aligned} \tag{22}$$

Based on the ψ's symmetry properties:

$$\hat{\boldsymbol{a}}_{j+n/2} = \sum_{i=0}^{n/2-1} \psi^{4ij+2i} a_{2i} - \psi^{2j+1} \sum_{i=0}^{n/2-1} \psi^{4ij+2i} a_{2i+1} \mod q \tag{23}$$

Let $A_j = \sum_{i=0}^{n/2-1} \psi^{4ij+2i} a_{2i}$ and $B_j = \sum_{i=0}^{n/2-1} \psi^{4ij+2i} a_{2i+1}$, equations (22) and (23) become:

$$\begin{aligned}\hat{\boldsymbol{a}}_j &= A_j + \psi^{2j+1} B_j \mod q \\ \hat{\boldsymbol{a}}_{j+n/2} &= A_j - \psi^{2j+1} B_j \mod q \end{aligned} \tag{24}$$

Notice that A_j and B_j can be obtained as $n/2$ points NTT. If n is power-of-two, the process can be repeated for all the coefficients. Figure 4 shows the visualization of Equation (24), usually called *CT butterfly* as a reference to its proposer, Cooley and Tukey [23].

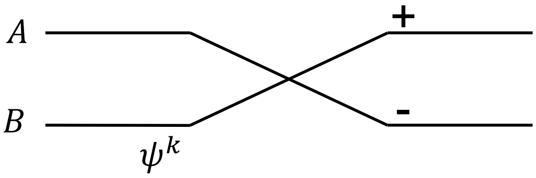


Fig. 4: Cooley-Tukey (CT) butterfly unit for calculating NTT.

One can configure several butterfly units to calculate the entire n length of NTT.

Example 4.1. *From Example 3.8, one can calculate the NTT by the matrix multiplication:*

$$\hat{\boldsymbol{g}} = \begin{bmatrix} \psi^0 & \psi^1 & \psi^2 & \psi^3 \\ \psi^0 & \psi^3 & \psi^6 & \psi^9 \\ \psi^0 & \psi^5 & \psi^{10} & \psi^{15} \\ \psi^0 & \psi^7 & \psi^{14} & \psi^{21} \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \end{bmatrix}$$

Based on the ψ periodicity:

$$\hat{\boldsymbol{g}} = \begin{bmatrix} \psi^0 & \psi^1 & \psi^2 & \psi^3 \\ \psi^0 & \psi^3 & \psi^6 & \psi^1 \\ \psi^0 & \psi^5 & \psi^2 & \psi^7 \\ \psi^0 & \psi^7 & \psi^6 & \psi^1 \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \end{bmatrix}$$

Based on the ψ symmetry:

$$\hat{\boldsymbol{g}} = \begin{bmatrix} \psi^0 & \psi^1 & \psi^2 & \psi^3 \\ \psi^0 & \psi^3 & -\psi^2 & \psi^1 \\ \psi^0 & -\psi^1 & \psi^2 & -\psi^3 \\ \psi^0 & -\psi^3 & -\psi^2 & \psi^1 \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \end{bmatrix}$$

Breaking down for each element:

$$\begin{aligned}
\hat{g}_0 &= 1\psi^0 + 2\psi^1 + 3\psi^2 + 4\psi^3\\
\hat{g}_1 &= 1\psi^0 + 2\psi^3 - 3\psi^2 + 4\psi^1\\
\hat{g}_2 &= 1\psi^0 - 2\psi^1 + 3\psi^2 - 4\psi^3\\
\hat{g}_3 &= 1\psi^0 - 2\psi^3 - 3\psi^2 + 4\psi^3
\end{aligned}$$

Factoring:

$$\begin{aligned}
\hat{g}_0 &= \psi^0(1 + 3\psi^2) + \psi^1(2 + 4\psi^2)\\
\hat{g}_1 &= \psi^0(1 - 3\psi^2) + \psi^3(2 + 4\psi^2)\\
\hat{g}_2 &= \psi^0(1 + 3\psi^2) - \psi^1(2 - 4\psi^2)\\
\hat{g}_3 &= \psi^0(1 - 3\psi^2) - \psi^3(2 - 4\psi^2)
\end{aligned} \tag{25}$$

The idea is to calculate similar terms in the brackets once and then distribute the results instead of calculating them multiple times. Figure 5 shows the visualization of Equation (25).

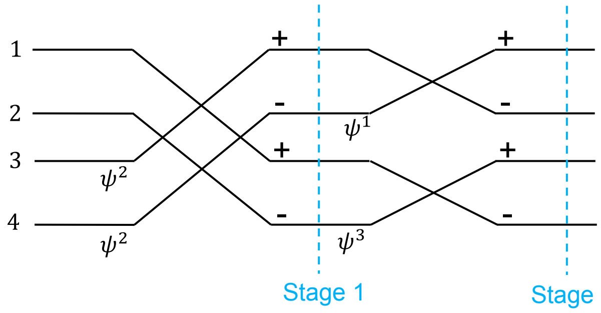


Fig. 5: Cooley-Tukey butterflies for $n = 4$ and $[1, 2, 3, 4]$ as its input.

The number of stages required is $\log_2(n)$*. For our case here, as* $n = 4$*, two stages are required. For this example, the result of stage 1 is* $[2469, 5853, 5214, 1832]$*, and stage 2 is* $[1467, 3471, 2807, 7621]$*. By reordering the result of stage 2, we can get the correct NTT result:* $[1467, 2807, 3471, 7621]$

The order of the results of CT-Butterfly is called *bit-reversed order* (BO), while the correct order of the NTT is called *normal order* (NO). We will discuss the ordering in more detail in Subsection 4.4.

Example 4.2. *Redoing Example 3.9, using the same butterfly configuration as Figure 5 with* $[5, 6, 7, 8]$ *as the input, the result of stage 1 is* $[643, 4027, 7048, 3666]$*, and stage 2 is* $[2489, 6478, 7489, 6607]$*. Reorder it to normal order for the NTT result:* $[2489, 7489, 6478, 6607]$.

However, to calculate INTT, one will need another but similar ”divide and conquer” approach.

4.2 Gentleman-Sande (GS) Algorithm for Fast-INTT

For the INTT, instead of dividing the summation by its index parity, it is separated by the lower and upper half of the summation. From equation (16) and ignoring n^{-1} term:

$$\begin{aligned}
\boldsymbol{a}_i &= \sum_{j=0}^{n-1} \psi^{-(2i+1)j}\hat{a}_j \mod q\\
&= \left[\sum_{j=0}^{\frac{n}{2}-1} \psi^{-(2i+1)j}\hat{a}_j + \sum_{j=\frac{n}{2}}^{n-1} \psi^{-(2i+1)(j+\frac{n}{2})}\hat{a}_{(j+\frac{n}{2})}\right] \mod q\\
&= \psi^{-i}\left[\sum_{j=0}^{\frac{n}{2}-1} \psi^{-2ij}\hat{a}_j + \sum_{j=0}^{\frac{n}{2}-1} \psi^{-2i(j+\frac{n}{2})}\hat{a}_{(j+\frac{n}{2})}\right] \mod q
\end{aligned}$$

Based on the periodicity and symmetry of ψ^{-1}, for the even term:

$$\boldsymbol{a}_{2i} = \psi^{-2i}\left[\sum_{j=0}^{\frac{n}{2}-1}\psi^{-4ij}\hat{a}_j + \sum_{j=0}^{\frac{n}{2}-1}\psi^{-4i(j+\frac{n}{2})}\hat{a}_{(j+\frac{n}{2})}\right] \mod q$$

$$\boldsymbol{a}_{2i} = \psi^{-2i}\sum_{j=0}^{\frac{n}{2}-1}\left[\hat{a}_j + \hat{a}_{(j+\frac{n}{2})}\right]\psi^{-4ij} \mod q \tag{26}$$

Doing the same derivation for the odd term:

$$\boldsymbol{a}_{2i+1} = \psi^{-2i}\sum_{j=0}^{\frac{n}{2}-1}\left[\hat{a}_j - \hat{a}_{(j+\frac{n}{2})}\right]\psi^{-4ij} \mod q \tag{27}$$

Let $A_i = \sum_{j=0}^{\frac{n}{2}-1}\hat{a}_j\psi^{-4ij}$ and $B_i = \sum_{j=0}^{\frac{n}{2}-1}\hat{a}_{j+\frac{n}{2}}\psi^{-4ij}$,

4.3 Gentleman-Sande (GS) Algorithm for Fast-INTT

For the INTT, instead of dividing the summation by its index parity, it is separated by the lower and upper half of the summation. From equation (16) and ignoring n^{-1} term:

$$\begin{aligned}
\boldsymbol{a}_i &= \sum_{j=0}^{n-1}\psi^{-(2i+1)j}\hat{a}_j \mod q\\
&= \left[\sum_{j=0}^{\frac{n}{2}-1}\psi^{-(2i+1)j}\hat{a}_j + \sum_{j=\frac{n}{2}}^{n-1}\psi^{-(2i+1)(j+\frac{n}{2})}\hat{a}_{(j+\frac{n}{2})}\right] \mod q\\
&= \psi^{-i}\left[\sum_{j=0}^{\frac{n}{2}-1}\psi^{-2ij}\hat{a}_j + \sum_{j=0}^{\frac{n}{2}-1}\psi^{-2i(j+\frac{n}{2})}\hat{a}_{(j+\frac{n}{2})}\right] \mod q
\end{aligned}$$

Based on the periodicity and symmetry of ψ^{-1}, for the even term:

$$\boldsymbol{a}_{2i} = \psi^{-2i}\left[\sum_{j=0}^{\frac{n}{2}-1}\psi^{-4ij}\hat{a}_j + \sum_{j=0}^{\frac{n}{2}-1}\psi^{-4i(j+\frac{n}{2})}\hat{a}_{(j+\frac{n}{2})}\right] \mod q$$

$$\boldsymbol{a}_{2i} = \psi^{-2i}\sum_{j=0}^{\frac{n}{2}-1}\left[\hat{a}_j + \hat{a}_{(j+\frac{n}{2})}\right]\psi^{-4ij} \mod q \tag{28}$$

Doing the same derivation for the odd term:

$$\boldsymbol{a}_{2i+1} = \psi^{-2i}\sum_{j=0}^{\frac{n}{2}-1}\left[\hat{a}_j - \hat{a}_{(j+\frac{n}{2})}\right]\psi^{-4ij} \mod q \tag{29}$$

Let $A_i = \sum_{j=0}^{\frac{n}{2}-1}\hat{a}_j\psi^{-4ij}$ and $B_i = \sum_{j=0}^{\frac{n}{2}-1}\hat{a}_{j+\frac{n}{2}}\psi^{-4ij}$, Equation (28) and (29) become:

$$\begin{aligned}
\boldsymbol{a}_{2i} &= (A_i + B_i)\psi^{-2i} \mod q\\
\boldsymbol{a}_{2i+1} &= (A_i - B_i)\psi^{-2i} \mod q
\end{aligned} \tag{30}$$

Notice that A_i and B_i can be obtained as $n/2$ points INTT. If n is power-of-two, the process can be repeated for all the coefficients. Figure 4 shows the visualization of Equation (32), usually called *GS butterfly* as a reference to its proposer, Gentleman and Sande [24].

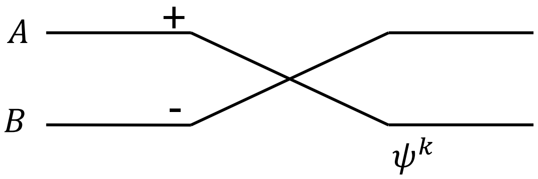


Fig. 6: Gentleman-Sande (GS) butterfly unit for calculating INTT.

Because the separation is done differently, GS butterflies' input is usually in bit-reversed order (BO) and the output is in normal order (NO).

Example 4.3. *Repeating example 3.10, let* $NTT^{\psi}(\boldsymbol{g}) = \hat{\boldsymbol{g}} = [1467, 2807, 3471, 7621]$*, the INTT can be calculated by using matrix multiplication:*

$$\boldsymbol{g} = n^{-1}\begin{bmatrix}\psi^{-0} & \psi^{-0} & \psi^{-0} & \psi^{-0}\\ \psi^{-1} & \psi^{-3} & \psi^{-5} & \psi^{-7}\\ \psi^{-2} & \psi^{-6} & \psi^{-10} & \psi^{-14}\\ \psi^{-3} & \psi^{-9} & \psi^{-15} & \psi^{-21}\end{bmatrix}\begin{bmatrix}1467\\2807\\3471\\7621\end{bmatrix}$$

Based on ψ^{-1} *periodicity:*

$$\boldsymbol{g} = n^{-1}\begin{bmatrix}\psi^{-0} & \psi^{-0} & \psi^{-0} & \psi^{-0}\\ \psi^{-1} & \psi^{-3} & \psi^{-5} & \psi^{-7}\\ \psi^{-2} & \psi^{-6} & \psi^{-2} & \psi^{-6}\\ \psi^{-3} & \psi^{-1} & \psi^{-7} & \psi^{-5}\end{bmatrix}\begin{bmatrix}1467\\2807\\3471\\7621\end{bmatrix}$$

Based on ψ^{-1} *symmetry:*

$$\boldsymbol{g} = n^{-1}\begin{bmatrix}\psi^{-0} & \psi^{-0} & \psi^{-0} & \psi^{-0}\\ \psi^{-1} & \psi^{-3} & -\psi^{-1} & -\psi^{-3}\\ \psi^{-2} & -\psi^{-2} & \psi^{-2} & -\psi^{-2}\\ \psi^{-3} & \psi^{-1} & -\psi^{-3} & -\psi^{-1}\end{bmatrix}\begin{bmatrix}1467\\2807\\3471\\7621\end{bmatrix}$$

Breaking down for each element:

$$\begin{aligned}g_0 &= [1467\psi^{-0} + 2807\psi^{-0} + 3471\psi^{-0} + 7621\psi^{-0}]n^{-1}\\ g_1 &= [1467\psi^{-1} + 2807\psi^{-3} - 3471\psi^{-1} - 7621\psi^{-3}]n^{-1}\\ g_2 &= [1467\psi^{-2} - 2807\psi^{-2} + 3471\psi^{-2} - 7621\psi^{-2}]n^{-1}\\ g_3 &= [1467\psi^{-3} + 2807\psi^{-1} - 3471\psi^{-3} - 7621\psi^{-1}]n^{-1}\end{aligned}$$

Factoring:

$$\begin{aligned}g_0 &= [(1467 + 3471)\psi^{-0} + (2807 + 7621)\psi^{-0}]\psi^{-0}n^{-1}\\ g_1 &= [(1467 - 3471)\psi^{-1} + (2807 - 7621)\psi^{-3}]\psi^{-0}n^{-1}\\ g_2 &= [(1467 + 3471)\psi^{-0} - (2807 + 7621)\psi^{-0}]\psi^{-2}n^{-1}\\ g_3 &= [(1467 - 3471)\psi^{-1} - (2807 - 7621)\psi^{-3}]\psi^{-2}n^{-1}\end{aligned} \tag{31}$$

Similar to NTT, the idea is to calculate the similar terms in the brackets once, then distribute the results instead of calculating them multiple times. By first reordering the input, we can visualize Equation 33 as shown in Figure 9.

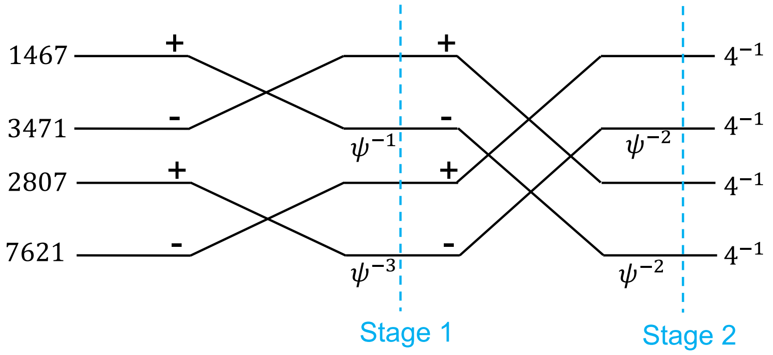


Fig. 7: Gentleman-Sande butterflies for $n = 4$ and $[1467, 2807, 3471, 7621]$ reordered as bit-reversed order as its input.

The result of stage 1 is $[4938, 4025, 2747, 3664]$*, and stage 2 is* $[4, 8, 12, 16]$*. After scaling with a* $4^{-1} = 5761$ *factor, we can get the INTT result of* $[1, 2, 3, 4]$.

Example 4.4. *Redoing Example 3.11, using the same butterfly configuration as Figure 9, reordering the input from normal order* $[2489, 7489, 6478, 6607]$ *to bit-reversed order* $[2489, 6478, 7489, 6607]$*. The result of stage 1 is* $[1286, 373, 6415, 7332]$*, the result of stage 2 is* $[20, 14, 28, 32]$*, and the INTT result after scaling is* $[5, 6, 7, 8]$.

For polynomial multiplication, one can use CT butterflies to transform both inputs to the NTT domain, then use element-wise multiplication for the NTT outputs. The result is then transformed back using GS butterflies to perform INTT. As the butterflies reduce the mathematical operation in a quasilinear scale, the complexity of the polynomial multiplication is reduced from $O(n^2)$ to $O(n \log n)$. The larger the polynomial degree, the larger the speed and cost gain [25]. Equation (28) and (29) become:

$$\begin{aligned} \boldsymbol{a}_{2i} &= (A_i + B_i)\psi^{-2i} \mod q \\ \boldsymbol{a}_{2i+1} &= (A_i - B_i)\psi^{-2i} \mod q \end{aligned} \tag{32}$$

Notice that A_i and B_i can be obtained as $n/2$ points INTT. If n is power-of-two, the process can be repeated for all the coefficients. Figure 4 shows the visualization of Equation (32), usually called *GS butterfly* as a reference to its proposer, Gentleman and Sande [24].

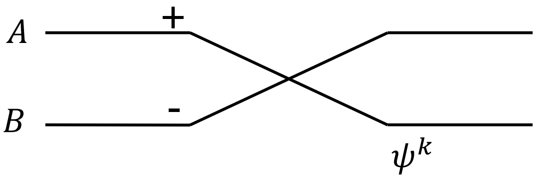


Fig. 8: Gentleman-Sande (GS) butterfly unit for calculating INTT.

Because the separation is done differently, GS butterflies' input is usually in bit-reversed order (BO) and the output is in normal order (NO).

Example 4.5. *Repeating example 3.10, let* $NTT^{\psi}(\boldsymbol{g}) = \hat{\boldsymbol{g}} = [1467, 2807, 3471, 7621]$*, the INTT can be calculated by using matrix multiplication:*

$$\boldsymbol{g} = n^{-1} \begin{bmatrix} \psi^{-0} & \psi^{-0} & \psi^{-0} & \psi^{-0} \\ \psi^{-1} & \psi^{-3} & \psi^{-5} & \psi^{-7} \\ \psi^{-2} & \psi^{-6} & \psi^{-10} & \psi^{-14} \\ \psi^{-3} & \psi^{-9} & \psi^{-15} & \psi^{-21} \end{bmatrix} \begin{bmatrix} 1467 \\ 2807 \\ 3471 \\ 7621 \end{bmatrix}$$

Based on ψ^{-1} periodicity:

$$\boldsymbol{g} = n^{-1} \begin{bmatrix} \psi^{-0} & \psi^{-0} & \psi^{-0} & \psi^{-0} \\ \psi^{-1} & \psi^{-3} & \psi^{-5} & \psi^{-7} \\ \psi^{-2} & \psi^{-6} & \psi^{-2} & \psi^{-6} \\ \psi^{-3} & \psi^{-1} & \psi^{-7} & \psi^{-5} \end{bmatrix} \begin{bmatrix} 1467 \\ 2807 \\ 3471 \\ 7621 \end{bmatrix}$$

Based on ψ^{-1} symmetry:

$$\boldsymbol{g} = n^{-1} \begin{bmatrix} \psi^{-0} & \psi^{-0} & \psi^{-0} & \psi^{-0} \\ \psi^{-1} & \psi^{-3} & -\psi^{-1} & -\psi^{-3} \\ \psi^{-2} & -\psi^{-2} & \psi^{-2} & -\psi^{-2} \\ \psi^{-3} & \psi^{-1} & -\psi^{-3} & -\psi^{-1} \end{bmatrix} \begin{bmatrix} 1467 \\ 2807 \\ 3471 \\ 7621 \end{bmatrix}$$

Breaking down for each element:

$$\begin{aligned} g_0 &= [1467\psi^{-0} + 2807\psi^{-0} + 3471\psi^{-0} + 7621\psi^{-0}]n^{-1} \\ g_1 &= [1467\psi^{-1} + 2807\psi^{-3} - 3471\psi^{-1} - 7621\psi^{-3}]n^{-1} \\ g_2 &= [1467\psi^{-2} - 2807\psi^{-2} + 3471\psi^{-2} - 7621\psi^{-2}]n^{-1} \\ g_3 &= [1467\psi^{-3} + 2807\psi^{-1} - 3471\psi^{-3} - 7621\psi^{-1}]n^{-1} \end{aligned}$$

Factoring:

$$\begin{aligned} g_0 &= [(1467 + 3471)\psi^{-0} + (2807 + 7621)\psi^{-0}]\psi^{-0}n^{-1} \\ g_1 &= [(1467 - 3471)\psi^{-1} + (2807 - 7621)\psi^{-3}]\psi^{-0}n^{-1} \\ g_2 &= [(1467 + 3471)\psi^{-0} - (2807 + 7621)\psi^{-0}]\psi^{-2}n^{-1} \\ g_3 &= [(1467 - 3471)\psi^{-1} - (2807 - 7621)\psi^{-3}]\psi^{-2}n^{-1} \end{aligned} \tag{33}$$

Similar to NTT, the idea is to calculate the similar terms in the brackets once, then distribute the results instead of calculating them multiple times. By first reordering the input, we can visualize Equation 33 as shown in Figure 9.

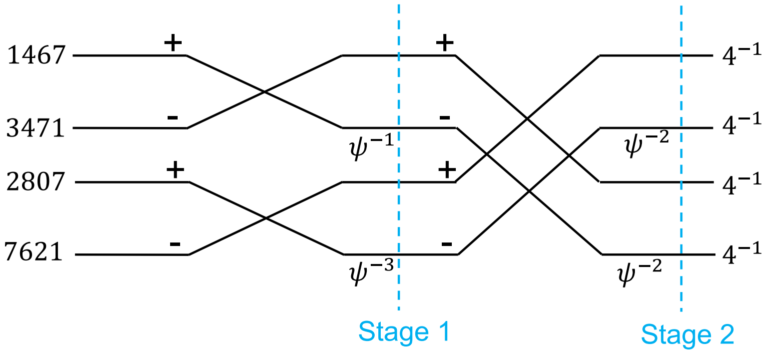


Fig. 9: Gentleman-Sande butterflies for $n = 4$ and $[1467, 2807, 3471, 7621]$ reordered as bit-reversed order as its input.

The result of stage 1 is $[4938, 4025, 2747, 3664]$, and stage 2 is $[4, 8, 12, 16]$. After scaling with a $4^{-1} = 5761$ factor, we can get the INTT result of $[1, 2, 3, 4]$.

Example 4.6. *Redoing Example 3.11, using the same butterfly configuration as Figure 9, reordering the input from normal order $[2489, 7489, 6478, 6607]$ to bit-reversed order $[2489, 6478, 7489, 6607]$. The result of stage 1 is $[1286, 373, 6415, 7332]$, the result of stage 2 is $[20, 14, 28, 32]$, and the INTT result after scaling is $[5, 6, 7, 8]$.*

For polynomial multiplication, one can use CT butterflies to transform both inputs to the NTT domain and then use element-wise multiplication for the NTT outputs. The result is then transformed back using GS butterflies to perform INTT. As the butterflies reduce the mathematical operation in a quasilinear scale, the complexity of the polynomial multiplication is reduced from $O(n^2)$ to $O(n \log n)$. The larger the polynomial degree, the larger the speed and cost gain [25].

Example 4.7. *From example 4.1, we get that the NTT transformation of* $[1, 2, 3, 4]$ *in bit-reversed order is* $[1467, 3471, 2807, 7621]$. *From example 4.2, we get the NTT transformation of* $[5, 6, 7, 8]$ *is* $[2489, 6478, 7489, 6607]$ *in bit-reversed order. Using element-wise multiplication for those two results, we get* $[2888, 2851, 6407, 2992]$ *in bit-reversed order. Transforming back the results using GS-butterfly, we will get* $[7625, 7645, 2, 60]$ *or* $[-56, -36, 2, 60]$ *when written using negative numbers. Which is the same result as Example 3.12.*

4.4 Normal Order and Bit-Reversed Order

As encountered in Subsection 4.1 and 4.3, typically, the input of CT Butterfly is in Normal Order (NO), and the output is in Bit-reversed Order (BO). Conversely, the input of GS Butterfly is in BO, and the output is in NO. This section clarifies the formal definition of Normal and Bit-reversed Order and provides examples for $n = 4$ and $n = 8$.

Definition 4.1. *Let n be a power of two, and b is a non-negative integer with $b < n$. The* ***bit-reversal*** *of b is defined as:*

$$
\begin{aligned}
&brv_n(b_{\log n-1}2^{\log n-1} + \cdots + b_1 2 + b_0)\\
&= b_0 2^{\log n-1} + \cdots + b_{\log n-2}2 + b_{\log n-1}
\end{aligned}
$$

Where b_i is the i-th bit of the binary expansion of b [26].

Example 4.8. *Consider $n = 4$, the index of the array in the normal order is* $[0, 1, 2, 3]$. *Table 2 shows the index binary representation in* $\log_2 n = 2$ *bit, their bit-reversal in binary, and their decimal representation.*

Table 2: Normal and bit-reversed order for $n = 4$

Index	Index in binary	Bit-reversal	Decimal of bit-reversal
0	00	00	0
1	01	10	2
2	10	01	1
3	11	11	3

From the table, we know that the index of the normal order is $[0, 1, 2, 3]$ *and the index of the bit-reversed order is* $[0, 2, 1, 3]$

Example 4.9. *Similarly, when considering $n = 8$, we can construct a similar table with* $\log_2 n = 3$ *as the length of binary representation. We will get the NO index is* $[0, 1, 2, 3, 4, 5, 6, 7]$, *and the BO index is* $[0, 4, 2, 6, 1, 5, 3, 7]$.

Table 3: Normal and bit-reversed order for $n = 8$

Index	Index in binary	Bit-reversal	Decimal of bit-reversal
0	000	000	0
1	001	100	4
2	010	010	2
3	011	110	6
4	100	001	1
5	101	101	5
6	110	011	3
7	111	111	7

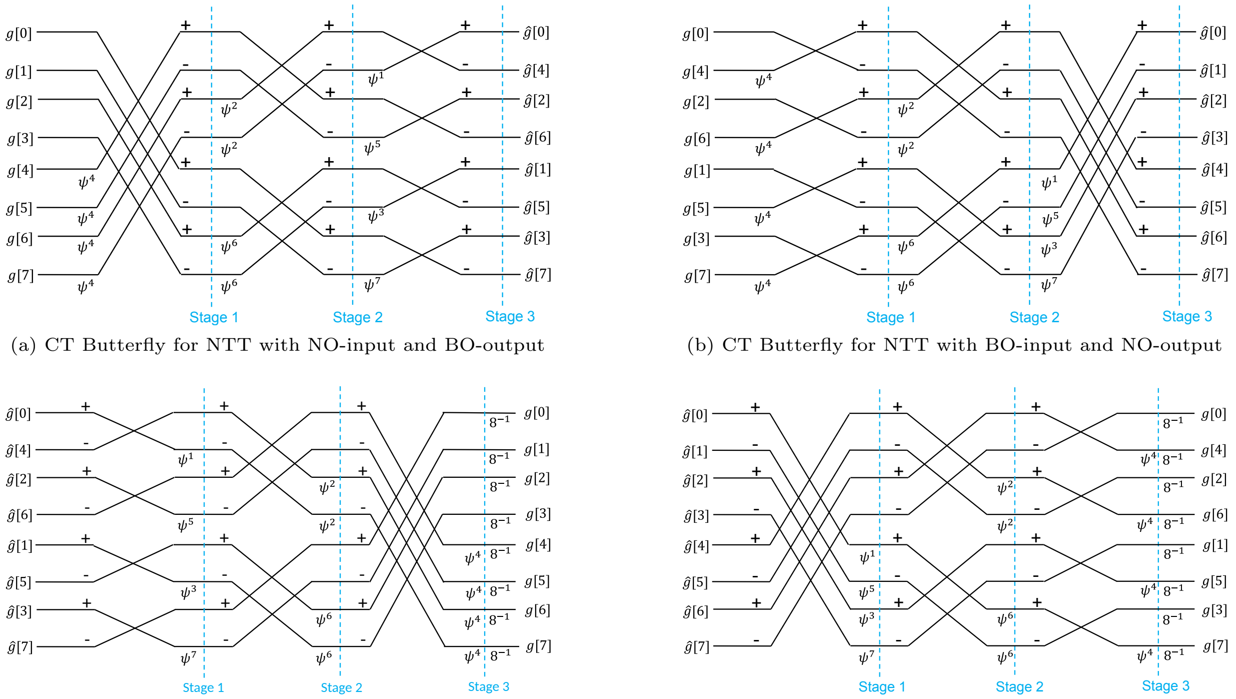
Example 4.10. *Redoing the previous examples for $n = 16$ and 4 as the length of binary representation.*

Therefore the NO index is $[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11 \quad 12, 13, 14, 15]$ *and the BO index is* $[0, 8, 4, 12, 2, 10, 6, 14, 1, 9,\ 5, 13, 3, 11, 7, 15]$.

Table 4: Normal and bit-reversed order for $n = 16$

Index	Index in binary	Bit-reversal	Decimal of bit-reversal
0	0000	0000	0
1	0001	1000	8
2	0010	0100	4
3	0011	1100	12
4	0100	0010	2
5	0101	1010	10
6	0110	0110	6
7	0111	1110	14
8	1000	0001	1
9	1001	1001	9
10	1010	0101	5
11	1011	1101	13
12	1100	0011	3
13	1101	1011	11
14	1110	0111	7
15	1111	1111	15

Typical NTT-CT Butterfly configuration has NO-input and BO-output, while INTT-GS configuration usually has BO-input and NO-output. However, one can reconfigure the CT butterfly to have BO-input & NO-output and GS butterfly to have NO-input & BO-output. Figure 10 shows all possible configurations for NTT CT and INTT GS Butterfly for $n = 8$.



(a) CT Butterfly for NTT with NO-input and BO-output

(b) CT Butterfly for NTT with BO-input and NO-output

(c) GS Butterfly for INTT with BO-input and NO-output

(d) GS Butterfly for INTT with NO-input and BO-output

Fig. 10: All possible CT and GS butterfly configurations for $n = 8$

Using normal order as NTT input is called *decimation in time*, while bit-reversed order input is called *decimation in frequency* [27].

Another thing to notice is that the power of ψ follows the bit-reversed order index. The set of all the exponentiation of ψ is called *twiddle factors*.

5 Summary

We introduced the basic concepts of linear, cyclic, and negacyclic convolutions via traditional schoolbook algorithms, traditional NTT, its inverse (INTT), and FFT-like versions of NTT/INTT. We also provided consistent toy examples through different concepts and algorithms to understand the basics of NTT.

References

[1] Satriawan, A., Syafalni, I., Mareta, R., Anshori, I., Shalannanda, W., Barra, A.: Conceptual review on number theoretic transform and comprehensive review on its implementations. IEEE Access (2023)

[2] Nussbaumer, H.J.: Elements of number theory and polynomial algebra. Fast Fourier Transform and Convolution Algorithms, 4–31 (1982)

[3] Convolution and polynomial multiplication in matlab. https://www.mathworks.com/ help/ matlab/ ref/ conv.html

[4] Numpy convolution. https://numpy.org/ doc/ stable/ reference/ generated/ numpy.convolve.html

[5] Modulo-n circular convolution in matlab. https://www.mathworks.com/ help/ signal/ref/cconv.html

[6] Karatsuba, A.A., Ofman, Y.P.: Multiplication of many-digital numbers by automatic computers. In: Doklady Akademii Nauk, vol. 145, pp. 293–294 (1962). Russian Academy of Sciences

[7] Weimerskirch, A., Paar, C.: Generalizations of the karatsuba algorithm for efficient implementations. Cryptology ePrint Archive (2006)

[8] Toom, A.L.: The complexity of a scheme of functional elements realizing the multiplication of integers. In: Soviet Mathematics Doklady, vol. 3, pp. 714–716 (1963)

[9] Cook, S.A., Aanderaa, S.O.: On the minimum computation time of functions. Transactions of the American Mathematical Society **142**, 291–314 (1969)

[10] Syafalni, I., Jonatan, G., Sutisna, N., Mulyawan, R., Adiono, T.: Efficient homomorphic encryption accelerator with integrated prng using low-cost fpga. IEEE Access **10**, 7753–7771 (2022)

[11] Avanzi, R., Bos, J., Ducas, L., Kiltz, E., Lepoint, T., Lyubashevsky, V., Schanck, J.M., Schwabe, P., Seiler, G., Stehlé, D.: Crystals-kyber algorithm specifications and supporting documentation (version 1.0). NIST Post-Quantum Cryptography Standardization Process (2017)

[12] Avanzi, R., Bos, J., Ducas, L., Kiltz, E., Lepoint, T., Lyubashevsky, V., Schanck, J.M., Schwabe, P., Seiler, G., Stehlé, D.: Crystals-kyber algorithm specifications and supporting documentation (version 2.0). NIST Post-Quantum Cryptography Standardization Process (2019)

[13] Sympy 1.11 documentation. https://docs.sympy.org/ latest/ modules/ ntheory.html#sympy.ntheory.residue_ntheory.nthroot_mod

[14] Agarwal, R.C., Burrus, C.S.: Number theoretic transforms to implement fast digital convolution. Proceedings of the IEEE **63**(4), 550–560 (1975)

[15] Nussbaumer, H.: Fast polynomial transform algorithms for digital convolution. IEEE Transactions on Acoustics, Speech, and Signal Processing **28**(2), 205–215 (1980)

[16] Schonhage, A.: Schnelle multiplikation grosser zahlen. Computing **7**, 281–292 (1971)

[17] Pollard, J.M.: The fast fourier transform in a finite field. Mathematics of computation **25**(114), 365–374 (1971)

[18] Dimitrov, V., Cooklev, T., Donevsky, B.: Generalized fermat-mersenne number theoretic transform. IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing **41**(2), 133–139 (1994)

[19] Rader, C.M.: Discrete convolutions via mersenne transrorms. IEEE Transactions on Computers **100**(12), 1269–1273 (1972)

[20] Ducas, L., Kiltz, E., Lepoint, T., Lyubashevsky, V., Schwabe, P., Seiler, G., Stehlé, D.: Crystals-dilithium: A lattice-based digital signature scheme. IACR Transactions on Cryptographic Hardware and Embedded Systems, 238–268 (2018)

[21] Fouque, P.-A., Hoffstein, J., Kirchner, P., Lyubashevsky, V., Pornin, T., Prest, T., Ricosset, T., Seiler,

G., Whyte, W., Zhang, Z., et al.: Falcon: Fast-fourier lattice-based compact signatures over ntru. Submission to the NIST's post-quantum cryptography standardization process **36**(5) (2018)

[22] Avanzi, R., Bos, J., Ducas, L., Kiltz, E., Lepoint, T., Lyubashevsky, V., Schanck, J.M., Schwabe, P., Seiler, G., Stehlé, D.: Crystals-kyber algorithm specifications and supporting documentation (version 3.0). NIST Post-Quantum Cryptography Standardization Process (2021)

[23] Cooley, J.W., Tukey, J.W.: An algorithm for the machine calculation of complex fourier series. Mathematics of computation **19**(90), 297–301 (1965)

[24] Gentleman, W.M., Sande, G.: Fast fourier transforms: for fun and profit. In: Proceedings of the November 7-10, 1966, Fall Joint Computer Conference, pp. 563–578 (1966)

[25] Heckbert, P.: Fourier transforms and the fast fourier transform (fft) algorithm. Computer Graphics **2**, 15–463 (1995)

[26] Liang, Z., Zhao, Y.: Number theoretic transform and its applications in lattice-based cryptosystems: A survey. arXiv preprint arXiv:2211.13546 (2022)

[27] Saidi, A.: Decimation-in-time-frequency fft algorithm. In: Proceedings of ICASSP'94. IEEE International Conference on Acoustics, Speech and Signal Processing, vol. 3, p. 453 (1994). IEEE