Article

Quantum Circuit Design of Toom 3-Way Multiplication

Harashta Tatimma Larasati 1,2,†, Asep Muhamad Awaludin 1,†, Janghyun Ji 1 and Howon Kim 1,*

1 School of Computer Science and Engineering, Pusan National University, Busan 609735, Korea; harashta@pusan.ac.kr (H.T.L.); asep.muhamad11@pusan.ac.kr (A.M.A.); jj0819@pusan.ac.kr (J.J.)
2 School of Electrical Engineering and Informatics, Institut Teknologi Bandung, Bandung 40116, Indonesia
* Correspondence: howonkim@pusan.ac.kr
† These authors contributed equally to this work.

Abstract: In classical computation, Toom–Cook is one of the multiplication methods for large numbers which offers faster execution time compared to other algorithms such as schoolbook and Karatsuba multiplication. For the use in quantum computation, prior work considered the Toom-2.5 variant rather than the classically faster and more prominent Toom-3, primarily to avoid the nontrivial division operations inherent in the latter circuit. In this paper, we investigate the quantum circuit for Toom-3 multiplication, which is expected to give an asymptotically lower depth than the Toom-2.5 circuit. In particular, we designed the corresponding quantum circuit and adopted the sequence proposed by Bodrato to yield a lower number of operations, especially in terms of nontrivial division, which is reduced to only one exact division by 3 circuit per iteration. Moreover, to further minimize the cost of the remaining division, we utilize the unique property of the particular division circuit, replacing it with a constant multiplication by reciprocal circuit and the corresponding swap operations. Our numerical analysis shows that the resulting circuit indeed gives a lower asymptotic complexity in terms of Toffoli depth and qubit count compared to Toom-2.5 but with a large number of Toffoli gates that mainly come from realizing the division operation.

Keywords: complexity analysis; quantum circuit; Toom–Cook multiplication; Toom-3; integer division; constant multiplication

1. Introduction

Quantum arithmetic circuits are considered as the basic building blocks to realize various quantum algorithms [1,2]. After the prominent proposal of Shor’s algorithm [3] in 1994 which shows the apparent advantage of quantum computers in cracking the existing cryptosystems (i.e., Rivest–Shamir-Adleman (RSA) and elliptic curve-based cryptography (ECC)), substantial research efforts have been made to design an efficient quantum circuit for modular exponentiation operation, which is the essential component of the algorithm. Thus, various proposed methods have focused on building such specialized circuit and its underlying components, such as adder, modular adder, and modular multiplier by a constant [2,4–7], tailored to certain characteristics of modular arithmetic operations for achieving further simplifications [8].

However, as researchers explore wider area of studies, many other quantum algorithms being discovered require a more general arithmetic approach. Among other components, integer multiplication is one of the essential functions that also underlies a more complex operation, such as the polynomial multiplication. In fact, various quantum algorithms, such as the one proposed by Childs et al. for nonlinear structures [9], Fourier Transform computation [10], as well as matrix inversion, can be reduced to an integer multiplication [8].

Adapted from classical approach, quantum integer multiplication can be simply realized using the naive schoolbook multiplication algorithm [8,11], which takes only around $4n + 1$ qubits, with $n$ representing the operand length. However, the depth of the circuit...
scales quadratically to \( n \), which would be unfavorable for large numbers since circuit depth directly affects the runtime and also correlates to error rates [12]. Thus, a few proposals [8,11,13] have explored other multiplication algorithms which traditionally offers better asymptotic time complexity, i.e., classical Karatsuba [14] and Toom–Cook multiplication [15,16] for use in quantum computation. Karatsuba multiplication itself is a fast multiplication technique proposed by Anatoly Karatsuba in 1962 which splits an \( n \)-digit multiplication into three submultiplications and some additions whereas Toom–Cook, or Toom-\( k \) multiplication, is the generalization of Karatsuba algorithm to \( k \) smaller multiplications for achieving an even lower asymptotic complexity than Karatsuba. Classically, higher \( k \) yields better complexity for sufficiently large integers.

Earlier, Kowada et al. [13] analyzed the reversible Karatsuba circuit for the use in either classical reversible or quantum computation. Their work shows that the reversible version of Karatsuba is still able to maintain asymptotically lower runtime than the naïve multiplication. To be precise, their analysis for sequential approach shows a similar complexity for depth, space (qubit count), and circuit size, i.e., \( O(n \log_2 3) \approx O(n^{1.585}) \). Subsequently, Parent et al. [8] improved the work of [13] and presented a more comprehensive discussion from the quantum computation’s viewpoint. Beside providing the corresponding quantum circuit, the authors of [8] explained the method to optimize the algorithm’s recursive implementation to yield an optimal time-space trade off, namely via the reversible pebble games. First introduced by Bennet [17] for classical reversible computation, employing this approach further reduces the depth and space to \( O(n^{1.158}) \) and \( O(n^{1.427}) \), respectively, while circuit size remains.

In terms of Toom–Cook multiplication, there has only been one proposal to examine this algorithm in the quantum realm, that is by Dutta et al. [11]. Presenting the circuit for Toom-2.5 and conducting analysis similar to [8], they obtained the asymptotic complexity for depth, space, and size of \( O(n^{1.143}) \), \( O(n^{1.404}) \), and \( O(n^{1.547}) \), which outperforms the quantum Karatsuba circuit of [8,13]. In their paper, they stated that the primary reason why they utilize the less familiar Toom 2.5-way rather than the more commonly-used Toom 3-way multiplication is to avoid employing the nontrivial division circuit required by the latter method, which they argued to induce higher cost.

Nevertheless, we believe that it would be beneficial to examine the Toom 3-way multiplication for the use in quantum computation for several reasons. First, Toom 3-way multiplication is classically known to yield lower computational complexity than Toom 2.5-way and more often found in the software implementation [11,18], such as in the GNU multi-precision library [19]. In addition, considering the mathematical equations, most of the higher-order Toom–Cook multiplications make use of some nontrivial divisions. If such operation needs to be completely avoided, this may hinder the advancement for higher-order Toom-\( k \) for the quantum case. In contrast, analyzing its cost in the quantum case can provide insights and open the possibility to the higher Toom-\( k \) multiplication. Additionally, despite the view that division circuit is expensive, there have been several attempts to achieve relatively efficient quantum dividers, either for binary integer [20–22] or floating point arithmetic case [23]. Furthermore, the nontrivial division operation inherent to Toom 3-way (and higher order) multiplication is not a general one; instead, it uses exact integer division, which we later found to enable some further optimization to the circuit. In particular, for the case of division by 3 as in Toom-3, we may treat it as a constant multiplication by the reciprocal of 3. Using the constant multiplication circuit by [24], we find that Toom 3-way multiplication still give lower asymptotic Toffoli depth and space compared to [11] but with a higher cost in the number of Toffoli gates.

In this paper, we investigate Toom 3-way multiplication for the use in quantum computation. In particular, we provide the quantum circuit for the Toom-3 multiplication. Moreover, we adopt the classical sequencing from Bodrato [25] for our circuit construction steps, which results in a lower number of operation and primarily, the reduction of nontrivial division to only one for each multiplication step. Furthermore, we propose to regard the exact division by 3 circuit as a constant multiplication by reciprocal to enable
employing a more efficient quantum subroutine proposed by [24]. Subsequently, we analyze the complexity of the circuit in terms of recursive approach, as in [8,11]. The result of analysis shows that even though the division circuit required in Toom-3 does contributes to a higher Toffoli count, it still acquires better asymptotic complexity in terms of Toffoli depth and qubit count, hence still giving a competitive advantage compared to existing multiplication algorithms.

The contribution of this paper can be summarized as follows:

• We design the quantum circuit for Toom-3 multiplication which minimizes the division circuit from four to only one by employing the classical sequence proposed by Bodrato [25].

• We propose to use a constant multiplication circuit to perform the division by 3 operation. In particular, the division is treated as a constant multiplication by reciprocal.

• We analyze the complexity of the circuit, which shows that the proposed quantum circuit for Toom-3 still has a competitive advantage in terms of Toffoli depth and qubit count.

The remainder of the paper is summarized as follows. Section 2 describes the existing quantum multiplication methods as well as the brief theoretical aspects of classical Toom 3-way multiplication. Subsequently, Section 3 details our proposed methods whereas Section 4 contains the complexity analysis of the proposed Toom 3-way multiplication, along with its comparison with existing methods and some discussions. Lastly, Section 5 concludes the paper.

2. Related Work

2.1. Multiplication Methods in Quantum Computation

Proposed multiplication methods in early quantum computation are mostly focused on the specific use in Shor’s algorithm for factoring problem (i.e., that can be used for breaking the RSA cryptosystem). In that case, the aim is to build an efficient circuit that performs the modular exponentiation function $|x⟩ → |a^x \mod N⟩$, which essentially can be achieved by a sequence of modular multiplications. Additionally, the random value $a$ and the modulus $N$ are predetermined before running the algorithm, which translates to an exact sequence of underlying multiplication circuits. Hence, former methods mainly perform a multiplication by constant in a modular arithmetic fashion, requiring only one quantum input, such as [2,6,7]. Then, newer proposals offer more enhanced techniques by leveraging the specific property the constant to yield a more optimal circuit such as [4,5,24].

On the other hand, general multiplication with two quantum inputs (sometimes referred to as quantum-quantum multiplication [4]) have not been widely addressed. Just after the other variant of Shor’s algorithm, i.e., for the elliptic curve discrete logarithm problem (ECDLP) that can crack the ECC cryptosystem, started to explicitly materialize in the literature by Proos and Zalka [26], the need for quantum-quantum multiplier started to gain attention. The classically-inspired double-and-add method as first proposed in [26] becomes one of the earliest designs for such circuit. Later, the design is explicitly constructed by Roetteler et al. [27], who also proposed their own multiplication method based on Montgomery multiplication. In [28], Haner et al. improved the work of [27] to use a windowing approach and adaptive uncomputation placement to lower the overall depth of the elliptic curve scalar multiplication circuit. However, note that the multiplications for Shor’s algorithm employ a modular arithmetic approach, which has slightly different characteristics compared to the nonmodular one.

In terms of the nonmodular approach, the authors of [8] illustrated the standard case of two-inputs quantum multiplication using controlled addition circuit based on the popular Cuccaro adder [29], which scales to $O(n^2)$ in both Toffoli depth and size and scales linearly in space (i.e., qubit count) for $n$-bit operands. Earlier, Kowada et al. [13] provided the description of Karatsuba multiplication circuit whose main aim is for a classical reversible computation, but the authors also mentioned their secondary motivation, that is for possible implementation in quantum computers. Additionally, they analyzed four versions of
Karatsuba algorithm based on the position of multiplication gates and the garbage disposal scheme. Their analysis shows that among the analyzed version, parallel multiplication approach using Bennett’s first scheme for garbage disposal gives the lower complexity: $O(n)$ for time (depth), and $O(n \log_2 3)$ for both space (qubit count) and gate size. Later, Parent et al. [8] proposed their Karatsuba algorithm approach for quantum computation, which yields a lower cost depth and space compared to that of [13]. This is achieved by analyzing the method called the reversible pebble games to determine the most efficient number of recursion, which eventually yields a lower complexity: $O(n^{1.158})$ and $O(n^{1.427})$ for depth and space, respectively. Additionally, the author of [30] extends the reversible Karatsuba for binary finite fields.

Following a similar analysis approach to [8], Dutta et al. [11] described the quantum circuit for Toom–Cook multiplication method, which is the generalization of Karatsuba algorithm and classically gives a lower complexity. Instead of leveraging Toom 3-way multiplication as the more commonly used algorithm, they proposed a quantum circuit for Toom 2.5-way to avoid division circuit. Using their approach, the input is split into two and three parts, with bit size of $\frac{n}{2}$ and $\frac{n}{3}$, respectively. Their method is applied in recurrence, with the second time alternating the input size (i.e., the the smaller limbs will now be split into two parts and the larger limbs will be split into three parts), yielding sixteen submultiplications, each of size $\frac{n}{6}$. Their analysis shows that Toom 2.5-way multiplication, implemented recursively with certain cutoff value, gives even lower asymptotic complexity than the Karatsuba in [8]. Precisely: the cost are bounded by $O(n^{1.143})$, $O(n^{0.3816})$, and $O(n^{1.404})$ for Toffoli depth, Toffoli count, and qubit count.

In relation to Dutta et al.’s work, it is worth noting that this paper does not directly improve or modify the Toom-2.5 circuit in [11]. Rather, we examine another variant in the Toom–Cook algorithm’s family, namely the Toom-3. It is described by different mathematical equations, which translates to different underlying operations (e.g., the number of multiplications and additions/subtractions, the existence of division operation).

Additionally, we follow the sequence of operation of Toom 3-way multiplication presented by Bodrato in [25], which was based on his previous work with Zanoni [31] on finding the optimal inversion matrix for Toom–Cook by leveraging optimized exhaustive searches. Their proposed sequence results in a lower number of operations compared to what was implemented in GNU multi-precision library [31]. For the use in the quantum case, we arrange the operations into a quantum circuit and add the necessary blocks, such as “copy” (which is not required for the classical case) using CNOT gates to fan out specific values to new registers, which is required for carrying out quantum Toom-3 multiplication.

Nevertheless, even with a reduced number of operations, there is still one division operation that needs to be performed. This is not favorable since a division circuit tends to incur a higher cost, even when compared to a multiplication. However, by taking into account that the division required is in the form of exact division, we propose to treat it as a multiplication by reciprocal. This approach enables the use of a constant multiplication circuit with lower costs, as proposed by [24].

Additionally, we examine the final addition operation (i.e., in the recomposition step), which in principle requires four $\frac{2n}{3}$-bit addition circuits. By observing that some of the outputs do not overlap, we propose replacing the four circuits with one $2n$ adder, which slightly reduces the Toffoli counts. Lastly, we analyze the complexity of our approach in terms of Toffoli count, along with time-space trade off for the recursive implementation.

2.2. Toom–Cook Multiplication Method

2.2.1. General Toom–Cook Multiplication

Toom–Cook multiplication algorithm [15,16], sometimes referred interchangeably to as Toom-k or Toom k-way multiplication, is a method to reduce the complexity of a multiplication for larger numbers. Proposed by Andrei L. Toom in 1963 [15], the algorithm is essentially a generalization of Karatsuba multiplication. While in Karatsuba, the operand input is split into half and the multiplication is performed using three submultiplications
Toom–Cook multiplication lets the number of splits $k$ vary with operand bitlength $n$, choosing increasingly larger $k$ as $n$ grows to obtain a lower complexity [32]. In 1966, Stephen A. Cook [16] then showed in his dissertation that this method can be adapted to fast computer programs [32].

The main idea of Toom–Cook algorithm is to treat the desired multiplication of two operands as a multiplication of two degree-$(n - 1)$ polynomials, in which by utilizing the principle taken from linear algebra, can be done using only $(2n - 1)$ coefficient multiplications. That is, any degree-$n$ polynomial can be uniquely determined by its evaluation at $n + 1$ distinct points [33], which directly translates to the number of required multiplications. Hence, Toom-$k$ splits the input operands into $k$ parts and reduces the number of multiplications from $k^2$ to $2k - 1$. That is, Toom-2.5 requires four evaluation points (i.e., four submultiplications) whereas Toom-3 requires five. Theoretically, a higher value of $k$ yields lower time complexity in classical computers, which corresponds to lower circuit depth in quantum computation [34,35]. However, since there is overhead from the suboperations apart from the multiplications itself, Toom–Cook multiplication will give faster runtime when $n$ is sufficiently large [33].

Toom–Cook, along with other fast multiplication method such as Karatsuba, is often used in a recursive manner [33], creating a tree structure. As an example, the fully recursive implementation of Toom-3 is as shown in Figure 1, where $T$ denotes the Toom-3 multiplication, $n$ represents the bit length for each level, and $N$ indicates the total depth of the tree. Executing the full recursive multiplication (i.e., until the last branch, $n = 1$) gives better time complexity but suffers from a much larger space usage. Since naive multiplication is more space efficient [8], for a balanced performance between space and time, one may select a crossover point in which the recursive call will halt and the naive multiplication will be used [13], in which the point is denoted as $k$ in the figure.

Note that in the case of quantum, the computation is not as straightforward and the resource counts can be higher since we need to deal with quantum-related property of the circuit, such as reversibility, which requires performing uncomputation to clear up garbage outputs, i.e., outputs that are neither one of the primary inputs nor a useful output but must exist in the quantum circuit to preserve reversibility [22]. Hence, resource consumption in quantum computation tends to be higher than the classical counterparts.

![Figure 1. Recursion tree structure of the Toom-3 Multiplication.](image)

2.2.2. Toom 3-Way Multiplication

Toom 3-way multiplication splits a large integer multiplication to five smaller multiplications, then carries out the operations per part. The general method to implement...
Toom 3-way multiplication method is by: (1) splitting the inputs (i.e., the numbers to be multiplied or input operands) \(x\) and \(y\), each into three smaller parts of length \(\frac{n}{3}\), then (2) evaluating each of them on the five predetermined evaluation points, here we use \(\{0, 1, -1, -2, \infty\}\); (3) performing point-wise multiplication, continued by (4) interpolation, then at the end, carrying out (5) recomposition to form the final result.

Here, we briefly describe the notation used in this paper to avoid confusion. \(x\) denotes the full digit input; \(x_0, x_1, x_2\) represent the split inputs, whereas \(x(0), x(1), x(-1), x(-2), x(\infty)\) indicate the result of evaluating \(x\) for the corresponding evaluation points. Likewise, the notation of \(y\) is identical to that of \(x\).

For the first step, the splitting of Toom 3-way multiplication can be done as follows:

\[
x = x_22^{2j} + x_12^j + x_0
\]
\[
y = y_22^{2j} + y_12^j + y_0
\]

with radix \(j\) is precomputed as

\[
j = \max\left\{\left\lfloor \frac{\log_2 x}{3} \right\rfloor, \left\lfloor \frac{\log_2 y}{3} \right\rfloor \right\}
\]

For the second step, evaluation of point \(x\) is:

\[
x(0) = x_0
\]
\[
x(1) = x_2 + x_1 + x_0
\]
\[
x(-1) = x_2 - x_1 + x_0
\]
\[
x(-2) = 4x_2 - 2x_1 + x_0
\]
\[
x(\infty) = x_2
\]

Similarly, evaluation of point \(y\) is described below:

\[
y(0) = y_0
\]
\[
y(1) = y_2 + y_1 + y_0
\]
\[
y(-1) = y_2 - y_1 + y_0
\]
\[
y(-2) = 4y_2 - 2y_1 + y_0
\]
\[
y(\infty) = y_2
\]

Subsequently, one round (nonrecursive) of Toom 3-way multiplication uses five multiplications with smaller bit length. Multiplying each part of Equations (3) and (4) (i.e., point-wise), the result of each part is as presented in Equation (5), indicated as the notion \(P, Q, R, S,\) and \(T\):

\[
P = x_0y_0
\]
\[
Q = (x_2 + x_1 + x_0)(y_2 + y_1 + y_0)
\]
\[
R = (x_2 - x_1 + x_0)(y_2 - y_1 + y_0)
\]
\[
S = (4x_2 - 2x_1 + x_0)(4y_2 - 2y_1 + y_0)
\]
\[
T = x_2y_2
\]
Next, interpolation is the reverse of the point-wise multiplication results, which can be presented in a matrix form as presented in Equation (6). Then, the result of interpolation steps, here indicated as \( A, B, C, D, \) and \( E \), follows the mathematical equation as presented in Equations (7)–(11).

\[
\begin{pmatrix}
E \\
D \\
C \\
B \\
A
\end{pmatrix} =
\begin{pmatrix}
1 & 0 & 0 & 0 \\
1 & 1 & 1 & 1 \\
1 & -1 & -1 & 1 \\
1 & -2 & 4 & -8 \\
0 & 0 & 0 & 0 & 1
\end{pmatrix}
-1
\begin{pmatrix}
P \\
Q \\
R \\
S \\
T
\end{pmatrix}
\]

\[
\begin{pmatrix}
P \\
Q \\
R \\
S \\
T
\end{pmatrix} =
\begin{pmatrix}
\frac{1}{2} & \frac{1}{2} & -1 & \frac{1}{6} & -2 \\
-1 & \frac{1}{2} & \frac{1}{2} & 0 & -1 \\
-\frac{1}{2} & 1 & -1 & 2 & 0 \\
0 & 0 & 0 & 0 & 1
\end{pmatrix}
\]

(6)

Then, the final multiplication result can be obtained as:

\[
xy = A2^{4j} + B2^{3j} + C2^{2j} + D2^j + E
\]

(12)

3. Proposed Method

In this paper, we design a quantum circuit for Toom 3-way multiplication. In particular, we discuss the steps to achieve lower number of operation for a more efficient quantum circuit. Furthermore, we provide the circuit diagram for evaluation, point-wise multiplication, and interpolation steps, whereas the first (splitting) and the last step (recomposition) is explained in the narrative. Moreover, we elaborate on the techniques that can be utilized for realizing the underlying components of the quantum Toom-3 multiplication (e.g., addition and division circuit). Additionally, we discuss several adjustments of the underlying components to achieve a more efficient process. Specifically, we replace the division by 3 circuit to a constant multiplication by reciprocal, whose circuit is based on [24], to maintain a lower depth. In addition, we employ only one adder of size \(2^n\) for the recomposition step instead of four of size \(2^{3n}\), which slightly reduces the number of required components.

3.1. Efficient Scheduling

Borrowing from the hardware terms, to reduce the number of operations, it is essential to employ an efficient scheduling (or sequencing) of the operations, merging similar operations whose values are overlapped. Regarding the number of operations that need to be performed, note that different from the classical implementation in which several values can be operated at once (e.g., ternary adders, carry-save adder tree (CSAT) for addition operation), common quantum approaches still leverage two-input operation. Following this concept, in the theoretical, unoptimized Toom 3-way multiplication method as presented in Section 2.2.2, there exists twelve \(\frac{4n}{3}\) additions and five \(\frac{2n}{3}\) submultiplications from the initial up to the point-wise multiplication stage (Equation (5)). At the interpolation step (Equations (7)–(11)), there are eleven \(\frac{4n}{3}\) additions/subtractions. Additionally, taking into account the coefficients of \(P, Q, R, S, T\) in those equations, there are two multiplications
by 2, five divisions by 2, three divisions by 6, and one division by 3, each of size $\frac{2n}{3}$. It is worth noting that in quantum, the first two operations are considered trivial since they can be implemented conveniently by a doubling and halving circuit, while the latter two are nontrivial since there has been no straightforward way to perform such operations. Lastly, at the recomposition stage, another four $\frac{2n}{3}$ additions are required to combine the submultiplications’ outcomes into the final multiplication result.

In this paper, we adopt the sequence proposed by Bodrato [25] for classical Toom 3-way multiplication, which is known to give a much lower number of operations. Additionally, since there are differences from the classical approach, we adjust the sequence and components for use in quantum computation.

Let

\[
i_1 = \frac{S - Q}{3}
\]

(13)

\[
i_2 = R - P
\]

(14)

\[
i_3 = \frac{Q - R}{2}
\]

(15)

Then, the original mathematical formulation of Toom-3 interpolation can be rewritten as:

\[
A = T
\]

(16)

\[
B = \frac{i_2 - i_1}{2} + 2T
\]

(17)

\[
C = i_3 + i_2 - T
\]

(18)

\[
D = i_3 - B
\]

(19)

\[
E = P
\]

(20)

As can be inferred from Equations (17)–(19), a substantial simplification to the interpolation stage can be obtained. In particular, interpolation step requires only eight additions/subtractions. More importantly, as indicated by Equations (13) and (17), the execution of the costly division operation can be reduced to only one time, i.e., one division by 3.

3.2. Quantum Circuit for Toom 3-Way Multiplication

The high-level picture of the resulting quantum circuit is as illustrated in Figure 2. Similar to that of the classical circuit, the overall steps consist of splitting, evaluation of points, point-wise multiplication, interpolation, and recomposition. In the figures, each line corresponds to a quantum register with size displayed as subscripts of ket notation at the left side of the circuit. Meanwhile, the symbols inside the ket represent the input quantum state of each step. Additionally, the red triangles at each function blocks indicate the input and output of each corresponding operation. Specifically, triangles at the left of a block show where the input enters and triangles at the right side show where the output resides. Note that the ancillary registers are not shown for simplicity. For the sake of clarity, we break down the circuit for each part, as presented in Figures 3–6.
Figure 2. Overall Quantum Circuit for Toom-3 Multiplication.

For the input splitting step, it is implicitly done by placing the $n$-bit number $x$ as the first multiplicand in the corresponding register labeled $x_0$, $x_1$, and $x_2$, slicing $x$ into three parts with length of $n/3$ each, padded in the most significant bit (i.e., $x_2$ and $y_2$) when necessary. Subsequently, the evaluation of each $x$ and $y$ point is presented in Figures 3 and 4, all of which run in parallel since they run on different registers.

Figure 3. Quantum Circuit for Evaluation of Point X.

Figure 4. Quantum Circuit for Evaluation of Point Y.
Next, the point-wise multiplication is illustrated in Figure 5. Similar to the previous step, the submultiplication operations are also performed in parallel. This part is the core of a Toom–Cook multiplication circuit, which gives the advantage of reducing the depth of multiplication compared to the naïve method. Subsequently, interpolation step is as illustrated in Figure 6.

The outcomes of interpolation, i.e., A, B, C, D, E, are that each has the size of $\frac{2^n}{3}$-bit. As for the recomposition (i.e., the final step), it essentially combines the interpolation results to yield the final result. Naively, it requires four $\frac{2^n}{3}$-bit addition circuits. However, for those digits, the addition overlaps each other by $\frac{n}{3}$ bits. We find that instead of performing a series of additions as described above, we can realign the intermediate output, resulting in two $2n$-bit numbers. That is, A, C, and E can be concatenated since they do not overlap each other; the case is similar for B and D. This method reduces the number of adder circuits to only one $2n$-bit adder, which in our calculation, is more efficient. Precisely, taking one $n$-bit adder’s cost of $2^n$ for both Toffoli depth and count \([8,11]\), the cost of four $\frac{2^n}{3}$-bit additions are $16 \frac{3}{2}n$ whereas it is only $4n$ for one $2n$-bit adder.
Circuit Components

In general, six underlying operations are required for implementing the quantum Toom 3-way multiplication: copy, addition, subtraction, doubling/halving, division, and the underlying multiplication itself. Copy block is realized through a series of CNOT gates, targeting a series of “fresh” qubits initialized in state $|0\rangle$. These operations are required since once the quantum states are consumed, they cannot be used by other operations. Addition can be realized through any available addition circuit; in our case we choose the ripple-carry adder proposed by Cuccaro et al. [29]. One can use other existing adder variants, either another binary arithmetic-based adder such as proposed by Takahashi et al. [36] or the Fourier-based arithmetic utilizing a set of rotation gates such as proposed by Draper et al. [37]. Subsequently, subtraction is performed simply by the reversed adder (for the binary arithmetic approach) or by the corresponding inverse circuit (for Fourier-based circuit). Regarding the doubling and halving, they are used to efficiently realize multiplication or division by 2, similar to that in classical circuit. In quantum computing, they can be achieved quite easily and efficiently by a cyclic qubit shifts through a series of qubit swappings, either using CNOT gates [27] or via qubit relabeling [38,39].

Additionally, since binary arithmetic is used in this paper, the radix or word size $j$ is fixed to $n/3$ rather than varying as previously shown in Equation (2).

Furthermore, special attention should be given to the division circuit since it is commonly known to be resource-expensive, whose cost can be as high or even higher than the multiplication itself, such as in [24]. Fortunately, the division required in the Toom-3 is an exact division, whose output is always an integer instead of a fraction, and there is no remainder. Secondly, it is a division by constant. Classical computation has employed several different techniques to make the operation efficient, such as by evaluating the division as a multiplication by the number’s inverse. However, since extracting the value’s information from the preceding quantum circuit is not straightforward, this approach is not favorable. Rather than using this approach, we employ another method that suits this special case of integer division, that is by treating it as a multiplication by the reciprocal of 3 followed by a binary shifting. Proposed by Alverson in 1991 [40], this approach has been adopted and widely used, such as in the GNU multi-precision library [19] and various patents.

Regarding the case of Toom-3, we require an exact division by 3 circuit. Since 3 is a prime number, its reciprocal has a period (i.e., always repeats). If we convert its decimal fraction (0.333...) to binary, we obtain an unending 01010101. Thus, multiplying this reciprocal by a number will be equivalent to dividing the number by 3, adding “1” at the end for rounding function [19,40]. Note that the result needs to be right-shifted by $(n+1)$-bit to get the final result. For instance, let 27 be the dividend in a 16-bit system. Dividing it by 3 equals to multiplying it with the reciprocal of 3 (i.e., 0xAAAB in hexadecimal), which yields 1179657. Shifting right the intermediate result by 17 will obtain 9, which is the exact quotient of 27/3. Then, for the quantum case, we can use the multiplication by constant circuit proposed in [24], whose Toffoli depth, Toffoli count, and qubit count are $8n$, $4n(n+1)$, and $3n + 1$, respectively. Note that the depth of this constant multiplication circuit is around four times of an adder (whose depth is bounded by $2n$ [8,11]), which is lower than $n^2$ as in the base-case multiplication.

4. Cost Analysis

In this section, we analyze the complexity of the quantum Toom 3-way multiplication circuit. In particular, the metrics used are the Toffoli count, qubit count, and Toffoli depth of the multiplication, in the form of space-time complexity trade off when performed recursively. In this study, we focus on Toffoli calculation since it incurs higher cost than other classical reversible gates counterparts such as NOT and CNOT [41,42]. Furthermore, Toffoli gates correspond directly to the depth of the T gate (often referred to as T-depth) in the Clifford + T library, which is currently the most favorable fault-tolerant implementation for a quantum algorithm, that directly affects run times [12,43]. Hence, this gate has been commonly used as the primary metric for assessing the cost of quantum circuits
and algorithms, such as used in [4,5,8,11,30] to name a few. Note that the discussion concentrates on the asymptotic performance rather than for small-number multiplication.

4.1. Gate Count

In the implementation, Toom–Cook circuit (as well as other fast multiplication circuit such as Karatsuba) is generally performed recursively until a certain criteria is met (e.g., until a specified multiplication size is attained, until a balanced time and space complexity is achieved, etc.). For multiplication, the basic assumption is that one Toffoli gate can be used to perform one-bit number multiplication [8,11], which will be the base scenario of our cost calculation. In addition, the cost of an in-place adder \( A_n \) can be bounded by at most 2\( n \) Toffoli gates [8], with \( n \) is the bit size of the larger addend. In terms of division circuit, we consider the implementation of constant multiplication in [24] with the Toffoli count of 4\( n(n + 1) \), which is competitive to the lowest cost of performing division in quantum, as found in other implementations [20–23].

Let \( T_n \) denote the cost of Toom 3-way multiplication for two large \( n \)-bit numbers. Accordingly, \( A_n \) and \( D_n \) represent the cost of \( n \)-bit addition/subtraction and \( n \)-bit division, respectively. As discussed in the Section 3, realizing \( n \)-bit Toom 3-way multiplication requires a total of five \( \frac{2}{3} \) submultiplications, three adder types of different length (ten \( \frac{3}{2} \)-bit, eight \( \frac{5}{2} \)-bit, and one \( 2n \)-bit addition/subtraction), and one \( \frac{2}{3} \)-bit division (which is carried out by the constant multiplication circuit). Then, the Toffoli cost of \( n \)-bit Toom 3-way multiplication can be calculated as in Equation (21).

\[
T_n = 5T_{\frac{n}{2}} + 10A_{\frac{n}{2}} + 8A_{\frac{2n}{3}} + A_{2n} + D_{\frac{2}{3}n} \tag{21}
\]

For a recursive implementation, the cost expands to Equation (22).

\[
T_n = 5^{\log_5 n} T_1 + 10\left(A_{\frac{n}{2}} + 5A_{\frac{n}{3}} + \ldots + 5^{\log_5 (n-1)} A_1 \right) + 8\left(A_{\frac{2n}{3}} + 5A_{\frac{2n}{6}} + \ldots + 5^{\log_5 (n-1)} A_2 \right) + (A_{2n} + 5A_{2n} + \ldots + 5^{\log_5 (n-1)} A_6) + (D_{\frac{2n}{3}} + 5D_{\frac{2n}{6}} + \ldots + 5^{\log_5 (n-1)} D_2) \tag{22}
\]

Substituting the Toffoli cost of \( A_n = 2n \) and \( D_n = 4n(n + 1) \), the equation can be rewritten as

\[
T_n = 5^{\log_5 n} + \sum_{i=0}^{\log_5 (n-1)} \left[ 24n \left( \frac{5}{3} \right)^i + 16n^2 \left( \frac{5}{9} \right)^i \right] \tag{23}
\]

Using geometric series \( \sum_{i=0}^{m-1} r^i = \frac{1-r^m}{1-r} \), we obtain the Toffoli cost of recursive implementation as presented in Equation (24).

\[
T_n = 5^{\log_5 n} + 24n \left( \frac{1 - (5/3)^{\log_5 n}}{1 - 5/3} \right) + 16n^2 \left( \frac{1 - (5/9)^{\log_5 n}}{1 - 5/9} \right) \tag{24}
\]

This obtained result is still not considering the uncomputation, which is essential in the workings of quantum computer. Taking it into account will require roughly doubling the previously acquired cost [8,11]. In this case, the “clean” cost would be as presented in Equation (25).

\[
T_n^{\text{(clean)}} = 8n^2 + 66n^{\log_5 5} - 72n \tag{25}
\]
4.2. Space-Time Complexity Analysis

The recursive approach intrinsic to the algorithm may result in an enormous space consumption (i.e., qubit count) if not bounded. To improve the asymptotic performance in terms of space, one can employ the so-called reversible pebble game. Introduced by Bennet [17] in 1989, the reversible pebble game is a modification of the traditional pebble game for use in the field of reversible computing, which allows analysis of time and space complexity as well as to achieve a time-efficient space-restricted computation [44]. This approach enables one to analyze techniques to minimize scratch space with the downside of needing to recompute intermediate results [8].

To calculate the optimized multiplication cost, we follow the steps presented in [8,11]. Additionally, the depth of our employed constant multiplication scales in linear fashion, similar to adder, thus treated similarly and omitted from the multiplication depth. In terms of the proposed Toom-3 circuit, the recursive implementation of five parallel multiplications forms a quinary tree structure. For an \( n \)-sized input at level \( l \), there are \( 5^l \) nodes with size \( 3^{-l}n \), with the total circuit cost of that level is \( n(\frac{5}{3})^l \). Consequently, the cost of full quinary tree is as presented in Equation (26).

\[
n^\sum_{i=0}^{N} \left(\frac{5}{3}\right)^i, \quad N = \log_3 n \tag{26}
\]

To achieve a balanced performance, we should seek the appropriate tree height \( k \) which makes the size of the below subtrees roughly equal to the tree above them. That is,

\[
n^\sum_{i=0}^{N-k-1} \left(\frac{5}{3}\right)^i = \frac{1}{3^{N-k}} \sum_{i=0}^{k-1} \left(\frac{5}{3}\right)^i \tag{27}
\]

Similar to Equation (25), the geometric series identity allows us to obtain the bound:

\[
k \leq \frac{N}{2 - \log_3 \frac{5}{3}} \approx 0.759N \tag{28}
\]

Thus, the space can be reduced to:

\[
QC = O \left( n \left(\frac{5}{3}\right)^{\frac{\log_3 \frac{5}{3}}{\log_3 \frac{5}{3}} \log_3 n} \right) \approx O(n^{1.353}), \tag{29}
\]

which is lower than the original required space:

\[
n^\sum_{k=0}^{\log_3 n - 1} \left(\frac{5}{3}\right)^k = \left(1 - \left(\frac{5}{3}\right)^{\log_3 n} \right) \left(1 - \frac{5}{3}\right) \tag{30}
\]

which bounds to

\[
O(n^{\log_3 5}) \approx O(n^{1.465}). \tag{31}
\]

Subsequently, the time complexity is often considered as the Toffoli depth of the circuit [11,12]. In this case, it can be calculated as the product of number of subtrees \( S_k \) and its corresponding depth \( D_k \) at the \( k \)-th level [11]. Then, the overall Toffoli depth \( T_d \) is as presented in Equation (32). The complexity analysis of Toom 3-way multiplication
As shown in the figures, the resulting quantum circuit for Toom 3-way multiplication does give lower asymptotic Toffoli depth and qubit count compared to prior work. Specifically, in terms of Toffoli depth, the cost scales $O(n^{1.112})$ to the multiplication length whereas Toom-2.5, Karatsuba, and naïve schoolbook multiplication cost $O(n^{1.143})$, $O(n^{1.158})$ and $O(n^2)$, respectively. Similarly, the qubit count also tops Toom-2.5 and Karatsuba with $O(n^{1.353})$ compared to $O(n^{1.404})$ and $O(n^{1.427})$, respectively. Note that for qubit count, schoolbook multiplication does incur a linear cost, meaning lower than all of the non-naïve multiplications; this is expected since no parallelization is enacted. Nevertheless, in terms of Toffoli count, Toom-3 circuit still scales quadratically as in the naïve method. This is heavily contributed by the division operation (or the constant multiplication in our case) that exist in Toom-3, which requires quadratic number of gates for its operation. Note that we use the cost of (unoptimized) general constant multiplication circuit in the complexity calculation to show the worse scenario. For a non-asymptotic performance consideration, an optimization to reduce the gate count in the constant multiplication by roughly a half can be employed since the constant that we use is fixed (i.e., the reciprocal, 0101...011), eliminating the gates required for the case of bit “0”. Then, the Toffoli count for the constant multiplication can be reduced to as low as $4n(\frac{1}{2}n + 2) = 2n^2 + 8$.

Regarding the possibility of employing other techniques to achieve a more efficient circuit, in classical computation, some authors proposed the division-free Toom–Cook multiplication, such as [18,45]. The methods generally revised the inversion matrix of Toom–Cook, multiplying it by 3, which results in the absence of division operation but multiply all of the values in the interpolation steps. The division is then performed at the end of
the circuit or fed directly for the higher-level function such as a modular multiplication. However, our calculation finds this is unsuitable for quantum usage since it incurs even higher additional cost. In particular, there are numerous constant multiplications required for the coefficient multiplication, and there will still be a division at the end, which nullifies the advantage of the so-called division-free itself. Nonetheless, this approach may be advantageous if the multiplication is not a standalone operation but incorporated into a more advanced operation such as in modular exponentiation.

In summary, Toom 3-way multiplication is competitive for achieving a lower asymptotic runtime and qubit count, with the tradeoff of higher number of Toffoli gates employed in the circuit. Nevertheless, division operation has been a bottleneck in lowering the Toffoli counts. Since higher-order Toom–Cook multiplication (e.g., Toom–4, Toom–5, etc.) also employ nontrivial divisions, it is essential to pursue a more efficient division method to enable further exploration in Toom–Cook multiplication, which we will investigate in the future research.

5. Conclusions

In this paper, we have explored the classical Toom 3-way multiplication for use in quantum computation, which shows a lower asymptotic depth compared to the prior quantum Toom 2.5-way multiplication method. In particular, we designed the quantum circuit and adopted the sequence proposed by Bodrato in the classical computation to yield a lower number of operations, especially in terms of nontrivial division. This efficient sequence has managed to reduce the need for such division circuits from four to only one per iteration, i.e., to one exact division by 3. Furthermore, we replace the division circuit with a constant multiplication by reciprocal circuit and the corresponding swap operations. The result shows that the Toom 3-way multiplication circuit does give a lower asymptotic complexity in terms of Toffoli depth and qubit count but with a tradeoff of a large number of Toffoli counts required for the division operation. Therefore, achieving a more efficient method for division would be of great importance to be pursued in the future.

Author Contributions: Conceptualization, H.T.L.; methodology, H.T.L. and A.M.A.; software, H.T.L. and A.M.A.; validation, J.J. and H.K.; formal analysis, A.M.A. and H.T.L.; resources, J.J.; data curation, H.T.L.; writing—original draft preparation, H.T.L. and A.M.A.; writing—review and editing, H.T.L. and A.M.A.; visualization, H.T.L. and A.M.A.; supervision, H.K.; project administration, J.J.; funding acquisition, H.K. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by Institute for Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No.2019-0-00033, Study on Quantum Security Evaluation of Cryptography based on Computational Quantum Complexity) and this work was also supported by Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (2019-0-01343, Regional Strategic Industry Convergence Security Core Talent Training Business).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: H.T.L. would like to thank Muhammad Firdaus for his assistance in addressing the reviewers’ feedback. Additionally, all authors would like to thank the anonymous reviewers for their constructive suggestions, which improve the quality of this work.

Conflicts of Interest: The authors declare no conflict of interest.

References
1. Takahashi, Y. Quantum arithmetic circuits: A survey. IEICE Trans. Fundam. Electron. Commun. Comput. Sci. 2009, 92, 1276–1283. [CrossRef]
2. Vedral, V.; Barenco, A.; Ekert, A. Quantum networks for elementary arithmetic operations. Phys. Rev. A 1996, 54, 147. [CrossRef]
35. Preskill, J. Lecture Notes for Ph219/CS219: Quantum Information and Computation Chapter 5; California Institute of Technology: Pasadena, CA, USA, 2015.

36. Takahashi, Y.; Tani, S.; Kunihiro, N. Quantum addition circuits and unbounded fan-out. Quantum Inf. Comput. 2010, 10, 872–890.

37. Draper, T.G. Addition on a quantum computer. arXiv Prepr. 2000, arXiv:quant-ph/0008033.

38. Žnidarič, M.; Giraud, O.; Georgeot, B. Optimal number of controlled-NOT gates to generate a three-qubit state. Phys. Rev. A 2008, 77, 032320. [CrossRef]

39. Chao, R.; Reichardt, B.W. Fault-tolerant quantum computation with few qubits. NPJ Quantum Inf. 2018, 4, 1–8. [CrossRef]

40. Alverson, R. Integer division using reciprocals. IEEE Symp. Comput. Arith. 1991, 1, 186–190.

41. Shende, V.V.; Markov, I.L. On the CNOT-cost of TOFFOLI gates. arXiv 2008, arXiv:0803.2316.

42. Jang, K.; Choi, S.; Kwon, H.; Kim, H.; Park, J.; Seo, H. Grover on Korean Block Ciphers. Appl. Sci. 2020, 10, 6407. [CrossRef]

43. Abdessaied, N.; Amy, M.; Soeken, M.; Drechsler, R. Technology mapping of reversible circuits to Clifford+ T quantum circuits. In Proceedings of the 2016 IEEE 46th international symposium on multiple-valued logic (ISMVL), Sapporo, Japan, 18–20 May 2016; pp. 150-155.

44. Královič, R. Time and space complexity of reversible pebbling. RAIRO-Theor. Inform. Appl. Inform. Théorique Appl. 2004, 38, 137–161. [CrossRef]

45. Ding, J.; Li, S.; Gu, Z. High-speed ECC processor over NIST prime fields applied with Toom–Cook multiplication. IEEE Trans. Circuits Syst. I Regul. Pap. 2018, 66, 1003–1016. [CrossRef]