Explicit cost analysis of Toom-4 multiplication for incomplete NTT in lattice-based cryptography

Sakura Oku and Momonari Kudo

Department of Information and Communication Engineering
Fukuoka Institute of Technology
3-30-1 Wajiro-higashi, Higashi-ku, Fukuoka 811-0295, Japan
s24b2043@bene.fit.ac.jp, m-kudo@fit.ac.jp

May 12, 2026

Abstract

Polynomial multiplication is fundamental in lattice-based cryptography. While the Number Theoretic Transform (NTT) enables fast multiplication, it imposes constraints on the modulus of the coefficient field. Hafiz et al. (2025) addressed this limitation by analyzing the incomplete NTT, which combines a truncated NTT with conventional multiplication methods.

In this work, we revisit Toom-4 multiplication in the context of incomplete NTT. Although Toom-4 is asymptotically faster than Karatsuba, its precise cost has not been expressed in a form compatible with the incomplete NTT framework.

We present a concrete Toom-4 implementation and derive explicit operation counts that separate additions/subtractions and multiplications over the coefficient field. Our analysis based on addition chains yields a simple cost model for incomplete NTT.

Using this model, we analyze hybrid strategies combining Toom-4, Karatsuba, and incomplete NTT. We identify parameter ranges where Toom-4 is advantageous and validate the predicted behavior experimentally.

1 Introduction

In practical lattice-based cryptosystems such as Kyber (ML-KEM) [6, 1] and Dilithium (ML-DSA) [9, 2], computations are performed over residue rings of the form $R_q = \mathbb{F}_q[x]/\langle x^n+1\rangle$, where q is an odd prime and n is a 2-power. The full, or complete, Number Theoretic Transform (NTT) provides fast polynomial multiplication in this setting, but requires the condition $2n \mid (q-1)$. This restriction can limit the choice of parameters and affect other design goals such as security and efficiency.

To address this issue, Hafiz et al. [10] formulated and analyzed the incomplete NTT, which relaxes this constraint by combining a truncated NTT with conventional polynomial multiplication methods. Their work provides a cost model showing that such hybrid strategies can improve performance when the full, or complete, NTT is not available. In this context, it is natural to consider fast multiplication algorithms such as Karatsuba [11] and Toom–Cook [14, 7] within their incomplete NTT framework.

While the complexity of Karatsuba is well understood, the precise cost of Toom-4 – especially its constant factors – has not been expressed in a form compatible with the cost model of incomplete NTT. Although prior work, such as Bodrato and Zanoni [5], studied optimized evaluation and interpolation schemes for Toom–Cook multiplication, these analyses are not directly suitable for integration into the incomplete NTT framework of Hafiz et al. [10].

The main contribution of this paper is to bridge this gap. We present a concrete Toom-4 implementation and derive explicit operation counts separately tracking additions/subtractions and multiplications in the coefficient field. Our analysis is based on addition-chain constructions, which enable a systematic determination of constant factors. The resulting cost model can be directly integrated into the operation-count framework of incomplete NTT. Our contribution is not a new multiplication algorithm, but a precise constant-level analysis of Toom-4 multiplication tailored to the incomplete NTT framework. To the best of our knowledge, such an explicit Toom-4 cost model has not been previously integrated into the incomplete NTT framework.

Using this refined model, we analyze hybrid multiplication strategies combining Toom-4, Karatsuba, and incomplete NTT. This allows us to identify parameter ranges where Toom-4 becomes advantageous and to determine the optimal recursion depth. We also provide a Python implementation, together with verification scripts for the addition-chain counts, and experimentally confirm the predicted behavior for NTT-unfriendly prime moduli near the Dilithium modulus. The Toom-4/Karatsuba hybrid is most effective when the admissible NTT depth is small. The implementation and the scripts for verifying the addition-chain optimality are available at [13].

2 Preliminaries

This section reviews incomplete NTT and Toom–Cook multiplication used throughout the paper.

2.1 Notation

Let K be a field, and $K[x]$ the univariate polynomial ring over K. For $\phi(x) \in K[x]$, we write $\langle \phi(x) \rangle$ for the principal ideal generated by $\phi(x)$. For $f(x) \in K[x]$ and nonzero $\phi(x) \in K[x]$, the remainder of $f(x)$ modulo $\phi(x)$ is denoted by $f(x) \bmod \phi(x)$. We identify an element of $K[x]/\langle \phi(x) \rangle$ with its canonical representative of degree less than $\deg(\phi)$. For a prime power q, we denote by $\mathbb{F}_q$ the finite field of order q. When q is fixed, let A and M denote one addition/subtraction and one multiplication in $\mathbb{F}_q$, respectively.

In lattice-based cryptography, one often works over residue rings of the form $\mathbb{F}_q[x]/\langle \phi(x) \rangle$. A typical choice is the $2n$-th cyclotomic polynomial $\phi(x) = x^n + 1$ with n a 2-power, since such rings admit efficient arithmetic and fast polynomial multiplication. This setting is used in Ring-LWE and Module-LWE based cryptosystems such as Kyber (ML-KEM) [6, 1] and Dilithium (ML-DSA) [9, 2]. Hence, fast multiplication in $R_q = \mathbb{F}_q[x]/\langle x^n + 1 \rangle$ is of central practical importance.

2.2 Incomplete Number Theoretic Transform

The *Number Theoretic Transform (NTT)* is a finite-field analogue of the Fast Fourier Transform (FFT) [8, 3]; see also [12]. It is used to realize fast polynomial multiplication over residue rings, with $O(n \log_2 n)$ arithmetic operations over $\mathbb{F}_q$ for input degree less than n. Let q be an odd

prime, let $n = 2^k$, and put $R_q = \mathbb{F}_q[x]/\langle x^n + 1\rangle$. The full, or complete, NTT over R_q requires a primitive $2n$-th root of unity in $\mathbb{F}_q$, equivalently $2n \mid (q-1)$. The *incomplete NTT* [10] relaxes this condition by using only depth ℓ, where $0 \le \ell \le k$ and $2^{\ell+1} \mid (q-1)$, and then combining the transform with another conventional polynomial multiplication method. The case $\ell = k$ gives the full NTT, while $\ell = 0$ gives no transform. Moreover, incomplete NTT can be implemented non-recursively and in-place on the coefficient representation of R_q, see, e.g., [1, 10].

Assume that $2^{\ell+1} \mid (q-1)$. We choose and fix a primitive $2^{\ell+1}$-th root of unity $\zeta_{2^{\ell+1}}$ in $\mathbb{F}_q$. Then $\zeta_{2^{\ell+1}}^{2j+1}$ $(0 \le j \le 2^\ell - 1)$ are exactly the primitive $2^{\ell+1}$-th roots of unity, and we have $x^n + 1 = \prod_{j=0}^{2^\ell-1} \left(x^{n/2^\ell} - \zeta_{2^{\ell+1}}^{2j+1}\right)$. By the Chinese Remainder Theorem (CRT), we also obtain the ring isomorphism

$$R_q = \mathbb{F}_q[x]/\langle x^n + 1\rangle \cong \bigoplus_{j=0}^{2^\ell-1} \mathbb{F}_q[x]/\langle x^{n/2^\ell} - \zeta_{2^{\ell+1}}^{2j+1}\rangle,$$

where the ring isomorphism is explicitly given by

$$\mathrm{NTT}_{(\ell,k)} : f \mapsto (f \bmod x^{n/2^\ell} - \zeta_{2^{\ell+1}}^{2j+1})_{j=0}^{2^\ell-1},$$

where we identify an element of R_q with its canonical representative of degree less than n.

Then, the procedures of incomplete NTT are summarized as follows:

Outline of Incomplete NTT: Given $f, g \in R_q$ and an integer ℓ with $0 \le \ell \le k$, the following procedures compute $fg \in R_q$:

1. Compute $\boldsymbol{f} := \mathrm{NTT}_{(\ell,k)}(f)$ and $\boldsymbol{g} := \mathrm{NTT}_{(\ell,k)}(g)$, each entry of which is a polynomial of degree less than $n/2^\ell$.
2. Compute the product $\boldsymbol{h} := \boldsymbol{f} * \boldsymbol{g}$ in the codomain of $\mathrm{NTT}_{(\ell,k)}$ by entry-wise polynomial multiplications with another multiplication method.
3. Compute $\mathrm{NTT}_{(\ell,k)}^{-1}(\boldsymbol{h})$, and output it.

The maps $\mathrm{NTT}_{(\ell,k)}$ and $\mathrm{NTT}_{(\ell,k)}^{-1}$ are computed using the same radix-2 butterfly operations as in standard NTT implementations; see, e.g., [1, 10]. Each level processes n coefficients using $n/2$ multiplications by twiddle factors and n additions/subtractions, yielding a cost of $(\ell n/2)(2\mathsf{A}+\mathsf{M})$ for each transform. In the inverse transform, the divisions by 2 can be postponed and combined into a single multiplication by $2^{-\ell} \in \mathbb{F}_q$. This results in an additional $(n\,\mathbf{1}_{\ell>0})\mathsf{M}$, where $\mathbf{1}_{\ell>0} = 1$ if $\ell > 0$ and 0 otherwise. Hence we have the following theorem:

Theorem 2.2.1 (cf. [10], Observations 3 & 4)**.** *Let $C(n;\ell)$ denote the operation cost of multiplying two elements in one CRT component $\mathbb{F}_q[x]/\langle x^{n/2^\ell} - \zeta_{2^{\ell+1}}^{2j+1}\rangle$, which is independent of j in the operation-count model used here. Suppose that $2^{-\ell} \in \mathbb{F}_q$ and the powers of $\zeta_{2^{\ell+1}}$ are computed in advance. Then, the total operation cost of incomplete NTT is*

$$3(\ell n/2)(2\mathsf{A} + \mathsf{M}) + 2^\ell C(n;\ell) + (n\,\mathbf{1}_{\ell>0})\mathsf{M}. \tag{2.1}$$

In practice, incomplete NTT is used with an intermediate depth ℓ and combined with another multiplication method such as Karatsuba [11] or Toom–Cook [14, 7] for the smaller subproblems. For example, Kyber (ML-KEM) [6, 1] uses $(n, q) = (256, 3329)$ with depth $\ell = 7$, whereas Dilithium (ML-DSA) [9, 2] uses $(n, q) = (256, 8380417)$ with depth $\ell = 8$ supporting the full NTT.

2.3 Toom–Cook polynomial multiplication

We describe Toom–Cook polynomial multiplication [14, 7] with a splitting parameter s (Toom-s method). Let $s, d > 1$, let $\{\alpha_0, \alpha_1, \ldots, \alpha_{2s-2}\}$ be a set of $2s-1$ distinct points in $K \cup \{\infty\}$, and let $f, g \in K[x]$ be polynomials of degree less than d. If necessary, replace d by the smallest integer $d' \geq d$ such that $s \mid d'$, and still denote d' by d.

Splitting: Write $f = \sum_{i=0}^{s-1} f_i x^{id/s}$ and $g = \sum_{i=0}^{s-1} g_i x^{id/s}$ with $f_i, g_i \in K[x]$ and $\deg(f_i), \deg(g_i) < d/s$, and write them as $F(X) = \sum_{i=0}^{s-1} f_i X^i$ and $G(X) = \sum_{i=0}^{s-1} g_i X^i$ in $K[x][X]$ by putting $X = x^{d/s}$.

Evaluation: For each $0 \leq i \leq 2s-2$, compute $F(\alpha_i)$ and $G(\alpha_i)$, which are polynomials in x of degree less than d/s.

Recursive multiplication: Compute recursively the products $M_i = F(\alpha_i)G(\alpha_i) \in K[x]$ for $0 \leq i \leq 2s-2$.

Interpolation: Recover $H(X) \in K[x][X]$ of degree at most $2s-2$ such that $H(\alpha_i) = M_i$ for $0 \leq i \leq 2s-2$.

Re-composition: We obtain $f(x)g(x) = H(x^{d/s})$.

The case $s = 2$ yields Karatsuba multiplication [11]. When working over $\mathbb{F}_q[x]/\langle x^d - \omega \rangle$ with $\omega \in \mathbb{F}_q \setminus \{0\}$, one finally reduces modulo $x^d - \omega$. This reduction costs $(d-1)(\mathsf{A}+\mathsf{M})$, which is reduced to $(d-1)\mathsf{A}$ if $\omega = \pm 1$. It is well-known that the asymptotic complexity of Toom-s is $O(d^{\log_s(2s-1)})$. Since our target applications involve residue rings defined by 2-power cyclotomic polynomials and their hybridization with NTT, the degrees d of the resulting subproblems are also powers of 2. Therefore, Toom-4 is more compatible with recursive splitting than Toom-3, since the latter requires padding at multiple recursion levels in power-of-two settings. In this paper, we focus on precise cost estimates including constant factors, especially for the case $s = 4$ (Toom-4) and its hybridizations.

3 Main results

In this section, we first derive explicit operation counts, including constant factors, for our concrete Toom-4 implementation. We then analyze its hybridization with Karatsuba multiplication and incomplete NTT over the residue ring R_q.

3.1 Explicit Toom-4 cost model for incomplete NTT

Here, we present a concrete Toom-4 implementation together with an explicit complexity analysis including constant factors. For finite-field polynomial multiplication inside CRT components of incomplete NTT, we require operation counts compatible with the cost model of [10]. The choice of evaluation points and the evaluation/interpolation schedules significantly affects the constant factors of Toom–Cook multiplication; see, e.g., [5]. Our purpose here is not to optimize Toom–Cook implementations in full generality, but to obtain a field-operation count suitable for hybridization with incomplete NTT. For this reason, we adopt integer evaluation points, avoid fractional constants, and implement the scalar multiplications in interpolation via addition chains, postponing all divisions to a final scaling step, thereby yielding a simple and explicit operation count.

Assume that the characteristic of K is neither 2, 3, nor 5. We use the seven evaluation points $\{0, \pm 1, \pm 2, 3, \infty\}$. We also assume that the input size d is divisible by 4; otherwise, zero-padding is applied as described in Section 2.3.

Splitting: We split the input polynomials $f(x)$ and $g(x)$ in $K[x]$ of degree less than d as described in Section 2.3, say

$$\begin{aligned} f(x) &= f_0(x) + f_1(x)x^{d/4} + f_2(x)x^{2d/4} + f_3(x)x^{3d/4}, \\ g(x) &= g_0(x) + g_1(x)x^{d/4} + g_2(x)x^{2d/4} + g_3(x)x^{3d/4}, \end{aligned}$$

where f_i and g_i are polynomials of degree less than $d/4$. Put

$$F(X) := f_0 + f_1 X + f_2 X^2 + f_3 X^3, \qquad G(X) := g_0 + g_1 X + g_2 X^2 + g_3 X^3.$$

Evaluation: We compute

$$\begin{gathered} F(0) = f_0,\ F(\infty) = f_3,\ F(3) = f_0 + 3f_1 + 9f_2 + 27f_3 \\ F(\pm 1) = f_0 \pm f_1 + f_2 \pm f_3,\ F(\pm 2) = f_0 \pm 2f_1 + 4f_2 \pm 8f_3 \end{gathered}$$

and similarly for $G(0), G(\pm 1), G(\pm 2), G(3), G(\infty)$, where $F(\infty)$ and $G(\infty)$ denote the leading coefficients of $F(X)$ and $G(X)$, respectively, as polynomials in X. The values $F(\pm 1)$, $F(\pm 2)$, and $F(3)$ can be computed as follows:

1. $a_1 := f_0 + f_2$; $a_2 := f_1 + f_3$; $a_{3,\pm} := a_1 \pm a_2$ $(= F(\pm 1))$;
2. $a_4 := f_3 + f_3$ $(= 2f_3)$; $a_5 := a_4 + f_3$ $(= 3f_3)$;
3. $a_{6,\pm} := a_{3,\pm} \pm a_5$ $(= f_0 \pm f_1 + f_2 \pm 4f_3)$;
4. $a_{7,\pm} := a_{6,\pm} + f_2$ $(= f_0 \pm f_1 + 2f_2 \pm 4f_3)$;
5. $a_{8,\pm} := 2a_{7,\pm}$ $(= 2f_0 \pm 2f_1 + 4f_2 \pm 8f_3)$;
6. $a_{9,\pm} := a_{8,\pm} - f_0$ $(= F(\pm 2))$;
7. $a_{10} := 2a_5$ $(= 6f_3)$;
8. $a_{11} := a_{9,+} - f_0 + a_{10}$ $(= 2f_1 + 4f_2 + 14f_3)$;
9. $a_{12} := 2a_{11}$ $(= 4f_1 + 8f_2 + 28f_3)$;
10. $a_{13} := a_{12} + a_{3,-}$ $(= F(3))$.

Applying the same procedure to G, the evaluation step requires $19 \times 2 = 38$ additions/subtractions of polynomials of degree less than $d/4$, that is, $38(d/4)$ additions/subtractions in K.

Recursive multiplication: Compute $M_\alpha := F(\alpha)G(\alpha)$ for $\alpha \in \{0, \pm 1, \pm 2, 3, \infty\}$, where $H(X) := F(X)G(X) \in K[x][X]$.

Interpolation: Writing $H(X) = \sum_{i=0}^{6} h_i X^i$ for $h_i \in K[x]$ with $\deg(h_i) < 2d/4$, the conditions $H(\alpha) = M_\alpha$ for $\alpha \in \{0, \pm 1, \pm 2, 3\}$ together with $H(\infty) = h_6 = M_\infty$ yield a linear system in $h_0, \ldots, h_6$. Its coefficient matrix is a slightly modified Vandermonde matrix, whose rows

correspond to the evaluations at $\alpha = 0, 1, -1, 2, -2, 3, \infty$ in this order. The inverse matrix of the coefficient matrix is

$$\frac{1}{120}\begin{pmatrix} 120 & 0 & 0 & 0 & 0 & 0 & 0 \\ -40 & 120 & -60 & -30 & 6 & 4 & -1440 \\ -150 & 80 & 80 & -5 & -5 & 0 & 480 \\ 50 & -70 & -5 & 35 & -5 & -5 & 1800 \\ 30 & -20 & -20 & 5 & 5 & 0 & -600 \\ -10 & 10 & 5 & -5 & -1 & 1 & -360 \\ 0 & 0 & 0 & 0 & 0 & 0 & 120 \end{pmatrix}$$

Using this inverse matrix, each h_i can be expressed as a linear combination of the values $M_\alpha = H(\alpha)$. For example,

$$h_1 = 120^{-1}\big(-40H(0) + 120H(1) - 60H(-1) - 30H(2) + 6H(-2) + 4H(3) - 1440H(\infty)\big),$$

and $h_2, \ldots, h_5$ are obtained similarly.

Hence the nontrivial interpolation coefficients have denominator 120, so we work with the scaled quantities $h'_i := 120h_i$ throughout the recursion. Thus, after L recursive Toom-4 levels, the output is scaled by 120^L and recovered by a single final multiplication by the precomputed inverse $120^{-L} \in K$. The recursion is terminated at a suitable base case, where Karatsuba or schoolbook multiplication is used.

For this interpolation step, we first compute several scalar multiples of each $H(\alpha)$ appearing in the above inverse matrix. Thus, rather than generating a single scalar multiple, we consider addition chains for a prescribed set of target multiples. For a finite set S of positive integers, an *addition chain for* S is a sequence $1 = b_0, b_1, \ldots, b_r$ such that each b_m with $1 \le m \le r$ is of the form $b_i + b_j$ for some $0 \le i, j < m$, and $S \subset \{b_0, \ldots, b_r\}$. Using such an addition chain, the required scalar multiples can be computed with r additions.

The following chains minimize the number of additions needed to generate all required scalar multiples within the prescribed target sets. The optimality of these chains for the prescribed target sets was verified by exhaustive enumeration up to the relevant lengths, together with elementary lower-bound arguments for the remaining large targets. The verification scripts are included in our public repository [13].

$$\begin{aligned} H(0) &: 1,\ 2,\ 4,\ 8,\ \mathbf{10},\ 20,\ \mathbf{30},\ \mathbf{40},\ \mathbf{50},\ 100,\ \mathbf{120},\ \mathbf{150}; \\ H(1) &: 1,\ 2,\ 4,\ 8,\ \mathbf{10},\ \mathbf{20},\ 30,\ 40,\ \mathbf{70},\ \mathbf{80},\ \mathbf{120}; \\ H(-1) &: 1,\ 2,\ 4,\ \mathbf{5},\ 10,\ \mathbf{20},\ 40,\ \mathbf{60},\ \mathbf{80}; \\ H(2) &: 1,\ 2,\ 4,\ \mathbf{5},\ 10,\ 20,\ \mathbf{30},\ \mathbf{35}; \\ H(-2) &: 1,\ 2,\ 4,\ \mathbf{5},\ \mathbf{6}; \qquad H(3) : 1,\ 2,\ \mathbf{4},\ \mathbf{5}; \\ H(\infty) &: 1,\ 2,\ 4,\ 8,\ 10,\ 20,\ 40,\ 80,\ \mathbf{120},\ 240,\ \mathbf{360},\ \mathbf{480}, \\ &\quad \mathbf{600},\ 1080,\ \mathbf{1440},\ \mathbf{1800}. \end{aligned}$$

The boldface entries indicate the scalar multiples appearing in the above inverse matrix. Thus, all required scalar multiples are obtained with $11 + 10 + 8 + 7 + 4 + 3 + 15 = 58$ additions of polynomials of degree less than $2d/4$, namely $58(2d/4)$ additions in K. This explicit control of scalar multiplications is crucial in the incomplete NTT setting, where constant factors directly affect the optimal hybrid strategy.

Once these scalar multiples are prepared, the scaled coefficients h'_i are recovered by additions/subtractions. For instance, the scaled coefficient h'_1 is obtained with 6 additions/subtractions once the relevant scalar multiples have been computed. Similarly, h'_2, h'_3, h'_4, and h'_5 are obtained with 5, 6, 5, and 6 additions/subtractions, respectively. Hence the polynomials $h'_1, \ldots, h'_5$ are recovered with 28 additions/subtractions of polynomials of degree less than $2d/4$, that is, $28(2d/4)$ additions/subtractions in K.

Re-composition: At the scaled level, recovering $120H(x^{d/4})$ from $h'_0, \ldots, h'_6$ requires 6 additions on overlapping blocks of size $d/4$, namely $6(d/4)$ additions in K. Indeed, since each h'_i has degree less than $2d/4$, the supports of h'_0, $h'_1 x^{d/4}$, $\ldots$, $h'_6 x^{6d/4}$ overlap only with adjacent blocks. Hence the recomposition can be carried out with 6 additions on overlapping blocks of size $d/4$.

Collecting the above counts, one Toom-4 level requires $38(d/4) + (58+28)(2d/4) + 6(d/4) = 54d$ additions in K, where the three terms correspond to evaluation, interpolation, and recomposition, respectively. The final rescaling by the precomputed inverse of the accumulated factor to the output polynomial requires $2d-1$ further multiplications in K.

Proposition 1. *Let $T_{\mathsf{A}}(d)$ and $T_{\mathsf{M}}(d)$ denote the numbers of additions/subtractions and multiplications in K required by the recursive part of the proposed Toom-4 implementation on inputs of size d, excluding the final rescaling. Then we have $T_{\mathsf{A}}(d) = 7\,T_{\mathsf{A}}(d/4) + 54d$ and $T_{\mathsf{M}}(d) = 7\,T_{\mathsf{M}}(d/4)$. In addition, the final rescaling step requires $2d-1$ further multiplications in K.*

The claim follows directly from the above construction by counting the number of arithmetic operations in each step.

Remark 3.1.1. Prior work on efficient Toom–Cook multiplication [5, 4] studied optimized evaluation and interpolation schemes, including fractional evaluation points such as $\pm 1/2$. In contrast, we restrict to integer evaluation points and postpone all divisions to a final scaling step, which simplifies the operation-count model over $\mathbb{F}_q$ and facilitates integration into the incomplete NTT framework. We also tested fractional evaluation points; after clearing denominators and applying the same addition-chain optimization, they required more additions/subtractions than the present choice. Thus, the set $\{0, \pm 1, \pm 2, 3, \infty\}$ of evaluation points gives a simpler and cheaper interpolation schedule in our setting.

3.2 Hybrid multiplication over a residue ring

From now on, let q be an odd prime and set $K = \mathbb{F}_q$. As in Section 2.2, let $n = 2^k$, assume that $2^{\ell+1} \mid (q-1)$, and put $R_q = \mathbb{F}_q[x]/\langle x^n + 1\rangle$. In this subsection, we instantiate the abstract cost function $C(n;\ell)$ in Theorem 2.2.1 by a concrete hybrid multiplication method over R_q. Recall that n is a power of 2 in our setting (Section 2.2). Throughout this subsection, let d denote the subproblem size; in particular, $d = n/2^\ell$ is also a power of 2.

Toom-4/Karatsuba hybrid. We apply Toom-4 recursively for L levels with Karatsuba multiplication as the base case. Let $T_{\mathsf{M}}(d;L)$ and $T_{\mathsf{A}}(d;L)$ denote the numbers of multiplications and additions/subtractions in $\mathbb{F}_q$ for multiplying two polynomials of degree less than d; the cost of the reduction modulo $x^d - \omega$ is counted separately. By Proposition 1 and summing the

geometric series of the $54d$-terms, we have

$$T_{\mathsf{M}}(d;L) = 7^L\, T_{\mathsf{M}}^{\mathrm{Kar}}(d/4^L) + (2d-1)\mathbf{1}_{L>0},$$
$$T_{\mathsf{A}}(d;L) = 7^L\, T_{\mathsf{A}}^{\mathrm{Kar}}(d/4^L) + 72d\big((7/4)^L - 1\big),$$

where $\mathbf{1}_{L>0} = 1$ if $L > 0$ and 0 otherwise. Here $T_{\mathsf{M}}^{\mathrm{Kar}}(d')$ and $T_{\mathsf{A}}^{\mathrm{Kar}}(d')$ denote the operation counts of Karatsuba multiplication for input size d', estimated by $(d')^{\log_2 3}$ and $8((d')^{\log_2 3} - d')$ respectively. These counts follow from the standard recursive operation count for Karatsuba multiplication; see, e.g., the proof of Theorem 8.3 in [15]. Consequently we obtain

$$T_{\mathsf{M}}(d;L) = 7^L (d/4^L)^{\log_2 3} + (2d-1)\mathbf{1}_{L>0},$$
$$T_{\mathsf{A}}(d;L) = 7^L \cdot 8\big((d/4^L)^{\log_2 3} - d/4^L\big) + 72d\big((7/4)^L - 1\big).$$

Incomplete NTT with Toom-4/Karatsuba hybrid. As in Subsection 2.2, let ℓ be the NTT depth. Following [10], let α and μ denote the costs of one addition/subtraction and one multiplication in $\mathbb{F}_q$, respectively. We normalize by μ and put $w := \alpha/\mu$. For one subproblem of size d, define

$$C_w(d;L) := T_{\mathsf{M}}(d;L) + w\, T_{\mathsf{A}}(d;L). \tag{3.1}$$

Substituting this into the cost formula (2.1) of Theorem 2.2.1 with $d = n/2^\ell$, we obtain the following weighted cost model:

$$C_w(n;\ell,L) = 2^\ell C_w(n/2^\ell;L) + (3\ell n/2)(1+2w) + (2n - 2^\ell)\mathbf{1}_{\ell>0} + w(n-2^\ell), \tag{3.2}$$

where the last two terms account for the final scaling and the reduction modulo $x^{n/2^\ell} - \zeta_{2^{\ell+1}}^{2j+1}$ in each CRT component.

For a fixed triple (n,ℓ,w), only the first term in (3.2) depends on L. Therefore, for each pair (n,ℓ), the optimal recursion depth is obtained by minimizing (3.1) with $d = n/2^\ell$.

3.3 Parameter optimization and comparison

We evaluate the cost model in Subsection 3.2. To highlight the effect of multiplication algorithms, we fix $w = 0.2$ (i.e., $\mu/\alpha = 5$ as in [10]), which reflects a typical setting where additions/subtractions are cheaper than multiplications.

For $w = 0.2$, Toom-4 becomes effective only for sufficiently large subproblems: Minimizing $C_w(d;L)$ in (3.1) over L shows that the optimal recursion depth is $L = 0$ for $d \le 128$, $L = 1$ for $d = 256$, and $L = 2$ for $d = 512, 1024$. In particular, for $d \le 128$, the hybrid method reduces to pure Karatsuba. Since Karatsuba has a lower cost than schoolbook multiplication in our model for all parameter sizes considered here, we omit schoolbook from the comparison.

We compare three multiplication strategies under the same NTT depth ℓ: Karatsuba ($L = 0$), pure Toom-4, and the proposed Toom-4/Karatsuba hybrid. Here, "pure Toom-4" means that Toom-4 is applied as long as possible and Karatsuba multiplication is used at the final base level. The maximal recursion depth is denoted by $L_{\max}$, while L_{opt} denotes the value minimizing $C_w(d;L)$ in (3.1). The results for $n \in \{256, 512, 1024\}$ are shown in Table 1.

Table 1 shows that, for fixed ℓ, the proposed hybrid method consistently improves on pure Toom-4 and, for sufficiently small ℓ, also improves on Karatsuba. This suggests that, beyond previous analyses focusing mainly on schoolbook or Karatsuba multiplication, Toom-4 can provide additional gains in the limited-depth regime. As ℓ increases, the subproblem size $d = n/2^\ell$

Table 1: Weighted cost $C_w(n;\ell,L)$ with $w = 0.2$ and $n \in \{256, 512, 1024\}$ for multiplication over $\mathbb{F}_q[x]/\langle x^n + 1\rangle$. Here ℓ is the NTT depth, $d = n/2^\ell$ is the subproblem size, and $w = 0.2$ is the relative cost of additions/subtractions. The columns correspond to Karatsuba ($L = 0$), pure Toom-4 ($L = L_{\max}$), and the hybrid method ($L = L_{\text{opt}}$). Costs are measured in units of one multiplication in $\mathbb{F}_q$.

n	ℓ	d	Karatsuba	Toom-4	Hybrid
256	0	256	16700.0	33851.0	15877.8 ($L_{\text{opt}} = 1$)
	1	128	12061.2	20834.4	12061.2 ($L_{\text{opt}} = 0$)
512	0	512	50458.8	73945.8	44781.0 ($L_{\text{opt}} = 2$)
	1	256	35497.2	69799.2	33852.8 ($L_{\text{opt}} = 1$)
	2	128	25197.6	42744.0	25197.6 ($L_{\text{opt}} = 0$)
1024	0	1024	152093.6	246333.8	120521.4 ($L_{\text{opt}} = 2$)
	1	512	105114.0	152088.0	93758.4 ($L_{\text{opt}} = 2$)
	2	256	73144.8	141748.8	69856.0 ($L_{\text{opt}} = 1$)
	3	128	52545.6	87638.4	52545.6 ($L_{\text{opt}} = 0$)

becomes smaller, and the optimal choice eventually becomes $L = 0$, so that the hybrid method coincides with Karatsuba.

Thus, for parameter sets with large admissible NTT depths, such as the recommended parameters of Kyber, the improvement by Toom-4 is limited. In contrast, when the admissible depth is restricted by the 2-adic valuation of $q - 1$, the proposed hybrid yields a noticeable reduction in cost.

4 Experimental validation

We provide an experimental validation of the proposed hybrid method. The goal is not to obtain precise quantitative agreement with the theoretical cost model, but to confirm the qualitative behavior predicted in Section 3.3. All experiments were conducted on a standard laptop environment (Python 3, single-threaded); see [13] for details. We implemented incomplete NTT together with Karatsuba, pure Toom-4, and the proposed hybrid method. Polynomial multiplication is performed over $\mathbb{F}_q[x]/\langle x^n + 1\rangle$ with $n \in \{256, 512, 1024\}$. To make the limited-depth regime visible, we choose primes q of comparable size but with different 2-adic valuations of $q - 1$, which directly control the maximal admissible depth of incomplete NTT. For a prime q, the maximal admissible incomplete NTT depth is $\ell_{\max} = \min\{v_2(q-1) - 1,\ \log_2 n\}$, where $v_2(q-1)$ denotes the 2-adic valuation of $q - 1$. While the case of Kyber was experimentally studied in [10], we choose NTT-unfriendly prime moduli (primes for which the admissible NTT depth is limited) of comparable size near the Dilithium modulus: $q = 8380403, 8380381, 8380249, 8380369$, for which $\ell_{\max} = 0, 1, 2, 3$, respectively. For each (n, q), we apply incomplete NTT with depth $\ell = \ell_{\max}$, and report the median running time over 100 random inputs.

Table 2 reports the running times. For each instance, the hybrid method uses the optimal recursion depth L_{opt} for the corresponding subproblem size. Note that when $L_{\text{opt}} = 0$, the hybrid method reduces to Karatsuba; this explains the identical entries in the corresponding rows. The results are consistent with the theoretical prediction. When the admissible NTT depth is very limited (e.g., $\ell_{\max} = 0$), the hybrid method significantly outperforms Karatsuba.

Table 2: Running time (ms) for multiplication over $\mathbb{F}_q[x]/\langle x^n+1\rangle$. Here $\ell_{\max} = \min\{v_2(q-1)-1,\ \log_2 n\}$ is the maximal admissible NTT depth, and $d = n/2^{\ell_{\max}}$ is the subproblem size. For incomplete NTT with depth $\ell = \ell_{\max}$, we compare Karatsuba ($L = 0$), pure Toom-4, and the hybrid method ($L = L_{\mathrm{opt}}$). Each value is the median over 100 random inputs.

n	q	$\ell_{\max}$	d	Karatsuba	Toom-4	Hybrid
256	8380403	0	256	16.722	32.873	14.182
	8380381	1	128	11.130	15.405	11.130
512	8380403	0	512	50.738	57.431	36.901
	8380381	1	256	33.975	66.562	28.998
	8380249	2	128	22.812	31.415	22.812
1024	8380403	0	1024	152.838	237.411	104.767
	8380381	1	512	101.967	114.807	74.587
	8380249	2	256	68.567	133.386	58.385
	8380369	3	128	45.749	63.358	45.749

As the admissible depth increases, the advantage gradually diminishes, and eventually disappears when the subproblem size becomes too small. This supports the conclusion that the proposed method is mainly beneficial in the limited-depth regime of incomplete NTT.

The implementation used for the experiments is publicly available at [13].

5 Conclusion and future work

In this paper, we presented a detailed complexity analysis of Toom-4 polynomial multiplication including constant factors, and integrated it into the incomplete NTT framework over $R_q = \mathbb{F}_q[x]/\langle x^n+1\rangle$. Based on this analysis, we derived explicit cost formulas and identified the optimal recursion depth in a weighted cost model. Our current implementation is in Python; however, the proposed cost model is independent of implementation details and is intended to guide optimized low-level implementations.

Our results show that Toom-4 provides additional gains mainly in the limited-depth regime of incomplete NTT, where the admissible depth is restricted by the 2-adic valuation of $q-1$. In particular, the hybrid method is most effective when the transform depth is minimal, highlighting a practically relevant regime where Toom-4 becomes advantageous.

Future work includes optimized low-level implementations (e.g., in C or C++) and further evaluation in concrete lattice-based cryptographic schemes.

Acknowledgment

The authors thank Masaya Yasuda for helpful comments. Preliminary results of this work were presented at the poster session of the 2026 Symposium on Cryptography and Information Security (SCIS 2026), a Japanese conference on cryptography and information security, where the authors received valuable comments and suggestions, particularly on hybridization with NTT. This work was partially supported by Comprehensive Research Organization, Fukuoka Institute of Technology. This work was also partially supported by JST CREST Grant Number JPMJCR2113 and JSPS KAKENHI Grant Number 25K00140.

References

[1] National Institute of Standards and Technology: "Module-Lattice-Based Key-Encapsulation Mechanism Standard", FIPS 203, 2024.

[2] National Institute of Standards and Technology: "Module-Lattice-Based Digital Signature Standard", FIPS 204, 2024.

[3] R. Agarwal and C. Burrus: "Fast Convolution using fermat number transforms with applications to digital filtering", In: IEEE Transactions on Acoustics, Speech, and Signal Processing, **22**, no. 2, pp. 87–97, 1974.

[4] M. Bodrato, "Towards optimal Toom–Cook multiplication for univariate and multivariate polynomials in characteristic 2 and 0", In: Proc. WAIFI 2007, LNCS 4547, pp. 116–133, 2007.

[5] M. Bodrato and A. Zanoni, "Integer and polynomial multiplication: towards optimal Toom–Cook matrices", In: International Symposium on Symbolic and Algebraic Computation, ISSAC 2007, pp. 17–24, 2007.

[6] J. Bos et al.: "CRYSTALS-Kyber: A CCA-Secure Module-Lattice-Based KEM", 2018 IEEE 10th EuroS&P, pp. 353–367, 2018.

[7] S. A. Cook: "On the Minimum Computation Time of Functions", PhD thesis, Harvard University, 1966.

[8] J. W. Cooley and J. W. Tukey: "An Algorithm for the Machine Calculation of Complex Fourier Series", Mathematics of Computation, **19**, No. 90, pp. 297–301, 1965.

[9] L. Ducas et al.: "CRYSTALS-Dilithium: A Lattice-Based Digital Signature Scheme", IACR Transactions on Cryptographic Hardware and Embedded Systems, Vol. 2018 (1), pp. 238–268, 2018.

[10] S. M. Hafiz et al.: "Incompleteness in Number-Theoretic Transforms: New Tradeoffs and Faster Lattice-Based Cryptographic Applications", 2025 IEEE 10th EuroS&P, pp. 565–584, 2025.

[11] A. Karatsuba and Y. Ofman: "Multiplication of many-digital numbers by automatic computers", Proceedings of USSR Academy of Sciences, **145** (7), pp. 293–294, 1962.

[12] Z. Liang and Y. Zhao: "Number Theoretic Transform and Its Applications in Lattice-based Cryptosystems: A Survey", Cryptology ePrint Archive, Paper 2022/1682, 2022.

[13] S. Oku and M. Kudo: "Implementation of Toom-4 Hybrid Multiplication for Incomplete NTT", GitHub repository, https://github.com/Sakura-Oku/toom4_incomplete_ntt, 2026.

[14] A. Toom: "The complexity of a scheme of functional elements realizing the multiplication of integers", In: Soviet Mathematics-Doklady, **7**, pp. 714–716, 1963.

[15] J. von zur Gathen and J. Gerhard, Modern Computer Algebra, 3rd ed., Cambridge University Press, 2013.