IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 14 January 2026, accepted 9 February 2026, date of publication 18 February 2026, date of current version 23 February 2026.

Digital Object Identifier 10.1109/ACCESS.2026.3665887

RESEARCH ARTICLE

An Evaluation of Computational Complexity of Shor's Algorithm on Surface Codes for Factorization

KENTO OONISHI[1] AND NOBORU KUNIHIRO[2], (Member, IEEE)
[1]Mitsubishi Electric Corporation, Kamakura 247-8501, Japan
[2]University of Tsukuba, Tsukuba 305-8573, Japan

Corresponding author: Kento Oonishi (Onishi.Kento@ap.MitsubishiElectric.co.jp)

This work was supported in part by Japan Society for the Promotion of Science (JSPS) KAKENHI under Grant JP21H03440, and in part by Japan Science and Technology CREST under Grant JPMJCR2113.

ABSTRACT Shor's algorithm is a quantum algorithm that can efficiently solve the problems of integer factorization and discrete logarithms, which potentially break currently used public-key cryptosystems such as RSA and elliptic curve cryptography. However, a significant challenge in solving these problems is the noise inherent in quantum computers. Previous research advanced fault-tolerant techniques such as surface codes to mitigate this noise in quantum computers. Analyzing the security of currently used public-key cryptosystems on fault-tolerant quantum computers is essential. In this paper, we estimate the computational cost required to solve integer factorization using Shor's algorithm on surface codes. We evaluated computational resources required for surface codes and magic state distillation and derived the computational cost of Shor's algorithm based on these evaluations. Our computational complexity analysis indicates that there is no significant difference in the order of complexity. However, it underscores the paramount importance of optimizing the coefficient part for future efficiency improvements.

INDEX TERMS Shor's algotirhm, integer factorization, surface code, magic state distillation, security evaluation.

I. INTRODUCTION

A. BACKGROUND

In recent years, accelerated advancements have been achieved in quantum computers. For example, IBM released Condor in 2023, which is a quantum computer with 1,121 qubits and it plan to incorporate error correction into its future development plans to address the expanding scale of quantum computers [1]. Furthermore, Google is advancing proof-of-concept studies on error-correcting codes for quantum computers [2]. As such, both IBM and Google are striving to develop fault-tolerant quantum computers, advancing toward the realization of large-scale quantum computing.

Quantum computers can significantly affect the society by potentially compromising the currently used public-key cryptosystems using Shor's algorithm [3]. Current public-key cryptosystems include RSA [4] and elliptic curve cryptosystems [5], [6], which rely on the computational difficulty of certain mathematical problems as the basis for its security. For example, RSA encryption is based on the difficulty of the integer factorization problem, whereas elliptic curve cryptography relies on the discrete logarithm problem. Shor's algorithm can efficiently solve both the integer factorization and discrete logarithm problem, underpinning the security of these cryptographic systems. Consequently, Shor's algorithm poses a potential threat to the current information society, necessitating the evaluation of its computational complexity.

However, Shor's algorithm is yet to be realized on a sufficiently large scale to compromise cryptographic systems by solving substantial integer factorization or discrete logarithm problems. This can be attributed to the limited number of qubits available in current quantum computers and the presence of noise within these systems. Quantum computers currently under development are referred to as noisy

The associate editor coordinating the review of this manuscript and approving it for publication was Siddhartha Bhattacharyya.

intermediate-scale quantum (NISQ) computers, wherein each qubit is subject to significant noise. Consequently, it is imperative to perform calculations while mitigating these noise factors to achieve high-precision outputs in large-scale quantum computations, such as those required by Shor's algorithm.

Research on error correction methods is being actively pursued to realize large-scale, fault-tolerant quantum computations, including Shor's algorithm. Surface codes [7] are well known because of its relatively high feasibility, which requires an accuracy of approximately 99% for each gate, making them more practical compared to other techniques. Surface codes employ a large number of noisy qubits (physical qubits) to construct qubits with reduced noise (logical qubits). Many existing studies on the computational complexity of Shor's algorithm [8], [9] have conducted analyses incorporating surface codes. Google [2] successfully conducted proof-of-concept studies on small-scale surface codes, and further advancements in this research are anticipated. Therefore, the computational complexity analysis of Shor's algorithm under surface codes is of paramount importance.

In this paper, we discuss Shor's algorithm for solving the integer factorization problem. Shor's algorithm comprises modular exponentiation and the quantum Fourier transform with the primary computational cost being attributed to modular exponentiation [10]. Therefore, a meticulous evaluation of the computational complexity of modular exponentiation is crucial for estimating the overall computational complexity of Shor's algorithm. Modular exponentiation is implemented through repeated additions or multiplications [3], [11]. The main implementation methods for addition or multiplication include the Ripple-Carry [8], [12], [13], [14], Carry-Lookahead [15], [16], and Fourier-basis methods [10], [17], [18]. Existing research focused on reducing high-cost T gates used in magic state distillation incorporating surface codes [7], [19], [20] and magic state cultivation [21]. Cost reductions for individual methods have been explored; however, comparative evaluations between these methods remain insufficient. Computational costs in terms of physical qubits, considering surface codes, have not been evaluated for the cost-reduced Carry-Lookahead [16] and Fourier-basis methods [10]. Moreover, existing evaluations of computational costs have not adequately addressed theoretical analyses that encompass surface codes and magic distillation, making it critically important to assess their overhead with precision. Therefore, this paper aims to evaluate computational costs in terms of physical qubits and conduct comparative analyses between these methods.

B. CONTRIBUTIONS

In this study, we evaluate the computational cost of Shor's algorithm to factor an n-bit number considering physical qubits. We assess the computational cost of four primary construction methods of Shor's algorithm: the Ripple-Carry (two variants) [8], [13], Carry-Lookahead [16], and Fourier-basis methods [10]. We discuss three types of magic state distillation [7], [19], [20] and magic state cultivation [21] on surface codes [7] to conduct these computational cost evaluations. We conduct a theoretical analysis of computational complexity encompassing surface codes and magic state distillation, while also comparing the computational complexities across multiple methods. The structure of this paper is illustrated in Figure 1.

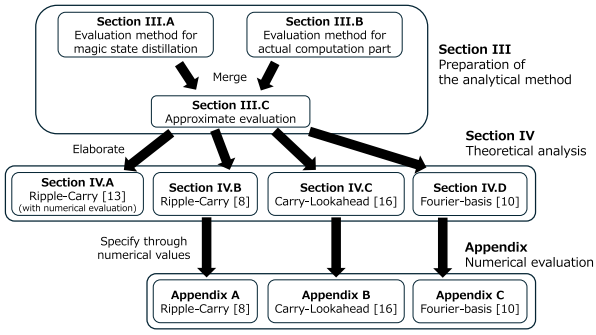

FIGURE 1. Schematic of the structure of this paper.

In Section III, we present the method for evaluating computational complexity related to surface codes and magic state distillation, and we discuss the overhead of additional computational costs required by these methods. In this study, we employ a composite evaluation approach using multiple evaluation methods [9], [19], [22]. We demonstrate that the overhead in terms of time and space complexities is at most poly(n), which implies that Shor's algorithm is efficient even when considering the overhead with physical qubits.

In Section IV, we evaluate the computational cost of Shor's algorithm across various implementations and compare them. In our evaluation, we conduct a meticulous scheduling that takes into account the execution time of quantum gates, thereby enabling a more precise evaluation. Our study reveals that the Carry-Lookahead method by Oonishi et al. [16] is the most efficient in terms of computational cost. Actual computed numerical values corroborated a similar reduction; however, in practice, the difference in KQ_p, product of the number of physical qubits and lattice steps, compared to the next best method, namely, the Ripple-Carry methods by Gidney and Ekerå [8], was only a factor between one to two, indicating minimal disparity. This can be attributed to the difference in KQ_p being only $\Theta(\log n)$. This underscores the paramount importance of optimizing the coefficient part for future efficiency improvements.

II. PRELIMINARIES

In this section, we first introduce the gate set addressed in this paper. Next, we introduce the computational complexity metrics used in this paper. Subsequently, we explain surface codes, magic state distillation, and magic state cultivation. Moreover, we explain the existing Shor's algorithm. Finally,

we introduce related works on evaluation method for the computational complexity of Shor's Algorithm.

A. GATE SET

In this paper, we utilize the following gate set.

- **Clifford gate:** $H = \frac{1}{\sqrt{2}}\begin{pmatrix}1 & 1\\ 1 & -1\end{pmatrix}$, $S = \begin{pmatrix}1 & 0\\ 0 & \mathrm{i}\end{pmatrix}$, and $\mathrm{CNOT} = \begin{pmatrix}1 & 0 & 0 & 0\\ 0 & 1 & 0 & 0\\ 0 & 0 & 0 & 1\\ 0 & 0 & 1 & 0\end{pmatrix}$
- **Non-Clifford gate:** $T = \begin{pmatrix}1 & 0\\ 0 & \exp(\mathrm{i}\pi/4)\end{pmatrix}$

Although not included in the aforementioned gate set, Pauli gates $X = \begin{pmatrix}0 & 1\\ 1 & 0\end{pmatrix}$, $Y = \begin{pmatrix}0 & -\mathrm{i}\\ \mathrm{i} & 0\end{pmatrix}$, and $Z = \begin{pmatrix}1 & 0\\ 0 & -1\end{pmatrix}$ can be expressed as a product of Clifford gates. When utilizing surface codes, the computational cost of the T gate increases significantly because of magic state distillation. Therefore, we focus on the T gate and evaluate the associated computational costs.

B. METRIC FOR EVALUATING COMPUTATIONAL COMPLEXITY

In this section, we introduce the computational complexity evaluation metrics KQ_p, product of the number of physical qubits and lattice steps, utilized in this study. Traditionally, the computational cost of quantum circuits is primarily evaluated using metrics such as the number of qubits, depth of the quantum circuit, and number of quantum gates. However, these metrics are insufficient when considering the use of surface codes [7]. If the evaluation metrics for quantum circuits are limited to either the number of qubits or depth of the quantum circuit, reducing one metric can increase the other. When evaluating the number of quantum gates, although error correction via surface codes is consistently applied across all qubits, the computational costs associated without quantum gates are not included. Given these considerations, this paper considers KQ [23], which is the product of the number of logical qubits and circuit depth, as the evaluation metric. KQ is a useful metric for evaluating computational cost because it can encompass the trade-off between the number of qubits and depth of the quantum circuit and the error correction performed using surface code in regions where quantum gates are not present.

We utilized the quantum computing model proposed by Jones et al. [9] to evaluate computational cost. Jones et al. proposed a model that structures a quantum computer into physical, virtual, quantum error correction, logical, and application layers, with the computational scale increasing in the order listed. An overview of processes performed at each layer are presented below.

- The physical layer handles the physical operations of the quantum computer.
- The virtual layer realizes quantum gates on physical qubits.
- The quantum error correction layer implements logical qubits through error correction using surface codes with numerous physical qubits [7].
- The logical layer achieves universal computation, including T gate magic state distillation [7].
- Finally, the application layer executes practical computations, such as Shor's algorithm.

In the application layer of Jones et al.'s quantum computing model, performing practical computations necessitates computational overheads for error correction such as magic state distillation and surface codes. Appropriately selecting the method of magic state distillation and code distance of the surface code is essential to achieve high-precision calculations [22]. Reducing the KQ of the actual computation part, i.e. the product of the number of logical qubits and lattice steps, lowers the required precision for each quantum gate, which decreases the circuit scale required for surface codes and magic state distillation. However, in practice, the circuit depth of the actual computation part, and thus the KQ, is affected by the waiting time for magic state distillation processes [10], [16]. Increasing the number of T gates generated at once reduces the circuit depth and KQ of the actual computation part. Therefore, the scheduling method of the quantum circuit varies significantly depending on the number of T gates generated at once.

Based on the aforementioned discussion, this study uses KQ_p as an evaluation metric. We assume that the size of the surface code, i.e., the number of physical qubits used per logical qubit, differs between the actual computation and magic state distillation parts. Consequently, we utilize the number of physical qubits as the metric for qubit count.

C. SURFACE CODES, MAGIC STATE DISTILLATION, AND MAGIC STATE CULTIVATION

Surface codes, magic state distillation, and magic state cultivation are crucial for error correction. By accurately estimating these computational costs, the overhead required for the transition from NISQ computers to fault-tolerant quantum computers will be appropriately assessed. This section provides an overview of the computational costs for surface codes, magic state distillation, and magic state cultivation.

Firstly, in surface codes, the code distance d determines the accuracy and computational cost of each quantum gate. Increasing the code distance enhances the redundancy for a single logical qubit, thereby increasing the computational cost while improving the accuracy of each quantum gate. The changes in accuracy with varying code distances have been simulated by Fowler et al. [22] and Jones et al. [9]. These studies demonstrate that increasing the code distance d results in an exponential decrease in the error rate of the output logical qubits. In error-correcting codes with a code distance of d, approximately $d/2$ errors can be corrected. Therefore, under an error rate of p, the probability of encountering uncorrectable errors is approximately $p^{d/2}$. Consequently, the error

rate in surface codes exponentially decreases with increasing code distance d, which is consistent with simulation results. To realize one logical qubit using the aforementioned surface code, qubits are arranged on a two-dimensional lattice with a side length of d. Measurements related to the stabilizers are performed on each lattice to detect and correct errors. Including the bits that store the measurement results for each lattice, a surface code with a code distance of d requires $2(d+1)^2$ physical qubits to represent a single logical qubit.

Next, we discuss magic state distillation. Magic state distillation is a technique used to create high-fidelity quantum states from multiple error-prone quantum states, enabling the realization of accurate non-Clifford gates. This process is particularly crucial for surface codes, where naive methods can only achieve low-fidelity non-Clifford gates. The initial concept of magic state distillation was proposed by Bravyi and Kitaev [24]. They evaluated the probability of obtaining the desired output state from magic state distillation, thereby allowing for the estimation of its computational cost. Magic state distillation is performed on quantum states encoded with quantum error correction codes such as surface codes, and the output can be categorized into following three scenarios, the desired quantum state, a quantum state within the code space but with errors, and a quantum state outside the code space. If the output state falls outside the code space, the distillation is considered a failure, and the process must be repeated. In the other two cases, the computation proceeds, and if the initial quantum state has a sufficiently high fidelity, the distillation process yields a higher-fidelity quantum state. Magic state distillation is applied to T gates [7], [20] and CCZ gates [19], [20]. Additionally, the computational cost for generating T gates is reduced through magic state cultivation [21] on color codes [25], rather than surface codes. This method generates high-fidelity T gates by gradually increasing the code distance. When the error rate of physical quantum gates is at 10^{-3}, the direct application of Shor's algorithm remains fraught with errors, necessitating a subsequent application of magic state distillation [14]. However, as the error rate of physical quantum gates decreases, the error associated with the generated T gates significantly diminishes [21]. Therefore, the computational complexity of Shor's algorithm will be substantially reduced in the future, because additional magic state distillations are not required.

D. EXISTING CONSTRUCTION METHODS OF SHOR'S ALGORITHM

Shor's algorithm is composed of modular exponentiation and the quantum Fourier transform, and existing research focuses on optimizing the computationally intensive modular exponentiation. The primary implementation methods for modular exponentiation include the Ripple-Carry [8], [12], [13], Carry-Lookahead [15], [16], and Fourier-basis methods [10], [17], [18]. These methods can handle the carry generated during addition differently, as indicated below.

- **Ripple-Carry method :** [8], [12], [13] Compute the carry sequentially from the lower bits.
- **Carry-Lookahead method :** [15], [16] Precompute all carries before starting the actual computation.
- **Fourier-basis method :** [10], [17], [18] Use the quantum Fourier transform (QFT) to compute carries independently for each qubit.

Although other adder circuits have been proposed, this study focuses on these recently optimized methods as the primary subject of research. Consideration of other adder circuits remains a topic for future work.

The differences in how these methods handle carries result in variations in computational complexity. Table 1 summarizes the computational complexity, focusing on the T and Toffoli gates, when factoring an n-bit composite number. KQ_T represents the product of the number of logical qubits and T-depth. Similarly, $\mathrm{KQ}_{\mathrm{Toffoli}}$ represents the product of the number of logical qubits and Toffoli-depth. Table 1 includes the estimated overall computational costs based on addition. For the Ripple-Carry method [8], the computational costs for the non-dominant terms are significant within the range of $n = 2048$ considered, and these details are included.

1) RIPPLE-CARRY METHOD

The Ripple-Carry method [12] is a technique that sequentially calculates the carry from the lower bits.

Initially, the majority (MAJ) circuit, which computes the majority of the three qubits, is iterated n times to sequentially compute the carry for each of the n bits. The MAJ circuit calculates and stores the carry generated by adding the carry from the previous digit and the two qubits of the same digit. The MAJ circuitis are employed because a carry occurs when two or more of these three qubits are one. After computing the carry for all n bits, the carry is removed and the computation result is stored. The unmajority and add (UMA) circuit is used to this computation.

The computation time for addition of n-bit numbers is $O(n)$. In Shor's algorithm, which factors n-bit composite numbers, addition is repeated $O\left(n^2\right)$ times, thereby resulting in a total computation time of $O\left(n^3\right)$ with $3n$ qubits.

In addition to the aforementioned computational methods, further improvements have been proposed, such as the implementation methods by Gidney [13] and Gidney and Ekerå [8]. Gidney [13] proposed an implementation method to reduce the computational cost of Toffoli gates in the MAJ and UMA circuits. A naive Toffoli gate requires seven T gates [26], necessitating a total of $14nT$ gates. Gidney reduced computation time by introducing an efficient Toffoli gate when the initial value of the carry storage was $|0\rangle$. In Gidney's method, the number of T gates required for the MAJ circuit is reduced to four by preparing n auxiliary qubits with an initial value of $|0\rangle$, and furthermore, no T gates are required for the UMA circuit. Furthermore, Gidney and Ekerå [8] proposed a method to parallelize the MAJ and UMA circuits, significantly reducing the circuit depth at

TABLE 1. Major computational cost of T or Toffoli gates in Shor's algorithm for factoring the n-bit number. In this table, the computational complexity related to the Toffoli gate is calculated only for cases where the Toffoli gate includes non-Clifford gates.

Method	Ripple-Carry [13]	Ripple-Carry [8]	Carry-Lookahead [16]	Fourier-basis [10]
#(Logical Qubits)	$4n$	$3n + 0.002n \log n$	$5n$	$2n$
#(T gates)	$36n^3$	Not evaluated	$129n^3$	$27n^3 \log n$
#(Toffoli gates)	$9n^3$	$0.3n^3 + 0.0005n^3 \log n$	$32.25n^3$	Not evaluated
T-depth	$18n^3$	Not evaluated	$36n^2 \log n$	$27n^2 \log n$
Toffoli-depth	$9n^3$	$500n^2 + n^2 \log n$	$18n^2 \log n$	Not evaluated
KQ_T	$72n^4$	Not evaluated	$180n^3 \log n$	$54n^3 \log n$
$\mathrm{KQ}_{\mathrm{Toffoli}}$	$36n^4$	$0.002n^3 (\log n)^2$	$90n^3 \log n$	Not evaluated

the expense of increasing the order of qubits. Notably, their research suggested parameters tailored for the factorization of 2048-bit composite numbers.

2) CARRY-LOOKAHEAD METHOD

The Carry-Lookahead method [15] is a technique that calculates all carries in advance. This method reduces computation time by storing information indicating the propagation of carries in auxiliary qubits as follows:

- $|p_{i,j}\rangle$: $|p_{i,j}\rangle$ indicates the carry from the i-th bit propagates to the j-th bit. If $p_{i,j} = 1$ and the carry from the i-th bit is one, then the carry for the j-th bit will be one.
- $|g_{i,j}\rangle$: $|g_{i,j}\rangle$ indicates the carry to the j-th bit when considering all information from the i-th bit onward. $|g_{0,j}\rangle$ indicates the original carry for the j-th bit.

Under $i < k < j$, the propagation status of the carry is calculated using AND operations, namely Toffoli gates, as $p_{i,j} = p_{i,k} \wedge p_{k,j}$ and $g_{i,j} = g_{j,k} \oplus (g_{i,k} \wedge p_{k,j})$. The computation is parallelized by the divide-and-conquer method, and the computation time for the addition of n-bit numbers becomes $O(\log n)$. In Shor's algorithm, which performs the factorization of n-bit composite numbers by repeating addition $O(n^2)$ times, the total computation time becomes $O(n^2 \log n)$.

Oonishi et al. [16] proposed an improved method for the aforementioned computation technique. In this method, the computational complexity was reduced by utilizing Gidney's Toffoli gate [13] instead of the naive Toffoli gate. Furthermore, the computational complexity was reduced by optimizing comparison circuits for controlled modular addition.

3) FOURIER-BASIS METHOD

The Fourier-basis method [17] employs the quantum Fourier transform (QFT) to independently calculate carries for each qubit. The process is

$$|x\rangle \to \frac{1}{2^n} \sum_{y=0}^{2^n-1} \exp\left(2\pi \mathrm{i} \frac{xy}{2^n}\right) |y\rangle \tag{1}$$

$$\to \frac{1}{2^n} \sum_{y=0}^{2^n-1} \exp\left(2\pi \mathrm{i} \frac{xy}{2^n}\right) \exp\left(2\pi \mathrm{i} \frac{ay}{2^n}\right) |y\rangle \tag{2}$$

$$\to |x + a\rangle . \tag{3}$$

The QFT and inverse QFT is performed in (1) and (3), respectively. In (2), addition is achieved by applying a phase shift $\exp(2\pi \mathrm{i} ay/2^n)$ in the Fourier domain. The phase shift can be decomposed into phase gates for each qubit of $|y\rangle$, which enables carries to be calculated independently. Consequently, the circuit depth for phase gates in the factorization algorithm of the Fourier-basis method is $O(n^2)$, and each phase gate requires $O(\log n)$ T gates [27]. Therefore, the computational time for factoring an n-bit number is $O(n^2 \log n)$.

A modified Shor's algorithm of Oonishi and Kunihiro [10] have been proposed to enhance the aforementioned computational technique. Their methods focus on efficiency gained by recognizing and omitting small phase gates having a negligible impact on the computational results.

E. RELATED WORKS ON EVALUATION METHOD FOR THE COMPUTATIONAL COMPLEXITY OF SHOR'S ALGORITHM

Numerous evaluations of the computational cost of Shor's algorithm have been proposed. Many of these focus on optimizing computational algorithms and evaluating complexity in the context of logical qubits. Moreover, various methods for complexity evaluation have been developed that specifically consider physical architectures and error corrections. This section discusses the complexity evaluation methods that are grounded in specific physical architectures.

First, we introduce key existing studies that have evaluated the computational complexity of Shor's algorithm on surface codes, specifically the works of Fowler et al. [28] and Jones et al. [9]. In Fowler et al.'s study [28], the implementation methods for each gate within surface codes were discussed, followed by an evaluation of the computational cost of factoring a 2000-bit composite number. This study evaluated the complexity of Shor's algorithm by Vedral et al. [29] using the Ripple-Carry addition proposed by Cuccaro et al. [12]. In Jones et al. [9] study, a complexity evaluation was performed based on the computational model described in Section II-B. This study assessed the computational complexity based on the Carry-Lookahead method proposed by Van Meter et al. [30], particularly evaluating the computation time under the constrained number of qubits.

In recent years, the evaluation have been conducted within architectures specifically designed for the Ripple-Carry method [8], [14]. Initially, the study by Gidney and Ekerå [8] diverged from that of Jones et al. [9] by placing

significant emphasis on measurement time referred to as reaction time, in their complexity evaluations. This study reduced computation time by employing an AutoCCZ [31] that incorporates delayed choice multiplex/demultiplex construction. In the Ripple-Carry method, Toffoli gates, which must typically be applied sequentially, are executed in parallel through the use of multiplexers, followed by sequential measurements. Moreover, the design of the quantum chip was conducted to ensure that the reaction time of these sequential measurements constitutes the primary component of the overall computation time. Gidney [14] further reduced the number of physical qubits by efficient construction and magic state cultivation [21]. This research applied Jones's magic state distillation [32] to generate CCZ gates after magic state cultivation, which enables the realization of Toffoli gates more efficiently than traditional magic state distillation.

In addition to the aforementioned studies, complexity evaluation of the Fourier-basis Shor's algorithm have also been conducted [33]. This research evaluated two variants of the Fourier-basis Shor's algorithm [34], [35], grounded in the surface codes proposed by Horsman et al. [36] and the error correction discussed by Folwer and Gidney [37]. Their study provided an explicit formulation of the computational cost, accompanied by an analysis based on the derived equations.

However, existing studies primarily focus on numerical evaluation, and the evaluation of theoretical computational complexity is insufficient. Furthermore, these studies tend to concentrate on a single addition method without comparing multiple approaches. In light of these considerations, this study aims to conduct a comprehensive complexity evaluation of surface codes and magic state distillation, while also comparing the computational performance of various methods under the same configurations of surface codes and magic state distillation.

III. EVALUATION METHOD FOR THE COMPUTATIONAL COMPLEXITY OF SHOR'S ALGORITHM CONSIDERING SURFACE CODES

We evaluate the computational complexity of Shor's algorithm considering surface codes, magic state distillation, and magic state cultivation. In Gidney's recent study [14], an computational model was utilized that optimally arranges physical qubits on a substrate tailored for 2048-bit input, employing different surface codes designated for compute, hot storage, and cold storage according to the frequency of qubit usage. Additionally, similar to Gidney and Ekerå [8], the error rate of physical gates was set at 0.1%, with a surface code cycle of 10^{-6} seconds and a measurement time of 10^{-5} seconds.

In comparison to the aforementioned evaluation method, our study employs a model that integrates three evaluation methods [9], [19], [22] to conduct a comprehensive complexity evaluation. To theoretically minimize KQ_p, the quantum computing model proposed by Jones et al. [9] is considered, supplemented by the other two methods. This model imposes no constraints on the connectivity of physical qubits; instead, it estimates the computation time for CNOT and H gates to be proportional to the code distance of the surface code. Our study uses the aforementioned model to achieve optimal scheduling based on the computation times for magic distillation and actual computation. The error rate of physical gates remains at 0.1% and the surface code cycle at 10^{-6} seconds, consistent with existing research. However, the model employed in our study estimates the computation time for CNOT and H gates to be longer in comparison to the measurement time, distinguishing it from models [8], [14] where measurement time is considered the primary computational cost. In consideration of the aforementioned points, this study will evaluate computational costs without utilizing AutoCCZ, directly employing quantum states generated through magic state distillation or magic state cultivation within the quantum circuit. In particular, regarding the quantum circuit proposed by Gidney and Ekerå [8], the evaluation of computational costs will differ from methods that consider measurement time as the primary computational duration, namely considering sequentially executed Toffoli gates.

In the ensuing discussion, magic state distillation will be considered to encompass magic state cultivation as well, unless explicitly stated otherwise, given that both serve the identical function of enhancing the fidelity of T or Toffoli gates. The specific evaluation procedure is then as follows:

1) We calculate the required fidelity for T or Toffoli gates and determine the computational cost of magic state distillation. The error rate of the magic state distillation part is defined as the product of the error rate of T or Toffoli gates output by each magic state distillation and the number of T or Toffoli gates.
2) Based on the computational cost of magic state distillation, compute the KQ and code distance of the surface code for the actual computation part. KQ is dependent on the computation time of the magic state distillation, making it challenging to calculate precisely. We vary the code distance d_a of the surface code in the actual computation part and seek the minimum d_a such that the sum of the error rate of the actual computation part derived from KQ and the error rate of the magic state distillation part calculated in Step 1) is less than one.
3) Considering both the actual computation part and the magic state distillation part, we calculate the number of physical qubits and computation time.

In this section, we first elucidate the method for evaluating computational cost in magic state distillation. Subsequently, we delineate the method for evaluating computational cost in surface codes. Finally, we present an approximate evaluation of the computational complexity. In this analysis, we demonstrate that the overhead in terms of both time and space complexities is at most poly(n).

A. EVALUATION METHOD FOR THE COMPUTATIONAL COST OF MAGIC STATE DISTILLATION

In this section, we present the method for evaluating the computational cost of magic state distillation, which utilizes multiple low-fidelity $|A\rangle = |0\rangle + \exp(\pi i/4)|1\rangle$ states to generate high-fidelity $|A\rangle$ states [7] or $|CCZ\rangle$ states [19]. When the error rate of each input logical qubit is p, if p is below a certain threshold, the fidelity of the $|A\rangle$ or $|CCZ\rangle$ states can be improved. Quantum states with the desired fidelity can be generated by repeatedly applying magic state distillation.

To calculate the computational cost associated with magic state distillation, it is necessary to determine the number of iterations of magic state distillation, the required number of input $|A\rangle$ states by considering the possibility of failure, and the code distance of the surface code utilized for each $|A\rangle$ state. We first introduce the two types of magic state distillation utilized in this study [7], [19] and the method for evaluating computational cost. For Litinski's magic state distillation [20] addressed in this paper, the numerical values from Table 1 of the original paper, which have already undergone a similar evaluation in this section, will be directly applied; thus, no further discussion will be undertaken in this section. Similarly, the computational cost illustrated in Figure 1 of existing research [21] will be directly applied to Gidney et al.'s magic state cultivation; thus, no further discussion will be undertaken in this section.

1) MAGIC STATE DISTILLATION UTILIZED IN THIS STUDY

We discuss the following two types of magic state distillation:

- One high-fidelity $|A\rangle$ state is generated from 15 $|A\rangle$ states [7]. The error rate is improved from p to $35p^3$; however, there is a $15p$ probability of failure.
- One high-fidelity $|CCZ\rangle$ state is generated from eight $|A\rangle$ states, and $|CCZ\rangle$ state subsequently converts it into two $|A\rangle$ states [19]. The error rate is improved from p to $28p^2$, with an $8p$ probability of failure. During the conversion to two $|A\rangle$ states, one previously generated $|A\rangle$ state is consumed to produce three $|A\rangle$ states. Consequently, the error accumulates, and the fidelity of the generated $|A\rangle$ states degrades with each conversion.

The magic state distillation generating $|CCZ\rangle$ state [19] is applied only in the final step of the iterative process. The magic state distillation generating $|A\rangle$ state [7] is utilized for the preceding steps for improving the fidelity of $|A\rangle$ states.

2) METHOD FOR EVALUATING THE COMPUTATIONAL COST OF MAGIC STATE DISTILLATION

This study uses the evaluation method of Gidney and Fowler [19]. However, when an alternative evaluation method from another reference is used, it will be stated explicitly.

First, we assume the error rate of physical gates p_g to be $p_g = 10^{-3}$. Under this assumption, the fidelity of the $|A\rangle$ states input into the magic state distillation process is 2×10^{-3}, as per the preparation method reported by Li [38]. Subsequently, for each magic state distillation, we calculate the following two types of errors and add them to determine the error rate from magic state distillation.

- **Distillation error:** These are residual errors after error correction has been performed via magic state distillation when $|A\rangle$ states with an error rate p are input. As mentioned earlier, the method proposed by Fowler et al. [7] results in residual errors with an error rate of $35p^3$, while the method proposed by Gidney and Fowler [19] results in residual errors with an error rate of $28p^2$.
- **Topological error:** These errors occur when physical quantum gates are applied within the surface code. In this study, based on the research by Fowler et al. [22], the topological error is calculated as $200d_m\left(100p_g\right)^{(d_m+1)/2}$, when the code distance of the surface code in magic state distillation is d_m. This value is provisionally determined as a common value to be used in both magic state distillations. Besides the coefficient part of 200, it is almost identical to the formula proposed by Fowler et al.

We determine if the achieved fidelity meets the required precision for T gates or Toffoli gates by calculating the error rate of the quantum state after magic state distillation in this manner. If the required precision is not met, we will prepend the magic state distillation method proposed by Fowler et al. [7], and subsequently recalculate the fidelity of the output quantum state. Once the required precision is achieved, the magic state distillation process is considered complete, and the computational cost is derived accordingly. Then, the number of input $|A\rangle$ states must be determined to ensure that the magic state distillation outputs the desired quantum state.

In the following discussion, we treat p_g as a constant. While the value of p_g may vary depending on the environment, it is unlikely to increase significantly with circuit size, and is expected to change within a constant factor range. The influence of the constant factor of p_g does not alter the computational complexity of m or d_m, affecting only the constant portion.

B. EVALUATION METHOD FOR THE COMPUTATIONAL COST OF SURFACE CODES IN THE ACTUAL COMPUTATION PART

Based on the existing study by Jones et al. [9], we determine the code distance d_a of the surface code for the actual computation part using KQ. We evaluate the computational cost in surface codes by utilizing the accuracy evaluation formula for logical qubits provided by Jones et al., which is given by

$$\varepsilon_L = C_1\left(C_2\frac{\varepsilon_V}{\varepsilon_{\text{thresh}}}\right)^{(d_a+1)/2}, \tag{4}$$

where ε_V represents the error rate of physical gates, and $\varepsilon_{\text{thresh}}$ represents the threshold error rate of physical gates,

both of which can be considered constants. C_1 and C_2 are also constants. Based on the existing research [9], we set $\varepsilon_V = 10^{-3}$, $\varepsilon_{\text{thresh}} = 9 \times 10^{-3}$, $C_1 = 0.13$, and $C_2 = 0.61$. Subsequently, we combine the error rate derived from magic state distillation with the error rate from the surface code, $\varepsilon_L \times \mathrm{KQ}$. We set the code distance d_a of the surface code to ensure that the overall error rate of the algorithm remains below one. When the code distance of the surface code is d_a, the required number of physical qubits is $2\,(d_a+1)^2$.

C. APPROXIMATE EVALUATION OF THE COMPUTATIONAL COMPLEXITY OF SHOR'S ALGORITHM

In this section, we provide an estimation of the computational cost associated with Shor's algorithm. We evaluate the computational complexity related to magic state distillation, and subsequently, we assess the computational complexity associated with surface codes. In both cases, the time and space complexities are poly(n). Therefore, even when considering these overheads, Shor's algorithm can be efficiently implemented on a quantum computer.

1) APPROXIMATE EVALUATION OF THE COMPUTATIONAL COMPLEXITY OF MAGIC STATE DISTILLATION

In this section, we provide an approximate evaluation of the computational complexity of magic state distillation. We estimate the number of repetitions, necessary number of logical qubits, and code distance of the surface code. In this estimation, the sum of distillation and topological errors should not exceed one. However, these errors are determined by discrete parameters, and therefore, minor adjustments to these parameters can ensure that the sum of these errors remains below one. Therefore, we evaluate these errors independently.

We discuss the number of repetitions. When the magic state distillation by Fowler et al. is repeated m times, the error rate of the T gate improves from p to

$$35^{(3^m-1)/2}p^{3^m} = \left(\sqrt{35}p\right)^{3^m}/\sqrt{35}. \tag{5}$$

If magic state distillation protocols [7], [19] are applied as the final step, the error rate is generally improved from p to ap^b, where a and b are positive real numbers. Then, the error rate of the T gate becomes $a\left(\left(\sqrt{35}p\right)^{3^m}/\sqrt{35}\right)^b$. In Shor's algorithm for factoring an n-bit number, the acceptable error rate for the T gate is $1/\mathrm{poly}(n)$, namely the reciprocal of the number of T gates. When the number of T gates is cn^k where c and k are positive real numbers, m satisfies

$$a\left(\left(\sqrt{35}p\right)^{3^m}/\sqrt{35}\right)^b \le \frac{1}{c}n^{-k} \tag{6}$$

$$\Leftrightarrow 3^m \ge \log_{\frac{1}{\sqrt{35}p}} \frac{(ac)^{1/b}\,n^{k/b}}{\sqrt{35}} \tag{7}$$

$$\Leftrightarrow 3^m \ge \frac{k}{b}\log_{\frac{1}{\sqrt{35}p}} n + C \left(\text{where } C = \log_{\frac{1}{\sqrt{35}p}} \frac{(ac)^{1/b}}{\sqrt{35}}\right) \tag{8}$$

$$\Leftrightarrow m \ge \log_3\left(\log_{\frac{1}{\sqrt{35}p}} n + \frac{bC}{k}\right) + \log_3 \frac{k}{b}, \tag{9}$$

and the right-hand side of (9) can be approximated by $\log_3 \log n$. Consequently, in Shor's algorithm, the number of repetitions of magic state distillation is $m = \Theta(\log\log n)$, and this remains nearly constant regardless of the specific implementation of Shor's algorithm.

Although this discussion focuses on the number of repetitions of magic state distillation, it is necessary to estimate the computational cost by considering the failure probability in each repetition. Let q denote the failure rate of magic state distillation. The expected number of trials required until a success is given by $1/(1-q)$. The expected number of initial input T gates required for obtaining the desired state is at most

$$\left(\frac{15}{1-15p}\right)^m \approx (\log n)^{\log_3(15/(1-15p))} \approx (\log n)^{2.5}. \tag{10}$$

Finally, we evaluate the computational complexity related to the code distance of the surface code. The error rate when the code distance of the surface code is d_m is given by $200d_m\,(0.1)^{(d_m+1)/2}$. We select the minimum d_m that satisfies

$$200d_m\,(0.1)^{(d_m+1)/2} \le \frac{1}{\mathrm{poly}(n)} \tag{11}$$

at the final magic state distillation. A higher error rate is allowed in input side, namely small code distance. Thus, if we set $d_m = 2k\log n/\log 10 - 1$, the left-hand side of (11) becomes $1/\mathrm{poly}(n)$ as derived from

$$\begin{aligned} &200\left(2k\frac{\log n}{\log 10} - 1\right)(0.1)^{k\log_{10} n} \\ &= \frac{400k}{\log 10}\frac{\log n}{n^k} - \frac{200}{n^k}. \end{aligned} \tag{12}$$

In this scenario, $d_m = \Theta(\log n)$. If we choose a smaller value,

$$d_m = \frac{2k\log n}{\log 10 \cdot \omega(1)} - 1, \tag{13}$$

the coefficient $200d_m$ on the left-hand side of (11) becomes $o(\log n)$, and $(0.1)^{(d_m+1)/2} = n^{-k/\omega(1)}$. The reciprocal of this number is $n^{k/\omega(1)}$, which is smaller than poly(n). Consequently, the left-hand side of (11) becomes greater than $1/\mathrm{poly}(n)$. Therefore, for the final magic state distillation, the code distance of the surface code is $d_m = \Theta(\log n)$, and the number of physical qubits required per logical qubit is $\Theta\left((\log n)^2\right)$. The other stage requires low precision and less physical qubits required per logical qubit.

From the above discussion, the number of logical qubits required for magic state distillation is $O\left((\log n)^{2.5}\right)$, and each logical qubit requires $O\left((\log n)^2\right)$ physical qubits. Consequently, the overhead in terms of physical qubits for magic state distillation, i.e., spatial complexity, is poly(n). Although magic state distillation is repeated $\log_3 \log n$ times, the number of magic state distillations executed in parallel is $O\left((\log n)^{2.5}\right)$. Each magic state distillation is composed of a constant number of Clifford gates, and the circuit depth

is $O(d_m) = O(\log n)$ [9]. Even if magic state distillation is not executed in parallel, the computational time required to generate a single $|A\rangle$ or $|CCZ\rangle$ state is at most

$$\log_3 \log n \times O\left((\log n)^{2.5}\right) \times O(\log n) = O\left((\log n)^4\right). \tag{14}$$

The number of T gates used in Shor's algorithm is poly(n), and the overhead in terms of time complexity attributed to the magic state distillation is at most poly(n). Therefore, the overhead in time and space complexities is at most poly(n).

2) APPROXIMATE EVALUATION OF THE COMPUTATIONAL COMPLEXITY OF SURFACE CODES FOR ACTUAL COMPUTATION PART

We evaluate the computational complexity related to the code distance of the surface code for the actual computation part. The number of logical qubits and time complexity required for magic state distillation per T gate are both poly(n). In addition, the number of gates in the actual computation part is poly(n). Therefore, even when considering the overhead caused by magic state distillation, the KQ of the actual computation part remains poly(n). Let us denote the KQ of the actual computation part as cn^k. From

$$C_1\left(C_2\frac{\varepsilon_V}{\varepsilon_{\text{thresh}}}\right)^{(d_a+1)/2} \leq \frac{1}{c}n^{-k} \tag{15}$$

$$\Leftrightarrow d_a \geq 2\log_{\frac{\varepsilon_{\text{thresh}}}{C_2\varepsilon_V}} C_1 c n^k - 1, \tag{16}$$

the code distance d_a of the surface code for the actual computation part is $\Theta(\log n)$. In this case, the number of physical qubits required per logical qubit is $\Theta\left((\log n)^2\right)$. Each magic state distillation comprises a constant number of Clifford gates, and the circuit depth is $\Theta(d_a) = \Theta(\log n)$. Therefore, the overhead in terms of both time and space complexities is at most poly(n).

IV. COMPARISON OF COMPUTATIONAL COSTS IN SHOR'S ALGORITHM

In this section, we evaluate and compare the computational complexity on the surface code for each method listed in Table 1. We address two types of Ripple-Carry methods, whereas only one type of Carry-Lookahead and Fourier-basis methods. The two types of Ripple-Carry Methods are treated as distinct implementations due to substantial improvements in the latter [8], which render it structurally different from the former. It is anticipated that similar enhancements could be applied to the Carry-Lookahead and Fourier-basis methods, although this remains a subject for future works.

The number of T or CCZ gates executed in parallel varies significantly across different methods, thereby leading to substantial differences in computational complexity. For instance, the Ripple-Carry method [13] tends to have a longer computation time because of its sequential execution of circuits, resulting in a lower number of T gates executed in parallel. In contrast, methods such as the Ripple-Carry [8], Carry-Lookahead [16], and Fourier-basis methods [10] enhance the parallelism of T gates, reducing computation time and KQ. Even within the same implementation of Shor's algorithm, variations in the number of T or CCZ gates executed in parallel can change both the time and spatial complexities of the algorithm. Therefore, discussions related to the number of T or CCZ gates executed in parallel are of paramount importance. The complexity analysis conducted in this study focuses on the major components derived from the T or CCZ gates. In this study, we evaluate the computational costs associated with the generation of T gates or CCZ gates using three different magic state distillation protocols [7], [19], [20] and magic state cultivation [21].

In this study, we numerically evaluate the computational costs associated with the two foundational magic state distillation protocols [7], [19] under the following three scenarios:

- When the number of qubits is minimized, i.e., when only one T or CCZ gate is generated at a time,
- When the computation time is minimized, and
- When the KQ_p is minimized.

Additionally, we provide a theoretical evaluation of the computational complexity under conditions that minimize KQ_p. Furthermore, this study numerically evaluates the computational cost of Litinski's magic state distillation [20] and magic state cultivation [21] under conditions that minimize KQ_p, allowing for a comparison with the other two methods.

TABLE 2. Major computational time of magic state distillation.

Method	Magic State		Actual Computation	
	CNOT	H	CNOT	H
T gate [7]	5	0	1	0
CCZ gate [19]	7	3	2	2
T gate [19]	11	5	1	0

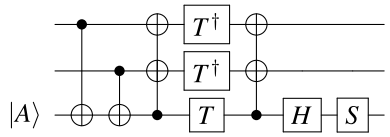

FIGURE 2. Gidney's relative-phase Toffoli gate [13]. Within the *T* gate, the circuit depth comprises of one CNOT and one *S* gate. Furthermore, the *S* gate has a circuit depth equivalent to two CNOT and two *H* gates. Integrating these components, the Toffoli gate has a circuit depth equivalent to nine CNOT and five *H* gates. When applying the *S* gate, an additional qubit is required to utilize the preserved quantum state.

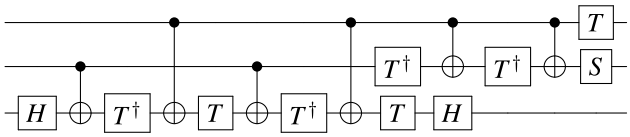

FIGURE 3. Standard Toffoli gate [26]. This Toffoli gate has a circuit depth equivalent to six CNOT gates, one *H* gate, and six *T* gates. When decomposed into CNOT and *H* gates, it corresponds to 24 CNOT and 13 *H* gates.

TABLE 3. Magic state distillation from Litinski's result [20].

Protocol	Output error	Qubits	Lattice steps
$(15\text{-to-}1)^6_{13,7,7} \times (8\text{-to-}CCZ)_{25,15,15}$	5.2×10^{-11}	47,000	60.0
$(15\text{-to-}1)^6_{11,5,5} \times (15\text{-to-}1)_{25,11,11}$	2.7×10^{-12}	30,700	82.5
$(15\text{-to-}1)^6_{13,5,5} \times (15\text{-to-}1)_{29,11,13}$	3.3×10^{-14}	39,100	97.5
$(15\text{-to-}1)^6_{17,7,7} \times (15\text{-to-}1)_{41,17,17}$	4.5×10^{-20}	73,400	128

TABLE 4. Magic state cultivation from Gidney et al.'s result [21].

ID	Output error	(Qubits)×(Lattice steps)
GSJ0	6×10^{-5}	4000
GSJ1	7×10^{-6}	5000
GSJ2	2×10^{-7}	20000
GSJ3	3×10^{-8}	30000
GSJ4	10^{-8}	40000
GSJ5	10^{-9}	100000

The following outlines the method for calculating computational costs. In the following calculation method, the following variables are introduced:

- n: Bit length of the composite number
- d_m: The code distance of the surface code for the magic state distillation part
- d_a: The code distance of the surface code for the actual computation part
- t_m: The minimum execution time of magic state distillation in terms of lattice steps
- t_g: The interval between the execution of T or CCZ gates in terms of lattice steps, which includes multiple T gates such as a Toffoli gate
- q_m: The number of physical qubits required for a magic state distillation
- q_a: The number of physical qubits required per logical qubit for the actual computation
- n_T: The number of $|A\rangle$ states generated simultaneously through magic state distillation
- n_{CCZ}: The number of $|CCZ\rangle$ states generated simultaneously through magic state distillation

An overview of the numerical computation method for calculating costs is presented as follows in steps 1) to 4). However, when applying this computational method to Litinski's magic state distillation [20] and Gidney et al.'s magic state cultivation [21], the utilization of pre-calculated computational costs will lead to the substitution of computational steps, as will be elaborated upon later.

1) We set the parameters from previous studies:
   - Quantum gate error rate: 10^{-3} [8]
   - Error rate of the inputted $|A\rangle$ state: $2 \cdot 10^{-3}$ [38]
   - Surface code cycle: 10^{-6} [s] [8]
   - Lattice steps for one H gate: $13\lceil d/8 \rceil$ [9]
   - Lattice steps for one CNOT gate: $13\lceil d/4 \rceil$ [9]
2) We calculate d_ms, namely the code distances of the surface code in each step of the magic state distillation.
   a) We calculate the required precision for the T gates or CCZ gates.
   b) In accordance with Section III-A2), we determine d_ms. d_ms are determined from the magic state distillation closest to the input. Specifically, it is assumed that the code distance of the subsequent stages of magic state distillation is sufficiently large, meaning the topological error is zero.
3) We search the minimal size of the surface code within the actual computation part from $d_a = 1$.
   a) We calculate t_m. As shown in Table 2, magic state distillation consists of computations within the distillation process and the transfer of the generated quantum states to the actual computation part. The computational cost of the actual computation part pertains to the cost required to convert the $|A\rangle$ and $|CCZ\rangle$ states obtained through magic state distillation into T gates (one CNOT gate) and CCZ gates (two CNOT and two H gates [19]), respectively. Therefore, t_m is calculated by summing the respective computation times. When the number of qubits is minimized, the value of t_m is calculated under the following conditions:
      - Taking into account the possibility of failure in magic state distillation, the required quantum states for the subsequent magic state distillation step are created one by one.
      - Once the quantum states are passed to the next magic state distillation step, the generation of new quantum states begins immediately.

      Otherwise, while generating T gates in the final stage, it is possible to parallelize the generation of T gates in the other stages. Consequently, the leading term of the circuit depth is determined by the final stage.
   b) We calculate t_g. When calculating t_g, it is assumed that each gate constituting the cycle incurs the following computational costs.
      - The S gate is decomposed into two CNOT gates and two H gates.
      - The T gate is decomposed into one CNOT gate and one S gate, taking into account the computational cost of applying the $|A\rangle$ state.
      - When applying the CCZ gate, two CNOT gates and two H gates are used to call the $|CCZ\rangle$ state. Additionally, two H gates are used to convert the CCZ gate into a Toffoli gate.

      The value of t_g varies depending on the specific adder circuit utilized.
      - In the Ripple-Carry method by Gidney [13], Toffoli gate depicted in Figure 2 is used. Moreover, two CNOT gates is required additionally.
      - In the Ripple-Carry method by Gidney and Ekerå [8], Toffoli gate depicted in Figure 3 is

used. Moreover, two CNOT gates is required additionally. Toffoli gate depicted in Figure 2 cannot be used because auxiliary qubits are not initialized to $|0\rangle$.
- In the Carry-Lookahead method [16], Toffoli gate depicted in Figure 2 is used.
- In the Fourier-basis method [10], the cycle time for a T gate consists of the time for one T gate and one H gate.

c) Depending on the optimization criteria, either n_T or n_{CCZ} is set as follows:
- In the case where the number of qubits is minimized, we set $n_T = 1$ or $n_{CCZ} = 1$. We assume that magic state distillation is not executed in parallel to minimize the number of qubits.
- In the case where the computation time is minimized, magic state distillation is executed in parallel to the maximum extent, ensuring that T or CCZ gates in the actual computation are executed immediately. If x represents the number of T or CCZ gates executed within the interval t_g, we set n_T or n_{CCZ} as xt_m/t_g.
- In the case where the KQ_p is minimized as existing studies by Oonishi et al. [16], we express KQ_p as a function of n_T or n_{CCZ} and calculate the value of n_T or n_{CCZ} that minimizes KQ_p.

d) We calculate the number of logical qubits, circuit depth, and KQ.

e) We calculate total error by adding the followings:
- Error in the actual computation part: $\varepsilon_L \times$ KQ (called "actual error" in the following discussion)
- Error in the all magic state distillation process (called "magic error" in the following discussion)

If total error is more than one, $d_a = d_a + 1$ and we go to a).

4) We calculate all error rate, the number of physical qubits, computation time and KQ_p.

We will now describe the modifications in the computational methods employed in Litinski's magic state distillation [20] and Gidney et al.'s magic state cultivation [21].

In Litinski's magic state distillation, we primarily use the magic state distillation method outputting $|CCZ\rangle$ state at the first row of Table 3 (extracted from Table 1 of the original paper [20]). However, if this results in an error rate exceeding one, we will employ a more precise magic state distillation. Since t_m and q_m are directly taken from the original paper, we bypass step 2) partially and proceed with the calculations.

In Gidney et al.'s magic state cultivation, a computational cost evaluation will be conducted based on Table 4 (extracted from Figure 1 of the original paper [21]). However, as noted in existing research, magic state cultivation alone fails to achieve the precision required for Shor's algorithm. Thus, we replace the first stage of magic state distillation with magic state cultivation, following the existing analytical method [14].

The aforementioned numerical calculations involve exploratory computations, such as exhaustive searches concerning d_a, which cannot be directly applied to theoretical complexity analysis. In the theoretical complexity analysis, we conduct analyses tailored to each specific scenario.

In this paper, we evaluate and compare the various methods listed in Table 1. The theoretical KQ_p obtained when minimizing KQ_p is as indicated in Table 5. For the factorization of a 2048-bit number, the numerically computed values for computation time, physical qubits, and KQ_p are as presented in Tables 6, 7, and 8, respectively. Numerical evaluations were also conducted for the cases of $n = 1024$ and $n = 4096$, yielding trends that are largely consistent with the discussions that follow. Other numerical experimental results are presented in Section IV-A and the Appendices A–C.

First, based on Table 5, we conduct a comparative analysis of computational complexity. As indicated in Table 5, within the same method, the order of computational complexity does not vary with the type of magic state distillation; only the coefficient part changes. The minimum value of KQ_p is achieved by the Carry-Lookahead method proposed by Oonishi et al. [16]; however, differences in KQ_p among the Ripple-Carry method by Gidney and Ekerå, Carry-Lookahead method, and Fourier-Basis method are at most $\log n$. This Ripple-Carry method by Gidney and Ekerå incorporates various innovations to significantly reduce computational complexity compared to Gidney's original Ripple-Carry method, which aims to shorten computation time. Some of these innovations, although incurring overhead, can be applied to the Carry-Lookahead method. As a future challenge, evaluating the computational complexity by applying Gidney and Ekerå's innovations to the Carry-Lookahead method is necessary to determine the optimal method.

Next, we will discuss the indicators for minimization, taking into account the computational costs that have been actually calculated. Table 6 indicates that conserving the number of qubits leads to impractical computation times. According to Table 8, this also results in an increase in KQ_p. Furthermore, from Table 7, there are instances where allocating qubits to the magic state distillation part reduces the total number of physical qubits in the Ripple-Carry method. In the Ripple-Carry method, the circuit scale required for magic state distillation is significantly smaller compared to that of the actual computation part. Consequently, by allocating a few qubits to the magic state distillation, the KQ, that is, the code distance of the surface code in the actual computation part, is reduced. Therefore, it is preferable to utilize a large number of qubits for minimizing computation time.

Finally, we will discuss the KQ_p associated with each method. As indicated in Table 8, the relative magnitudes

TABLE 5. KQ_p of Shor's algorithm for factoring the n-bit number when minimizing KQ_p.

Method	Magic state distillation	
	[7]	[19]
Ripple-Carry [13]	$27625.03n^4 (\log n)^3$	$12277.79n^4 (\log n)^3$
Ripple-Carry [8]	$0.80n^3 (\log n)^5$	$0.15n^3 (\log n)^5$
Carry-Lookahead [16]	$24819.36n^3 (\log n)^4$	$8632.82n^3 (\log n)^4$
Fourier-basis [10]	$1220733.91n^3 (\log n)^{4.5}$	$12386453.49n^3 (\log n)^{4.5}$

TABLE 6. The number of days of Shor's algorithm for factoring the 2048-bit number.

Adder	Minimize	Magic state distillation			
		[7]	[19]	[20]	[21]
Ripple-Carry [13]	#(Qubit)	1.57×10^4	3.22×10^3	–	–
	Time	1.26×10^3	5.58×10^2	–	–
	KQ_p	1.26×10^3	5.58×10^2	1.26×10^3	5.12×10^2
Ripple-Carry [8]	#(Qubit)	9.20×10^2	1.08×10^2	–	–
	Time	46.2	9.48	–	–
	KQ_p	46.2	9.48	8.69	8.69
Carry-Lookahead [16]	#(Qubit)	5.64×10^4	1.31×10^4	–	–
	Time	10.4	3.75	–	–
	KQ_p	7.39	1.92	8.19	1.50
Fourier-basis [10]	#(Qubit)	1.29×10^5	4.90×10^5	–	–
	Time	2.02×10^{-4}	2.02×10^{-4}	–	–
	KQ_p	8.24	24.7	3.99	8.25

TABLE 7. The number of physical qubits of Shor's algorithm for factoring the 2048-bit number.

Adder	Minimize	Magic state distillation			
		[7]	[19]	[20]	[21]
Ripple-Carry [13]	#(Qubit)	1.79×10^7	1.90×10^7	–	–
	Time	1.60×10^7	1.81×10^7	–	–
	KQ_p	1.60×10^7	1.81×10^7	1.58×10^7	1.38×10^7
Ripple-Carry [8]	#(Qubit)	1.12×10^7	1.27×10^7	–	–
	Time	1.00×10^7	1.16×10^7	–	–
	KQ_p	1.00×10^7	1.16×10^7	8.43×10^6	8.45×10^6
Carry-Lookahead [16]	#(Qubit)	2.37×10^7	2.37×10^7	–	–
	Time	1.56×10^9	1.87×10^9	–	–
	KQ_p	3.62×10^7	3.84×10^7	1.66×10^7	1.75×10^7
Fourier-basis [10]	#(Qubit)	8.96×10^6	9.61×10^6	–	–
	Time	4.14×10^{18}	8.08×10^{19}	–	–
	KQ_p	3.75×10^8	2.39×10^9	5.66×10^7	8.06×10^7

TABLE 8. KQ_p of Shor's algorithm for factoring the 2048-bit number.

Adder	Minimize	Magic state distillation			
		[7]	[19]	[20]	[21]
Ripple-Carry [13]	#(Qubit)	2.43×10^{22}	5.29×10^{21}	–	–
	Time	1.74×10^{21}	8.73×10^{20}	–	–
	KQ_p	1.74×10^{21}	8.73×10^{20}	1.71×10^{21}	6.12×10^{20}
Ripple-Carry [8]	#(Qubit)	8.88×10^{20}	1.19×10^{20}	–	–
	Time	4.00×10^{19}	9.47×10^{18}	–	–
	KQ_p	4.00×10^{19}	9.47×10^{18}	6.32×10^{18}	6.34×10^{18}
Carry-Lookahead [16]	#(Qubit)	1.16×10^{23}	2.68×10^{22}	–	–
	Time	1.40×10^{21}	6.04×10^{20}	–	–
	KQ_p	2.31×10^{19}	6.37×10^{18}	1.18×10^{19}	2.27×10^{18}
Fourier-basis [10]	#(Qubit)	1.00×10^{23}	4.07×10^{23}	–	–
	Time	7.22×10^{25}	1.41×10^{27}	–	–
	KQ_p	2.67×10^{20}	5.12×10^{21}	1.92×10^{19}	5.75×10^{19}

under each experimental condition in Table 8 corresponds to the relative magnitudes in Table 5. The Ripple-Carry [8], [13] and Carry-Lookahead methods [16] experienced a reduction in both computational time and KQ_p by using the magic state distillations of Gidney and Fowler [19]. However, in the Fourier-basis method [10], both computational time and KQ_p increased by using the magic state distillations of Gidney and Fowler. The Fourier-basis method requires converting

Toffoli gates into T gates, and the associated conversion loss leads to an increase in computational cost. In this study, the computational cost evaluation of Litinski's magic state distillation was couducted based on Table 3, which has limited candidates for magic state distillation that output the $|CCZ\rangle$ state, Thus, the methods proposed by Gidney's Ripple-Carry [13] and Carry-Lookahead [16] could not apply magic state distillation that produces the $|CCZ\rangle$ state, and the KQ_p values for these methods are elevated; however, if a magic state distillation capable of generating a more accurate $|CCZ\rangle$ state could be constructed, it is anticipated that the KQ_p would decrease. Moreover, with respect to the magic state cultivation proposed by Gidney et al., there was a reduction in KQ_p when compared to both Fowler et al.'s magics state distillation and the magic state distillation by Gidney and Fowler. However, when compared to Litinski's magic state distillation, the KQ_p values were found to be larger in the application of $|CCZ\rangle$ states to Gidney and Ekerå's Ripple-Carry method and the application of $|A\rangle$ states to Fourier-basis method. This discrepancy arises from the fact that, even if the first stage of magic state distillation is optimized, the computational complexity of the second stage remains substantial when compared to the jointly optimized two-stage magic state distillation by Litinski.

In this section, we evaluate the theoretical computational complexity minimizing KQ_p. We focus on the computation time of each gate and precisely evaluate computational complexity by assessing parameters such as n_T and n_{CCZ}. Furthermore, we present an example of numerical evaluations for Gidney's Ripple-Carry method. Other numerical evaluations are discussed in the Appendices A–C. In the evaluation, the displayed numerical values are rounded, and the actual calculations are performed using the values without rounding.

A. RIPPLE-CARRY METHOD PROPOSED BY GIDNEY

In this section, we evaluate the computational cost of the Ripple-Carry adder proposed by Gidney [13]. We assess the computational cost when using the three magic state distillation methods [7], [19], [20] and the magic state cultivation [21]. We demonstrate that this Ripple-Carry method performs better when using the magic state distillation proposed by Gidney and Fowler in a theoretical computational complexity. It is necessary to consider ancilla qubits related to the S gate. However, the number of parallel S gates is negligible compared to the qubits used in the actual computation part and not included in the following discussion. We present the computational process for the case where $n = 2048$.

1) USING THE MAGIC STATE DISTILLATION METHOD PROPOSED BY FOWLER et al.

a: EVALUATION OF THE COMPUTATIONAL COST OF MAGIC STATE DISTILLATION

Table 1 shows that the number of T gates is $36n^3 = 3.09 \times 10^{11}$. As indicated in Table 9, the code distances, determined greedily to be as small as possible from the input side, are 15 and 30, respectively. As presented in Table 9, for the first iteration of magic state distillation, even if the topological error is relatively large, the second iteration minimizes this error. Therefore, the code distance of the surface code in the first iteration of magic state distillation can be set to a smaller value. The aforementioned observations are also applicable to other cases. We then evaluate the computational cost.

TABLE 9. Progression of error rates for the Ripple-Carry method proposed by Gidney [13] when using the magic state distillation method proposed by Fowler et al. [7] where $n = 2048$.

Iteration	1	2
Code distance	15	30
Input error	2.00×10^{-3}	3.03×10^{-5}
Topological error	3.00×10^{-5}	1.90×10^{-12}
Distillation error	2.80×10^{-7}	9.72×10^{-13}
Output error	3.03×10^{-5}	2.87×10^{-12}

b: CASE WHERE THE NUMBER OF QUBITS IS MINIMIZED

From Table 9, the lattice steps required for magic state distillation per T gate are calculated as follows. To obtain the desired $|A\rangle$ state, the second stage of magic state distillation is performed $1/\left(1 - 15 \times 3.03 \times 10^{-5}\right)$ times. The first stage of magic state distillation is performed $15/\left(1 - 15 \times 2.00 \times 10^{-3}\right)$ times for the second stage. In these stages, each magic state distillation requires 15 $|A\rangle$ states and 15 logical qubits. While generating T gates in the second stage, it is possible to parallelize the generation of T gates in the first stage. The code distance of the surface code in the actual computation that ensures the sum of the error rates from the magic state distillation and actual computation parts remains below one is 32. Then,

$$\begin{aligned} t_m = &\frac{1}{1 - 15 \times 3.03 \times 10^{-5}} \times \left(\frac{15}{1 - 15 \times 2.00 \times 10^{-3}} - 1 \right) \\ &\times 5 \times 13 \times \left\lceil \frac{15}{4} \right\rceil \\ &+ \frac{1}{1 - 15 \times 3.03 \times 10^{-5}} \times 5 \times 13 \times \left\lceil \frac{30}{4} \right\rceil + 13 \times \left\lceil \frac{32}{4} \right\rceil \\ = &\ 4386.56. \end{aligned} \tag{17}$$

Because the number of T gates is 3.09×10^{11}, the circuit depth is $3.09 \times 10^{11} \times 4386.56 = 1.36 \times 10^{15}$.

In conclusion, the error rate, computation time, number of physical qubits, and KQ_p are as follows:

- Error rate: 96.2%
- Computation time:

$$1.36 \times 10^{15} \times 10^{-6}\ [\mathrm{s}] = 15700.12\ [\mathrm{days}]. \tag{18}$$

- Number of physical qubits:

$$\begin{aligned} &15 \times 2\,(15+1)^2 + 15 \times 2\,(30+1)^2 \\ &\quad + 4n \times 2\,(32+1)^2 = 1.79 \times 10^7. \end{aligned} \tag{19}$$

TABLE 10. Qubit minimized computational cost for the Ripple-Carry method proposed by Gidney [13] when using the magic state distillation method proposed by Fowler et al. [7].

n	1024	2048	4096
d_ms	14 and 31	15 and 30	16 and 32
Magic error	97.2%	88.7%	59.8%
d_a	31	32	33
Actual error	1.8%	7.5%	31.2%
Error rate	99.0%	96.2%	91.0%
Time (day)	1.96×10^3	1.57×10^4	1.26×10^5
#(physical qubits)	8.43×10^6	1.79×10^7	3.79×10^7
KQ_p	1.43×10^{21}	2.43×10^{22}	4.13×10^{23}

TABLE 11. Time minimized computational cost for the Ripple-Carry method proposed by Gidney [13] when using the magic state distillation method proposed by Fowler et al. [7].

n	1024	2048	4096
d_ms	14 and 31	15 and 30	16 and 32
Magic error	97.2%	88.7%	59.8%
d_a	29	30	31
Actual error	2.1%	8.8%	36.7%
Error rate	99.3%	97.5%	96.5%
Time (day)	1.57×10^2	1.26×10^3	1.01×10^4
#(physical qubits)	7.61×10^6	1.60×10^7	3.39×10^7
KQ_p	1.03×10^{20}	1.74×10^{21}	2.94×10^{22}

- KQ_p:

$$1.79 \times 10^7 \times 1.36 \times 10^{15} = 2.43 \times 10^{22}. \quad (20)$$

The computational costs for similar numerical calculations when $n = 1024$ and 4096 are shown in Table 10. As shown in Table 10, the code distance of the surface code remains almost unchanged despite variations in n. This is because the code distance is $\Theta(\log n)$.

c: CASE WHERE THE COMPUTATION TIME IS MINIMIZED

The calculation of t_m is conducted for the MAJ circuit, where T gates are invoked. The Toffoli gate depicted in Figure 2 is utilized, which consumes four $|A\rangle$ states. In our evaluation, the minimal code distance of the surface code in the actual computation should be 30. Then,

$$t_m = 5 \times 13 \times \left\lceil \frac{30}{4} \right\rceil + 13 \times \left\lceil \frac{30}{4} \right\rceil = 624, \quad (21)$$

$$t_g = 11 \times 13 \times \left\lceil \frac{30}{4} \right\rceil + 5 \times 13 \times \left\lceil \frac{30}{8} \right\rceil = 1404. \quad (22)$$

Therefore, the quantum circuit can be executed without any waiting time for T gate generation because of magic state distillation by setting the parallel execution of magic state distillation to generate $4t_m/t_g = 1.78\,T$ gates. In this context, to obtain the desired $|A\rangle$ state, the second stage of magic state distillation requires the input of

$$1.78 \times \frac{15}{1 - 15 \times 3.03 \times 10^{-5}} = 26.68 \quad (23)$$

$|A\rangle$ states and the same number of logical qubits. For the first stage of magic state distillation, the input requirement is

$$26.68 \times \frac{15}{1 - 15 \times 2.00 \times 10^{-3}} = 412.56 \quad (24)$$

$|A\rangle$ states and the same number of logical qubits. Here, four T gates are executed within the time interval t_g, and therefore, the circuit depth is given by $36n^3 \times 1404/4 = 1.09 \times 10^{14}$.

In conclusion, the error rate, computation time, number of physical qubits, and KQ_p, calculated similarly as when the number of qubits are optimized, are as follows:

- Error rate: 97.5%
- Computation time: 1256.28 [days]
- Number of physical qubits: 1.60×10^7
- KQ_p: 1.74×10^{21}

The computational costs for similar numerical calculations when $n = 1024$ and 4096 are shown in Table 11. Comparing Tables 10 and 11, optimizing computation time also results in a reduction in the number of physical qubits. In the current addition circuit, the physical qubit count required for magic state distillation is sufficiently small, leading to a direct correlation where the decrease in the code distance d_a of the actual computation results in a reduction in the number of qubits.

d: CASE FOR THE MINIMIZATION OF KQ_p

In the Ripple-Carry method proposed by Gidney, the computational segment utilizing T gates is confined to the MAJ circuit. Consequently, KQ_p is represented as

$$\mathrm{KQ}_p = (4nq_a + q_m n_T) \frac{36n^3}{n_T} t_m \quad (25)$$

$$= \frac{144n^4 q_a t_m}{n_T} + 36n^3 q_m t_m, \quad (26)$$

where t_m, q_a, and q_m are inherently determined by n_T. Once n_T is fixed, the code distance d_a of the surface code is determined, and using this d_a, t_m, q_a, and q_m are determined subsequently. Since d_a takes discrete values, even a slight change in n_T does not alter t_m, q_a, and q_m. Here, the KQ of the actual computation part is $144n^4 q_a t_m/n_T$. As n_T increases, both the KQ and the accompanying decrease in t_m reduce KQ. Thus, as n_T increases, the code distance d_a of the actual computation part, and consequently, t_m, q_a, and q_m decrease. Therefore, n_T that mininizes KQ_p closely aligns with the scenario where the computation time is minimized. At this point, $n_T = 4t_m/t_g$, and therefore, we proceed to evaluate the computational complexity based on this consideration.

We demonstrate that physical qubits used for magic state distillation are significantly fewer than those used for actual computation. The logical qubits required for the actual computation are $4n$, and $q_a = \Theta\left((\log n)^2\right)$. Therefore, the numbr of physical qubits required for the actual computation are $\Theta\left(n\,(\log n)^2\right)$. According to Section III-C1, the number of logical qubits required for a single instance of magic state distillation is $O\left((\log n)^{2.5}\right)$, and $q_m = O\left((\log n)^{4.5}\right)$. The number of parallel executions of magic state distillation is determined by t_g and t_m, which are determined by the code distances of the surface codes for the actual computation

and final magic state distillation, respectively. Because t_g is proportional to the code distance of the surface code for the actual computation, $t_g = \Theta(\log n)$. While generating T gates in the final stage of the magic state distillation, it is possible to parallelize the generation of T gates in the other stages, and $t_m = \Theta(\log n)$ based on the final stage of magic state distillation. Therefore, the number of parallel executions of magic state distillation n_T is $4t_m/t_g = \Theta(1)$, and the number of physical qubits allocated for magic state distillation is $O\left((\log n)^{4.5}\right)$. Thus, qubits required for magic state distillation can be neglected compared to those required for the actual computation.

Based on the above discussion, we evaluate the computational complexity by considering only the actual computation part. The code distance of the surface code for the actual computation is $\Theta(\log n)$, circuit depth is $\Theta\left(n^3 \log n\right)$, and KQ is $\Theta\left(n^4 \log n\right)$. Calculating the exact code distance under a certain constant k, we have

$$C_1\left(C_2\frac{\varepsilon_V}{\varepsilon_{\text{thresh}}}\right)^{(d_a+1)/2} \leq \frac{1}{kn^4\log n} \tag{27}$$

$$\Leftrightarrow \frac{d_a+1}{2} \geq \log_{\frac{\varepsilon_{\text{thresh}}}{C_2\varepsilon_V}} C_1 kn^4 \log n \tag{28}$$

$$\Leftrightarrow \frac{d_a+1}{2} \geq \frac{4}{\log \frac{\varepsilon_{\text{thresh}}}{C_2\varepsilon_V}} \log n + o(\log n). \tag{29}$$

Thus,

$$d_a \geq \frac{8}{\log \frac{\varepsilon_{\text{thresh}}}{C_2\varepsilon_V}} \log n \approx 2.06 \log n. \tag{30}$$

Therefore, when evaluating the computational complexity with KQ_p minimized, we obtain the following:

- Number of physical qubits:

$$4n \times 2 \times (2.06 \log n)^2 = 33.96n(\log n)^2. \tag{31}$$

- Circuit depth:

$$9n^3 \times 13 \times \left(\frac{11 \times 2.06 \log n}{4} + \frac{5 \times 2.06 \log n}{8}\right) = 813.54n^3 \log n. \tag{32}$$

- KQ_p: $27625.03n^4 (\log n)^3$

2) USING THE MAGIC STATE DISTILLATION METHOD PROPOSED BY GIDNEY AND FOWLER

In this section, we evaluate computational complexity similar to that in Section IV-A1. The magic state distillation of Gidney and Fowler [19] pertains to the *CCZ* gate, specifically the Toffoli gate, and therefore, we focus on the computational complexity evaluation related to the Toffoli gate.

a: EVALUATION OF THE COMPUTATIONAL COST OF MAGIC STATE DISTILLATION

Table 1 shows that the number of Toffoli gates is $9n^3 = 7.73 \times 10^{10}$. In this case, two iterations of magic state distillation are required to ensure that the error rate is less than one. Code distances determined greedily to be as small as possible from the input side are 19 and 31. The error rates for each magic state distillation are presented in Table 12. We then evaluate the computational cost.

TABLE 12. Progression of error rates for the Ripple-Carry method proposed by Gidney [13] when using the magic state distillation method proposed by Gidney and Fowler [19] where $n = 2048$.

Iteration	1	2
Code distance	19	31
Input error	2.00×10^{-3}	6.60×10^{-7}
Topological error	3.80×10^{-7}	6.20×10^{-13}
Distillation error	2.80×10^{-7}	1.22×10^{-11}
Output error	6.60×10^{-7}	1.28×10^{-11}

b: CASE WHERE THE NUMBER OF QUBITS IS MINIMIZED

As indicated in Table 12, lattice steps required for magic state distillation per $|CCZ\rangle$ state are calculated as follows. To obtain the desired $|A\rangle$ state, the second stage of magic state distillation is performed $1/\left(1 - 8 \times 6.60 \times 10^{-7}\right)$ times, and each requires 8 $|A\rangle$ states and 15 logical qubits. Next, the first stage of magic state distillation is performed $8/\left(1 - 15 \times 2.00 \times 10^{-3}\right)$ times for second stage, and each requires 15 $|A\rangle$ states and 15 logical qubits. The code distance of the surface code in the actual computation that ensures the sum of the error rates from the magic state distillation and actual computation part remains below one is 33. Consequently, calculating the overall circuit depth of the magic state distillation as lattice steps yields

$$\begin{aligned}&\frac{1}{1 - 8 \times 6.60 \times 10^{-7}} \times \left(\frac{8}{1 - 15 \times 2.00 \times 10^{-3}} - 1\right) \\ &\quad \times 5 \times 13 \times \left\lceil \frac{19}{4} \right\rceil \\ &\quad + \frac{1}{1 - 8 \times 6.60 \times 10^{-7}} \\ &\quad \times \left(7 \times 13 \times \left\lceil \frac{31}{4} \right\rceil + 3 \times 13 \times \left\lceil \frac{31}{8} \right\rceil\right) \\ &\quad + 2 \times 13 \times \left\lceil \frac{33}{4} \right\rceil + 2 \times 13 \times \left\lceil \frac{33}{8} \right\rceil = 3603.43. \end{aligned} \tag{33}$$

Therefore, given that the number of Toffoli gates is $9n^3$, the circuit depth is $9n^3 \times 3603.43 = 2.79 \times 10^{14}$.

In conclusion, similar to calculation in Section IV-A1, the error rate, computation time, number of physical qubits, and KQ_p are as follows:

- Error rate: 99.5%
- Computation time: 3224.29 [days]
- Number of physical qubits: 1.90×10^7
- KQ_p: 5.29×10^{21}

The computational costs for similar numerical calculations when $n = 1024$ and $n = 4096$ are shown in Table 13.

c: CASE WHERE THE COMPUTATION TIME IS MINIMIZED

The code distance of the surface code in the actual computation that ensures the sum of the error rates from the magic state distillation and actual computation part remains

TABLE 13. Qubit minimized computational cost for the Ripple-Carry method proposed by Gidney [13] when using the magic state distillation method proposed by Gidney and Fowler [19].

n	1024	2048	4096
d_ms	18 and 28	19 and 31	12, 20, and 31
Magic error	71.6%	99.1%	73.2%
d_a	28	33	34
Actual error	19.9%	0.4%	10.9%
Error rate	91.4%	99.5%	84.0%
Time (day)	3.84×10^2	3.22×10^3	1.69×10^5
#(physical qubits)	6.93×10^6	1.90×10^7	4.02×10^7
KQ_p	2.30×10^{20}	5.29×10^{21}	5.86×10^{23}

TABLE 14. Time minimized computational cost for the Ripple-Carry method proposed by Gidney [13] when using the magic state distillation method proposed by Gidney and Fowler [19].

n	1024	2048	4096
d_ms	18 and 28	19 and 31	12, 20, and 31
Magic error	71.6%	99.1%	73.2%
d_a	27	32	31
Actual error	12.7%	0.3%	16.3%
Error rate	84.3%	99.4%	89.5%
Time (day)	64.0	5.58×10^2	4.47×10^3
#(physical qubits)	6.64×10^6	1.81×10^7	3.50×10^7
KQ_p	3.67×10^{19}	8.73×10^{20}	1.35×10^{22}

below one is 32. In this context, $t_m = 1196$ and $t_g = 624$. By setting the parallel execution of magic state distillation to generate $t_m/t_g = 1.92$ Toffili gates, the quantum circuit can be executed without any waiting time for Toffoli gate generation attributed to magic state distillation. Similar to calculation in Section IV-A1, to obtain the desired $|CCZ\rangle$ state, the second stage of magic state distillation requires an input of 15.33 $|A\rangle$ states and $15.33 \times 15/8 = 28.75$ logical qubits. For the first stage of magic state distillation, the input requirement is $237.11|A\rangle$ states and the same number of logical qubits. Here, since one Toffoli gate is executed within the time interval t_g, the circuit depth is given by $9n^3 \times 624 = 4.82 \times 10^{13}$.

In conclusion, similar to calculation in Section IV-A1, the error rate, computation time, number of physical qubits, and KQ_p are as follows:

- Error rate: 99.4%
- Computation time: 558.35 [days]
- Number of physical qubits: 1.81×10^7
- KQ_p: 8.73×10^{20}.

The computational costs for similar numerical calculations when $n = 1024$ and $n = 4096$ are shown in Table 14. Similar to Section IV-A1, a comparison of Tables 13 and 14 reveals that optimizing computation time also results in a reduction in the number of physical qubits.

d: CASE FOR THE MINIMIZATION OF KQ_P

The order of computational complexity follows a similar argument to that in Section IV-A1. By focusing on the changes in the costs of Toffoli gates and calculating the computational costs in a manner analogous to Section IV-A1, the results are as follows where KQ_p is minimized:

- Number of physical qubits: $33.96n\,(\log n)^2$
- Circuit depth: $361.57n^3 \log n$
- KQ_p: $12277.79n^4\,(\log n)^3$

3) USING THE MAGIC STATE DISTILLATION METHOD PROPOSED BY LITINSKI

In this section, we examine the extent to which computational efficiency can be improved using magic state distillation as employed in the existing research by Litinski [20]. The experimental results are presented in Table 15. When comparing the results in Table 15 with those in Tables 11 and 14, it is observed that for 1024 bits, Table 15 yields the minimum KQ_p, whereas for 2048 bits and 4096 bits, the minimum KQ_p is found in Table 14. This dependency on the output format of the magic state distillation suggests that using the magic state distillation to produce $|CCZ\rangle$ states via the Ripple-Carry method results in a reduction in computational costs.

4) USING THE MAGIC STATE CULTIVATION METHOD PROPOSED BY GIDNEY et al.

In this section, we evaluate computational cost based on the utilization of Gidney et al.'s magic state cultivation. Building upon the results of the evaluations from the previous sections, this analysis evaluates the computational costs associated with applying Gidney and Fowler's magic state distillation to generate $|CCZ\rangle$ states following magic state cultivation. The results of the computational cost evaluation are presented in Table 16. Table 16 indicates that KQ_p is minimized when outputted error is small in magic state cultivation.

For the case of $n = 2048$ in Table 16, the error rate of the T gates outputted from the first stage of magic state distillation is at most 10^{-8}, while the distillation error of the generated $|CCZ\rangle$ states is 2.8×10^{-15}, which is sufficiently smaller than the values indicated in Table 12. Consequently, the code distance for magic state distillation is smaller compared to Table 12, and the topological error becomes dominant. Therefore, unless the error rate in the output of magic state cultivation decreases further, a substantial reduction in computational complexity is not anticipated.

B. RIPPLE-CARRY METHOD PROPOSED BY GIDNEY AND EKERÅ

In this section, we evaluate the computational cost of applying two different magic state distillation methods to the Ripple-Carry adder proposed by Gidney and Ekerå [8]. We assess the computational cost when using the magic state distillation method porposed by Fowler et al [7]. Next, we evaluate the computational cost when using the magic state distillation methods proposed by Gidney and Fowler [19]. In this section, we discuss the theoretical computational complexity under minimization of KQ_p. The numerical computational complexity will be addressed in the Appendix A.

TABLE 15. KQ_p minimized computational cost for the Ripple-Carry method proposed by Gidney [13] when using the magic state distillation method proposed by Litinski [20].

n	1024	2048	4096
Distilllation Type	$(15\text{-to-}1)^6_{13,7,7} \times (8\text{-to-}CCZ)_{25,15,15}$	$(15\text{-to-}1)^6_{11,5,5} \times (15\text{-to-}1)_{25,11,11}$	$(15\text{-to-}1)^6_{13,5,5} \times (15\text{-to-}1)_{29,11,13}$
Magic error	50.3%	83.5%	8.2%
d_a	26	30	31
Actual error	48.8%	8.8%	36.7%
Error rate	99.1%	92.3%	44.8%
Time (day)	64.0	1.26×10^3	1.01×10^4
#(physical qubits)	6.00×10^6	1.58×10^7	3.36×10^7
KQ_p	3.32×10^{19}	1.71×10^{21}	2.92×10^{22}

TABLE 16. KQ_p minimized computational cost for the Ripple-Carry method proposed by Gidney [13] when using the magic state cultivation method proposed by Gidney et al. [21]. d_m denotes the code distance in Gidney and Fowler's magic state distillation [19] applicable to the T gates that have undergone magic state cultivation.

n	1024	2048	4096
ID	GSJ3 and GSJ4	GSJ4 and GSJ5	GSJ5
d_m	27	29	31
Magic error	52.2%	44.9% (GSJ4), 44.8% (GSJ5)	38.3%
d_a	27	28	31
Actual error	12.7%	52.9%	16.3%
Error rate	64.9%	97.8%	54.7%
Time (day)	64.0	5.12×10^2	4.47×10^3
#(physical qubits)	6.47×10^6	1.38×10^7	3.36×10^7
KQ_p	3.57×10^{19}	6.12×10^{20}	1.30×10^{22}

TABLE 17. Qubit minimized computational cost for the Ripple-Carry method proposed by Gidney and Ekerå [8] when using the magic state distillation method proposed by Fowler et al. [7].

n	1024	2048	4096
d_ms	14 and 26	14 and 28	15 and 29
Magic error	43.3%	77.6%	99.7%
d_a	27	29	35
Actual error	17.2%	18.7%	0.095%
Error rate	60.5%	96.3%	99.8%
Time (day)	1.14×10^2	9.20×10^2	7.50×10^3
#(physical qubits)	4.88×10^6	1.12×10^7	3.21×10^7
KQ_p	4.82×10^{19}	8.88×10^{20}	2.08×10^{22}

TABLE 18. Time minimized computational cost for the Ripple-Carry method proposed by Gidney and Ekerå [8] when using the magic state distillation method proposed by Fowler et al. [7].

n	1024	2048	4096
d_ms	14 and 26	14 and 28	15 and 29
Magic error	43.3%	77.6%	99.7%
d_a	25	27	32
Actual error	25.5%	13.9%	0.15%
Error rate	68.8%	91.5%	99.8%
Time (day)	11.5	46.2	2.06×10^2
#(physical qubits)	4.34×10^6	1.00×10^7	2.77×10^7
KQ_p	4.32×10^{18}	4.00×10^{19}	4.93×10^{20}

TABLE 19. Qubit minimized computational cost for the Ripple-Carry method proposed by Gidney and Ekerå [8] when using the magic state distillation method proposed by Gidney and Fowler [19].

n	1024	2048	4096
d_ms	16 and 28	17 and 31	19 and 28
Magic error	99.7%	99.7%	62.9%
d_a	29	31	29
Actual error	0.12%	0.15%	34.3%
Error rate	99.9%	99.8%	97.2%
Time (day)	11.3	1.08×10^2	8.42×10^2
#(physical qubits)	5.60×10^6	1.27×10^7	2.23×10^7
KQ_p	5.48×10^{18}	1.19×10^{20}	1.63×10^{21}

TABLE 20. Time minimized computational cost for the Ripple-Carry method proposed by Gidney and Ekerå [8] when using the magic state distillation method proposed by Gidney and Fowler [19].

n	1024	2048	4096
d_ms	16 and 28	17 and 31	19 and 28
Magic error	99.7%	99.7%	62.9%
d_a	28	29	27
Actual error	0.085%	0.19%	20.9%
Error rate	99.8%	99.9%	83.8%
Time (day)	2.17	9.48	34.8
#(physical qubits)	5.38×10^6	1.16×10^7	2.04×10^7
KQ_p	1.01×10^{18}	9.47×10^{18}	6.12×10^{19}

This method, compared to Gidney's method, achieves a significant reduction in computation time by adding a few extra qubits as follows:

- Multiple inputs are provided as superpositions of quantum states, reducing the number of adder circuits required. In the following configuration, many inputs are combined, and the depth becomes almost $1/25$.
- The adder circuits are executed in a divided manner, which reduces the overall circuit depth of Shor's algorithm. In the following configuration, $n/1024 = 2$ adder circuits are executed in parallel.
- The coset representation by Zalka [39] is utilized to reduce the computational cost of the modular arithmetic.

Due to these optimizations, the overall computation time and KQ_p of Shor's algorithm are reduced while the number of

TABLE 21. KQ_p minimized computational cost for the Ripple-Carry method proposed by Gidney and Ekerå [8] when using the magic state distillation method proposed by Litinski [20].

n	1024	2048	4096
Distilllation Type	$(15\text{-to-}1)^6_{13,7,7} \times (8\text{-to-}CCZ)_{25,15,15}$	$(15\text{-to-}1)^6_{13,7,7} \times (8\text{-to-}CCZ)_{25,15,15}$	$(15\text{-to-}1)^6_{11,5,5} \times (15\text{-to-}1)_{25,11,11}$
Magic error	1.70%	13.6%	39.7%
d_a	23	25	28
Actual error	57.9%	38.5%	28.9%
Error rate	59.6%	52.1%	68.7%
Time (day)	1.77	8.69	1.85×10^2
#(physical qubits)	3.59×10^6	8.43×10^6	2.09×10^7
KQ_p	5.51×10^{17}	6.32×10^{18}	3.34×10^{20}

TABLE 22. KQ_p minimized computational cost for the Ripple-Carry method proposed by Gidney and Ekerå [8] when using the magic state cultivation method proposed by Gidney et al. [21]. d_m denotes the code distance in Gidney and Fowler's magic state distillation [19] applicable to the T gates that have undergone magic state cultivation.

n	1024	2048	4096
ID	GSJ2–5	GSJ3 and GSJ4	GSJ5
d_m	24	26	28
Magic error	49.7%	43.2%	37.2%
d_a	24	25	27
Actual error	15.1%	38.5%	20.9%
Error rate	64.8%	81.6%	58.1%
Time (day)	1.77	8.69	34.8
#(physical qubits)	3.90×10^6	8.45×10^6	1.96×10^7
KQ_p	5.98×10^{17}	6.34×10^{18}	5.90×10^{19}

TABLE 23. Qubit minimized computational cost for the Carry-Lookahead method proposed by Oonishi et al. [16] when using the magic state distillation method proposed by Fowler et al. [7].

n	1024	2048	4096
d_ms	15 and 29	16 and 31	16 and 33
Magic error	93.8%	73.1%	93.4%
d_a	32	33	36
Actual error	2.09%	8.72%	2.50%
Error rate	95.9%	81.8%	95.9%
Time (day)	7.03×10^3	5.64×10^4	4.58×10^5
#(physical qubits)	1.12×10^7	2.37×10^7	5.61×10^7
KQ_p	6.80×10^{21}	1.16×10^{23}	2.22×10^{24}

TABLE 24. Time minimized computational cost for the Carry-Lookahead method proposed by Oonishi et al. [16] when using the magic state distillation method proposed by Fowler et al. [7].

n	1024	2048	4096
d_ms	15 and 29	16 and 31	16 and 33
Magic error	93.8%	73.1%	93.4%
d_a	26	27	30
Actual error	4.95%	11.3%	1.93%
Error rate	98.7%	84.4%	95.3%
Time (day)	2.36	10.4	50.2
#(physical qubits)	6.93×10^8	1.56×10^9	3.27×10^9
KQ_p	1.41×10^{20}	1.40×10^{21}	1.42×10^{22}

physical qubits increases slightly. We evaluate the specific computational costs of Shor's algorithm proposed by Gidney and Ekerå. In this algorithm, various parameters are utilized. We employed the parameters used in their methods to evaluate the time and space complexities. However, in practice, these parameters need to be optimized according to the value of n. Considering such optimization remains a task for future work.

TABLE 25. KQ_p minimized computational cost for the Carry-Lookahead method proposed by Oonishi et al. [16] when using the magic state distillation method proposed by Fowler et al. [7].

n	1024	2048	4096
d_ms	15 and 29	16 and 31	16 and 33
Magic error	93.8%	73.1%	93.4%
d_a	25	26	29
Actual error	5.99%	14.1%	2.46%
Error rate	99.8%	87.2%	95.9%
Time (day)	1.64	7.39	36.5
#(physical qubits)	1.70×10^7	3.62×10^7	7.81×10^7
KQ_p	2.41×10^{18}	2.31×10^{19}	2.46×10^{20}

TABLE 26. Qubit minimized computational cost for the Carry-Lookahead method proposed by Oonishi et al. [16] when using the magic state distillation method proposed by Gidney and Fowler [19].

n	1024	2048	4096
d_ms	19 and 29	21 and 31	12, 21, and 32
Magic error	62.3%	97.6%	65.2%
d_a	29	33	35
Actual error	24.0%	2.02%	12.9%
Error rate	86.3%	99.6%	78.2%
Time (day)	1.42×10^3	1.31×10^4	6.16×10^5
#(physical qubits)	9.25×10^6	2.37×10^7	5.31×10^7
KQ_p	1.14×10^{21}	2.68×10^{22}	2.83×10^{24}

TABLE 27. Time minimized computational cost for the Carry-Lookahead method proposed by Oonishi et al. [16] when using the magic state distillation method proposed by Gidney and Fowler [19].

n	1024	2048	4096
d_ms	19 and 29	21 and 31	12, 21, and 32
Magic error	62.3%	97.6%	65.2%
d_a	23	27	27
Actual error	36.8%	1.86%	16.2%
Error rate	99.2%	99.5%	81.5%
Time (day)	0.68	3.75	16.4
#(physical qubits)	9.30×10^8	1.87×10^9	1.93×10^{10}
KQ_p	5.48×10^{19}	6.04×10^{20}	2.73×10^{22}

1) USING THE MAGIC STATE DISTILLATION METHOD PROPOSED BY FOWLER et al.

The number of T gates is

$$7\left(0.3n^3 + 0.0005n^3 \log n\right)$$

TABLE 28. KQ_p minimized computational cost for the Carry-Lookahead method proposed by Oonishi et al. [16] when using the magic state distillation method proposed by Gidney and Fowler [19].

n	1024	2048	4096
d_ms	19 and 29	21 and 31	12, 21, and 32
Magic error	62.3%	97.6%	65.2%
d_a	23	27	26
Actual error	17.0%	0.95%	34.0%
Error rate	79.3%	98.6%	99.2%
Time (day)	0.31	1.92	8.91
#(physical qubits)	1.69×10^7	3.84×10^7	2.48×10^8
KQ_p	4.58×10^{17}	6.37×10^{18}	1.91×10^{20}

$$= 2.1n^3 + 0.0035n^3 \log n, \tag{34}$$

and KQ_p is represented as

$$KQ_p \approx \frac{f(n, t_m, q_a)}{n_T} + g(n, t_m, q_m), \tag{35}$$

where $f(n, t_m, q_a)$ and $g(n, t_m, q_m)$ are functions. Similar to the discussion in Section IV-A1, KQ_p is minimized by maximizing the number of qubits used for magic state distillation. Therefore, the n_T that minimizes KQ_p closely aligns with the scenario where the computation time is minimized.

Based on the above considerations, we evaluate the computational complexity when KQ_p is minimized. In this method, the adder circuit is decomposed into adder circuits of 1024 qubits each, enabling the parallel execution of $n/1024$ adder circuits. In each quantum circuit, $7n/1024$ T gates are executed in parallel during time t_g. Therefore, $n_T = 7nt_m/1024t_g$. Considering that n_T has the aforementioned value, we proceed to discuss the computational complexity.

First, we discuss the actual computation part. Here, similar to the discussion in Section IV-A1, t_m and t_g are $\Theta(\log n)$. Therefore, the number of logical qubits in the actual computation part is $\Theta(n \log n)$, and the circuit depth is

$$\frac{2.1n^3 + 0.0035n^3 \log n}{n_T} t_m = \Theta\left(n^2 (\log n)^2\right), \tag{36}$$

which results in a KQ of $\Theta\left(n^3 (\log n)^3\right)$. Therefore, similar to (27)–(30), $d_a \geq 1.55 \log n$. Moreover, $q_a \approx 4.78 (\log n)^2$, and $t_g \approx 163.21 \log n$.

The number of T gates is $2.1n^3 + 0.0035n^3 \log n$, and the code distance of the surface code in the magic state distillation part is determined based on this number of T gates. However, there is a distillation error originating from the magic state distillation in addition to the topological error. The distillation error decreases at a constant rate regardless of d_m, and therefore, even if the code distance d_m is increased, there is a limit to the reduction in the error rate. To reduce computational complexity, the value of d_m should be set as small as possible such that the topological and distillation errors are almost the same. Based on the above considerations, we estimate the size of the surface code for each magic state distillation. The distillation error decreases cubically with the number of distillations. Here, considering that the topological error is approximately $(0.1)^{d_m/2}$, if we set the code distance of the surface code for the j-th round of magic state distillation to

$$d_m = 2 \times 3^j \log_{0.1}\left(0.002 \times \sqrt{35}\right) \approx 3.85 \times 3^j, \tag{37}$$

the topological and distillation errors are almost the same. Invoking the $|A\rangle$ state in the actual computation part requires one CNOT gate, $t_m \approx 67.65 \log n$. Furthermore,

$$q_m \approx \sum_{j=1}^{\log_3 \log n} 2 \times \left(\left(3.85 \times 3^j\right)^2 \times \prod_{k=j}^{\log_3 \log n} \left(\frac{15}{1 - \frac{15}{\sqrt{35}}\left(\sqrt{35}p\right)^{3^{k-1}}}\right)\right) \tag{38}$$

$$\approx 30 \times 3.85^2 \sum_{j=1}^{\log_3 \log n} \left(9^j \cdot 15^{\log_3 \log n - j}\right) \tag{39}$$

$$= 30 \times 3.85^2 \times 15^{\log_3 \log n} \sum_{j=1}^{\log_3 \log n} \left(\frac{3}{5}\right)^j \tag{40}$$

$$\approx 45 \times 3.85^2 \times 15^{\log_3 \log n} \tag{41}$$

$$\approx 668.35 (\log n)^{2.5}. \tag{42}$$

Based on the above considerations, we evaluate computational complexity. For determining the number of physical qubits, the actual computation part requires

$$(3n + 0.002n \log n) q_a = 0.0096n (\log n)^3, \tag{43}$$

and the magic state distillation part requires

$$q_m n_T = 1.89n (\log n)^{2.5}. \tag{44}$$

Therefore, the leading term is $0.0096n (\log n)^3$ from the actual computation part. Furthermore, the circuit depth is

$$\frac{2.1n^3 + 0.0035n^3 \log n}{n_T} t_m = 83.56n^2 (\log n)^2, \tag{45}$$

and KQ_p is $0.80n^3 (\log n)^5$.

2) USING THE MAGIC STATE DISTILLATION METHOD PROPOSED BY GIDNEY AND FOWLER

In this section, we evaluate computational complexity evaluation similar to that in Section IV-A2. We focus on evaluating the computational complexity related to the Toffoli gate.

We discuss n_{CCZ}, which minimizes KQ_p. Here, the order of computational complexity follows a similar argument to that in Section IV-B1. Then, q_a and q_m are almost the same as that in Section IV-B1, $t_m \approx 121.53 \log n$, $t_g \approx 30.13 \log n$, and $n_{CCZ} = nt_m/1024t_g \approx 0.0039n$.

When focusing on the coefficients of the leading terms where KQ_p is minimized, similar to Section IV-B1, the results are as follows:

TABLE 29. KQ_p minimized computational cost for the Carry-Lookahead method proposed by Oonishi et al. [16] when using the magic state distillation method proposed by Litinski [20].

n	1024	2048	4096
Distilllation Type	$(15\text{-to-}1)^6_{11,5,5} \times (15\text{-to-}1)_{25,11,11}$	$(15\text{-to-}1)^6_{13,5,5} \times (15\text{-to-}1)_{29,11,13}$	$(15\text{-to-}1)^6_{13,5,5} \times (15\text{-to-}1)_{29,11,13}$
Magic error	37.4%	3.66%	29.3%
d_a	24	25	27
Actual error	21.4%	60.0%	35.9%
Error rate	58.8%	63.6%	65.2%
Time (day)	1.52	8.19	36.2
#(physical qubits)	7.60×10^6	1.66×10^7	3.74×10^7
KQ_p	9.99×10^{17}	1.18×10^{19}	1.17×10^{20}

TABLE 30. KQ_p minimized computational cost for the Carry-Lookahead method proposed by Oonishi et al. [16] when using the magic state cultivation method proposed by Gidney et al. [21]. d_m denotes the code distance in Gidney and Fowler's magic state distillation [19] applicable to the T gates that have undergone magic state cultivation.

n	1024	2048	4096
ID	GSJ5	GSJ5	GSJ5
d_m	28	30	32
Magic error	61.3%	52.6%	44.9%
d_a	23	24	26
Actual error	17.1%	42.1%	33.9%
Error rate	78.4%	94.6%	78.8%
Time (day)	0.32	1.50	8.89
#(physical qubits)	8.04×10^6	1.75×10^7	3.91×10^7
KQ_p	2.20×10^{17}	2.27×10^{18}	3.00×10^{19}

TABLE 31. Qubit minimized computational cost for the Fourier-basis method proposed by Oonishi and Kunihiro [10] when using the magic state distillation method proposed by Fowler et al. [7].

n	1024	2048	4096
d_ms	15 and 30	16 and 32	16 and 36
Magic error	83.2%	61.7%	92.7%
d_a	31	32	36
Actual error	6.72%	30.8%	2.51%
Error rate	89.9%	92.4%	95.2%
Time (day)	1.47×10^4	1.29×10^5	1.15×10^6
#(physical qubits)	4.23×10^6	8.96×10^6	2.25×10^7
KQ_p	5.38×10^{21}	1.00×10^{23}	2.23×10^{24}

TABLE 32. Time minimized computational cost for the Fourier-basis method proposed by Oonishi and Kunihiro [10] when using the magic state distillation method proposed by Fowler et al. [7].

n	1024	2048	4096
d_ms	17 and 37	18 and 40	20 and 42
Magic error	98.3%	87.2%	60.9%
d_a	34	35	37
Actual error	1.62%	8.18%	11.3%
Error rate	99.96%	95.4%	72.2%
Time(day)	1.67×10^{-4}	2.02×10^{-4}	2.57×10^{-4}
Time (second)	14.4	17.4	22.2
#(physical qubits)	2.30×10^{17}	4.14×10^{18}	8.11×10^{19}
KQ_p	3.32×10^{24}	7.22×10^{25}	1.80×10^{27}

- Number of physical qubits: $0.0096n\,(\log n)^3$
- Circuit depth: $15.43n^2\,(\log n)^2$
- KQ_p: $0.15n^3\,(\log n)^5$

C. CARRY-LOOKAHEAD METHOD

In this section, we evaluate the computational cost of applying two different magic state distillation methods to the Carry-Lookahead adder proposed by Oonishi et al. [16]. Similar to Section IV-B, we assess the computational cost when using two magic state distillation methods [7], [19], and we discuss the minimization of KQ_p. The numerical computational complexity will be addressed in the Appendix B.

The Carry-Lookahead adder discussed in this section can be accelerated using the method described in Section IV-B. However, we focus on discussing the baseline method.

TABLE 33. KQ_p minimized computational cost for the Fourier-basis method proposed by Oonishi and Kunihiro [10] when using the magic state distillation method proposed by Fowler et al. [7].

n	1024	2048	4096
d_ms	15 and 30	16 and 32	16 and 36
Magic error	83.2%	61.7%	92.7%
d_a	24	26	28
Actual error	16.5%	10.1%	6.60%
Error rate	99.7%	71.8%	99.3%
Time (day)	2.31	8.24	27.3
#(physical qubits)	1.31×10^8	3.75×10^8	1.15×10^9
KQ_p	2.62×10^{19}	2.67×10^{20}	2.71×10^{21}

TABLE 34. Qubit minimized computational cost for the Fourier-basis method proposed by Oonishi and Kunihiro [10] when using the magic state distillation method proposed by Gidney and Fowler [19].

n	1024	2048	4096
d_ms	15, 32, and 54	16, 33, and 57	17, 35, and 60
Magic error	88.3%	69.1%	73.6%
d_a	32	33	36
Actual error	6.48%	30.3%	11.2%
Error rate	94.8%	99.4%	84.8%
Time (day)	5.46×10^4	4.90×10^5	5.13×10^6
#(physical qubits)	4.59×10^6	9.61×10^6	2.26×10^7
KQ_p	2.17×10^{22}	4.07×10^{23}	1.00×10^{25}

TABLE 35. Time minimized computational cost for the Fourier-basis method proposed by Oonishi and Kunihiro [10] when using the magic state distillation method proposed by Gidney and Fowler [19].

n	1024	2048	4096
d_ms	18, 37, and 66	19, 40, and 70	20, 44, and 76
Magic error	93.7%	96.2%	89.3%
d_a	33	36	38
Actual error	6.23%	2.13%	2.94%
Error rate	99.95%	98.3%	92.3%
Time(day)	1.67×10^{-4}	2.02×10^{-4}	2.57×10^{-4}
Time (second)	14.4	17.4	22.2
#(physical qubits)	4.31×10^{18}	8.08×10^{19}	1.45×10^{21}
KQ_p	6.22×10^{25}	1.41×10^{27}	3.23×10^{28}

TABLE 36. KQ_p minimized computational cost for the Fourier-basis method proposed by Oonishi and Kunihiro [10] when using the magic state distillation method proposed by Gidney and Fowler [19].

n	1024	2048	4096
d_ms	15, 32, and 54	16, 33, and 57	17, 35, and 60
Magic error	88.3%	69.1%	73.6%
d_a	25	26	28
Actual error	9.18%	22.7%	13.4%
Error rate	97.5%	91.9%	87.0%
Time (day)	7.47	24.7	78.0
#(physical qubits)	7.48×10^8	2.39×10^9	7.42×10^9
KQ_p	4.83×10^{20}	5.12×10^{21}	5.00×10^{22}

1) USING THE MAGIC STATE DISTILLATION METHOD PROPOSED BY FOWLER et al.

As discussed by Oonishi et al. [16], the Carry-Lookahead adder exhibits significant differences in the number of concurrently executed T gates. To minimize KQ_p, calculating n_T considering these differences is necessary, similar to existing research. The number of T gates that can be executed in parallel during time t_g is $n_T t_g/t_m$. In sections where more T gates are executed in parallel, a waiting is introduced because of magic state distillation. Here, the total number of T gates executed without any waiting time is

$$6 \sum_{j=0}^{\lfloor \log(n_T t_g/t_m) \rfloor} 2^j \approx \frac{12 n_T t_g}{t_m} \tag{46}$$

in each controlled modular adder, and the circuit depth is

$$\begin{aligned} &3n^2 \left(\frac{43n - 12 n_T t_g/t_m}{n_T} t_m + 6\, t_g \log \left(n_T \frac{t_g}{t_m} \right) \right) \\ &= 3n^2 \left(\frac{43n}{n_T} t_m - 12 t_g + 6\, t_g \log \left(n_T \frac{t_g}{t_m} \right) \right). \end{aligned} \tag{47}$$

In principle, it is necessary to consider the computational cost of ancillary qubits related to the S gate in addition to the actual computation and magic state distillation parts. In this case, it is necessary to apply one S gate to each of the maximum n_T parallelly executed T gates. Consequently, KQ_p is represented as

$$\begin{aligned} KQ_p = &(5nq_a + (q_m + q_a)\, n_T) \\ &\times 3n^2 \left(\frac{43n}{n_T} t_m - 12 t_g + 6\, t_g \log \left(n_T \frac{t_g}{t_m} \right) \right). \end{aligned} \tag{48}$$

However, in (48), parameters t_g, t_m, q_a, and q_m, which inherently depend on n_T, are included, thereby making optimization challenging. Once n_T is fixed, the code distance d_m and d_a of the surface code is determined, and using d_m and d_a, t_g, t_m, q_a, and q_m are subsequently determined. Since d_m and d_a take discrete values, even a slight change in n_T does not alter t_g, t_m, q_a, and q_m. Therefore, assuming that t_g, t_m, q_a, and q_m are constants independent of n_T, the value of n_T that minimizes KQ_p is almost

$$n_T = n \sqrt{\frac{215 t_m q_a}{6 t_g (q_m + q_a) \log n}}. \tag{49}$$

By focusing on variables within n_T, we calculate n_T.

We discuss the values of t_g, t_m, q_a, and q_m. Similar to the discussion in Sections IV-A1 and IV-B1, t_g and t_m are $\Theta(\log n)$, q_a is $\Theta\left((\log n)^2\right)$, and q_m is $\Theta\left((\log n)^{2.5}\right)$. Therefore, $n_T = \Theta\left(n\,(\log n)^{-0.75}\right)$. These values are calculated based on KQ and d_a derived from KQ. The ancilla qubits for the S gates can be neglected because $n_T = o(n)$, and therefore, the number of logical qubits is $5n$ in the actual computation part. From (47), the circuit depth is $\Theta\left(n^2\,(\log n)^2\right)$, which results in a KQ of $\Theta\left(n^3\,(\log n)^2\right)$. Therefore, since the degree of n is 3, we proceed in the same manner as in Section IV-B1, $d_a \geq 1.55 \log n$. Thus, similar to Section IV-B1, $t_m \approx 67.65 \log n$, $q_a \approx 4.78\,(\log n)^2$, and $q_m \approx 668.35\,(\log n)^{2.5}$. Furthermore, considering t_g as the execution interval of the T gates within the Toffoli gate as illustrated in Figure 2, $t_g \approx 57.75 \log n$. Therefore, $n_T \approx 0.55\, n\,(\log n)^{-0.75}$ from (49).

Based on the above considerations, we evaluate the computational complexity. For the number of physical qubits, the actual computation part requires

$$5nq_a + q_a n_T \approx 23.88 n\,(\log n)^2\,, \tag{50}$$

and the magic state distillation part requires

$$q_m n_T = 366.00 n\,(\log n)^{1.75}\,. \tag{51}$$

Therefore, the leading term is $23.88 n\,(\log n)^2$ from the actual computation part. Furthermore, the circuit depth is

$$3n^2 \times 6 t_g \log \left(n_T \frac{t_g}{t_m} \right) \approx 1039.52\, n^2\,(\log n)^2 \tag{52}$$

and KQ_p is $24819.36\, n^3\,(\log n)^4$.

2) USING THE MAGIC STATE DISTILLATION METHOD PROPOSED BY GIDNEY AND FOWLER

In this section, we evaluate computational complexity similar to that in Section IV-A2. We focus on the computational complexity evaluation related to the Toffoli gate. The number of Toffoli gates that can be executed in parallel during time t_g is $n_{CCZ} t_g/t_m$. The circuit depth becomes

$$3n^2 \left(\frac{10.75n}{n_{CCZ}} t_m - 12 t_g + 6\, t_g \log \left(n_{CCZ} \frac{t_g}{t_m} \right) \right). \tag{53}$$

Consequently, KQ_p is represented as

$$\begin{aligned} KQ_p = &(5nq_a + q_m n_{CCZ}) \\ &\times 3n^2 \left(\frac{10.75n}{n_{CCZ}} t_m - 12 t_g + 6\, t_g \log \left(n_{CCZ} \frac{t_g}{t_m} \right) \right). \end{aligned} \tag{54}$$

The value of n_{CCZ} that minimizes KQ_p is almost

$$n_{CCZ} = n \sqrt{\frac{215 t_m q_a}{24 t_g q_m \log n}}. \tag{55}$$

Here, focusing on the variables within n_{CCZ}, we calculate n_{CCZ}. t_m, q_a, and q_m are the same as that in Section IV-B2, and $t_g \approx 20.09 \log n$. Thus, $n_{CCZ} \approx 0.62\, n\,(\log n)^{-0.75}$.

TABLE 37. KQ_p minimized computational cost for the Fourier-basis method proposed by Oonishi and Kunihiro [10] when using the magic state distillation method proposed by Litinski [20].

n	1024	2048	4096
Distilllation Type	$(15\text{-to-}1)^6_{11,5,5} \times (15\text{-to-}1)_{25,11,11}$	$(15\text{-to-}1)^6_{13,5,5} \times (15\text{-to-}1)_{29,11,13}$	$(15\text{-to-}1)^6_{13,5,5} \times (15\text{-to-}1)_{29,11,13}$
Magic error	78.3%	8.42%	73.5%
d_a	23	24	27
Actual error	17.1%	42.7%	7.07%
Error rate	95.3%	51.1%	80.5%
Time (day)	1.20	3.99	14.4
#(physical qubits)	1.53×10^7	5.66×10^7	1.47×10^8
KQ_p	1.59×10^{18}	1.92×10^{19}	1.84×10^{20}

TABLE 38. KQ_p minimized computational cost for the Fourier-basis method proposed by Oonishi and Kunihiro [10] when using the magic state cultivation method proposed by Gidney et al. [21]. d_m denotes the code distance in Fowler et al.'s magic state distillation [7] applicable to the T gates that have undergone magic state cultivation.

n	1024	2048	4096
ID	GSJ1	GSJ1	GSJ2
d_m	30	32	34
Magic error	55.4%	54.7%	47.9%
d_a	24	25	27
Actual error	16.6%	38.9%	25.6%
Error rate	72.0%	93.6%	73.5%
Time (day)	2.32	8.25	27.5
#(physical qubits)	2.83×10^7	8.06×10^7	2.66×10^8
KQ_p	5.67×10^{18}	5.75×10^{19}	6.31×10^{20}

Therefore, when evaluating the computational complexity with KQ_p minimized, we obtain the following which is similar to that in Section IV-C1.

- Number of physical qubits: $23.88n\,(\log n)^2$
- Circuit depth: $361.57\,n^2\,(\log n)^2$
- KQ_p: $8632.82n^3\,(\log n)^4$

D. FOURIER-BASIS METHOD

In this section, we evaluate the computational cost of applying two different magic state distillation methods to the Fourier-basis adder proposed by Oonishi and Kunihiro [10]. First, we assess the computational cost when using the magic state distillation method proposed by Fowler et al [7]. Next, we evaluate the computational cost when using the magic state distillation methods proposed by Gidney and Fowler [19].

1) USING THE MAGIC STATE DISTILLATION METHOD PROPOSED BY FOWLER et al.

The number of T gates that can be executed in parallel during time t_g is $n_T t_g/t_m$. From the previous analysis [10], we consider $n\log n \le n_T t_g/t_m \le n^2$ for minimizing KQ_p. Then, the circuit depth becomes

$$\frac{27n^3\log n}{n_T}t_m + 297t_g n\log n. \tag{56}$$

Each phase gate requires two qubits to store the original computation bit values, in addition to one qubit for the execution of the S gate. Consequently, KQ_p is represented as

$$\begin{aligned}\mathrm{KQ}_p &= (2nq_a + (q_m + 3q_a)\,n_T) \\ &\quad \times \left(\frac{27n^3\log n}{n_T}t_m + 297t_g n\log n\right) \qquad (57)\\ &= \left(297t_g\,(q_m+3q_a)\,n\log n\right)n_T \\ &\quad + \frac{54t_m q_a n^4\log n}{n_T} + f\left(n, t_g, t_m, q_a, q_m\right), \qquad (58)\end{aligned}$$

and we assume that $f\left(n, t_g, t_m, q_a, q_m\right)$ is a constant independent of n_T, similar to that in Section IV-C1. The value of n_{CCZ} that minimizes KQ_p is almost

$$n_T = \sqrt{\frac{2t_m q_a}{11t_g\,(q_m+3q_a)}}\,n^{1.5}. \tag{59}$$

Here, focusing on the variables within n_T, we calculate n_T.

First, we discuss the value of t_g, t_m, q_a, and q_m. Similar to the discussion in Sections IV-A1 and IV-B1, t_g and t_m are $\Theta(\log n)$, q_a is $\Theta\left((\log n)^2\right)$, and q_m is $\Theta\left((\log n)^{2.5}\right)$. Therefore, $n_T = \Theta\left(n^{1.5}(\log n)^{-0.25}\right)$. These values are calculated based on KQ and d_a derived from KQ. The leading term of the number of logical qubits in the actual computation step is $3n_T = \Theta\left(n^{1.5}(\log n)^{-0.25}\right)$, from the ancilla qubits for the S gates. Furthermore, from (56), the circuit depth is $\Theta\left(n^{1.5}(\log n)^{2.25}\right)$, which results in a KQ of $\Theta\left(n^3(\log n)^2\right)$. Therefore, since the degree of n is 3, we proceed in the same manner as that in Section IV-B1, $d_a \ge 1.55\log n$. Thus, similar to Section IV-B1, $t_m \approx 67.65\log n$, $q_a \approx 4.78(\log n)^2$, and $q_m \approx 668.35(\log n)^{2.5}$. Moreover, $t_g \approx 22.60\log n$, and then, $n_T \approx 0.062\,n^{1.5}(\log n)^{-0.25}$.

Based on the above considerations, we evaluate the computational complexity. For the number of physical qubits, the actual computation part requires

$$2nq_a + 3q_a n_T \approx 0.89n^{1.5}(\log n)^{1.75}, \tag{60}$$

and the magic state distillation part requires

$$q_m n_T = 41.68n^{1.5}(\log n)^{2.25}. \tag{61}$$

Therefore, the leading term is $41.68n^{1.5}(\log n)^{2.25}$ from the magic state distillation part. Furthermore, the circuit depth is

$$\frac{27n^3\log n}{0.062\,n^{1.5}(\log n)^{-0.25}} \times 67.65\log n$$

$$\approx 29289.85\, n^{1.5} (\log n)^{2.25}, \tag{62}$$

and KQ_p is $1220733.91\, n^3 (\log n)^{4.5}$.

2) USING THE MAGIC STATE DISTILLATION METHOD PROPOSED BY GIDNEY AND FOWLER

In the Fourier-basis method, it is necessary to convert Toffoli gates into T gates when utilizing magic state distillation proposed by Gidney and Fowler [19]. However, compared to the method by Fowler et al. [7], it is more challenging to reduce the error rate. As demonstrated in the Appendix C2, for $n = 2048$, the number of repetitions for magic state distillation becomes three, which is evidently less efficient compared to that of the other methods.

Two $|A\rangle$ states can be generated within the time t_m by utilizing the magic state distillation proposed by Gidney and Fowler. The number of T gates that can be executed in parallel during time t_g is $2n_{CCZ}t_g/t_m$. From a previous analysis [10], we consider $n \log n \leq 2n_{CCZ}t_g/t_m \leq n^2$ for minimizing KQ_p. If we consider them as constants, the circuit depth becomes

$$\frac{27n^3 \log n}{n_{CCZ}} \frac{t_m}{2} + 297 t_g n \log n, \tag{63}$$

where $n \log n \leq 2n_{CCZ}t_g/t_m \leq n^2$. Similar to Section IV-D1, KQ_p is represented as

$$\mathrm{KQ}_p = \left(297 t_g (q_m + 3q_a)\, n \log n\right) n_{CCZ} + \frac{27 t_m q_a n^4 \log n}{n_{CCZ}} + f\left(n, t_g, t_m, q_a, q_m\right), \tag{64}$$

where $f\left(n, t_g, t_m, q_a, q_m\right)$ is a constant independent of n_{CCZ}. The value of n_{CCZ} that minimizes KQ_p is almost

$$n_{CCZ} = \sqrt{\frac{t_m q_a}{11 t_g (q_m + 3q_a)}} n^{1.5}. \tag{65}$$

By focusing on the variables within n_{CCZ}, we calculate n_{CCZ}.

First, we discuss the values of t_g, t_m, q_a, and q_m. When repeatedly generating T gates using the magic state distillation by Gidney and Fowler, errors accumulate because erroneous T gates are reused as inputs. If the error rate of the T gates generated in the initial magic state distillation is p, the total error rate when x T gates are generated becomes

$$\sum_{i=1}^{x/2} 2pi = \frac{x(x+2)}{4} p. \tag{66}$$

Equation (66) indicates that, although the required error rate decreases quadratically, it remains on the order of $1/\mathrm{poly}(n)$. Then, similar to the discussion in Section IV-D1, t_g and t_m are $\Theta(\log n)$, q_a is $\Theta\left((\log n)^2\right)$, q_m is $\Theta\left((\log n)^{2.5}\right)$, and $n_{CCZ} = \Theta\left(n^{1.5} (\log n)^{-0.25}\right)$. Here, these values are calculated based on KQ and d_a derived from KQ. The leading term of the number of logical qubits in the actual computation step is $3n_T = \Theta\left(n^{1.5} (\log n)^{-0.25}\right)$, from the ancilla qubits for S gates. Furthermore, from (63), the circuit depth is $\Theta\left(n^{1.5} (\log n)^{2.25}\right)$, which results in a KQ of $\Theta\left(n^3 (\log n)^2\right)$.

Therefore, since the degree of n is 3, we proceed in the same manner as in Section IV-B1, $d_a \geq 1.55 \log n$. Thus, similar to Section IV-D1, $t_g \approx 22.60 \log n$ and $q_a \approx 4.78 (\log n)^2$.

From (66), the required precision for the output of magic state distillation is squared compared to that in other cases. Considering that the topological error is approximately $(0.1)^{d_m/2}$, if we set the code distance of the surface code for the j-th round of magic state distillation to

$$d_m = 4 \times 3^j \log_{0.1}\left(0.002 \times \sqrt{35}\right) \approx 7.70 \times 3^i, \tag{67}$$

the topological and distillation errors are almost the same. Then, $t_m \approx 343.20 \log n$, and similar to (38)–(42), $q_m \approx 2673.42 (\log n)^{2.5}$. Therefore, $n_{CCZ} \approx 0.050\, n^{1.5} (\log n)^{-0.25}$.

Therefore, when evaluating the computational complexity with KQ_p minimized, we obtain the following, which is similar to that in Section IV-D1.

- Number of physical qubits: $132.76 n^{1.5} (\log n)^{2.25}$
- Circuit depth: $93299.68\, n^{1.5} (\log n)^{2.25}$
- KQ_p: $12386453.49 n^3 (\log n)^{4.5}$

V. CONCLUSION AND FUTURE WORKS

In this study, we evaluated the computational cost of Shor's algorithm for factoring an n-bit number considering physical qubits. We evaluated the computational cost of four primary construction methods of Shor's algorithm: Ripple-Carry [8], [13], Carry-Lookahead [16], and Fourier-basis methods [10]. To conduct these computational cost evaluations, we discussed the effect of three types of magic state distillation [7], [19], [20] and magic state cultivation [21] on surface code [7]. Through the aforementioned discussion, we conducted a theoretical analysis of computational complexity encompassing surface codes and magic state distillation, while also comparing the computational complexities across multiple methods.

In Section III, we presented the method for evaluating computational complexity related to surface codes and magic state distillation, and we discussed the overhead of additional computational costs required by these methods. We demonstrated that the overhead in terms of both time and space complexities is at most $\mathrm{poly}(n)$, i.e., Shor's algorithm remains efficient even when considering the overhead associated with physical qubits. This study demonstrated that the code distance of the surface code constituting each logical qubit is $\Theta(\log n)$.

In Section IV, we presented an evaluation of the computational cost of Shor's algorithm across various implementations and compared the computed computational costs. In our evaluation, we conduct a meticulous scheduling that takes into account the execution time of quantum gates, thereby enabling a more precise evaluation. Our results indicate that, in terms of computational complexity, the Carry-Lookahead method proposed by Oonishi et al. [16] is the most efficient. Actual computed numerical values corroborated a similar reduction; however, in practice, the difference in KQ_p

compared to the next best method, namely the Ripple-Carry methods proposed by Gidney and Ekerå [8], was only a factor between one to two, indicating minimal disparity. This can be attributed to the difference in KQ_p being merely $\Theta(\log n)$. This underscores the paramount importance of optimizing the coefficient part for future efficiency improvements.

As future work for this study, further optimization of each method can be considered. For the Ripple-Carry method proposed by Gidney and Ekerå [8], it is crucial to devise a configuration that minimizes the computational cost related to physical qubits. In addition, for the Carry-Lookahead and Fourier-basis methods, it is extremely important to estimate the extent to which computational complexity can be reduced by consolidating multiple computational circuits for acceleration. As demonstrated in this study, the method of Shor's algorithm considered efficient can easily change by minimizing the coefficient part. Therefore, further optimization focusing on the coefficient part remains a significant challenge for future research.

APPENDIX
ADDITIONAL NUMERICAL CALCULATION COST FOR SHOR'S ALGORITHM

In this section, we show all computational costs that were not covered in Section IV. For the sake of simplicity, this section will present only the types of magic state distillation, the code distance of surface codes, error rates, computation time, physical qubit count, and KQ_p, similar to Table 10.

A. RIPPLE-CARRY METHOD PROPOSED BY GIDNEY AND EKERÅ

1) MAGIC STATE DISTILLATION METHOD PROPOSED BY FOWLER et al.

The results minimizing the number of qubits are presented in Table 17, while the results minimizing time, specifically KQ_p, are shown in Table 18. Comparing Tables 17 and 18, it is evident that minimizing computation time also results in a reduction in the number of physical qubits, similar to Section IV-A1.

2) MAGIC STATE DISTILLATION METHOD PROPOSED BY GIDNEY AND FOWLER

The results minimizing the number of qubits are presented in Table 19, while the results minimizing time, specifically KQ_p, are shown in Table 20. Tables 19 and 20 demonstrate a reduction in KQ_p compared to Tables 17 and 18, indicating, as in Section IV-A, that the computational complexity is diminished when passing through the $|CCZ\rangle$ state.

3) MAGIC STATE DISTILLATION METHOD PROPOSED BY LITINSKI

The results using Litinski's magic state distillation are presented in Table 21. As in Section IV-A3, utilizing magic state distillation that outputs the $|CCZ\rangle$ state significantly reduces KQ_p.

4) MAGIC STATE CULTIVATION METHOD PROPOSED BY GIDNEY et al.

The results using Gidney et al.'s magic state cultivation distillation are presented in Table 22, which is similar calculated as in Section IV-A4.

B. CARRY-LOOKAHEAD METHOD

1) MAGIC STATE DISTILLATION METHOD PROPOSED BY FOWLER et al.

The results for minimizing the number of qubits are presented in Table 23, the results for minimizing computation time are shown in Table 24, and the results for minimizing KQ_p are displayed in Table 25. In minimizing computation time, a maximum of $2n$ Toffoli gates, or equivalently, up to $6nS$ gates, can be executed in parallel. Consequently, the number of logical qubits required for the actual computation increases from $5n$ to $11n$ for these S gates. In minimizing KQ, this study utilizes n_T as defined in (49). In this context, the theoretical values for q_m and q_a derived in Section IV-C1, specifically $q_m = 668.35(\log n)^{2.5}$ and $q_a = 4.78(\log n)^2$, are employed. This calculation method is consistently applied in Appendices B2–B4, and C as well. When minimizing KQ_p, the resulting values are nearly equivalent in magnitude to those minimized in the other two cases, leading to significantly enhanced computational efficiency.

2) MAGIC STATE DISTILLATION METHOD PROPOSED BY GIDNEY AND FOWLER

The results for minimizing the number of qubits are presented in Table 26, the results for minimizing computation time are shown in Table 27, and the results for minimizing KQ_p are displayed in Table 28. As in Appendix B1, when minimizing KQ_p, the resulting values are nearly equivalent in magnitude to those minimized in the other two cases, leading to significantly enhanced computational efficiency. Furthermore, with the exception of the case where $n = 4096$ and the number of magic state distillation rounds is three, the values of KQ_p have decreased, thereby improving computational efficiency from Appendix B1.

3) MAGIC STATE DISTILLATION METHOD PROPOSED BY LITINSKI

The results using Litinski's magic state distillation are presented in Table 29. According to Table 29, KQ_p decreases when compared to Table 25, which does not employ the $|CCZ\rangle$ state; however, when compared to Table 28, which utilizes the $|CCZ\rangle$ state, KQ_p does not exhibit a reduction.

4) MAGIC STATE CULTIVATION METHOD PROPOSED BY GIDNEY et al.

The results using Gidney et al.'s magic state cultivation distillation are presented in Table 30, which is similar calculated as in Section IV-A4.

C. FOURIER-BASIS METHOD

1) MAGIC STATE DISTILLATION METHOD PROPOSED BY FOWLER et al.

The results for minimizing the number of qubits are presented in Table 31, the results for minimizing computation time are shown in Table 32, and the results for minimizing KQ_p are displayed in Table 33. This study extends its focus on time minimization beyond merely scheduling magic state distillation; it also conducts a numerical evaluation of computational complexity to identify configurations that yield the shortest computation time, as demonstrated in the original paper [10]. Although this method increases the number of T gates to $36n^4 \log n$, the T-depth can be reduced to $264 (\log n)^2$ by executing $0.75n^4T$ gates in parallel. From the above, it can be concluded that minimizing KQ_p yields values that are nearly equivalent in magnitude to those minimized in the other two cases, resulting in significantly improved computational efficiency. According to Table 32, when minimizing computation time, a significant reduction in computation duration is achieved; however, this approach results in a substantial increase in KQ_p as suggested in the original paper [10].

2) MAGIC STATE DISTILLATION METHOD PROPOSED BY GIDNEY AND FOWLER

The results for minimizing the number of qubits are presented in Table 34, the results for minimizing computation time are shown in Table 35, and the results for minimizing KQ_p are displayed in Table 36. When employing the Fourier-basis method, the Toffoli gate and CCZ gate are not utilized; consequently, utilizing the $|CCZ\rangle$ state incurs additional costs when converting the $|CCZ\rangle$ state to a T gate. This results in an increase in the number of magic state distillation operations to three, thereby leading to an escalation in KQ_p.

3) MAGIC STATE DISTILLATION METHOD PROPOSED BY LITINSKI

The results using Litinski's magic state distillation are presented in Table 37. The Fourier-basis method performs operations based on the T gate; therefore, when utilizing Litinski's magic state distillation to generate T gates, KQ_p decreases in comparison to Tables 33 and 36.

4) MAGIC STATE CULTIVATION METHOD PROPOSED BY GIDNEY et al.

The results using Gidney et al.'s magic state cultivation are presented in Table 38. In the case of the Fourier-basis method, as discussed in the preceding sections, it is more cost-effective to perform Fowler et al.'s magic state distillation on the T gates after magic state cultivation. Consequently, this section evaluates the computational costs associated with the application of Fowler et al.'s magic state distillation [7], in contrast to Section IV-A4. For the Fourier-basis method, smaller circuit sizes, such as those of GSJ1 and GSJ2, result in lower computational costs for magic state cultivation compared to other approaches. This is attributable to the fact that the primary computational cost of the Fourier-basis method lies within the magic state distillation component; thus, reducing the computational cost of magic state distillation leads to an overall decrease in the computational cost of the Shor's algorithm.

REFERENCES

[1] (2024). *Expanding the IBM Quantum Roadmap to Anticipate the Future of Quantum-Centric Supercomputing*. Accessed: 15-. [Online]. Available: https://www.ibm.com/quantum/blog/ibm-quantum-roadmap-2025
[2] R. Acharya, "Suppressing quantum errors by scaling a surface code logical qubit," *Nature*, vol. 614, no. 7949, pp. 676–681, 2023.
[3] P. W. Shor, "Algorithms for quantum computation: Discrete logarithms and factoring," in *Proc. 35th Annu. Symp. Found. Comput. Sci.*, 1994, pp. 124–134.
[4] R. L. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," *Commun. ACM*, vol. 21, no. 2, pp. 120–126, 1978.
[5] N. Koblitz, "Elliptic curve cryptosystems," *Math. Comput.*, vol. 48, no. 177, pp. 203–209, 1987.
[6] V. S. Miller, "Use of elliptic curves in cryptography," in *Proc. Conf. theory Appl. Cryptograph. Techn.* Cham, Switzerland: Springer, 1985, pp. 417–426.
[7] A. G. Fowler, A. M. Stephens, and P. Groszkowski, "High-threshold universal quantum computation on the surface code," *Phys. Rev. A, Gen. Phys.*, vol. 80, no. 5, Nov. 2009, Art. no. 052312.
[8] C. Gidney and M. Ekera, "How to factor 2048 bit RSA integers in 8 hours using 20 million noisy qubits," *Quantum*, vol. 5, p. 433, Apr. 2021.
[9] N. C. Jones, R. Van Meter, A. G. Fowler, P. L. McMahon, J. Kim, T. D. Ladd, and Y. Yamamoto, "Layered architecture for quantum computing," *Phys. Rev. X*, vol. 2, no. 3, Jul. 2012, Art. no. 031007.
[10] K. Oonishi and N. Kunihiro, "Shor's algorithm using efficient approximate quantum Fourier transform," *IEEE Trans. Quantum Eng.*, vol. 4, pp. 1–16, 2023.
[11] M. Ekerå and J. Håstad, "Quantum algorithms for computing short discrete logarithms and factoring RSA integers," in *Proc. 8th Int. Workshop Post-Quantum Cryptography*. Cham, Switzerland: Springer, Jun. 2017, pp. 347–363.
[12] S. A. Cuccaro, T. G. Draper, S. A. Kutin, and D. P. Moulton, "A new quantum ripple-carry addition circuit," 2004, *arXiv: quant-ph/0410184*.
[13] C. Gidney, "Halving the cost of quantum addition," *Quantum*, vol. 2, p. 74, Jun. 2018.
[14] C. Gidney, "How to factor 2048 bit RSA integers with less than a million noisy qubits," 2025, *arXiv:2505.15917*.
[15] T. G. Draper, S. A. Kutin, E. M. Rains, and K. M. Svore, "A logarithmic-depth quantum carry-lookahead adder," *Quantum Inf. Comput.*, vol. 6, no. 4, pp. 351–369, Jul. 2006.
[16] K. Oonishi, T. Tanaka, S. Uno, T. Satoh, R. Van Meter, and N. Kunihiro, "Efficient construction of a control modular adder on a carry-lookahead adder using relative-phase Toffoli gates," *IEEE Trans. Quantum Eng.*, vol. 3, pp. 1–18, 2022.
[17] T. G. Draper, "Addition on a quantum computer," 2000, *arXiv: quant-ph/0008033*.
[18] R. Rines and I. Chuang, "High performance quantum modular multipliers," 2018, *arXiv:1801.01081*.
[19] C. Gidney and A. G. Fowler, "Efficient magic state factories with a catalyzed|CCZ to2|T transformation," *Quantum*, vol. 3, p. 135, Apr. 2019.
[20] D. Litinski, "Magic state distillation: Not as costly as you think," *Quantum*, vol. 3, p. 205, Dec. 2019.
[21] C. Gidney, N. Shutty, and C. Jones, "Magic state cultivation: Growing t states as cheap as CNOT gates," 2024, *arXiv:2409.17595*.
[22] A. G. Fowler, S. J. Devitt, and C. Jones, "Surface code implementation of block code state distillation," *Sci. Rep.*, vol. 3, no. 1, p. 1939, Jun. 2013.
[23] A. M. Steane, "Overhead and noise threshold of fault-tolerant quantum error correction," *Phys. Rev. A, Gen. Phys.*, vol. 68, no. 4, Oct. 2003, Art. no. 042322.
[24] S. Bravyi and A. Kitaev, "Universal quantum computation with ideal Clifford gates and noisy ancillas," *Phys. Rev. A, Gen. Phys.*, vol. 71, no. 2, Feb. 2005, Art. no. 022316.

[25] C. Gidney and C. Jones, "New circuits and an open source decoder for the color code," 2023, *arXiv:2312.08813*.
[26] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information*. Cambridge, U.K.: Cambridge Univ. Press, 2010.
[27] N. J. Ross and P. Selinger, "Optimal ancilla-free Clifford+ T approximation of Z-rotations," *Quantum Inf. Comput.*, vol. 16, no. 11, pp. 901–953, 2016.
[28] A. G. Fowler, M. Mariantoni, J. M. Martinis, and A. N. Cleland, "Surface codes: Towards practical large-scale quantum computation," *Phys. Rev. A, Gen. Phys.*, vol. 86, no. 3, Sep. 2012, Art. no. 032324.
[29] V. Vedral, A. Barenco, and A. Ekert, "Quantum networks for elementary arithmetic operations," *Phys. Rev. A, Gen. Phys.*, vol. 54, no. 1, pp. 147–153, Jul. 1996.
[30] R. Van Meter, T. D. Ladd, A. G. Fowler, and Y. Yamamoto, "Distributed quantum computation architecture using semiconductor nanophotonics," *Int. J. Quantum Inf.*, vol. 8, no. 2, pp. 295–323, Feb. 2010.
[31] C. Gidney and A. G. Fowler, "Flexible layout of surface code computations using AutoCCZ states," 2019, *arXiv:1905.08916*.
[32] C. Jones, "Low-overhead constructions for the fault-tolerant Toffoli gate," *Phys. Rev. A, Gen. Phys.*, vol. 87, no. 2, Feb. 2013, Art. no. 022328.
[33] J. Ha, J. Lee, and J. Heo, "Resource analysis of quantum computing with noisy qubits for Shor's factoring algorithms," *Quantum Inf. Process.*, vol. 21, no. 2, p. 60, Feb. 2022.
[34] S. Beauregard, "Circuit for Shor's algorithm using 2n+3 qubits," *Quantum Inf. Comput.*, vol. 3, no. 2, pp. 175–185, Mar. 2003.
[35] A. Pavlidis and D. Gizopoulos, "Fast quantum modular exponentiation architecture for Shor's factoring alogrithm," *Quantum Inf. Comput.*, vol. 14, no. 7, pp. 649–682, May 2014.
[36] D. Horsman, A. G. Fowler, S. Devitt, and R. V. Meter, "Surface code quantum computing by lattice surgery," *New J. Phys.*, vol. 14, no. 12, Dec. 2012, Art. no. 123011.
[37] A. G. Fowler and C. Gidney, "Low overhead quantum computation using lattice surgery," 2018, *arXiv:1808.06709*.
[38] Y. Li, "A magic state's fidelity can be superior to the operations that created it," *New J. Phys.*, vol. 17, no. 2, Feb. 2015, Art. no. 023037.
[39] C. Zalka, "Shor's algorithm with fewer (pure) qubits," 2006, *arXiv:quant-ph/0601097*.

KENTO OONISHI received the B.S. degree in engineering, the M.S. degree in information science and technology, and the Ph.D. degree in mathematical informatics from The University of Tokyo, Tokyo, Japan, in 2016, 2018, and 2021, respectively.

He has been a Researcher with Mitsubishi Electric Corporation, Kamakura, Japan, since 2021. His research interests include cryptography, quantum computation, and artificial intelligence.

NOBORU KUNIHIRO (Member, IEEE) received the B.E., M.E., and Ph.D. degrees in mathematical engineering and information physics from The University of Tokyo, Tokyo, Japan, in 1994, 1996, and 2001, respectively.

He was a Researcher with NTT Communication Science Laboratories, from 1996 to 2002. He was an Associate Professor with The University of Electro-Communications, from 2002 to 2008. He was an Associate Professor with The University of Tokyo, from 2008 to 2019. He has been a Professor with the University of Tsukuba, Tsukuba, Japan, since 2019. His research interests include cryptography, information security, and quantum computation.