Perspectives of Convolution and NTT under Prime Power Moduli

Yimeng He[1]

Nanyang Technological University, 50 Nanyang Ave, Singapore
yimeng002@e.ntu.edu.sg

Abstract. Abstract to be done. **(To reference)**

Keywords: NTT · Convolution · Galois Ring

1 Introduction

The convolution problem We are interested in the discrete circular convolution problem where the underlying modulus is a prime power. Specifically, given 2 sequences $\boldsymbol{a} = (a_0, \dots, a_{N-1})$, $\boldsymbol{b} = (b_0, \dots, b_{N-1}) \in \mathbb{Z}_{p^m}^N$ for prime power modulus p^m and length N. We aim to efficiently compute their circular convolution product mod p^m:

$$\boldsymbol{c} = \boldsymbol{a} \circledast \boldsymbol{b} \qquad \text{where } c_i = \sum_{i=0}^{N-1} a_i b_{N-i}\, \forall 0 \le i < N$$

Alternatively, we can recast the problem into one of wrapped polynomial multiplication. Let $f(x) = a_0 + a_1 x + \ldots + a_{N-1}x^{N-1}$, $g(x) = b_0 + b_1 x + \ldots + b_{N-1}x^{N-1} \in \mathbb{Z}_{p^m}[x]$, we want to efficiently calculate the product $h(x) = f(x)g(x) \pmod{p^m, x^N - 1}$.

There is another interpretation of the problem. We define the circulant matrix $\boldsymbol{H} \in \mathbb{Z}_{p^m}^{N \times N}$, $\boldsymbol{H}_{i,j} = a_{j-i \bmod N}$. The goal is to efficiently calculate the matrix vector product $\boldsymbol{c} = \boldsymbol{H} \cdot \boldsymbol{b} \in \mathbb{Z}_{p^m}^N$.

Classical Number Theoretic Transform (NTT) In the special(and important) case where the length $N = 2^n$ and the modulus $p = k2^n + 1$ is a prime, we can find a primitive N^{th} root of unity $\omega \in \mathbb{Z}_p^\times$. The above problem can be handled by the famous radix-2 Cooley-Tuckey algorithm under $O(N \log N)$ time **(To reference Cooley-Tuckey)**.

Let us briefly recall the classical NTT strategy:

1. Find a multiplicative generator $g \in \mathbb{Z}_p^\times$
2. Raise to a power $\omega = g^{\frac{p-1}{N}}$ so that ω is a primitive N^{th} root of unity
3. Use Cooley-Tuckey radix 2 algorithm to compute the fourier transform $\mathsf{FFT}(\boldsymbol{a})_i = \sum_{j=0}^{N-1} a_j \omega^{ij}$, $\mathsf{FFT}(\boldsymbol{b})_i = \sum_{j=0}^{N-1} b_j \omega^{ij}$
4. Perform the elementwise product $\boldsymbol{c}' = \mathsf{FFT}(\boldsymbol{a}) \odot \mathsf{FFT}(\boldsymbol{b})$
5. Finally, use the radix-2 algorithm again to calculate the inverse fourier transform $\boldsymbol{c} = \mathsf{InvFFT}(\boldsymbol{c}')$, $\boldsymbol{c}_i = \frac{1}{N}\sum_{j=0}^{N-1} c'_j \omega^{-ij}$

The more general case Though elegant, classical NTT imposes significant restrictions on the modulus and the convolution length. Since NTT has found important applications in Cryptography, Coding Theory, Communication Theory etc. there is an ongoing research attempting to work around this limitation. The aim is to efficiently compute the circular convolution / NTT under more general modulus and length.

This paper considers this general question and focuses on prime power modulus p^m and arbitrary length. We are mostly interested in a very small prime p: the most interesting case is $p = 2$, since this is the default base in most modern computer architectures.

To the best of our knowledge, the most practical "generic" solution to the arbirary modulus, arbitrary length problem makes use of multimodular NTT, a combination of classical NTT with Residue Number System and Chinese Remainder Theorem. See for example **(To reference Shoup multimodular NTT)**. We refer the reader to the related work section for a brief overview of this strategy.

Our perspectives and solutions This paper approaches the convolution / NTT problem from a perspective different from the multimodular method. In that method, we must lift both arguments to *integer sequences* and solve the integer circular convolution problem. Apart from bounding the size of the final result, the initial modulus plays almost no role in the multimodular NTT algorithm.

In contrast, we propose NTT-like strategies that closely follows the modular arithmetic. For simplicity, we focus on the prime power modulus p^m, $m \geqq 1$. At a high level, our strategy is the same as the classical NTT and its numerous variants. Namely:

1. Find a "suitable" set of roots of unity
2. Pad the arguments to "suitable" lengths
3. Employ "suitable" quasi-linear time algorithms to compute the fourier transform. pointwise multiply the intermediate result and use similar quasi-linear time algorithm to compute the inverse transform

We need a number of less well-known techniques to make the strategy work in our setting. We do not claim novelty over these techniques per se, but only over their combination in order to solve the general circular convolution problem. In more detail:

- Section **(To reference Section 1)** finds and calculates a class of roots of unity compatible with both NTT application and arithmetic of the ring $\mathbb{Z}_{p^m}$. The main idea is to find these roots in the Galois ring extension $\mathsf{GR}(p^m, r)$. For each $r \geqq 1$, there is a suitable root of unity of order $p^r - 1$. We will see why such a root is appropriate and provide efficient algorithms to calculate the root.
- In **(To reference Section 2)** we briefly recall how to pad arguments so that we can transform a circular convolution problem of length N to one of a more convenient length $N' > N$.
- Efficient convolution and NTT is the main topic of **(To reference Section 3)**. We suggest 2 algorithms based on factorization characteristics of N'
  - If $N' = q^e$ is a power of a small prime q, we take inspiration from **(To reference Fast circulant vector multiplication)** and propose an $O(N \log N)$ recursion algorithm to compute the circular convolution without doing the fourier and inverse fourier transforms.
  - If $N' = N_1 N_2$ where N_1, N_2 are coprime. We employ the Good-Thomas Prime Factorization technique **(To reference Good-Thomas)** and reduce the calculation of length N' fourier transform to 2 consecutive block-wise length N_1 and length N_2 fourier transforms.
  - Finally, we combine the 2 methods above to handle those N' having factorization $N' = q_1 \ldots q_{n-1} q_n^e$, where $e \geqq 1$ and $q_1, \ldots, q_n$ are distinct small primes
- The last section contains some benchmark results of a proof-of-concept Sagemath/Python implementation of our strategy. We will also discuss the strengths and weaknesses; the time complexities; and the applications of our approach

Applications NTT has become a cornerstone in modern cryptographic applications, particularly within Post-Quantum Cryptography and Homomorphic Encryption. It plays a vital role in lattice-based cryptography schemes such as Dilithium, Falcon, and Kyber **(To reference These)**. In Homomorphic Encryption schemes like BFV, BGV, and CKKS **(To reference These)**, NTT is also critical for such operations as modulus switching and ciphertext relinearization. However as hinted before, classical NTT imposes severe restrictions on the set of parameters, particularly the existence of suitable N^{th}th or $2N^{\text{th}}$ roots of unity (for negacyclic convolution) modulo a prime q. We believe that overcoming the limitation of classical NTT will also open up the range of parameter choices in those schemes and will help us find better and more secure instantiations.

If we focus on the power of 2 modulus, where $p = 2$, our perspective might also have implications in domains such as Coding Theory and Communication. Efficient computation of (linear) circular convolutions is integral to the encoding and decoding procedures of certain codes (e.x. cyclic and negacyclic code). In addition, our result might also testify to the perspective that some operations on finite fields have useful analogues in finite ring settings.

2 Related Works

To be done