NAIST-IS-DD0761205

Doctoral Dissertation

Studies on Methods for the Synthesis of Quantum Circuits

Yumi Nakajima

June 25, 2009

Department of Information Systems
Graduate School of Information Science
Nara Institute of Science and Technology

A Doctoral Dissertation
submitted to Graduate School of Information Science,
Nara Institute of Science and Technology
in partial fulfillment of the requirements for the degree of
Doctor of ENGINEERING

Yumi Nakajima

Thesis Committee:

Professor Yasuhiko Nakashima	(Supervisor)
Professor Hiroyuki Seki	(Co-supervisor)
Professor Shigeru Yamashita	(Ritsumeikan University)
Associate Professor Masaki Nakanishi	(Yamagata University)

Studies on Methods for the Synthesis of Quantum Circuits*

Yumi Nakajima

Abstract

A quantum computer is expected to solve some problems far faster than today's computers and may be a promising device with low power consumption. This computer is based on the physical theory of quantum mechanics. In the theory of quantum mechanics, quantum states and evolutions of quantum states are represented by vectors in a Hilbert space and unitary operators, respectively. A unitary matrix can then be regarded as an algorithm in the conventional computer. Since it is generally difficult to develop an arbitrary unitary matrix in a quantum system, quantum computation is carried out by combining elementary operations such as one- or two-qubit operations, where a qubit is the smallest unit of information in quantum computation. A unitary matrix is said to be performed effectively if there exists a quantum circuit composed of a polynomial number of elementary gates. Therefore, translating a unitary matrix into an efficient sequence of elementary gates is a fundamental problem in designing quantum circuits.

In this dissertation, the problem involved in synthesizing minimal quantum circuits for carrying out any (large) quantum operation is described. The author considers two quantum systems, namely, the two-level quantum system and the d-level quantum system, where d is any integer greater than two. Here, the two-level quantum system can be regarded as a quantum mechanical analogue of conventional computation, and the d-level quantum system can be regarded as a quantum mechanical analogue of conventional multilevel logic.

* Doctoral Dissertation, Department of Information Systems, Graduate School of Information Science, Nara Institute of Science and Technology, NAIST-IS-DD0761205, June 25, 2009.

First, the author proposes a new method for the synthesis of quantum circuits for the two-level quantum system, which is based on a divide-and-conquer strategy and the KAK matrix decomposition. Using this method, an arbitrary $2^n \times 2^n$ unitary matrix can be translated into a quantum circuit composed of $O(4^n)$ elementary gates. The size of the quantum circuit synthesized by the method described in this dissertation is the same as that of the size of the previous best-practice methods. However, the proposed method is advantageous for the synthesis of polynomial-size quantum circuits for the radix-two quantum Fourier transform (QFT), whereas other methods include some optimization and simplification techniques for the synthesis of polynomial-size quantum circuit.

Second, a method for the synthesis of quantum circuits for the d-level quantum system is described. Here, the author proposes a balanced partitioning method that is based on the the divide-and-conquer strategy. The previous methods used for computing the KAK decomposition are preferred for the two-level quantum system. However, to apply the KAK decomposition to the general d-level quantum system, the partition size has to be chosen carefully because the size of the quantum circuit produced by the synthesis method increases exponentially when an inappropriate partition size is selected. The synthesized quantum circuit is not asymptotically optimal except when d is a power of two. However, when the number of qudits n is small, where a qudit is the smallest unit of information in the d-level quantum system, the proposed method can synthesize the smallest quantum circuit as compared to those synthesized by the other synthesis methods.

Third, a new quantum circuit is developed for implementing the Aharonov-Jones-Landau (AJL) algorithm. The AJL algorithm approximates the Jones polynomial (knot invariant) at the k-th root of unity in $poly(n, m, k)$. Here, the input knot is given by an n-strand braid word of length m. The AJL algorithm is a polynomial-time quantum algorithm in the input size if m and k are bounded by polynomials in n (in this case, the input size is polynomial in n). The problem is intractable on conventional computers even under this assumption. The author slightly improves the performance of the AJL algorithm and shows that the performance of the AJL algorithm does not depend on k when $k \geq n/2 + 1$.

Keywords:

Quantum circuit synthesis, unitary matrix decomposition, quantum Fourier transform (QFT), AJL algorithm, Jones polynomial

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The information society grows rapidly with advances in computer technology and the Internet. The accessibility of information by the information society is enhanced by increasing the speed of computers. However, irrespective of how the performance of the computer is improved, there might still be many problems that are unsolvable in polynomial time under the framework of today's computers, such as, problems related to genetic analysis, design of a novel drug, and weather forecast.

A quantum computer may be a promising device for solving some problems far faster than today's computers. Shor's algorithm [43] for factoring large numbers on a quantum computer provides an exponential speed up over known conventional factorization algorithms. The key component of Shor's algorithm is the quantum Fourier transform (QFT), which amplifies probabilistic amplitudes using quantum interference. The QFT can be implemented using a polynomial number of elementary operations, whereas the conventional algorithms are believed to require exponential-time steps. This is the reason why Shor's algorithm makes it possible to exponentially speed up the factoring of very large numbers.

The quantum computer is based on the physical theory of quantum mechanics. In the theory of quantum mechanics, quantum states and evolutions of quantum states are represented by vectors in a Hilbert space and unitary operators, respectively. A unitary matrix can then be regarded as an algorithm in the conventional computer.

Since it is generally difficult to develop an arbitrary unitary matrix in a quantum system, quantum computation is carried out by combining elementary operations such as one- or two-qubit operations, where a qubit is the smallest unit of information in quantum computation. The sequence of elementary operations can be denoted by using a quantum circuit model. The concept of quantum circuits was introduced by Deutsch [17]; these circuits can be regarded as quantum analogues of conventional circuits. Conventional circuits are used to design conventional algorithms, and similarly, quantum circuits are used to design quantum algorithms and analyze their efficiencies in computation time.

In the quantum circuit model, each operation is denoted by a quantum gate. A set of quantum gates, which are used to develop an arbitrary unitary matrix, is termed universal gates or elementary gates. A set of arbitrary one-qubit gates and a two-qubit gate called the CNOT gate, where the CNOT gate is a quantum analogue of the conventional XOR gate, is an example of elementary gates. A unitary matrix is said to be performed effectively if there exists a quantum circuit composed of a polynomial number of elementary gates. Therefore, translating a unitary matrix into an efficient sequence of elementary gates is a fundamental problem in designing quantum circuits. The details about quantum circuits and elementary gates are provided in Chapter 2.

In this dissertation, two methods for synthesizing efficient quantum circuits that transform an arbitrary large unitary matrix into a quantum circuit consisting of elementary gates are proposed. Here, the author considers two quantum systems, namely, the two-level quantum system (Chapter 3) and the d-level quantum system (Chapter 4), where d is any integer greater than two.

The two-level quantum system is a quantum mechanical analogue of conventional computation. Here, the smallest unit of information in quantum computation is called a qubit, which is a quantum analogue of a conventional bit. A qubit represents two values ($|0\rangle$ and $|1\rangle$) simultaneously. A state of qubit is then described by a vector in the following two-dimensional space: $|\varphi\rangle = \alpha_0|0\rangle + \alpha_1|1\rangle$, where $|\alpha_0|^2 + |\alpha_1|^2 = 1$ for complex numbers α_0 and α_1, and $|0\rangle$ and $|1\rangle$ form a basis in the two-dimensional Hilbert space. An operation that performed on a qubit is described by a 2×2 unitary matrix. Consequently, a state of n qubits can be represented by a vector with a dimension of 2^n and an operation that is performed on n qubits is described by a $2^n \times 2^n$ unitary matrix.

The d-level quantum system can be regarded as a quantum analogue of conventional multilevel logic. Note that d is any integer greater than two. The smallest unit of information is known as a qudit, which represents the following d values simultaneously: $|0\rangle$, $|1\rangle$, $\cdots$, $|d-1\rangle$. These states form a basis in the d-dimensional Hilbert space, and that, a state of a qudit is described by a vector in a d-dimensional space: $|\varphi\rangle = \beta_0|0\rangle + \beta_1|1\rangle + \beta_2|2\rangle + \cdots + \beta_{d-1}|d-1\rangle$ such that $\sum_{j=0}^{d-1} |\beta_j|^2 = 1$ for complex numbers β_0, β_1, $\cdots$, β_{d-1}. An operation that is performed on a qudit is then described by a $d \times d$ unitary matrix. Consequently, a state of n qudits is described by a vector with a dimension of d^n and an operation that is performed on n qudits is described by a $d^n \times d^n$ unitary matrix.

In some cases, quantum computations such as the QFT [25] and the Schur transform [4, 5] may be described naturally as operations that are performed on the d-level quantum system (qudits). The number of coupled qudits used to perform a quantum operation is less than that used in the two-level quantum system (qubits) [7]. In addition, the approximation of the radix-d QFT, which is the QFT that is performed on n qudits, provides better approximation properties than the radix-two QFT because the magnitude of error decreases exponentially with d [50]. Therefore, in this dissertation, the author analyzes both quantum systems. The author will explain more details about the two-level quantum system and the d-level quantum system in Chapter 2.

Chapter 3 describes a method used for the synthesis of quantum circuits for the two-level quantum system by the KAK decomposition. The KAK decomposition is a generic matrix decomposition of the form $G = K_1AK_2$, derived from the $G = KAK$ theorem of the Lie group theory. The main objective of this chapter is to develop a concrete algorithm for computing the KAK decomposition.

Since the KAK decomposition is a comprehensive decomposition based on the Lie group theory, it involves Khaneja-Glaser decomposition (KGD [32, 33]), which is a group-theoretical decomposition that translates an arbitrary operation performed on n qubits into a time-optimal sequence of operations performed on NMR quantum computers. If the number of qubits n is small, all parameters appearing in the time-optimal sequence can be computed by solving a linear equation. However, since the number of parameters increases exponentially in n, to compute the KGD by solving linear equation does not seems be practical in general.

By using the synthesis method described in Chapter 3, the KGD can be computed by matrix decomposition. Since the KAK decomposition is a comprehensive decomposition, the proposed method might be useful for translating a time-optimal sequence of operations for other physical devices. By using the method described in this dissertation, an arbitrary $2^n \times 2^n$ unitary matrix can be translated into a quantum circuit composed of $O(4^n)$ elementary gates, which is a best-practice result of synthesizing quantum circuits for the two-level quantum system. This synthesis method is also useful for designing polynomial-size quantum circuits for the radix-2 QFT.

Chapter 4 describes a method for the synthesis of quantum circuits for the d-level quantum system by the Cosine-Sine decomposition (CSD). The CSD is the generalized singular value decomposition (SVD), and the algorithm for computing the CSD is well known in numerical linear algebra.

First, the CSD is carried out for synthesizing quantum circuit for the two-level quantum system. During the CSD, an input matrix G is partitioned into four sub-blocks and the CSD of G is achieved by computing the SVD of each partitioned matrix. In the two-level quantum system, the size of an input matrix is $2^n \times 2^n$. Therefore, the size of each partitioned matrix is identical, *i.e.*, each partitioned matrix is a $2^{n-1} \times 2^{n-1}$ matrix. Then, a product of block-diagonal matrices, where each block-diagonal matrix is composed of $2^{n-1} \times 2^{n-1}$ sub-blocks, is obtained by computing the SVD of each $2^{n-1} \times 2^{n-1}$ unitary matrix. This leads to an advantage in the quantum circuit design. Let $G = UDV$ be the CSD. Then, U and V correspond to controlled-unitary operations, which have one control qubit and $n-1$ target qubits. Furthermore, D is a controlled-unitary operation, which has $n-1$ controlled qubits and one target qubit. The formation of a sequence of controlled unitary operations, where each controlled unitary operation involves $n-1$ controlled qubits and one target qubit, is obtained by recursively carrying out the CSD on the sub-blocks of U and V, *i.e.*, the matrices of size greater than 2×2.

The above method is preferred for synthesizing quantum circuit for the two-level quantum system. However, to carry out the CSD in the arbitrary d-level quantum system, G cannot be partitioned into four equally-sized sub-blocks, because G is a $d^n \times d^n$ unitary matrix. In this case, the size of the circuit synthesized by the CSD depends on the selection of the partition size.

The main objective of Chapter 4 is to propose a balanced partitioning method for developing a recursive synthesis algorithm. Though the size of the quantum circuit synthesized by the proposed method is not asymptotically optimal, except when d is a power of two, the proposed method can synthesize the smallest quantum circuit; this circuit is smaller than those synthesized by other synthesis methods for the d-level quantum system. For example, when $d = 3$ and $n = 2$ (two qudits), the number of CINC gates, which is a generalized versions of the CNOT gate, is 36, whereas the previous method requires 156 CINC gates. In addition, the author shows that the proposed synthesis method is useful for designing polynomial-size quantum circuits for the radix-d QFT. The result described in Chapter 4 is a generalization of the result described in Chapter 3.

Chapter 5 describes the development of a new quantum circuit for implementing the Aharonov-Jones-Landau (AJL) algorithm. The AJL algorithm is a quantum algorithm that approximates the Jones polynomial (knot invariant) at a certain root of unity [3] in $O(mn \log^2 k)$ time. Here, the input knot is given by a closure of an n-strand braid word of length m. The AJL algorithm is a polynomial-time quantum algorithm in the input size if m and k are bounded by polynomials in n. The problem is still intractable on conventional computers even under this assumption [3, 18, 21].

The main objective of Chapter 5 is to propose a new method for implementing the AJL algorithm using $O(mn)$ elementary gates. This result shows that the performance of the AJL algorithm does not depend on k. Therefore, the difference between the proposed quantum circuit and and the previous quantum circuit for implementing the AJL algorithm has significant implication in quantum computational complexity.

Jozsa stated the following conjecture: Any polynomial-time quantum algorithm can be implemented with only logarithmic quantum layers interspersed with polynomial time conventional computations [29]. Although this remains unproven in general, Cleve and Watrous have shown that it holds for Shor's algorithm [16]. If there exists a quantum circuit with logarithmic depth in the input size for the AJL algorithm, then Jozsa's conjecture is affirmatively proved. Thus, it is interesting to develop a polynomial-size quantum circuit that performs the AJL algorithm. Although the proposed quantum circuit does not prove Jozsa's conjecture, the idea may be a step closer to the solution.

Finally, the dissertation concludes in Chapter 6 with a brief summary.

Chapter 2

Preliminaries

This section briefly reviews the fundamentals of quantum mechanics and quantum computation. For more comprehensive discussions, please refer to [39].

1. Linear Algebra and Quantum Computation

1.1 The Two-Level Quantum System

A quantum computer is based on the physical theory of quantum mechanics. In the theory of quantum mechanics, quantum states and evolutions of quantum states are represented by vectors in a Hilbert space and unitary operators, respectively. Here, vectors and unitary operators are described in a finite-dimensional Hilbert space consisting of a finite number of orthonormal vectors.

The smallest unit of information in quantum computation is known as a *qubit*, which is a quantum mechanical analogue of the conventional bit. Two possible states of a qubit are denoted as $|0\rangle$ and $|1\rangle$, which can be regarded as 0 and 1 for a conventional bit. The notation $|\cdot\rangle$ is known as the Dirac notation, and it is used as a standard notation for denoting states in quantum mechanics.

In general, $|0\rangle$ and $|1\rangle$ can be denoted as orthonormal basis of a two dimensional Hilbert space as follows:

$$|0\rangle \;=\; \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \qquad |1\rangle \;=\; \begin{pmatrix} 1 \\ 0 \end{pmatrix}. \tag{2.1}$$

An arbitrary one-qubit state can be expressed as the following linear combinations of these two states:

$$|\psi\rangle \;=\; \alpha_0|0\rangle + \alpha_1|1\rangle,$$

where α_0 and α_1 are complex numbers such that $|\alpha_0|^2 + |\alpha_1|^2 = 1$. Linear combinations of the above states is known as superpositions. A qubit can form two superposition states $|0\rangle$ and $|1\rangle$; this is the main difference between qubits and bits.

Then, the basis of n qubits is described as a tensor product of the basis $|0\rangle$ and $|1\rangle$. A two-qubit system has four computational basis obtained by $|0\rangle \otimes |0\rangle$, $|0\rangle \otimes |1\rangle$, $|1\rangle \otimes |0\rangle$, and $|1\rangle \otimes |1\rangle$, which correspond to four possible conventional two-bit states, *i.e.*, 00, 01, 10, and 11. Here, $\otimes$ denotes a tensor product. These vectors in the basis are often denoted as $|00\rangle$, $|01\rangle$, $|10\rangle$, and $|11\rangle$. An arbitrary state of two qubits can be denoted as

$$|\psi\rangle \;=\; \alpha_{00}|00\rangle + \alpha_{01}|01\rangle + \alpha_{10}|10\rangle + \alpha_{11}|11\rangle,$$

where $|\alpha_{00}|^2 + |\alpha_{01}|^2 + |\alpha_{10}|^2 + |\alpha_{11}|^2 = 1$ holds for complex numbers α_{00}, α_{01}, α_{10}, and α_{11}. Vector notations of these four states can also be obtained by computing the tensor product of two-dimensional vectors, as shown in (2.1), in the following manner:

$$|00\rangle \;=\; \begin{pmatrix} 0 \\ 1 \end{pmatrix} \otimes \begin{pmatrix} 0 \\ 1 \end{pmatrix} \;=\; \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}.$$

Therefore, an arbitrary state of n qubits is described as

$$|\psi\rangle \;=\; \sum_{j=0}^{2^n-1} \alpha_j |j\rangle,$$

where $\sum_{j=0}^{2^n-1} |\alpha_j|^2 = 1$. Here, j is the decimal representation of a binary number.

Quantum computation is an evolution of input and output quantum states, it can be expressed as a unitary matrix. For example, consider the quantum NOT operation σ_x, which is analogous to the conventional NOT operation. Let the input quantum state be $|0\rangle$. Then, the output state is $|1\rangle$. Then, the quantum NOT operation can be expressed as the following 2×2 unitary matrix:

$$\sigma_x = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}.$$

We can easily verify that U_{NOT} gives a quantum NOT operation by a simple linear algebra computation.

$$\sigma_x|0\rangle = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}.$$

Some frequently performed unitary operations on a qubit are listed below.

$$R_x(\theta) = \begin{pmatrix} \cos\frac{\theta}{2} & -i\sin\frac{\theta}{2} \\ -i\sin\frac{\theta}{2} & \cos\frac{\theta}{2} \end{pmatrix}, \tag{2.2}$$

$$R_y(\theta) = \begin{pmatrix} \cos\frac{\theta}{2} & -\sin\frac{\theta}{2} \\ \sin\frac{\theta}{2} & \cos\frac{\theta}{2} \end{pmatrix}, \tag{2.3}$$

$$R_z(\theta) = \begin{pmatrix} \exp\left(-i\frac{\theta}{2}\right) & 0 \\ 0 & \exp\left(i\frac{\theta}{2}\right) \end{pmatrix}. \tag{2.4}$$

These operations are known as rotation operations about the x, y, and z axes. The Hadamard gate H is the following 2×2 unitary matrix:

$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}.$$

H is often used to generate a quantum superposition state.

It is well known that an arbitrary 2×2 unitary matrix U (a single-qubit unitary operation) can be decomposed as follows:

$$U = \exp(-i\alpha)R_z(\beta)R_y(\gamma)R_z(\delta), \tag{2.5}$$

where α, β, γ, and δ are real-valued and $0 \leq \alpha, \beta, \gamma < \pi$.

The most important unitary operation on two qubits is the controlled-NOT operation known as the CNOT, where a NOT operation is performed on a target qubit when the controlled qubit is $|1\rangle$. The matrix notation of the CNOT is as follows:

$$U_{CNOT} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}. \quad (2.6)$$

The output of the target qubit is the conventional XOR operation between the controlled qubit and the target qubit. Therefore, the CNOT can be regarded as an analogue of the conventional XOR operation.

The following two-qubit operations are also frequently performed in this dissertation.

$$\chi_1^2 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}. \quad (2.7)$$

χ_1^2 represents the SWAP operation, which exchanges the first and the second qubits.

Similarly, an operation performed on n qubits is expressed as a $2^n \times 2^n$ unitary matrix.

1.2 The d-Level Quantum System

The d-Level Quantum System, where d is any integer greater than two, is considered as analogues of conventional multilevel logic. Here, the smallest unit of information is known as a *qudit*, which forms superpositions of d states $|0\rangle$, $|1\rangle$, $|2\rangle$, $\cdots$, $|d-1\rangle$, *i.e.*,

$$|\psi\rangle = \sum_{j=0}^{d-1} \alpha_j |j\rangle,$$

where $\sum_{j=0}^{d-1} |\alpha_j|^2 = 1$ for complex numbers α_0, α_1, $\cdots$, α_{j-1}. Here, a state of one qudit is denoted by a d-dimensional vector. For example, when $d = 3$, since $|0\rangle$,

$|1\rangle$, and $|2\rangle$ correspond to vectors

$$|0\rangle = \begin{pmatrix}1\\0\\0\end{pmatrix}, \qquad |1\rangle = \begin{pmatrix}0\\1\\0\end{pmatrix}, \qquad |2\rangle = \begin{pmatrix}0\\0\\1\end{pmatrix},$$

$|\psi\rangle$ can be denoted by a three-dimensional vector $(\alpha_0, \alpha_1, \alpha_2)^T$.

Then, an operation performed on a qudit is described by a $d \times d$ unitary matrix. The most important operations performed on two qudits is known as CINC [15]. CINC is the generalization of the CNOT, defined in Eq. (2.6), of the arbitrary d-level quantum systems. Let INC be the operation on one qudit as follows:

$$INC|a\rangle = |(a+1) \mod d\rangle.$$

Then, the CINC is a controlled unitary operation that applies INC to the target qudit when the control qudit is $|d-1\rangle$.

2. Elementary Gates and Quantum Circuits

A quantum circuit may be regarded as an analogue of a conventional logic circuit model. In a conventional computation, an arbitrary operation performed on n bits is realized by combining a set of elementary operations such as AND, OR, and NOT gate. A set of {AND, OR, NOT} gates or {NAND} is known as universal gates in conventional computation. Similarly, there are some sets of elementary quantum operations, which can form an arbitrary quantum operation that acts on n qubits. This set of elementary operation is known as universal quantum gates or elementary gates in quantum computation.

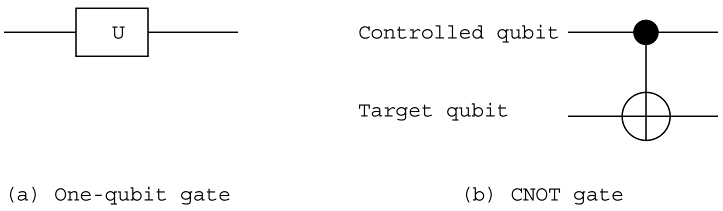
In the two-level quantum system, a set of arbitrary one-qubit operations and a two-qubit operation termed the CNOT is known as universal gates in quantum computation. The notation of quantum circuits was proposed by Deutsch [17]. As already mentioned in the previous section, since an arbitrary one-qubit operation is composed of three single-qubit rotations, as shown in (2.5), a set of arbitrary single-qubit rotations and the CNOT such as $\{R_z(\alpha), R_y(\beta), CNOT\}$ is also known as universal quantum gates.

In a quantum circuit model, each elementary gate is represented by a symbol, as shown in Fig. 2.1. Each line in the quantum circuit represents a wire in the quantum circuit. A wire corresponds to one qubit, and not a physical wire. Each symbol on each wire represents a quantum operation (gate), which is performed on a qubit that corresponds to the wire. An arbitrary unitary operation can be described in terms of a quantum circuit by combining the above symbols. For example, a single qubit operation U, defined in (2.5), can be described in terms of a quantum circuit such as that shown in Fig. 2.2. A quantum circuit is to be read from left to right. Therefore, in Fig. 2.2, $R_z(\delta)$,$R_y(\gamma)$, and $\exp(-i\alpha)R_z(\beta)$ are applied in this order. Note that the order of operations is reverse in the quantum circuit model.

Fig. 2.3 shows a simple example of a quantum circuit for a three-qubit system. Here, first, a one-qubit unitary operation U_1 is performed on the first qubit. Then, the CNOT is applied between the first qubit and the second qubit, and so on. Gates that are independently applied on different qubits, such as U_2 and U_3 shown in Fig. 2.3, can be applied in parallel. The size of the quantum circuit is equal to the number of gates appearing in the quantum circuit. As shown in Fig. 2.3, the size of the quantum circuit is 5. A unitary operation is efficient if there exist a quantum circuit composed of a polynomial number of elementary gates.

Since evolutions of quantum states are expressed by unitary matrices, these matrices can be regarded as an algorithm in a conventional computer. Since it is generally difficult to develop an arbitrary unitary matrix in a quantum system, quantum computation is carried out by combining elementary gates. An arbitrary operation performed on n qubits, *i.e.*, $2^n \times 2^n$ unitary matrix, involves $O(4^n)$ elementary gates. However, specific unitary operations such as the QFT, which is a key operation in Shor's factoring algorithm [43], involves a polynomial number of elementary gates. The QFT is implemented in $O(n^2)$ elementary operations. Therefore, a quantum computer can factor large integers in polynomial time. Therefore, quantum circuits are fundamental tools for evaluating the efficiency of quantum algorithms and for determining the advantages and limitations of quantum computation.

In the case of the arbitrary d-level quantum systems, a set of arbitrary operations performed on one qudit and CINC forms universal gates. A quantum



(a) One-qubit gate (b) CNOT gate

Figure 2.1. Representation of quantum elementary gates.

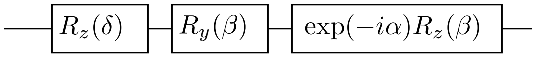


Figure 2.2. Quantum circuit for U defined by Eq. (2.5).

circuit can be described in a manner similar to that used to describe the quantum circuit model for the two-level quantum system, as described above. In the two-level quantum system, the QFT is well-known as a unitary operation that can be performed in polynomial time. On the other hand, the Schur transform [4,5] is a well-known operation that can be performed on the d-level quantum system; this operation can be performed to solve a hidden subgroup problem.

3. Uniformly Controlled Gates

Uniformly controlled gates play a key role throughout this thesis. Fig. 2.4 shows an example of a two-fold uniformly controlled one qubit gate in the two-level quantum system. Here, the black circle represents a controlled qubit of state $|1\rangle$, and the white circle represents a controlled qubit of state $|0\rangle$. As shown in this

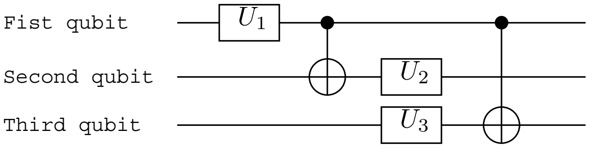


Figure 2.3. Example of a quantum circuit for a three-qubit system.

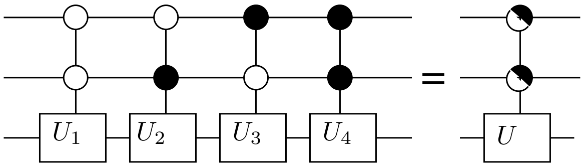


Figure 2.4. Two-fold uniformly controlled one-qubit gate, where U_1, U_2, U_3, U_4, and U are 2×2 unitary matrices.

figure, the uniformly controlled gate is a sequence of adjacent controlled gates with slightly different control node configurations. All possible combinations of the controlled states, such as $|00\rangle$, $|01\rangle$, $|10\rangle$, and $|11\rangle$, appearing in a uniformly controlled gate are shown in Fig. 2.4. In general, this sequence of controlled unitary operations can be denoted as a gate with half-moon controlled qubits as shown in the right-hand size of Fig. 2.4. The concept of uniformly controlled gates, half-moon controlled symbols, and an efficient implementation of a uniformly controlled gate was first introduced by Möttönen *et al.* in [37].

A uniformly controlled gate can be expressed by a block-diagonal unitary matrix. For example, Fig. 2.4 corresponds to the following matrix.

$$\begin{pmatrix} U_1 & 0 & 0 & 0 \\ 0 & U_2 & 0 & 0 \\ 0 & 0 & U_3 & 0 \\ 0 & 0 & 0 & U_4 \end{pmatrix},$$

where 0 is a 2×2 matrix with all zero elements.

To decompose a uniformly controlled gate into a sequence of elementary gates, Möttönen *et al.* carried out quantum multiplexor decomposition [42], which is expressed as follows:

$$\begin{pmatrix} U_1 & 0 \\ 0 & U_2 \end{pmatrix} = \begin{pmatrix} V & 0 \\ 0 & V \end{pmatrix} \begin{pmatrix} D & 0 \\ 0 & D^\dagger \end{pmatrix} \begin{pmatrix} W & 0 \\ 0 & W \end{pmatrix}.$$

Here, 0 represents a zero matrix, that is, a matrix with all zero elements. Since $U_1 U_2^\dagger = VDWW^\dagger DV^\dagger = VD^2V^\dagger$, V and D^2 are obtained by computing the

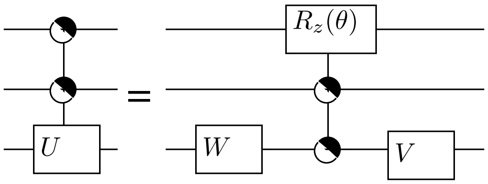


Figure 2.5. Quantum multiplexor decomposition of a two-fold uniformly controlled one-qubit gate.

eigenvalue decomposition of $U_1U_2^\dagger$. Then, W can be obtained from $D^\dagger V^\dagger U_1$ or $DV^\dagger U_2$.

The effect of quantum multiplexor decomposition is to cause decomposition, which involves gates with a reduced number of control nodes. Fig. 2.5 shows a quantum circuit obtained by carrying out quantum multiplexor decomposition on a two-fold uniformly controlled one-qubit gate, as shown in Fig. 2.4. Here, $R_z(\theta)$ is a 2×2 unitary matrix defined by Eq. (2.4).

Möttönen *et al.* carried out the above concept of decomposition of uniformly controlled gates on the basis of quantum multiplexor decomposition, as shown in Fig. 2.6.

During their effective decomposition, the locations of the controlled bits in the sequence of CNOT gates were determined by binary reflected Gray codes [40], as shown in Fig. 2.6. For example, in Fig. 2.6, consider a three-bit Gray code {000, 001, 011, 010, 110, 111, 101, 100 }. Then, the locations of the controlled bits correspond to the bits that change their states, as shown in Fig. 2.7. Note that similar decomposition holds for the other one-parameter rotation such as $R_x(\theta)$ and $R_y(\theta)$. The quantum circuit for an arbitrary $(n-1)$-fold uniformly controlled one-qubit gate is given in a similar manner. Furthermore, they showed that an arbitrary $(n-1)$-fold uniformly controlled one-qubit gate is composed of $O(2^n)$ elementary gates.

They also described a method to compute the rotation angles of the uniformly controlled one-qubit gates during effective decomposition, such as θ_1, θ_2, θ_3, $\cdots$, θ_8, as shown in Fig. 2.6. These parameters can be computed by solving a linear

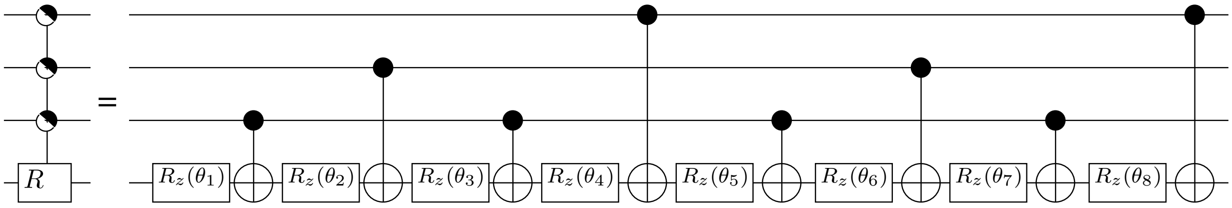
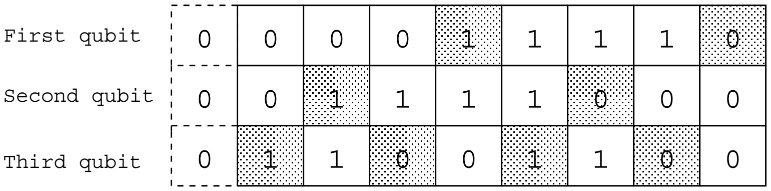


Figure 2.6. Quantum circuit that implements a three-fold uniformly controlled one-qubit gate, where $R = diag(R_z(\varphi_1),\ R_z(\varphi_2),\ R_z(\varphi_3),\ \cdots,\ R_z(\varphi_8))$.



First qubit	0	0	0	0	1	1	1	1	0
Second qubit	0	0	1	1	1	1	0	0	0
Third qubit	0	1	1	0	0	1	1	0	0

Figure 2.7. Relationship between the locations of the controlled bit and the Gray code.

equation. Consider the following uniformly controlled R_z gate $\vee_n^{n-1} R_z$:

$$\vee_n^k R_z \quad = \quad diag\left(R_z(\alpha_1), R_z(\alpha_2), R_z(\alpha_3), \cdot, R_z(\alpha_{2^k})\right),$$

which results from controlled-unitary operation performed on n qubits with k-controlled qubits. Consider the matrix M^k, where each (i, j) element is defined as follows:

$$M_{i,j}^k \quad = \quad (-1)^{b_{i-1} \cdot g_{j-1}}.$$

Here, b_j is the standard binary code representation of the integer i, and the dot in the exponent denotes the bitwise inner product of the binary vectors. g_j is the j-th gray code, such as $g_0 = 000$, $g_1 = 001$, $g_2 = 011$, $g_3 = 010$, as shown in Fig. 2.7. Then, all the parameters $\theta_1,\ \theta_2,\ \cdots,\ \theta_{2^k}$ appearing in the sequence of one-qubit rotations, as those shown in Fig. 2.6 can be computed as follows:

$$\begin{pmatrix} \theta_1 \\ \theta_2 \\ \theta_3 \\ \vdots \\ \theta_{2^k} \end{pmatrix} = M^k \begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \\ \vdots \\ \alpha_{2^k} \end{pmatrix}.$$

For more details, please see Ref. [37].

Uniformly controlled gates also play a key role in synthesizing a quantum circuit for the *d*-level quantum system. Details about the definition of these gates and the quantum circuit for uniformly controlled gates of the *d*-level quantum system will be discussed in Chapter 4.

4. The Cosine-Sine Decomposition (CSD)

The CSD is a well-known algorithm in numerical linear algebra that is used for computing the generalized singular value decomposition (GSVD). The CSD is the first matrix decomposition carried out for synthesis of quantum circuits [46].

Here, the two-level quantum system is considered. The CSD of an input matrix G can be expressed as follows:

$$\begin{array}{c} \\ 2^{n-1} \\ 2^{n-1} \end{array}\begin{array}{c} \begin{array}{cc} 2^{n-1} & 2^{n-1} \end{array} \\ \begin{pmatrix} G_{11} & G_{12} \\ G_{21} & G_{22} \end{pmatrix} \end{array} = \begin{pmatrix} U_1 & 0 \\ 0 & U_2 \end{pmatrix} \begin{pmatrix} C & -S \\ S & C \end{pmatrix} \begin{pmatrix} V_1 & 0 \\ 0 & V_2 \end{pmatrix} = U\Sigma V, \tag{2.8}$$

where

$$\begin{aligned} C &= diag\left[\cos(\zeta_1), \cos(\zeta_2), \cdots, \cos(\zeta_{2^{n-1}})\right], \\ S &= diag\left[\sin(\zeta_1), \sin(\zeta_2), \cdots, \sin(\zeta_{2^{n-1}})\right]. \end{aligned}$$

Then, $G_{11} = U_1CV_1$, $G_{12} = -U_1SV_2$, $G_{21} = U_2SV_1$, and $G_{22} = U_2CV_2$ are the SVD. Therefore, the CSD can be computed by applying the SVD to each sub-matrices.

5. Synthesis of Quantum Circuits using the CSD

In this section, the basic idea of the synthesis method for the two-level quantum system by means of the CSD is described. For more details, please see [13,37,42, 46,47].

When the CSD (2.8) is used to synthesize quantum circuit for the two-level quantum system, if the input matrix is a $2^n \times 2^n$ matrix, where n is the number of qubits, then the input matrix is partitioned into four equally sized matrices, *i.e.*, $m = 2^n$ and $\ell = 2^{n-1}$ in (2.8). For example when $n = 3$, the input matrix is an 8×8 unitary matrix, then in (2.8), each sub-block is of size 4×4 and

$$\begin{aligned} D_{11} &= D_{22} = diag(\cos\varphi_1, \cos\varphi_2, \cos\varphi_3, \cos\varphi_4), \\ D_{21} &= -D_{21} = diag(\sin\varphi_1, \sin\varphi_2, \sin\varphi_3, \sin\varphi_4). \end{aligned}$$

where the middle part D is an 8×8 block-diagonal matrix as follows:

$$D = \begin{pmatrix} \cos\varphi_1 & 0 & 0 & 0 & -\sin\varphi_1 & 0 & 0 & 0 \\ 0 & \cos\varphi_2 & 0 & 0 & 0 & -\sin\varphi_2 & 0 & 0 \\ 0 & 0 & \cos\varphi_3 & 0 & 0 & 0 & -\sin\varphi_3 & 0 \\ 0 & 0 & 0 & \cos\varphi_4 & 0 & 0 & 0 & -\sin\varphi_4 \\ \sin\varphi_1 & 0 & 0 & 0 & \cos\varphi_1 & 0 & 0 & 0 \\ 0 & \sin\varphi_2 & 0 & 0 & 0 & \cos\varphi_2 & 0 & 0 \\ 0 & 0 & \sin\varphi_3 & 0 & 0 & 0 & \cos\varphi_3 & 0 \\ 0 & 0 & 0 & \sin\varphi_4 & 0 & 0 & 0 & \cos\varphi_4 \end{pmatrix}. \tag{2.9}$$

Note that D is a sequence of unitary operations as shown in Fig. 2.8, where

$$R_y(\varphi_i) = \begin{pmatrix} \cos\varphi_i & -\sin\varphi_i \\ \sin\varphi_i & \cos\varphi_i \end{pmatrix}, \tag{2.10}$$

for $0 \leq \varphi_i < 2\pi$. This sequence of controlled unitary operations is called the uniformly controlled one-qubit operation, which is firstly introduced by Möttönen *et al.* [37].

Similarly, the rest of matrices U and V are also uniformly controlled operations, as shown in Fig. 2.9 (b). Since U_1, U_2, V_1, and V_2 are 4×4 unitary

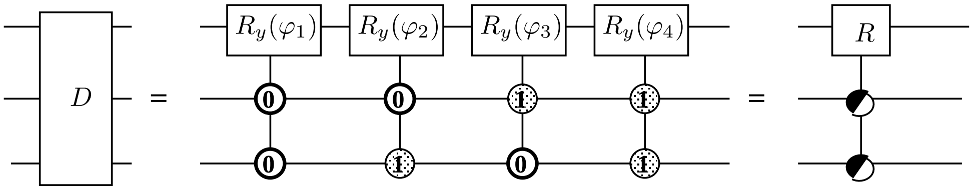


Figure 2.8. The quantum circuit corresponds to the unitary matrix D which is shown in (2.9), where $R_y(\varphi_i)$ $(i = 1, 2, 3, 4)$ is a 2×2 unitary matrix shown in (2.10).

matrices, the CSD is recursively applied to these sub-matrices. Finally, a sequence of uniformly controlled one-qubit operations, as shown in Fig. 2.9 (c), is obtained. Thus, the uniformly controlled operation is the fundamental gates for synthesizing quantum circuits by means of the CSD based approach.

As shown in Fig. 2.9, the size of the target qubits, *i.e.*, the number of qubits that the target operation $G_j^{(k)}$ is applied, decreases 1 as the recursion proceeds. On the other hand, the number of controlled unitary operations increases exponentially as the recursion proceeds. Let the input matrix be a $2^n \times 2^n$ unitary matrix, *i.e.*, an operation on n qubits, then the number of uniformly controlled one-qubit gates appearing in the quantum circuit produced by the synthesis method is $2^n - 1$.

To obtain a quantum circuit composed of elementary gates such as a sequence of CNOT gates and one-qubit operations, the translation method between a uniformly controlled one-qubit gate and a sequence of elementary gates proposed in [37] is needed.

As described above, to synthesize a quantum circuit from a $2^n \times 2^n$ unitary matrix, the CSD is first recursively applied to the sub-matrices appearing in matrix products until each matrix corresponds to a uniformly controlled one-qubit operation, and then the translation method between a uniformly controlled one-qubit operation and a sequence of elementary gates is used.

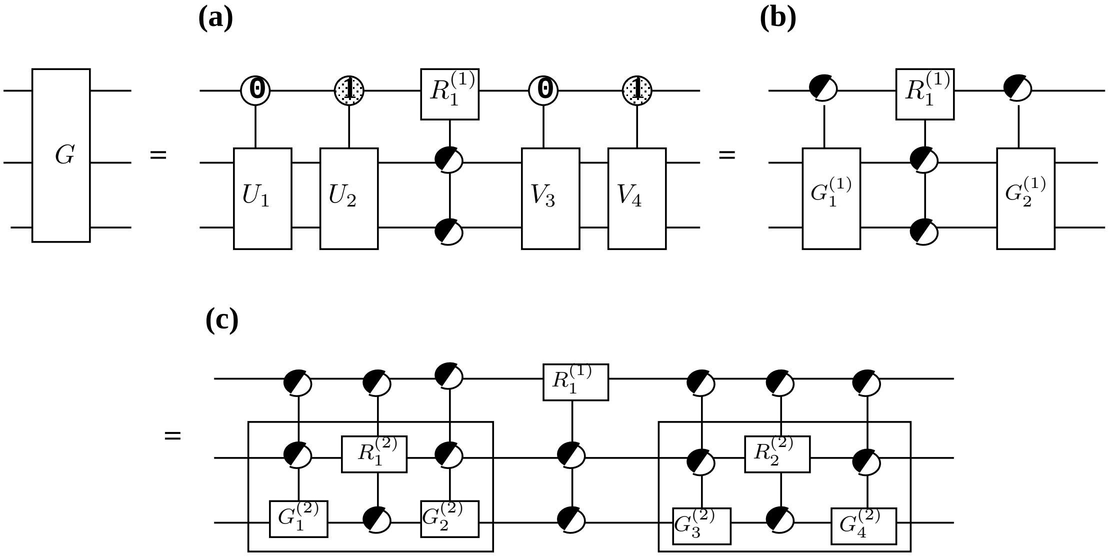


Figure 2.9. The synthesis of quantum circuits for the two-level quantum system when $n = 3$. Here, $G_i^{(k)}$ and $R_i^{(k)}$ denote the gates that are produced at the k-th recursion step.

Chapter 3

Synthesis Method for the Two-Level Quantum System

1. Introduction

Decomposing a unitary matrix into an efficient sequence of elementary gates is a fundamental problem in designing quantum circuits. There are two types of decomposition: One is exact decomposition where an arbitrary unitary matrix is decomposed precisely into a sequence of elementary gates, such as arbitrary single-qubit rotations and the CNOT gate. The other involves approximate strategies by which an arbitrary unitary matrix is decomposed approximately into a sequence of a fixed set of elementary gates, as shown in Solovay-Kitaev theorem (cf. [39], Appendix 3). In this chapter, exact decomposition is treated.

An exact decomposition based on the Cosine-Sine decomposition (CSD) has been studied [36,37,42,44,47]. The CSD-based algorithms are easy to implement on a computer because algorithms for calculating the generalized singular decomposition (GSVD) are well-known, and software libraries including the GSVD are available. Some CSD-based algorithms have been investigated with the aim of improving Barenco's result that an arbitrary $2^n \times 2^n$ unitary matrix is composed of $O(n^2 4^n)$ elementary gates [6]. And improvement to $O(4^n)$ elementary gates has been reported by Möttönen *et al.* [37,47] and by Shende *et al.* [42].

On the other hand, Khaneja and Glaser provided another kind of decomposition [33], which was later named the KGD. The KGD lies within the framework of the $G = KAK$ theorem (cf. [24], Theorem 8.6) in Lie group theory. This theorem shows that an element $g \in SU(2^n)$ is decomposed into matrix products $k_1 a k_2$ for some $k_1, k_2 \in \exp(\mathfrak{k})$ and $a \in \exp(\mathfrak{h})$. Here, $\mathfrak{su}(\mathfrak{2}^\mathfrak{n}) = \mathfrak{k} \oplus \mathfrak{m}$ is a Cartan decomposition in Lie algebra $\mathfrak{su}(\mathfrak{2}^\mathfrak{n})$, $\mathfrak{k}$ and $\mathfrak{m} = \mathfrak{k}^\perp$ are orthogonal vector spaces, and $\mathfrak{h}$ is a maximal Abelian subalgebra (a Cartan subalgebra) contained in $\mathfrak{m}$ (cf. [34], §VI.2). Matrices k_1, a, and k_2 are not uniquely determined from g. They depend on the selections of the bases of $\mathfrak{k}$, $\mathfrak{m}$, and $\mathfrak{h}$; besides, they are not determined even if bases are selected. Khaneja and Glaser provided a particular selection of bases of $\mathfrak{k}$, $\mathfrak{m}$, and $\mathfrak{h}$ in Ref. [33] so that the selection matches an NMR system, and they proved that a time-optimal control on a two-qubit NMR quantum computer can be obtained from the decomposition [32]. Thus, the KGD can be regarded as the $G = KAK$ theorem on the particular bases. It should be noted that the KGD does not give a unique translation of the input matrix into a quantum circuit.

Bullock [12] showed that the CSD can also be regarded in the framework of the $G = KAK$ theorem; i.e., the CSD uses the type-**AIII** KAK decomposition with the global Cartan decomposition Θ defined as $\Theta(X) = \sigma_{1z} X \sigma_{1z}$ for $X \in SU(2^n)$, where σ_{jz} denotes that the operation defined as the Pauli matrix σ_z acts on the j-th qubit. He also introduced a method that translates matrices U, Σ, and V in (2.8) into $k_1 \in \exp(\mathfrak{k})$, $a \in \exp(\mathfrak{h})$, and $k_2 \in \exp(\mathfrak{k})$, respectively, where $\mathfrak{k}$ and $\mathfrak{h}$ are the ones defined in the KGD. Here, the KGD corresponds to the $G = KAK$ decomposition with the selection of Θ defined as $\Theta(X) = \sigma_{nz} X \sigma_{nz}$. Thus, the KGD-based quantum circuit can be obtained by combining Bullock's translation and the CSD-based algorithms.

The author introduces a new algorithm that translates a $2^n \times 2^n$ unitary matrix into a quantum circuit according to the $G = KAK$ theorem. The algorithm can derive any matrix decomposition corresponding to the type-**AIII** KAK decompositions for the given global Cartan involution Θ. The algorithm contains, as its special cases, both the CSD and the KGD in the sense that it derives the same quantum circuits as the ones calculated by them if suitable Cartan involutions and square root matrices are selected. The author finds that $\Theta(X) = \sigma_{1z} X \sigma_{1z}$ derives the CSD and $\Theta(X) = \sigma_{nz} X \sigma_{nz}$ derives the KGD, where $X \in SU(2^n)$. The strategy utilized in the proposed algorithm is related to those used in Refs. [13, 14].

However, those strategies provided methods for computing the type-**AII** KAK decomposition; no translation between the type-**AII** KAK decompositions and the type-**AIII** KAK decompositions was provided. Furthermore, the method utilized in Ref. [13] is different from those presented here in the square root matrix calculations, i.e., methods for calculating m from m^2 (where $g = km$ is a global Cartan decomposition of the input matrix g). In the method proposed in Ref. [13], first, a square root matrix is calculated in Lie algebra level. And then it is translated into an element in Lie group level via exponential mapping. In contrast, a square root matrix is calculated directly at the Lie group level using the algorithm presented here.

Although the proposed algorithm contains the CSD and derives any matrix decomposition corresponding to the type-**AIII** KAK decompositions according to the given Cartan involution, the efficiency for calculating a decomposition is not sacrificed. The reason is as follows: Roughly speaking, to decompose g into k_1ak_2, the CSD-based algorithms apply the SVD to four $2^{n-1}\times2^{n-1}$ matrices (G_{11}, G_{12}, G_{21}, and G_{22} in Eq. (2.8)), while the proposed algorithm applies eigenvalue decomposition to $2^n \times 2^n$ matrix. Therefore, the efficiencies for computing the SVD on four $2^{n-1} \times 2^{n-1}$ matrices and for computing eigenvalue decomposition on one $2^n \times 2^n$ matrix are the same.

In addition, to determine a class of quantum circuits for a give class of matrices, the proposed algorithm might have an advantage over the CSD. In the CSD-based algorithms, it is difficult to formulate a class of matrices U_1, U_2, V_1, and V_2 such that relation (2.8) holds for a given class of input matrices. Actually, to reproduce the well-known QFT circuit by using the CSD [45,46], Tucci changes the rows and columns of the QFT matrices beforehand and makes each submatrix hold a convenient form, which can be written by the $(n-1)$-qubit QFT. It would not be possible to describe the general form of the decomposition when the input matrix does not have a convenient form like the QFT. On the other hand, the proposed algorithm does not require such a preliminary change of rows and columns. All matrices appearing through the algorithm can be described using the input matrix g, the given global Cartan involution Θ, and the eigenvalues and eigenvectors of these matrix products. This will be shown explicitly as an example of the QFT decomposition (Section 4).

This chapter is organized as follows: In the following section, some preliminaries about notations, the $G = KAK$ theorem, and the KGD will be covered. Section 3 describes a new algorithm for computing the decomposition follows from the $G = KAK$ theorem. Section 4 presents decompositions of the n-qubit QFT using the new algorithm and the previous CSD-based algorithms. The author shows that the new algorithm can produce the well-known QFT circuit by using these matrix decompositions.

2. Preliminaries

2.1 Notations

Let σ_x, σ_y, and σ_z denote the Pauli matrices and $I^{\otimes s}$ be a $2^s \times 2^s$ identity matrix ($I = 2^1 \times 2^1$). Here, $\sigma_{j\alpha}$ is used to denote the Pauli matrix acting on the j-th qubit; $\sigma_{j\alpha} = I^{\otimes(j-1)} \otimes \sigma_\alpha \otimes I^{\otimes(n-j)}$, ($\alpha = x, y$, or z). Let U_{CNOT} denote the standard CNOT gate, H denote a Hadamard gate, and $R_x(\zeta) = \exp(-i\zeta\sigma_x)$. All these notations follow those in Ref. [39].

2.2 The $G = KAK$ Theorem

The $G = KAK$ theorem for compact groups (cf. [24], Theorem 8.6) provides a framework for decomposing $g \in SU(2^n)$ into the following matrix products:

$$g \quad = \quad k_1 a k_2, \qquad k_1, k_2 \in \exp(\mathfrak{k}), \mathfrak{a} \in \exp(\mathfrak{h}) \subset \exp(\mathfrak{m}). \tag{3.1}$$

Here, $\mathfrak{su}(\mathfrak{2}^\mathfrak{n}) = \mathfrak{k} \oplus \mathfrak{m}$ is a Cartan decomposition, where $\mathfrak{k}$ and $\mathfrak{m} = \mathfrak{k}^\perp$ are orthogonal vector spaces, and $\mathfrak{h}$ is a Cartan subalgebra, that is, a maximal Abelian subalgebra contained in $\mathfrak{m}$.

Let θ denote the Cartan involution of its Lie algebra $\mathfrak{su}(\mathfrak{2}^\mathfrak{n})$; i.e., (i) $\theta^2 = I^{\otimes n}$ ($\theta \neq I^{\otimes n}$) and (ii) θ is an automorphism of the Lie algebra $\mathfrak{su}(\mathfrak{2}^\mathfrak{n})$. And let the global Cartan involution (cf. [34], p. 362) of $SU(2^n)$ be Θ. Then $\mathfrak{k}$ and $\mathfrak{m}$ have the following property:

$$\theta(x) = \begin{cases} x & \text{if } x \in \mathfrak{k} \\ -x & \text{if } x \in \mathfrak{m} \end{cases}, \qquad \Theta(X) = \begin{cases} X & \text{if } X \in \exp(\mathfrak{k}) \\ X^\dagger & \text{if } X \in \exp(\mathfrak{m}) \end{cases}. \tag{3.2}$$

Three types of $\mathfrak{k}$-algebra, named **AI**, **AII**, and **AIII**, arise for $\mathfrak{su}(2^n)$. Here, **AI**, **AII**, and **AIII** correspond to $\mathfrak{k} = \mathfrak{so}(2^n)$, $\mathfrak{k} = \mathfrak{sp}(2^n)$, and $\mathfrak{s}[\mathfrak{u}(\mathfrak{p}) \oplus \mathfrak{u}(\mathfrak{q})]$ $(p + q = 2^n)$, respectively (cf. [24], p. 518).

2.3 The Khaneja-Glaser Decomposition (KGD)

Khaneja and Glaser provided a particular selection of bases of $\mathfrak{k}$, $\mathfrak{m}$, and $\mathfrak{h}$ (cf. [33], Notation 3 and 5) so that the selection matches an NMR system. Notice that $\mathfrak{h}$ is used instead of $\mathfrak{h}(\mathfrak{n})$. Here, generators of $\mathfrak{k}$, $\mathfrak{m}$, and $\mathfrak{h}$, are denoted to be tensor products of the Pauli matrices;

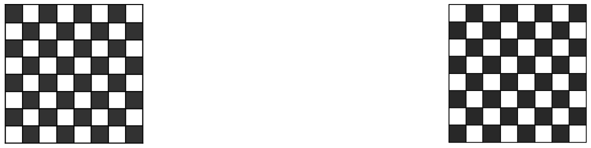
$$\begin{aligned} \mathfrak{k} &= \mathrm{span}\,\{\mathrm{A} \otimes \sigma_{\mathrm{z}}/2, \mathrm{B} \otimes \mathrm{I}, \mathrm{i}\sigma_{\mathrm{nz}}/2 \mid \mathrm{A}, \mathrm{B} \in \mathfrak{su}(2^{n-1})\}, && (3.3)\\ \mathfrak{m} &= \mathrm{span}\,\{\mathrm{A} \otimes \sigma_{\mathrm{x}}/2, \mathrm{B} \otimes \sigma_{\mathrm{y}}/2, \mathrm{i}\sigma_{\mathrm{nx}}/2, \mathrm{i}\sigma_{\mathrm{ny}}/2 \mid \mathrm{A}, \mathrm{B} \in \mathfrak{su}(2^{n-1})\}. && (3.4) \end{aligned}$$

Here, generators of $\mathfrak{k}$ and $\mathfrak{m}$ have a specific operation on the last qubit; i.e., σ_z or I for generators of $\mathfrak{k}$ and σ_x or σ_y for generators of $\mathfrak{m}$. Thus, to determine $\mathfrak{k}$ and $\mathfrak{m}$, the Cartan involution θ and the global Cartan involution Θ can be chosen as follows:

$$\theta(x) = \sigma_{nz} x \sigma_{nz}, \qquad \Theta(X) = \sigma_{nz} X \sigma_{nz}. \tag{3.5}$$

Since $\theta(\sigma_z) = \sigma_z$, $\theta(I) = I$, $\theta(\sigma_x) = -\sigma_x$, $\theta(\sigma_y) = -\sigma_y$, then it can be checked that the above $\mathfrak{k}$ and $\mathfrak{m}$ satisfy relation (3.2) when θ and Θ are chosen as (3.5).

For the number of qubits $n \geq 3$, $\mathfrak{k}$ and $\mathfrak{m}$ have specific patterns, as shown in Fig. 3.1, because all generators defined in (3.3) and (3.4) have these patterns. In Fig. 3.1, each square represents an element of an 8×8 matrix. The white elements are always zero, and the black elements take some value that depends on the input matrix. Note that in contrast to an element of $\exp(\mathfrak{k})$ taking the same pattern as an element of $\mathfrak{k}$, an element of $\exp(\mathfrak{m})$ does not take the same pattern as $\mathfrak{m}$. Therefore, the KAK decomposition can be applied recursively.



$\mathfrak{k}$ and $\exp(\mathfrak{k})$ $\qquad$ $\mathfrak{m}$

Figure 3.1. Patterns of an element of $\mathfrak{k}$, $\exp(\mathfrak{k})$ and $\mathfrak{m}$ for a three-qubit system.

3. Proposed Algorithm for Computing the KAK Decomposition

3.1 Algorithm

In this section, a new constructive algorithm that computes a decomposition based on the $G = KAK$ theorem is presented. Here, a Cartan subalgebra is chosen as $\widetilde{\mathfrak{h}}$, which is different from the $\mathfrak{h}$ used in the KGD. Since Cartan subalgebras are Abelian, they can translate each other by $\widetilde{\mathfrak{h}} = Ad_{T\in\exp(\mathfrak{k})}(\mathfrak{h})$. Here, T is fixed for given $\widetilde{\mathfrak{h}}$ and $\mathfrak{h}$. It should be noted that the $G = KAK$ decomposition for the fixed input $g \in SU(2^n)$ is not unique. The following theorem is the key to the new algorithm.

Theorem 1 *Let $g \in SU(2^n)$ be the input matrix. If g has a global Cartan decomposition $g = km$ $(k \in \exp(\mathfrak{k}), \mathfrak{m} \in \exp(\mathfrak{m}))$, then m^2 is uniquely determined by $m^2 = \Theta(g^\dagger)g$.*

Proof. From (3.2), $\Theta(g^\dagger)g = \Theta(m^\dagger k^\dagger)km = \Theta(m^\dagger)\,\Theta(k^\dagger)km = mk^\dagger km = m^2$. □

Theorem 1 shows that the fixed global Cartan involution Θ only determines m^2. Therefore, arbitrariness remains in the selection of m, and also k. Furthermore, a in (3.1) has also arbitrariness because it follows from a decomposition of m, $m = \widetilde{k}^\dagger a \widetilde{k}$, where $\widetilde{k} \in \exp(\mathfrak{k})$. (See, [34], §AII.3). With Theorem 1, the decomposition in (3.1) is computed as follows:

Algorithm 1 (The KAK decomposition)

Step 1 *Compute $m^2 = \Theta(g^\dagger)g$.*

Step 2 *Decompose $m^2 = pbp^\dagger$ such that $p \in \exp(\mathfrak{k})$ and $b \in \exp(\widetilde{\mathfrak{h}})$.*

Such decomposition always exists because $m^2 \in \exp(\mathfrak{m})$ (cf. [34], Proposition 7.29). Then, p is obtained by computing eigenvectors of m^2. Examples will be shown in Section 3.3.

Step 3 *Find y such that $y^2 = b$ and $y \in \exp(\widetilde{\mathfrak{h}})$.*

The y can be computed by replacing the diagonal blocks of b, when a suitable $\widetilde{\mathfrak{h}}$ is chosen. Examples of selections of $\widetilde{\mathfrak{h}}$ will be described in Section 3.3.

Step 4 *Compute $m = pyp^\dagger$. Here, $m \in \exp(\mathfrak{m})$ because $\Theta(m) = \Theta(p)\Theta(y)\Theta(p^\dagger) = py^\dagger p^\dagger \ = \ m^\dagger$.*

Step 5 *Compute $k = gm^\dagger$. Then, k always satisfies $k \in \exp(\mathfrak{k})$ because $(m^2)^\dagger = g^\dagger\Theta(g)$ and $\Theta(gm^\dagger) = \Theta(g)\Theta(m^\dagger) = g(m^2)^\dagger m = gm^\dagger$.*

Steps 2–4 provide a method for computing the square root of a matrix to find m from m^2. After these procedures,

$$g \ = \ kpyp^\dagger \ = \ \widetilde{k}yp^\dagger. \tag{3.6}$$

Here, $\Theta(\widetilde{k}) = \widetilde{k}$, $\Theta(y) = y^\dagger$, and $\Theta(p^\dagger) = p^\dagger$, so that the decomposition follows the $G = KAK$ theorem. All matrices that appear through the algorithm can be described by using g and Θ. T By using this property, the author shows that the proposed algorithm can automatically reproduce the well-known QFT circuit (See, Section 4.2).

3.2 Proof of Step 2 in Algorithm 1

In the previous section, the author shows that b is a block-diagonal matrix, each block of which is $R_x(2\zeta)$, where $\widetilde{\mathfrak{h}} = \mathrm{span}\{|\mathrm{j}\rangle\langle\mathrm{j}| \otimes \mathrm{i}\sigma_\mathrm{x} \mid \mathrm{j} = 0, \cdots, 2^{\mathrm{n}-1} - 1\} \subset \mathfrak{m}$. Here, the author provides two lemmas and proofs for the eigenvalues of $m^2 = \Theta(G^\dagger)G$, where $G \in SU(2^n)$ to prove that. Let $X = m^2 = \Theta(G^\dagger)G$.

Lemma 1 (Properties of complex eigenvalues of X)

1. *The number of the complex eigenvalues of X, with multiplicity counted, is even. They are of the form α_1, $\overline{\alpha_1}$, α_2, $\overline{\alpha_2}$, $\ldots$, α_t, $\overline{\alpha_t}$, repeated with multiplicity.*

2. *There exists unit length vectors $u_1,\ \ldots,\ u_{2t}$ that are mutually orthogonal and satisfy $\sigma_{nz}u_{2j-1} = u_{2j-1}$ and $\sigma_{nz}u_{2j} = -u_{2j}$. Furthermore, there exists $\zeta_1,\ \ldots,\ \zeta_t \in \mathbb{R}$ satisfying $Xu_{2j-1} = \cos(\zeta_j)u_{2j-1} - i\sin(\zeta_j)u_{2j}$ and $Xu_{2j} = -i\sin(\zeta_j)u_{2j-1} + \cos(\zeta_j)u_{2j}$.*

Proof. Let $\alpha_1,\ \ldots,\ \alpha_{2^n}$ be the eigenvalues of X, repeated with multiplicity. Then, except for the order, the elements of the sequence $\overline{\alpha_1},\ \ldots,\ \overline{\alpha_{2^n}}$ are equal to $\alpha_1,\ \ldots,\ \alpha_{2^n}$ because $\Theta(X) = X^\dagger$. Therefore, the number of the complex eigenvalues, counted with multiplicity, is even. Let the number of the complex eigenvalues be $2t$ and the complex eigenvalues be $\alpha_1,\ \overline{\alpha_1},\ \ldots, \alpha_t,\ \overline{\alpha_t}$. That is, the first statement holds.

Let u be an eigenvector of X corresponding to the eigenvalue α, then $\sigma_{nz}u$ is an eigenvector of X corresponding to the eigenvalue $\overline{\alpha}$ because $X(\sigma_{nz}u) = \sigma_{nz}X^\dagger u = \overline{\alpha_j}(\sigma_{nz}u)$. Now, let W be the eigenspace corresponding to a complex eigenvalue α and $\beta_1, \ldots, \beta_r$ be an orthonormal basis of W. Then, let $\sigma_{nz}\beta_1, \ldots, \sigma_{nz}\beta_r$ be an orthonormal basis of the eigenspace corresponding to the eigenvalue $\overline{\alpha}$. Therefore, $u_{2j-1} = \beta_j + \sigma_{nz}\beta_j$ and $u_{2j} = \beta_j - \sigma_{nz}\beta_j$ for $j = 1, \cdots, r$, span the eigenspace of X corresponding to the eigenvalues α and $\overline{\alpha}$. Furthermore, they are eigenvectors of σ_{nz} because $\sigma_{nz}u_{2j-1} = u_{2j-1}$ and $\sigma_{nz}u_{2j} = -u_{2j}$. On the other hand, the following relations hold:

$$\begin{aligned} Xu_{2j-1} &= \alpha\beta_j + \overline{\alpha}\sigma_{nz}\beta_j = \frac{\alpha+\overline{\alpha}}{2}u_{2j-1} + \frac{\alpha-\overline{\alpha}}{2}u_{2j} = \mathrm{Re}(\alpha)u_{2j-1} + i\mathrm{Im}(\alpha)u_{2j}, \\ Xu_{2j} &= \alpha\beta_j - \overline{\alpha}\sigma_{nz}\beta_j = \frac{\alpha-\overline{\alpha}}{2}u_{2j-1} + \frac{\alpha+\overline{\alpha}}{2}u_{2j} = i\mathrm{Im}(\alpha)u_{2j-1} + \mathrm{Re}(\alpha)u_{2j}. \end{aligned}$$

Therefore, put $\zeta = -\arg(\alpha)$. Then $\cos(\zeta) = \mathrm{Re}(\alpha)$ and $\sin(\zeta) = -\mathrm{Im}(\alpha)$. That is,

$$Xu_{2j-1} = \cos(\zeta)u_{2j-1} - i\sin(\zeta)u_{2j}, \qquad Xu_{2j} = -i\sin(\zeta)u_{2j-1} + \cos(\zeta)u_{2j}.$$

Similar arguments hold for the other complex eigenvalues. □

Lemma 2 (Properties of real eigenvalues of X) *Real eigenvalues of X are* ±1.

1. *The multiplicity of the eigenvalue 1 is even. There exists an orthonormal basis u_{2t+j} ($j = 1, \cdots, 2\mu$) of the eigenspace corresponding to the eigenvalue 1 that satisfies the $\sigma_{nz}u_{2t+2j-1} = u_{2t+2j-1}$ and $\sigma_{nz}u_{2t+2j} = -u_{2t+2j}$.*

2. *The multiplicity of the eigenvalue -1 is even. There exists an orthonormal basis $u_{2t+2\mu+j}$ $(j = 1, \cdots, 2\nu)$ of the eigenspace corresponding to the eigenvalue -1 that satisfies*
   $\sigma_{nz}u_{2t+2\mu+2j-1} = u_{2t+2\mu+2j-1}$ *and* $\sigma_{nz}u_{2t+2\mu+2j} = -u_{2t+2\mu+2j}$*.*

Proof. Lemma 1 implies that the product of all the complex eigenvalues is 1, and thus the product of all the real eigenvalues is 1. Therefore, the real eigenvalues of X are 1 or -1, thus both of the multiplicities of the eigenvalues 1 and -1 are even. Let W_1 and W_2 be the eigenspaces of X corresponding to the eigenvalues of 1 and -1, respectively. Then $\sigma_{nz}W_1 \subset W_1$ and $\sigma_{nz}W_2 \subset W_2$ hold because $\Theta(X) = X^\dagger$. Put $W = W_1 \oplus W_2$. Then $W^\perp$is the direct sum of the eigenspaces for the complex eigenvalues of X. The trace of $\sigma_{nz}|_W$ is 0 because Lemma 1 implies that the trace of $\sigma_{nz}|_{W^\perp}$ is 0, and this implies that the multiplicities of the eigenvalues of 1 and -1 of $\sigma_{nz}|_W$ are equal. Let $W_1 = W_{11} \oplus W_{12}$ and $W_2 = W_{21} \oplus W_{22}$, where W_{11} and W_{21} are the eigenspaces of σ_{nz} corresponding to the eigenvalue 1, and W_{12} and W_{22} are the eigenspaces of σ_{nz} corresponding to the eigenvalue -1. Let $\dim W_{ij}$ be d_{ij}. Then, $d_{11} + d_{21} = d_{12} + d_{22}$.

The similar arguments are applied for $X' = \Theta(G)G^\dagger$. Here, let $W_1' = W_{11}' \oplus W_{12}'$ and $W_2' = W_{21}' \oplus W_{22}'$, where W_{11}' and W_{21}' are the eigenspaces of σ_{nz} corresponding to the eigenvalue 1, and W_{12}' and W_{22}' are the eigenspaces of σ_{nz} corresponding to the eigenvalue -1. Let $\dim W_{ij}'$ be d_{ij}'. Then, $d_{11}' + d_{21}' = d_{12}' + d_{22}'$.

From Lemma 3 below, $d_{11} = d_{11}'$, $d_{12} = d_{12}'$, $d_{21} = d_{22}'$, and $d_{22} = d_{21}'$. Then, $d_{11} = d_{12}$ and $d_{21} = d_{22}$. Therefore, the statements of the lemma hold. □

Lemma 3 *Let $X = \Theta(G^\dagger)G$ and $X' = \Theta(G)G^\dagger$.*

1. *If $Xu = u$ and $\sigma_{nz}u = u$, then $X'(Gu) = Gu$ and $\sigma_{nz}(Gu) = Gu$.*

2. *If $Xu = u$ and $\sigma_{nz}u = -u$, then $X'(Gu) = Gu$ and $\sigma_{nz}(Gu) = -Gu$.*

3. *If $Xu = -u$ and $\sigma_{nz}u = u$, then $X'(Gu) = -Gu$ and $\sigma_{nz}(Gu) = -Gu$.*

4. *If $Xu = -u$ and $\sigma_{nz}u = -u$, then $X'(Gu) = -Gu$ and $\sigma_{nz}(Gu) = Gu$.*

Proof. First, the author proves the statements for the eigenvalues and eigenvectors of X'. Since $G^\dagger X'G = \Theta(X)$ holds, $(X'(Gu), Gu) = (G^\dagger X'Gu, u) =$

$(\Theta(X)u,u) = (X^{\dagger}u,u) = (u,Xu)$. Thus, the equation $Xu = \epsilon u$, where $\epsilon = \pm 1$, implies $(X'(Gu),Gu) = \epsilon(u,u)$. On the other hand, the Cauchy-Schwarz inequality implies $|(X'(Gu),Gu)| \leq \|X'(Gu)\| \cdot \|Gu\|$. Since the right-hand side is equal to $\|Gu\|^2 = \|u\|^2 = (u,u) = |(X'(Gu),Gu)|$, the equality $|(X'(Gu),Gu)| = \|X'(Gu)\| \cdot \|Gu\|$ holds. Therefore, $X'(Gu) = \alpha(Gu)$ for some $\alpha \in \mathbb{C}$; that is, Gu is an eigenvector of X' corresponding to the eigenvalue α. The equality $\alpha = \epsilon$ follows $\epsilon(u,u) = (X'(Gu),Gu) = (\alpha Gu,Gu) = \alpha(Gu,Gu) = \alpha(u,u)$.

To prove the statements for the eigenvalues and eigenvectors of σ_{nz}, the following equations are used.

$$(\sigma_{nz}(Gu),Gu) = (G^{\dagger}\sigma_{nz}Gu,u) = (\sigma_{nz}Xu,u) = (Xu,\sigma_{nz}u).$$

The equations $Xu = \epsilon u$ and $\sigma_{nz}u = \epsilon' u$, where ϵ, $\epsilon' = \pm 1$, imply $(\sigma_{nz}(Gu),Gu) = \epsilon\epsilon'(u,u)$. Similar arguments are made for the eigenvalues and eigenvectors of X', then $\sigma_{nz}(Gu) = \epsilon\epsilon'(Gu)$. □

3.3 Methods for Performing Steps 2 and 3 in Algorithm 1

Above, a method for computing the $G = KAK$ decomposition was provided but a concrete method for computing p, b, and y in steps 2 and 3 was not described. To show examples of such concrete methods, In this section, the author presents examples of such concrete methods for fixed Θ and $\widetilde{\mathfrak{h}}$. Here, examples for computing the KGD are treated, *i.e.*, Θ is chosen as in (3.5) . The author shows two methods that compute p, b, y for particular selections of $\widetilde{\mathfrak{h}}$; i.e., $\widetilde{\mathfrak{h}}_1 = \mathrm{span}\,\{|\mathrm{j}\rangle\langle\mathrm{j}| \otimes \mathrm{i}\sigma_{\mathrm{x}} \,|\, \mathrm{j} = 0, \cdots, 2^{\mathrm{n}-1} - 1\}$ and $\widetilde{\mathfrak{h}}_2 = \mathrm{span}\,\{|\mathrm{j}\rangle\langle\mathrm{j}| \otimes \mathrm{i}(\sigma_{\mathrm{x}} \otimes \sigma_{\mathrm{x}} + \sigma_{\mathrm{y}} \otimes \sigma_{\mathrm{y}}) \,|\, \mathrm{j} = 0, \cdots, 2^{\mathrm{n}-2} - 1\}$. Note that the second selection is not always possible because it demands that all eigenvalues appearing in m^2 in step 1 should be duplicated twice.

3.4 Example 1

Let Θ be as in (3.5), and $\widetilde{\mathfrak{h}}_1$ as $\mathrm{span}\,\{|\mathrm{j}\rangle\langle\mathrm{j}| \otimes \mathrm{i}\sigma_{\mathrm{x}} \,|\, \mathrm{j} = 0, \cdots, 2^{\mathrm{n}-1} - 1\}$. Then, p, b, and y in steps 2 and 3 can be computed as follows:

(i) Compute eigenvalue decomposition of m^2.

Let $m^2 = \widetilde{p}d\widetilde{p}^{\dagger}$ be eigenvalue decomposition and $\mu_1, \cdots, \mu_N$ be the columns of $\widetilde{p}$, where $N = 2^n$. Then, d is a diagonal matrix in which diagonals have eigenvalues of m^2 and all μ_j's are eigenvectors of m^2 and mutually orthogonal.

(ii) Normalize all μ_j's by

$$\nu_{2j-1} = \frac{\mu_j + \sigma_{nz}\mu_j}{\|\mu_j + \sigma_{nz}\mu_j\|}, \qquad \nu_{2j} = \frac{\mu_j - \sigma_{nz}\mu_j}{\|\mu_j - \sigma_{nz}\mu_j\|}. \tag{3.7}$$

(iii) For all ν_j's that are associated with imaginary eigenvalues,

(a) let W_1, W_2, and W_3 be sets of vectors such that

$$\begin{aligned} W_1 &= \{\nu_j \mid \sigma_{nz}\nu_j = \nu_j\}, \\ W_2 &= \{\nu_j \mid \sigma_{nz}\nu_j = -\nu_j\}, \\ W_3 &= \{\nu_j \mid \sigma_{nz}\nu_j \neq \pm\nu_j\}. \end{aligned}$$

(b) For each $w \in W_3$, compute

$$\nu^+ = \frac{w + \sigma_{nz}w}{\|w + \sigma_{nz}w\|}, \qquad \nu^- = \frac{w - \sigma_{nz}w}{\|w - \sigma_{nz}w\|}. \tag{3.8}$$

Here, $\| \cdot \|$ denotes the length of a vector. Then,

- if all elements in W_1 and u^+ are linearly independent, then $W_1 = W_1 \cup \{\nu^+\}$;
- if all elements in W_2 and u^- are linearly independent, then $W_2 = W_2 \cup \{\nu^-\}$.

(iv) Repeat steps (a) and (b) for all μ_j's that are associated with positive real eigenvalues.

(v) Repeat steps (a) and (b) for all μ_j's that are associated with negative real eigenvalues.

(vi) Let $p = (\upsilon_1, \upsilon_2, \cdots, \upsilon_N)$, where $\upsilon_{2j-1} \in W_1$ and $\upsilon_{2j} \in W_2$, for $j = 1, \cdots, N/2$.

The computation procedure follows from Section 3.2 Since $\sigma_{nz}\upsilon_{2j-1} = \upsilon_{2j-1}$

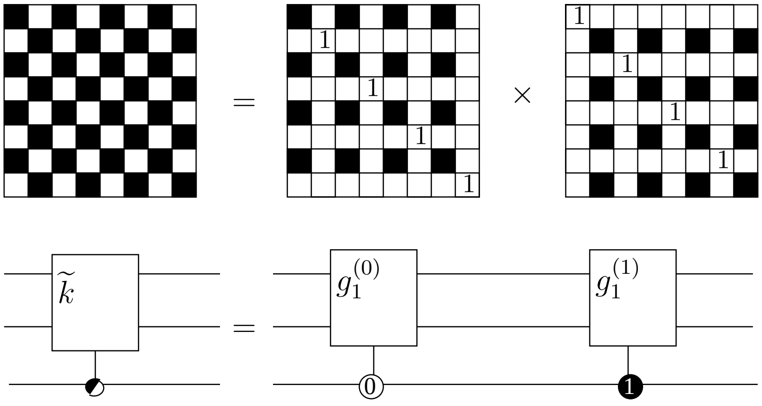


Figure 3.2. Image of decomposition of $\widetilde{k} \in \exp(\mathfrak{k})$ for a three qubit system.

and $\sigma_{nz}\upsilon_{2j} = -\upsilon_{2j}$, it can easily be checked that $\Theta(p) = p$. Then, $b = p^\dagger m^2 p$ satisfy $\Theta(b) = b^\dagger$ and $b \in \exp(\widetilde{\mathfrak{h}})$, where $b = \sum_{j=0}^{2^{n-1}} |j\rangle\langle j| \otimes R_x(2\zeta_j)$ $(0 \le \zeta_j < \pi)$.

(vii) Compute y by replacing all $R_x(2\zeta_j)$ appears in b with $R_x(\zeta_j)$.
Since $\Theta(R_x(2\zeta_j)) = R_x^\dagger(2\zeta_j)$, $R_x^2(\zeta_j) = R_x(2\zeta_j)$, then y satisfies $y^2 = b$ and $y \in \exp(\widetilde{\mathfrak{h}}_1)$.

In step (vi), one may notice that, when $R_x(\pi)$ appears in b, thus $R_y(\pi/2)$ is used instead of $R_x(\pi/2)$ as a replacement rule.

Since $\widetilde{k}$ and $p^\dagger \in$ are elements of $\exp(\mathfrak{k})$ that has the specific pattern as shown in Fig. 3.1, they have the following decomposition:

$$\widetilde{k} = g_1^{(0)} \otimes |0\rangle\langle 0| + g_1^{(1)} \otimes |1\rangle\langle 1|, \qquad p^\dagger = g_2^{(0)} \otimes |0\rangle\langle 0| + g_2^{(1)} \otimes |1\rangle\langle 1|, \tag{3.9}$$

where $g_1^{(j)}, g_2^{(j)} \in SU(2^{n-1})$ for $\widetilde{k}, p^\dagger \in SU(2^n)$ $(j = 0$ or $1)$. Here, $g_1^{(0)}$ and $g_2^{(0)}$ are composed of nonzero elements (black squares in Fig. 3.1) of odd rows, and $g_1^{(1)}$ and $g_2^{(1)}$ are composed of nonzero elements of even rows. Fig. 3.2 illustrates the decomposition of $\widetilde{k}$ for a three-qubit system.

An elements in $\exp(\mathfrak{h}_1)$ can be regarded as a uniformly controlled one-qubit operation as shown in Fig. 3.3. The KAK decomposition that is obtained through the above computation is then corresponds to a quantum circuit as shown in Fig. 3.4. This figure shows the image of a decomposition in (3.1) for a three-qubit system, when Θ is chosen as in (3.5) and a Cartan subalgebra $\mathfrak{h}_1$ as span $\{|\mathrm{j}\rangle\langle\mathrm{j}| \otimes$

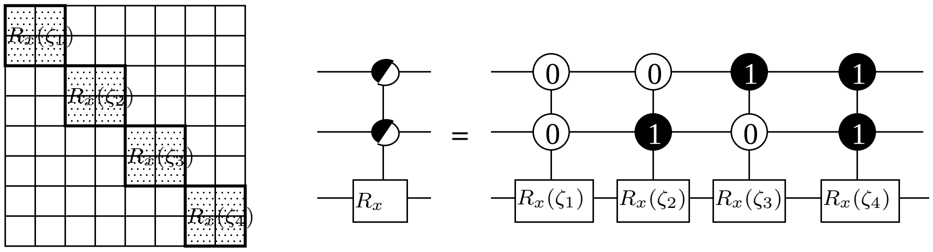


Figure 3.3. A quantum circuit for $y \in \exp(\mathfrak{h}_1)$.

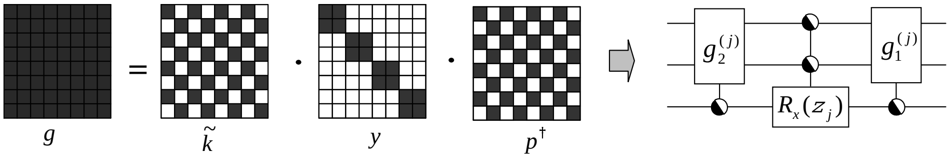


Figure 3.4. Image of a decomposition.

$\mathrm{i}\sigma_\mathrm{x} \mid \mathrm{j} = 0, \cdots, 2^{\mathrm{n}-1} - 1\}$. The matrices uses the same notation as in Fig. 3.1 to represent the properties. In the quantum circuit, the symbol of the control qubit represents the uniformly controlled rotations [36,38]. $g_\ell^{(j-1)} \in SU(4)$ ($\ell, j = 1$ or 2) are applied selectively; that is, $g_1^{(0)}$ and $g_2^{(0)}$ are applied when the third qubit is $|0\rangle$, whereas $g_1^{(1)}$ and $g_2^{(1)}$ are applied when it is $|1\rangle$.

By applying the decomposition in (3.6) recursively to elements $g_1^{(j)}, g_2^{(j)} \in SU(2^{n-1})$ (j=0 or 1), a sequence of uniformly controlled rotations like in Ref. [36] (in Fig. 13) is obtained, except that R_x is used instead of R_y in the proposed algorithm. The full decomposition of these uniformly controlled rotations into elementary gates has been provided by Möttönen *et al.* [36, 38]. Also, if the quantum multiplexor decomposition is applied to $\widetilde{k}$ and $p^\dagger$ in (3.1) after the order of qubits are changed, the produced circuit is the same as that in Fig. 2 in Ref. [42], except that rotation R_y is used instead of R_x. Therefore, the number of elementary gates needed to compose $g \in SU(2^n)$ in the proposed method is $O(4^n)$, which is the same as in Refs. [37, 42].

3.5 Example 2

Here, an another example of methods for computing p, b, and y in Section 3.1 will be described. Here, $\widetilde{\mathfrak{h}}_2 = \mathrm{span}\,\{|\mathrm{j}\rangle\langle\mathrm{j}| \otimes \mathrm{i}(\sigma_\mathrm{x} \otimes \sigma_\mathrm{x} + \sigma_\mathrm{y} \otimes \sigma_\mathrm{y}) \mid \mathrm{j} = 0, \cdots, 2^{\mathrm{n}-1} - 1\}$.

The Cartan involution Θ is the same as in (3.5). The decomposition of this type is chosen as an example of a decomposition of the QFT. In this case, only the steps (vi) and (vii) in Section 3.4 is replaced as follows:

(vi) Let $p = (\upsilon_1, \upsilon_2, \cdots, \upsilon_N)$, for $j = 1, \cdots, N/4$,

- $\nu_{4j-3} \in W_1$ and it is associated with positive eigenvalues,
- $\nu_{4j-2} \in W_2$ and it is associated with negative eigenvalues,
- $\nu_{4j-1} \in W_1$ and it is associated with negative eigenvalues,
- $\nu_{4j} \in W_2$ and it is associated with positive eigenvalues.

Then, p also satisfies $\Theta(p) = p$, and $b = p^\dagger m^2 p$ is a block-diagonal matrix, in which the diagonals are constructed from the 4×4 matrix

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(2\zeta_j) & i\sin(2\zeta_j) & 0 \\ 0 & i\sin(2\zeta_j) & \cos(2\zeta_j) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} = \exp(i\zeta_j(\sigma_x \otimes \sigma_x + \sigma_y \otimes \sigma_y)).$$

Here, the middle part of the above matrix is $R_x(2\zeta_j)$.

(vii) Compute y by replacing $R_x(2\zeta_j)$ in b with $R_x(\zeta_j)$.

4. Application to the Design of Circuits for the QFT

In this section, the author shows that the new algorithm presented in this chapter can automatically reproduce the well-known QFT circuit. All the matrices that appear through the proposed algorithm can be described using the input matrix g and Θ. In contrast, it is difficult to describe all the matrices that appear through the CSD-based algorithm because, as shown in (2.8), the input matrix g has to be divided into four square matrices and the SVD has to be applied to each partitioned matrix. It is difficult to formulate each partitioned matrix, *i.e.*, G_{11}, G_{12}, G_{21}, and G_{22}, and that makes it difficult to describe suitable the SVD for each partitioned matrix.

Fortunately, the n-qubit QFT is a very special matrix that has the following property: If the order of qubits is permuted, then each partitioned matrix can be described using the $(n-1)$-qubit QFT. By using the feature, the author provides a decomposition of the QFT by the CSD-based algorithm. This is shown in Section 4.3.

4.1 Notation

The QFT on n qubits, F_n, is a $2^n \times 2^n$ matrix such that

$$F_n \;=\; \left(\frac{1}{\sqrt{2^n}}\omega_n^{(j-1)(\ell-1)}\right)_{j\ell}, \qquad \text{where } \ \omega_{\rm n} = \exp\left(\frac{2\pi {\rm i}}{2^{\rm n}}\right). \qquad (3.10)$$

Let Q_n be a $2^n \times 2^n$ permutation matrix: $Q_n \;=\; \chi_{n-1}^n \cdots \chi_2^n \chi_1^n$, where χ_j^k is the SWAP gate applied to the j-th and the k-th qubits. Let $H_1 = H \otimes I^{\otimes(n-1)}$, then (3.10) is written as

$$F_n \;=\; \frac{1}{\sqrt{2}}\begin{pmatrix} F_{n-1} & \Omega_{n-1}F_{n-1} \\ F_{n-1} & -\Omega_{n-1}F_{n-1}\end{pmatrix} Q_n \;=\; H_1 D_n (I \otimes F_{n-1}) Q_n, \qquad (3.11)$$

where

$$D_n = \begin{pmatrix} I^{\otimes(n-1)} & 0 \\ 0 & \Omega_{n-1}\end{pmatrix}, \qquad \Omega_{n-1} = diag\,(1, \omega_n, \cdots, \omega_n^{2^{n-1}-1}).$$

This notation follows from Section 4.6.4 in Ref. [22].

4.2 Decomposition of the QFT by the Proposed Method

According to Section 3.1, the decomposition of F_n can be computed as follows:

1. Compute $m^2 = \Theta(F_n^\dagger)F_n$.
   Let S be $(I \otimes F_{n-1})Q_n$. Since $\Theta(H_1) = H_1$, $\Theta(D_n) = D_n$, and $\sigma_{1z}S = S\sigma_{nz}$, then $m^2 = S^\dagger \sigma_{1z}\sigma_{nz}S$. All column vectors of $S^\dagger$ are then eigenvectors of m^2 because $\sigma_{1z}\sigma_{nz}$ is a diagonal matrix in which diagonal elements are eigenvalues of m^2.

2. Decompose $m^2 = pbp^\dagger$ such that $\Theta(p) = p$ and $\Theta(b) = b^\dagger$.

   Let p be $S^\dagger Q_n$ (This selection was done so that p satisfies $\Theta(p) = p$ and follows Section 3.5), then $b = p^\dagger m^2 p = Q_n^\dagger \sigma_{1z}\sigma_{nz} Q_n = I^{\otimes(n-2)} \otimes$ $diag(1, -1, -1, 1)$.

3. Choose y such that $\Theta(y) = y^\dagger$. According to the step (vi) in Section 3.5, then

$$y = I^{\otimes(n-2)} \otimes \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & i & 0 \\ 0 & i & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} = I^{\otimes(n-2)} \otimes \exp(\pi(\sigma_x \otimes \sigma_x + \sigma_y \otimes \sigma_y)/4).$$

   This is obtained by replacing each 4×4 diagonal block of v, *i.e.*, $b = diag(1, -1, -1, 1)$, with $y = \exp(\pi(\sigma_x \otimes \sigma_x + \sigma_y \otimes \sigma_y)/4)$. It can easily be checked that $\Theta(y) = y^\dagger$ and $y^2 = b$, because $\Theta(y) = y^\dagger$ and $y^2 = b$.

4. Compute $m = pyp^\dagger$.

5. Compute $k = gm^\dagger = F_n m^\dagger$.

Then, $\widetilde{k} = kp = (F_n p y^\dagger p^\dagger)p = F_n p y^\dagger$, so the following decomposition is obtained.

$$\begin{aligned} F_n &= \widetilde{k} y p^\dagger = (H_1 D_n Q_n y^\dagger) y (Q_n^\dagger S) = H_1 D_n S, \\ &= H_1 D_n (I \otimes F_{n-1}) Q_n. \qquad (3.12) \end{aligned}$$

The similar decomposition is applied to F_j, for $j = n-1, n-2, \cdots, 2$. Next, the author shows a decomposition of D_n. D_n is controlled-Ω_j (where $j = n-1, n-2, \cdots, 2$), so it suffices to consider the decomposition of Ω_j. Since $\Omega_j \in \exp(\mathfrak{k})$ (it follows from $\Theta(\Omega_j) = \Omega_j$), the decomposition in (3.9) is applied to Ω_j. Consider Ω_3 as an example, then $\Omega_3 = g_1^{(0)} \otimes |0\rangle\langle 0| + g_1^{(1)} \otimes |1\rangle\langle 1|$, where $g_1^{(0)} = diag\ (1, \omega^2, \omega^4, \omega^6)$ and $g_1^{(1)} = diag\ (\omega, \omega^3, \omega^5, \omega^7) = \omega \cdot diag\ (1, \omega^2, \omega^4, \omega^6)$. Then, $\Omega_3 = diag\ (1, \omega^2, \omega^4, \omega^6) \otimes diag\ (1, \omega)$. Similarly, since $diag\ (1, \omega^2, \omega^4, \omega^6)$ is also an element of $\exp(\mathfrak{k})$, it is decomposed into $diag\ (1, \omega^4) \otimes diag\ (1, \omega^2)$. Therefore, Ω_{n-1} is composed of $n-1$ single-qubit rotations as follows:

$$\Omega_{n-1} = \begin{pmatrix} 1 & 0 \\ 0 & \omega_n^{2^{n-1}-1} \end{pmatrix} \otimes \cdots \otimes \begin{pmatrix} 1 & 0 \\ 0 & \omega_n^{2^{j-1}} \end{pmatrix} \otimes \cdots \otimes \begin{pmatrix} 1 & 0 \\ 0 & \omega_n^2 \end{pmatrix} \otimes \begin{pmatrix} 1 & 0 \\ 0 & \omega_n \end{pmatrix}. \qquad (3.13)$$

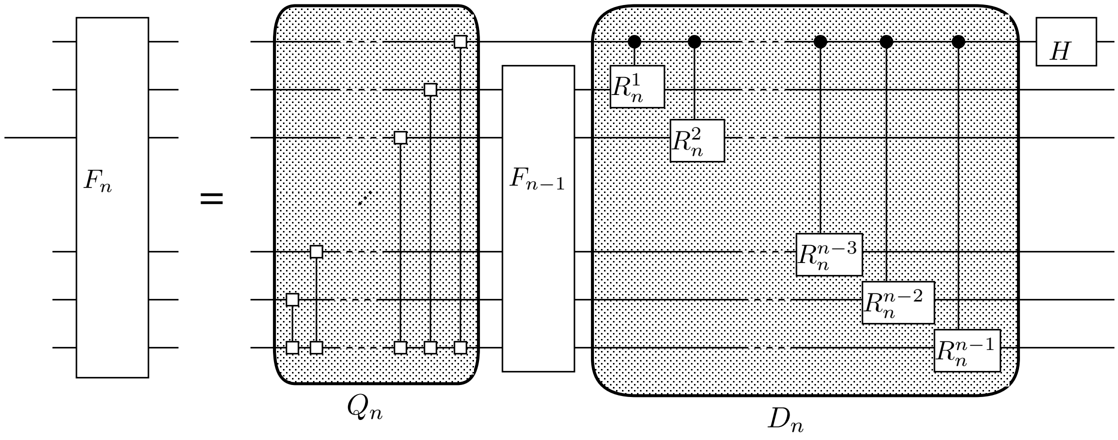


Figure 3.5. Decomposition of the QFT by the proposed algorithm.

The circuit obtained from the above decomposition is shown in Fig. 3.5. Here, $R_n^j = diag\ (1, \omega_n^j)$, i.e., a single-qubit rotation, and Q_n is composed of $n-1$ SWAP gates. A similar decomposition is applied to F_j (for $j = n-1, n-2, \cdots, 2$). Finally, a full decomposition of the QFT composed of n Hadamard gates, $\frac{1}{2}n(n-1)$ controlled-rotations, and $\lfloor \frac{n}{2} \rfloor$ SWAP gates is obtained. Here, SWAP gates that appeared in a sequence of permutations $Q_n Q_{n-1} \cdots Q_2$ were optimized. Thus, the number of elementary gates that appear in Fig. 3.5 is $O(n^2)$.

4.3 Decomposition of the QFT by the Previous Methods

As a comparison between the proposed methods and the previous methods, this section describes the decomposition of the QFT by the previous methods. Two methods are treated: One is the CSD provided in [36, 37, 44, 46] and the other is the quantum Shannon Decomposition (QSD) [42] .

Decomposition of the QFT by the CSD can easily be denoted if $Q_n^\dagger$ is applied to the input matrix F_n beforehand.

$$\begin{aligned} F_n Q_n^\dagger &= U \Sigma V \\ &= \begin{pmatrix} I^{\otimes n-1} & 0 \\ 0 & I^{\otimes n-1} \end{pmatrix} \cdot \frac{1}{\sqrt{2}} \begin{pmatrix} I^{\otimes (n-1)} & -I^{\otimes (n-1)} \\ I^{\otimes (n-1)} & I^{\otimes (n-1)} \end{pmatrix} \cdot \begin{pmatrix} F_{n-1} & 0 \\ 0 & -\Omega_{n-1} F_{n-1} \end{pmatrix}, \\ &= (H\sigma_z \otimes I^{\otimes (n-1)})(\sigma_z \otimes I^{\otimes (n-1)}) D_n (I \otimes F_{n-1}) \ = \ H_1 S. \qquad (3.14) \end{aligned}$$

It can easily be checked that (3.14) satisfies the definition of the CSD, where all θ_j's appear in c and s in (2.8) are $\pi/4$. Although $Q_j^\dagger$ have to be applied beforehand when the CSD is recursively applied to F_j $(j = 1, \cdots, n-1)$, the well-known QFT circuit can be obtained. Based on the feature, Tucci reproduced the well-known QFT circuit using the CSD-based algorithm [45, 46].

The QSD is a method that combines the CSD and the quantum multiplexor decomposition. First, the CSD for the input matrix is computed and then the quantum multiplexor decomposition (cf. [42], Theorem 12), *i.e.*,

$$\begin{pmatrix} U_1 & 0 \\ 0 & U_2 \end{pmatrix} = \begin{pmatrix} A & 0 \\ 0 & A \end{pmatrix} \begin{pmatrix} D & 0 \\ 0 & D^\dagger \end{pmatrix} \begin{pmatrix} B & 0 \\ 0 & B \end{pmatrix}, \tag{3.15}$$

is applied to U and V in (2.8). Here, $U_1 U_2^\dagger = A D^2 A^\dagger$ and $B = D A^\dagger U_2$. In the QFT, the decomposition in (3.15) is applied to V because U is an identity matrix in (3.14). Then, $A = I^{\otimes(n-1)}$, $D = \sqrt{-\Omega_{n-1}^\dagger}$, and $B = \sqrt{-\Omega_{n-1}} F_{n-1}$ in (3.15). Here, $\sqrt{-\Omega_{n-1}}$ is a $2^{n-1} \times 2^{n-1}$ diagonal matrix whose (j,j)-th component is $i\omega_{n+1}^{j-1}$, $(j = 1, \cdots, 2^{n-1})$. Therefore, the decomposition of the QFT by the QSD is as follows:

$$F_n Q_n^\dagger = \exp(i\sigma_y \otimes \delta_2) \exp(-i\sigma_z \otimes \delta_3)(I \otimes v_4), \tag{3.16}$$

where $v_4 = \sqrt{-\Omega_{n-1}} F_{n-1}$ and δ_2 and δ_3 are $2^{n-1} \times 2^{n-1}$ diagonal matrices. Each element of δ_2 is $\pi/2^{n-1}$ and each (j,j)-th element of δ_3 is $-j\pi/2^{n+1}$. Furthermore, $\sqrt{\Omega_{n-1}}$ is composed of $n-1$ single-qubit rotations as follows:

$$\sqrt{-\Omega_{n-1}} = i \begin{pmatrix} 1 & 0 \\ 0 & \omega_{n+1}^{2^{n-1}} \end{pmatrix} \otimes \cdots \otimes \begin{pmatrix} 1 & 0 \\ 0 & \omega_{n+1}^{2^{j-1}} \end{pmatrix} \otimes \cdots \otimes \begin{pmatrix} 1 & 0 \\ 0 & \omega_{n+1}^{2} \end{pmatrix} \otimes \begin{pmatrix} 1 & 0 \\ 0 & \omega_{n+1} \end{pmatrix}.$$

Here, (3.16) is also equal to the well-known QFT decomposition (3.11) after optimization as shown in Fig. 3.6. Here, $Q_n^\dagger$ is moved to the right-hand-side by inverting it. The dark-gray block in the middle circuit can be simplified. The simplified block is described by the same color in the rightmost circuit. Here, $D = \sqrt{-\Omega_{n-1}}$.

To obtain the decomposition in (3.16), note that Q_j $(j = 0, \cdots, n)$ must be applied beforehand for F_j.

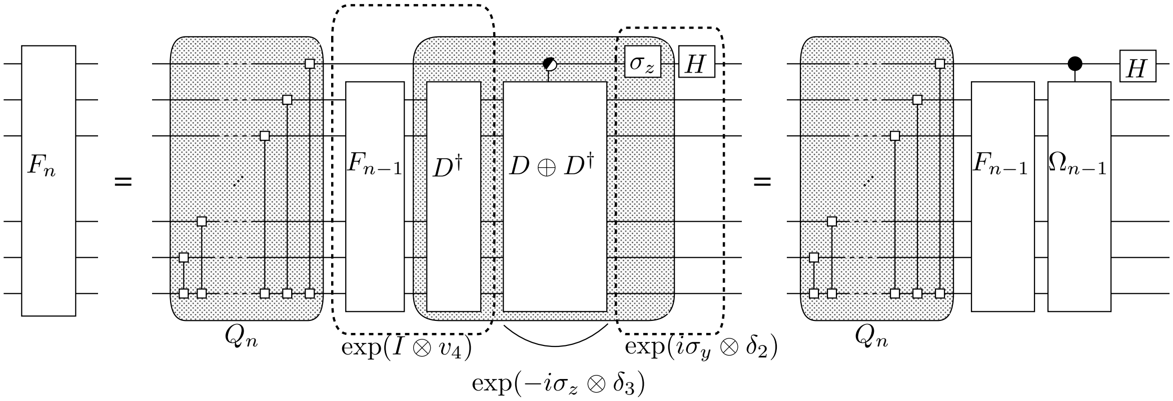


Figure 3.6. Decomposition of the QFT by the QSD.

5. Summary

The author introduces a new algorithm for computing any type-**AIII** KAK decomposition according to the given global Cartan involution Θ. Recursively performing the decomposition leads to a quantum circuit composed of uniformly controlled rotations. The proposed algorithm can derive any matrix decomposition corresponding to the type-**AIII** KAK decomposition, and it contains the CSD and the KGD as its special cases. This is because the proposed algorithm contains arbitrariness in selecting the Cartan subalgebra $\mathfrak{h}$ and a square root matrix m for the given Cartan involution Θ, where m is a matrix derived from a global Cartan decomposition $g = km$. Two methods for computing a square root matrix are also presented.

Although the correctness of the proposed algorithm depends on Lie group theory, the main methods involved are eigenvalue decomposition and a simple replacement rule. Thus, the decomposition can be computed without knowledge of Lie group theory.

As an example, the author shows that the new algorithm automatically reproduces the well-known QFT circuit for arbitrary n-qubits. When using the CSD-based algorithms, some permutations must be applied beforehand in order to reproduce the circuit. The same technique can not always be used to describe the canonical form of the decomposition for a given matrix because matrices do not always have a convenient form like the QFT. The proposed algorithm might be useful in showing the effectiveness of The $G = KAK$ matrix decompositions for other particular input matrices, because all matrices appearing through the algorithm can be described using an input matrix g and the given Cartan involution Θ.

Chapter 4

Synthesis Method for the d-Level Quantum System

1. Introduction

Quantum circuits are fundamental tools for describing quantum algorithms and for understanding the power and limitations of quantum computation. In general, a quantum circuit is described as a sequence of quantum operations (elementary gates) that act on one or two qubits, where a qubit is the quantum mechanical analogue of the conventional bit. However, in some cases, quantum computations, *e.g.*, the quantum Fourier transform (QFT) [25] and the Schur transform [4, 5], may be described naturally as operations that act on the d-level quantum system (qudits). The number of coupled qudits used to implement a quantum operation is less than that of qubits [7]. In addition, if the approximation of the radix-d QFT, *i.e.*, the QFT that acts on n qudits, is considered, it provides better approximation properties than the binary one because the error magnitude decreases exponentially with d [50].

In this chapter, the author focuses a method, which is based on matrix decompositions, for synthesizing quantum circuits for the d-level quantum system ($d > 2$). For $d = 2$ (qubits), there have been many studies of quantum circuit synthesis based on matrix decompositions, such as QR decomposition [44], the cosine-sine decomposition (CSD) [13, 37, 42, 46, 47], and the Khaneja-Glaser decomposition (KGD) [33]. The best-practice quantum circuit for an arbitrary operation on n qubits, which is composed of $O(4^n)$ elementary gates, was first found by means of the CSD [37, 42]. Later, Bullock showed that the KGD is a variant of the CSD and that they can be translated to each other [12]. Variants of the CSD, such as the KGD, are also useful for synthesizing asymptotically optimal quantum circuits for the two-level quantum system and those methods are useful for designing a polynomial-size quantum circuit for the radix-2 QFT in Chapter 3 and [45].

In contrast, for the arbitrary d-level quantum systems ($d > 2$), there have been some studies concerning synthesis methods of quantum circuits, but whether the CSD is useful for synthesizing quantum circuits is unclear. Non-CSD methods were provided by Brennen *et al*: spectral decomposition [15] and the 'Triangle' algorithm [10]. The latter algorithm is based on QR decomposition and Householder transformation. By using these methods, they showed that an arbitrary operation on n qudits can be implemented using $O(d^{2n})$ elementary gates. They defined the elementary gates as a set of arbitrary one-qudit operations and a two-qudit operation called the CINC gate, where $\text{INC}|a\rangle = |(a+1) \mod d\rangle$ and the CINC is a controlled unitary operation that applies INC to the target qudit when the control qudit is $|d-1\rangle$. However, in [10], they mentioned that, "Moreover, the current best-practice n-qubit circuits exploit the CSD, yet technical difficulties with the tensor product structure make it quite unclear whether this matrix decomposition is useful for qudits."

A synthesis method based on the CSD for the d-level quantum system has been proposed by Khan *et al.* [30, 31], but it is inefficient when $d > 2$. The size of the produced quantum circuit was not discussed in that paper, but it can be checked that it is $O(\alpha^n d^{2n})$, where $\alpha = 2^{d-1}/d$ and n is the number of qudits. Here, the elementary gates are the same as those in [10, 15]. The size of the produced circuit by Khan *et al.*'s CSD-based method is exponentially larger than that produced by Brennen *et al.*'s non-CSD based method.

Here, the author improves the CSD-based synthesis method for the d-level quantum system by using a balanced partitioning. With the proposed method, the size of the quantum circuit produced by the synthesis method is reduced to $O(n^{2+\log_2 d} d^{2n})$ when d is odd, to $O(d^{2n})$ when d is a power of two, and to $O(nd^{2n})$ otherwise, for the number n of qudits. Furthermore, when n is a small, the considerable number of two-qudit operations (CINC gates) appearing in the quantum circuit produced by the proposed method can be reduced. For example, when $d = 3$ and $n = 2$, the number of CINC gates used in the quantum circuit produced by the proposed method is 36, where 156 CINC gates are required in the quantum circuit produced by the previous best-practice method (*i.e.*, Brennen *et al.*'s non-CSD based method).

The main difference between the author's method and Khan *et al.*'s CSD-based method is the partition size. Let G be an $m \times m$ unitary matrix, then the CSD of G, denoted by $G = UDV$, is as follows:

$$\begin{array}{c} \\ \ell \\ m-\ell \end{array} \begin{array}{c} \begin{array}{cc} \ell & m-\ell \end{array} \\ \begin{pmatrix} G_{11} & G_{12} \\ G_{21} & G_{22} \end{pmatrix} \end{array} = \begin{pmatrix} U_1 & 0 \\ 0 & U_2 \end{pmatrix} \begin{pmatrix} D_{11} & D_{12} \\ D_{21} & D_{22} \end{pmatrix} \begin{pmatrix} V_1 & 0 \\ 0 & V_2 \end{pmatrix}.$$

Throughout this chapter, let ℓ be the partition size. The input matrix is partitioned into four sub-blocks and the size of each sub-block depends on ℓ. In the CSD described above, D_{ij} $(i, j = 1 \text{ or } 2)$ is a diagonal matrix and for each sub-block of G, $G_{ij} = U_i D_{ij} V_j$ $(i, j = 1 \text{ or } 2)$ is the singular value decomposition (SVD).

In the CSD-based method, the CSD is recursively applied to sub-blocks of matrices appearing in the previous applications of the CSD. For the two-level quantum system, let the input matrix be a $2^n \times 2^n$ matrix, where n denotes the number of qubits, then ℓ is chosen as 2^{n-1}. The input matrix is partitioned into four equally sized sub-blocks and U_1, U_2, V_1, and V_2 in (4.1) are $2^{n-1} \times 2^{n-1}$ unitary matrices. In the next step, these $2^{n-1} \times 2^{n-1}$ unitary matrices are decomposed with the partition size $\ell = 2^{n-2}$ and then, $2^{n-2} \times 2^{n-2}$ sub-matrices are appeared. These procedure will be repeated until each sub-block is of size 2×2. The number of repetitions is $n-1$ and the size of the quantum circuit produced by the synthesis method is $4^n - 1$. Note that the size of the quantum circuit produced by the CSD-based synthesis method increases exponentially in the number of repetitions of the CSD.

In contrast, for the arbitrary d-level quantum systems $(d > 2)$, the input matrix cannot partitioned into four equally sized sub-blocks. Since the CSD have to be repeated until each sub-block is of size $d \times d$, the number of repetitions depends on the largest sub-block of U_1, U_2, V_1, and V_2. Thus, to reduce the number of repetitions, it is better to choose the partition size ℓ such that the size of each sub-block is nearly-balanced and is a power of d. The former condition is required to reduce the number of repetitions. The latter condition is required to decompose $d \times d$ sub-blocks after some repetitions.

Let n be the number of qudits and suppose that the input matrix is a $d^n \times d^n$ unitary matrix. In Khan *et al.*'s method, $\ell = d^{n-1}$. Thus, the size of the largest sub-block is $(d-1)d^{n-1} \times (d-1)d^{n-1}$ and the size of the smallest sub-block is $d^{n-1} \times d^{n-1}$. Here, the author proposes a new partitioning method that is more efficient than the previous ones. The partition size is chosen as $\ell = d^{\lfloor n/2 \rfloor} \lfloor d^{\lceil n/2 \rceil}/2 \rfloor$. Then, the largest sub-block is $d^{\lfloor n/2 \rfloor} \lceil d^{\lceil n/2 \rceil}/2 \rceil \times d^{\lfloor n/2 \rfloor} \lceil d^{\lceil n/2 \rceil}/2 \rceil$ and the smallest sub-block is $d^{\lfloor n/2 \rfloor} \lfloor d^{\lceil n/2 \rceil}/2 \rfloor \times d^{\lfloor n/2 \rfloor} \lfloor d^{\lceil n/2 \rceil}/2 \rfloor$. Since the size of each sub-block is nearly-balanced compared to that of Khan *et al.*'s method, the considerable number of repetitions of the CSD can be reduced. The partition size proposed here is optimal when using the CSD-based approach to synthesize quantum circuit. The proposed algorithm for partitioning is also efficient for other variants of the CSD, such as a generalization of the decomposition for the two-level quantum system as shown in Chapter 3 to the d-level quantum system.

Furthermore, the CSD-based method can produce a polynomial-size quantum circuit for the general radix-d QFT. This is because the nested structure of the QFT is naturally represented by means of the CSD. The result was already known for the two-level quantum system, but the author shows for the first time that the CSD proposed in this chapter, which is based on a different partition size from that used in the two-level quantum system, derives a similar result for the arbitrary d-level quantum systems. The same argument does not work for Brennen *et al.*'s methods because QR decomposition is not suitable for describing a nested structure.

This chapter is organized as follows: Section 2 introduces some preliminary knowledge related to this chapter. Section 3 describes the previous synthesis method for the d-level quantum system proposed by Khan *et al.*. Section 4 describes the proposed synthesis method for the d-level quantum system by means of the CSD with balanced partitioning. Section 5 presents a comparison between the proposed method and the other synthesis methods for qudits. Section 6 presents an application to the design of quantum circuits for the radix-d QFT by using the proposed method. The author concludes this chapter in Section 7 with a brief summary.

2. Preliminaries

2.1 The Uniformly Controlled Gates for the d-Level Quantum System

The idea of uniformly controlled gate is also important for generalizing the synthesis method into the arbitrary d-level quantum systems, where $d > 2$. Throughout this chapter, a generalized notation for the uniformly controlled gate for the arbitrary d-level quantum systems is used.

Definition 1 *A k-fold uniformly controlled gate, denoted by $\vee_n^k W$, is defined as follows:*

$$\vee_n^k W \;=\; \sum_{j=0}^{d^k-1} |j\rangle\langle j| \otimes W_j, \;=\; diag(W_0, W_1, \cdots, W_{d^k-1}),$$

where $W = \{W_0, W_1, W_2, \cdots, W_{d^k-1}\}$, which is a set of $d^{n-k} \times d^{n-k}$ unitary matrices.

The uniformly controlled gate is a sequence of controlled gates that consists of all possible control node combinations, *e.g.*, $\vee_n^k W$ gate defined above involves d^k controlled unitary operations, where each controlled gate has k controlled qudits and $n - k$ target qudits, and the unitary operations $W_0, W_1, W_2, \cdots, W_{d^k-1} \in U(d^{n-k})$ are applied according to the controlled states (see Figure 4.1).

Suppose the elementary gates in the d-level quantum system are a set of one-qudit operations and two-qudit operations called the CINC gate and the CINC^{-1} gate, where the CINC gate is a generalization of the CNOT gate and the CINC^{-1} gate is the inverse of the CINC gate. To translate a uniformly controlled one-qudit gate into a sequence of elementary gates, the Householder transformation [10] is applied. Details will be shown in Section 2.2. With this translation method, a uniformly controlled one-qudit operation can be composed of $O(d^n)$ elementary gates.

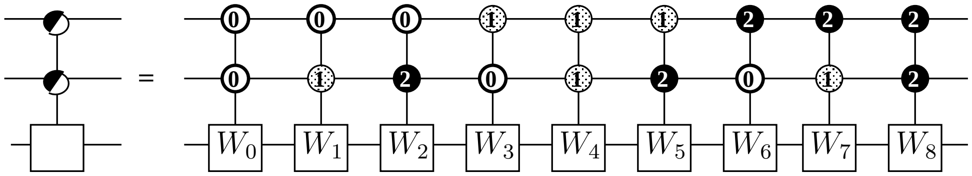


Figure 4.1. Notation of the 2-fold uniformly controlled one-qutrit gate ($d = 3$), $\vee_3^2 W$, where $W = \{W_0, W_1, W_2, \cdots, W_8\}$.

2.2 Structure of Quantum Circuit for the Uniformly Controlled Gates

An input matrix is translated into a sequence of uniformly controlled one-qudit gates by using the CSD. Here, the uniformly controlled one-qudit gate is denoted by $\vee_n^{n-1}(V)$. Section 2.1 showed that a uniformly controlled one-qudit gate can be composed of $O(d^n)$ elementary gates, and that this can be proved by using the result in [10]. This section describes the details of the structure of the quantum circuit for a uniformly controlled one-qudit gate and show that the size of the quantum circuit is $O(d^n)$.

In the two-level quantum system, a uniformly controlled one-qubit gate can be composed of $O(4^n)$ elementary gates. As shown in Section 3, the structure of the quantum circuit, which is composed of a sequence of elementary gates, for a uniformly controlled one-qubit gate is described using Gray code ordering [37]. Here, Gray code is used to define the positions of the control qubits.

A generalization of Gray code ordering for the d-level quantum system, called $\clubsuit$-sequence, was provided by Brennen *et al.* [10]. This is also used to define the positions of the control and target qudits. They use this ordering in the state synthesis algorithm. However, the method is also used to describe the structure of quantum circuits for a uniformly controlled one-qudit gate.

For example, consider when $n = 2$ and $d = 3$. The $\clubsuit$-sequence for $n = 2$ is $\{0\clubsuit, 1\clubsuit, 2\clubsuit, \clubsuit\clubsuit\}$. According to this ordering, a quantum circuit as shown in Figure 4.2 is obtained. Here, let U_4 be a diagonal matrix of the form $diag(c_1, c_2, c_3)$.

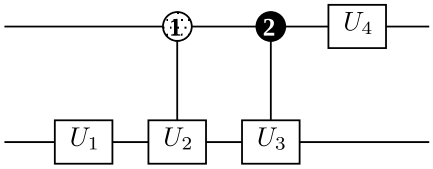


Figure 4.2. Structure of quantum circuit for a one-fold uniformly controlled one-qutrit gate, $\vee_2^n V$, where $V = V_0, V_1, V_2$.

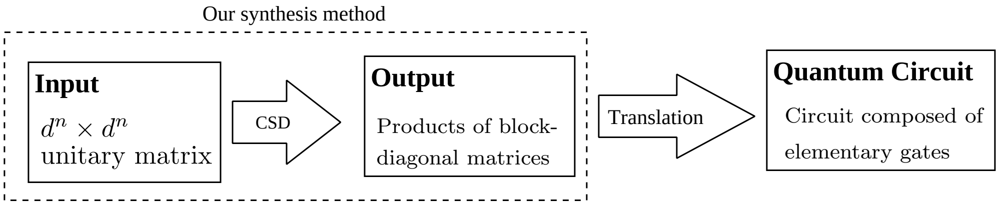


Figure 4.3. Overview of the proposed algorithm

Then, the quantum circuit corresponds to the following unitary operation:

$$\begin{pmatrix} c_1U_1 & 0 & 0 \\ 0 & c_2U_2U_1 & 0 \\ 0 & 0 & c_3U_3U_1 \end{pmatrix}.$$

This is a uniformly controlled one-qutrit gate $\vee_2^1 V = diag(V_0, V_1, V_2)$, where $V_0 = c_1U_1 \in U(d)$, $V_1 = c_2U_2U_1 \in U(d)$, and $V_2 = c_3U_3U_1 \in U(d)$.

Similarly, a sequence of controlled unitary operations defined by $\clubsuit$-sequence ordering can realize a uniformly controlled one-qudit gate. The number of elements in the $\clubsuit$-sequence is $(d^n - 1)/(d - 1)$, and the sequence is composed of operations that act on one- or two-qudit. Thus, the number of elementary gates used to implement a uniformly controlled one-qudit gate is $O(d^n)$. The details of the estimation of the number of elementary gates appearing in the $\clubsuit$-sequence are described in [10].

2.3 The CSD-Based Approach for the d-Level Quantum System

Figure 4.3 overviews the CSD-based approach for the arbitrary d-level quantum systems. The input is a $d^n \times d^n$ unitary matrix, which corresponds to an operation on n qudits. The output is a product of the $d^n \times d^n$ block-diagonal matrices, where the diagonals are composed of $d \times d$ sub-blocks. Here, each block-diagonal matrix corresponds to a uniformly controlled one-qudit gate, $\vee_n^{n-1}U$, where U is a set of $d \times d$ unitary matrices. Thus, an input $d^n \times d^n$ unitary matrix is translated into a quantum circuit composed of uniformly controlled one-qudit gates by using the CSD.

After that, the known translation between a uniformly controlled gate and a quantum circuit composed of elementary gates, as mentioned in Section 2.1, is applied.

2.4 Generalization of the CSD

As shown in Chapter 3 and Section 5, the input matrix is partitioned into four equally sized matrices in the CSD for the two-level quantum system. A problem arises when this approach is applied to the arbitrary d-level quantum systems because an input $d^n \times d^n$ unitary matrix cannot partitioned into equally sized sub-blocks.

Thus, in this chapter a generalized versions of the CSD are treated. Let the input matrix G be an $m \times m$ unitary matrix composed of four sub-blocks as follows:

$$G = \begin{array}{c|cc} & \ell & m-\ell \\ \hline \ell & G_{11} & G_{12} \\ m-\ell & G_{21} & G_{22} \end{array},$$

where G is partitioned into four blocks, each size of which is $\ell \times \ell$, $\ell \times (m-\ell)$, $(m-\ell) \times \ell$, $(m-\ell) \times (m-\ell)$. Here, ℓ is the partition size.

Let $G = UDV$ be the CSD of G, then

$$
\begin{aligned}
U &= \begin{array}{c c} & \begin{array}{cc} \ell & m-\ell \end{array} \\ \begin{array}{c} \ell \\ m-\ell \end{array} & \begin{pmatrix} U_1 & 0 \\ 0 & U_2 \end{pmatrix} \end{array}, \\
D &= \begin{array}{c c} & \begin{array}{ccc} \ell & \ell & m-2\ell \end{array} \\ \begin{array}{c} \ell \\ \ell \\ m-2\ell \end{array} & \begin{pmatrix} C & -S & 0 \\ S & C & 0 \\ 0 & 0 & I \end{pmatrix} \end{array}, \\
V &= \begin{array}{c c} & \begin{array}{cc} \ell & m-\ell \end{array} \\ \begin{array}{c} \ell \\ m-\ell \end{array} & \begin{pmatrix} V_1 & 0 \\ 0 & V_2 \end{pmatrix} \end{array},
\end{aligned}
$$

where C and S are diagonal matrices of the forms $C = diag(\cos\varphi_1, \cos\varphi_2, \cdots, \cos\varphi_\ell)$ and $S = diag(\sin\varphi_1, \sin\varphi_2, \cdots, \sin\varphi_\ell)$, and I is the $(m-2\ell) \times (m-2\ell)$ identity matrix. Note that if the input matrix G is partitioned into equally sized blocks, then the above D does not contain I, *i.e.*, D is a block-diagonal matrix composed of C and S blocks. Note that D can be written in the above form (4.1), when $2\ell \leq m$.

3. Previous Work

3.1 The Previous CSD-Based Method Proposed by Khan et al.

The CSD-based approach to synthesize quantum circuits for the d-level quantum system was first introduced by Khan *et al* in [30, 31]. Here, they choose the partition size $\ell = d^{n-1}$ at each recursion level.

Algorithm 2 (Matrix partition algorithm in [31])

1. *Compute* $n_1 = \lfloor \log_d m \rfloor$*, where* m *is the number of rows of the target matrix.*
2. *Compute* $\ell = d^{n_1 - 1}$.

Here, the target matrix means the matrix that the CSD will be applied to. Note that the target matrix is a square matrix, *i.e.*, the number of rows and the number of columns are the same size.

Let $U(m)$ be an $m \times m$ unitary matrix and $U(\ell) \oplus U(m-\ell)$ be a block-diagonal matrix of the form (4.1). Consider when $d = 3$ and $n = 3$, the algorithm proceeds as follows:

Step 1. Compute the CSD of $U(27)$ with the partition size 9, *i.e.*,

$$U(27) = [U(9) \oplus U(18)] \cdot [U(3)^{\otimes 9}] \cdot [U(9) \oplus U(18)].$$

Step 2.

1. Compute the CSD of $U(18)$ appearing in Step 1 with the partition size 9, *i.e.*,

$$U(18) = [U(9) \oplus U(9)] \cdot [U(3)^{\otimes 6}] \cdot [U(9) \oplus U(9)].$$

2. Compute the CSD of $U(9)$ appearing in Step 1 with the partition size 3, *i.e.*,

$$U(9) = [U(3) \oplus U(6)] \cdot [U(3) \oplus U(3) \oplus U(3)] \cdot [U(3) \oplus U(6)].$$

Step 3.

1. Compute the CSD of $U(9)$ appearing in Step 2 with the partition size 3, *i.e.*,

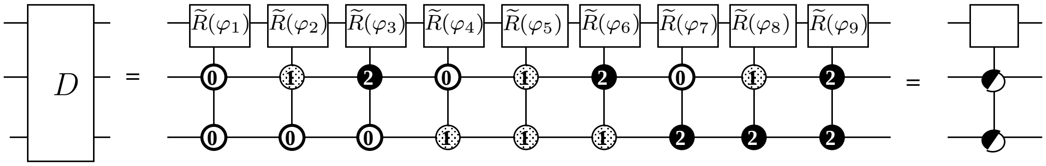
$$U(9) = [U(3) \oplus U(6)] \cdot [U(3) \oplus U(3) \oplus U(3)] \cdot [U(3) \oplus U(6)].$$

2. Compute the CSD of $U(6)$ appearing in Step 2 with the partition size 3, *i.e.*,

$$U(6) = [U(3) \oplus U(3)] \cdot [U(3) \oplus U(3)] \cdot [U(3) \oplus U(3)].$$

Step 4. Compute the CSD of $U(6)$ appearing in Step 3 with the partition size 3, *i.e.*,

$$U(6) = [U(3) \oplus U(3)] \cdot [U(3) \oplus U(3)] \cdot [U(3) \oplus U(3)].$$



D
=
$\widetilde{R}(\varphi_1)$
$\widetilde{R}(\varphi_2)$
$\widetilde{R}(\varphi_3)$
$\widetilde{R}(\varphi_4)$
$\widetilde{R}(\varphi_5)$
$\widetilde{R}(\varphi_6)$
$\widetilde{R}(\varphi_7)$
$\widetilde{R}(\varphi_8)$
$\widetilde{R}(\varphi_9)$
0
1
2
0
1
2
0
1
2
0
0
0
1
1
1
2
2
2
=

Figure 4.4. The quantum circuit corresponds to D in the CSD of G, *i.e.*, $G = UDV$, when $d = 3$, $n = 3$.

In the following steps, the decompositions of sub-matrices appearing at the previous step are described. Here, the middle part D is always of the form $[U(3)^{\otimes 9}]$, where

$$[U(3)^{\otimes 9}] \;=\; [U(3), U(3), U(3), U(3), U(3), U(3), U(3), U(3), U(3)],$$

and it is a uniformly controlled gate, as shown in Figure 4.4, where $\widetilde{R}(\varphi_j)$ is a 3×3 unitary matrix as follows:

$$\widetilde{R}(\varphi_j) \;=\; \begin{pmatrix} \cos\varphi_j & -\sin\varphi_j & 0 \\ \sin\varphi_j & \cos\varphi_j & 0 \\ 0 & 0 & 1 \end{pmatrix}. \tag{4.1}$$

Here, $j = 1, 2, 3, \cdots, 9$ and $0 \leq \varphi_j < 2\pi$.

In general, let n be the number of qudits, then the input matrix is a $d^n \times d^n$ unitary matrix. First, the CSD is applied to the input matrix with the partition size d^{n-1}. Then, the smallest sub-block is of size $d^{n-1} \times d^{n-1}$ and the largest sub-block is of size $(d-1)d^{n-1} \times (d-1)d^{n-1}$. Next, the CSD is applied to these $(d-1)d^{n-1} \times (d-1)d^{n-1}$ sub-blocks with the partition size d^{n-1}. After $d-1$ applications of the CSD, then all sub-blocks are of size $d^{n-1} \times d^{n-1}$. In the next step, the CSD is applied to a $d^{n-1} \times d^{n-1}$ sub-block with the partition size d^{n-2}. Similarly, the CSD is applied to the largest sub-block with partition size d^{n-2} until all sub-blocks appearing in the matrix products are of size $d^{n-2} \times d^{n-2}$. In each $d-1$ repetition of the CSD, the size of each sub-block appearing in the matrix products decreases $1/d$. Thus, to decompose a $d^n \times d^n$ unitary matrix into a sequence of block-diagonal matrices, where the size of each sub-block appearing in the matrix products is $d \times d$, $(d-1)(n-1)$ times of repetitions of the CSD is required. Since the uniformly controlled gate increases exponentially in the number of repetitions, thus the number of uniformly controlled gates appearing in the produced quantum circuit is $2^{(d-1)(n-1)+1} - 1$. For more details, please see in Section 3.2.

3.2 Gate Counts for Khan et al.'s Method

This section presents an estimation of the number of elementary gates appearing in the quantum circuit produced by the CSD proposed by Khan *et al.* [30, 31]. Since the translation between a uniformly controlled one-qudit gate into a sequence of elementary gates is the same as that used in the author's method, the number of uniformly controlled one-qudit gates $f(n)$ appearing in the quantum circuit produced by the author's method and that of the circuit produced by Khan *et al.*'s will be treated in this section.

In Khan *et al.*'s method, the partition size of (4.1) is chosen as $\ell = d^{n-1}$. Then, the diagonal part D is a uniformly controlled one-qudit operation, where each one-qudit operation is of the form

$$\begin{pmatrix} \cos\phi_j & -\sin\phi_j & 0 \\ \sin\phi_j & \cos\phi_j & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

Then, U and V in the CSD (4.1) are recursively decomposed until each sub-block size is $d^{n-1} \times d^{n-1}$. In each step, the partition size is chosen as $\ell = d^{n-1}$. Thus, the number of repetitions is $d - 1$. Then, a quantum circuit shown in the first line in Figure 4.5 is obtained.

Let $f(n)$ be the number of uniformly controlled one-qudit operations used to implement an arbitrary operation on n qudits. Let a_n be the total number of $G_j^{(d)}$ gates appearing in the circuit at the first line in Figure 4.5. Here, the number of $B_j^{(h)}$ gates is $a_n - 1$. Then,

$$f(n) \;=\; a_n f(n-1) + a_n - 1, \tag{4.2}$$

where $a_n = 2^{d-1}$ and $f(1) = 1$.

Let $g(n) = f(n) + 1$. Then,

$$g(n) \;=\; 2^{d-1} g(n-1) \;=\; 2^{(d-1)n} g(0) \;=\; 2^{(d-1)n}.$$

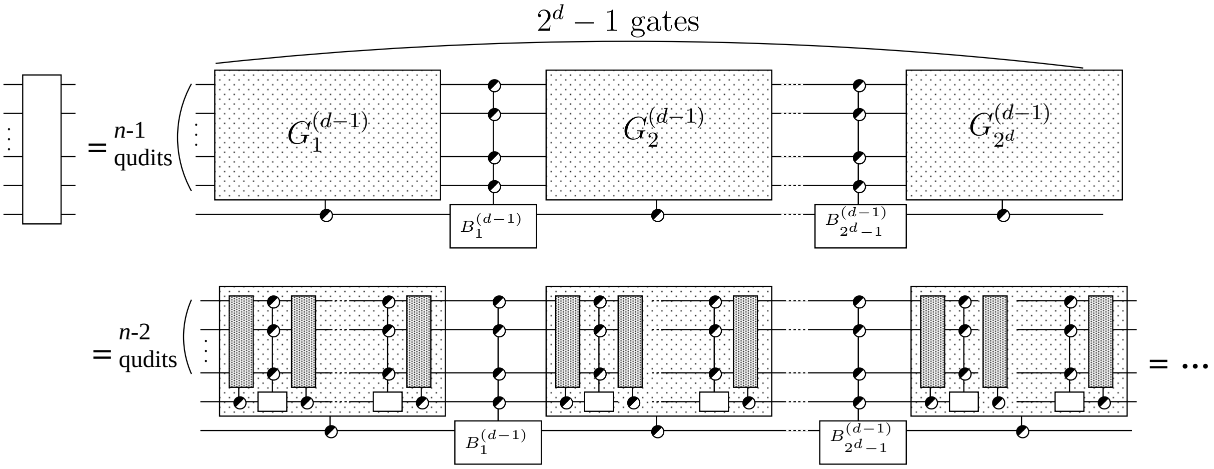


Figure 4.5. A quantum circuit obtained after $d-1$ repetitions of the CSD proposed by Khan *et al.* [30, 31].

Thus, $g(n) = 2^{(d-1)n}$ and $f(n) = 2^{(d-1)n} - 1$ can be bounded by $O(2^{(d-1)n})$. Since a uniformly controlled one-qudit operation can be implemented using $O(d^n)$ elementary gates, the number of elementary gates appearing in the quantum circuit produced by Khan *et al.*'s method is $O(2^{(d-1)n}d^n)$. This can be rewrited as $O(\alpha^n d^{2n})$, where $\alpha = 2^{d-1}/d$.

4. Description of the Proposed Method

4.1 Balanced Partitioning

The main difference between the previous CSD-based approach proposed by Khan *et al.* [30,31] and the proposed CSD-based approach is the choice of the partition size.

As discussed in Section 2, the number of uniformly controlled one-qudit gates increases exponentially in the number of repetitions. Thus, the choice of partition size has therefore significant effect to reduce the size of the quantum circuit produced by the CSD-based synthesis methods.

To produce a sequence of uniformly controlled one-qudit operations, the CSD is recursively applied to each sub-block until the size of each sub-block is $d \times d$.

Thus, the number of repetitions will increase as the size of sub-matrix gets large. In other words, the number of repetitions depends on the size of the largest sub-block in the CSD. Therefore, the best way to reduce the depth is to balance the size of each sub-block.

As mentioned in Section 2, for the two-level quantum system, the partition size is chosen as a half of the size of the target matrix (*i.e.*, the matrix that will be decomposed). In other words, the partition size is balanced at each recursion step. However, the partition size cannot be balanced perfectly when d is not a power of two. Consider when a $d^n \times d^n$ unitary matrix is decomposed, an operation on n qudits, and the partition size ℓ is chosen as $\lfloor d^n/2 \rfloor$ by following the strategy for the two-level case. Then, the size of each sub-block at this decomposition is nearly balanced. However, it cannot reach a $d \times d$ matrix at the bottom level of the recursion by applying this "nearly-balanced" CSD recursively. Thus, it may be inefficient to synthesize quantum circuits for the d-level quantum system by applying this "nearly-balanced" CSD like the two-level case.

Considering the above discussion, the author proposes an efficient partitioning method called balanced partitioning. The partition size ℓ in (4.1) is chosen as follows:

Algorithm 3 (Balanced partitioning)

1. *Compute $n' = \lfloor \log_d m \rfloor$, where m is the number of rows of the target matrix.*

2. *If $d^{n'} = m$ then compute $\ell = d^{\lfloor n'/2 \rfloor} \lfloor d^{\lceil n'/2 \rceil}/2 \rfloor$.*
3. *Else compute $c = m/d^{\lfloor n'/2 \rfloor}$ and $\ell = d^{\lfloor n'/2 \rfloor} \lfloor c/2 \rfloor$.*

4.2 Example

Consider when $d = 3$ and $n = 3$ and the partitioning method presented in Algorithm 2 is applied. The input matrix is 27×27 unitary matrix and is partitioned into four sub-blocks with the partition size $\ell = 12$. Then, the size of sub-blocks $U_1^{(1)}, V_1^{(1)}$ are both 12×12 and the size of $U_2^{(1)}$ and $V_2^{(1)}$ are both 15×15. Here, in the previous method described in Section. 3.1, the size of $U_1^{(1)}$ and $V_1^{(1)}$ are both

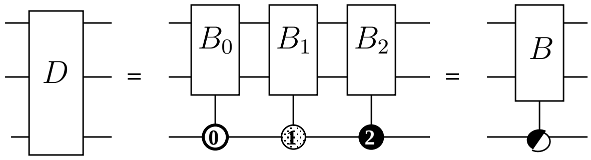


Figure 4.6. Gate structure of D obtained by the CSD with balanced partitioning method, when $d = 3$ and $n = 3$.

9×9, and the size of $U_2^{(1)}$ and $V_2^{(1)}$ are both 18×18, thus the size of sub-blocks in the proposed method are balanced compared to the previous method.

Here, the middle part D is

$$\begin{array}{c} \\ 12 \\ 12 \\ 3 \end{array} \begin{array}{c} \begin{array}{ccc} 12 & 12 & 3 \end{array} \\ \begin{pmatrix} C & -S & 0 \\ S & C & 0 \\ 0 & 0 & I \end{pmatrix} \end{array}.$$

To the simplicity of the notation, consider the matrix $\widetilde{D}$ instead of D, where $\widetilde{D}$ is the same opeartion as D if the first and the third qudits are swapped. In other words, let χ_n^3 be the SWAP operation that acts on the first and the third qudits. Then, $\widetilde{D} = \chi_1^3 D \chi_1^3$ is as follows:

$$\widetilde{D} \;=\; diag(B_0, B_1, B_2),$$

where B_0, B_1, and B_2 are 9×9 unitary matrices such that

$$B_j \;=\; diag(R_y(\varphi_{4j+1}), R_y(\varphi_{4j+2}), R_y(\varphi_{4j+3}), R_y(\varphi_{4j+4}), 1), \qquad j = 0, 1, 2,$$

where $R_y(\varphi_j)$ is a 2×2 unitary matrix defined in (2.10). Thus, D is a 1-fold controlled unitary two-qudit operation, as shown in Figure 4.6.

Let $U(m)$ be an $m \times m$ unitary matrix and $U(\ell) \oplus U(m-\ell)$ be a block-diagonal matrix of the form (4.1). Consider the case when $d = 3$ and $n = 3$, the proposed algorithm proceeds as follows:

Step 1. Compute the CSD of $U(27)$ with the partition size 12, *i.e.*,

$$U(27) \;=\; [U(12) \oplus U(15)] \cdot [U(9) \oplus U(9) \oplus U(9)] \cdot [U(12) \oplus U(15)].$$

Step 2.

1. Compute the CSD of $U(15)$ appearing in Step 1 with the partition size 6, *i.e.*,

$$U(15) = [U(6) \oplus U(9)] \cdot [U(5) \oplus U(5) \oplus U(5)] \cdot [U(6) \oplus U(9)].$$

2. Compute the CSD of $U(12)$ appearing in Step 1 with the partition size 6, *i.e.*,

$$U(12) = [U(6) \oplus U(6)] \cdot [U(4) \oplus U(4) \oplus U(4)] \cdot [U(6) \oplus U(6)].$$

Step 3.

1. Compute the CSD of $U(9)$ appearing in Step 2 with the partition size 3, *i.e.*,

$$U(9) = [U(3) \oplus U(6)] \cdot [U(3) \oplus U(3) \oplus U(3)] \cdot [U(3) \oplus U(6)].$$

2. Compute the CSD of $U(6)$ appearing in Step 2 with the partition size 3, *i.e.*,

$$U(6) = [U(3) \oplus U(3)] \cdot [U(2) \oplus U(2) \oplus U(2)] \cdot [U(3) \oplus U(3)].$$

Here, in each step of the above, the decompositions of sub-matrices appearing at the previous step are described. The size of the sub-block appearing in the middle part do not seem those of operations on qudits. For example, in Step 2, 5×5 sub-blocks appear in the decomposition of $U(15)$ and 4×4 sub-blocks appear in the decomposition of $U(12)$. However, by combining these sub-blocks, *i.e.*, by combining the decomposition described in Step 2.1 and Step 2.2 as the decomposition of $[U(12) \oplus U(15)]$, then the middle part is $[U(9) \oplus U(9) \oplus U(9)]$, where each $U(9) = [U(4) \oplus U(5)]$, thus it is a one-fold uniformly controlled two qudit operation.

The partition size proposed here is optimal when using the CSD-based approach to synthesize quantum circuit. Consider the above example, if $U(27)$ is divided into $U(13) \oplus U(14)$, then the size of each sub-block is approximately balanced. However, by repeating this partitioning, the produced block-diagonal matrix does not correspond to a uniformly controlled one-qudit gate. Thus, by dividing $U(27)$ into $U(12) \oplus U(15)$ is the best choice under the condition that the produced matrices correspond to uniformly controlled one-qudit gates after repetitions.

The number of steps, where it is 3 in the above example, denotes the number of repetitions of the CSD. Compared to the previous approach described in Section 3.1, the balanced partitioning method can reduce the number of repetitions.

In general, for arbitrary d and the number of qudits n, the size of the circuit is $O(n^{2+\log_2 d} d^{2n})$ when d is odd, to $O(d^{2n})$ when d is a power of two, and to $O(nd^{2n})$ otherwise. Here, Khan *et al.*'s method needs $O(\alpha^n d^{2n})$, where $\alpha = 2^{d-1}/d$. For more details, please see Section 4.3.

4.3 Estimation of Gate Counts by the Proposed Method

Suppose the input matrix is a $d^n \times d^n$ unitary matrix, *i.e.*, an operation on n qudits. The dimension of I contained in D (4.1) is

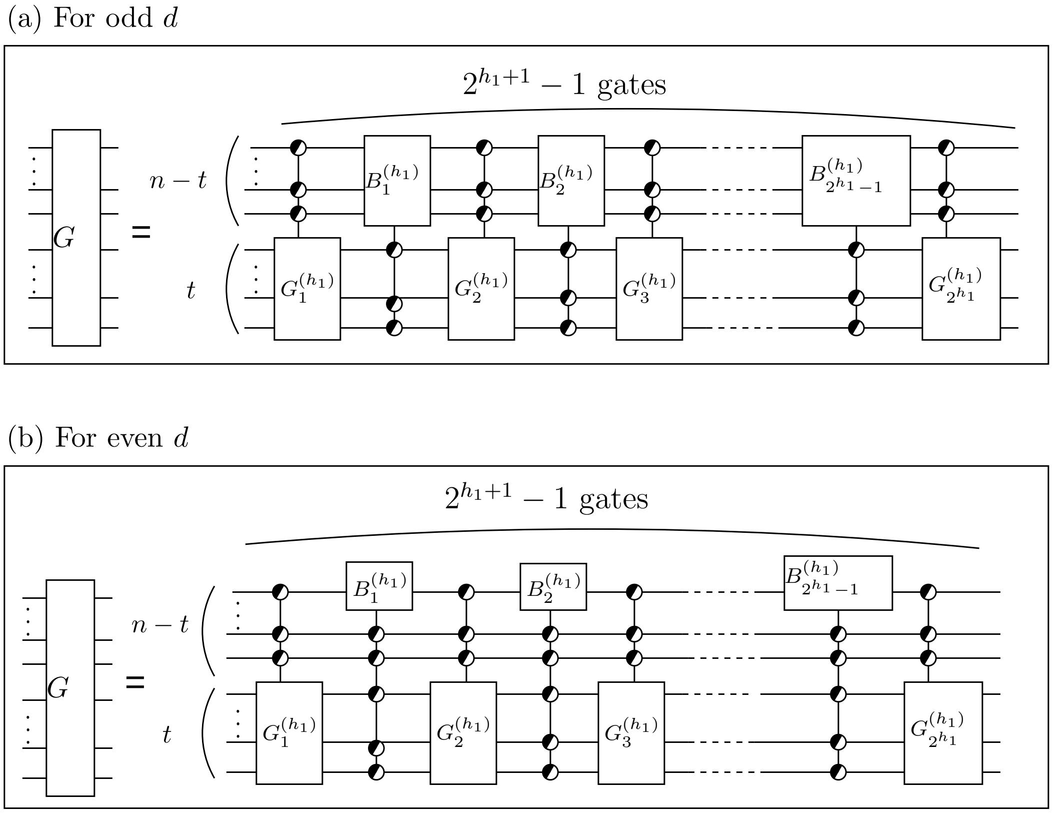
$$d^n - 2d^{\lfloor n/2 \rfloor} \lfloor d^{\lceil n/2 \rceil}/2 \rfloor \quad = \quad \begin{cases} d^{\lfloor n/2 \rfloor}, & \text{when } d \text{ is odd;} \\ 0, & \text{when } d \text{ is even,} \end{cases}$$

Then, the number of sub-blocks of the form $R_y(\varphi_j)$ that are contained in D is $d^{\lfloor n/2 \rfloor} \lfloor d^{\lceil n/2 \rceil}/2 \rfloor$. For example when $d = 3$ and $n = 3$, there is 12 sub-block of the form $R_y(\varphi_j)$. To the simplicity of the notation, $\widetilde{D} = \chi_1^n D \chi_1^n$ is used instead of D, then

$$\begin{aligned} \widetilde{D} \;&=\; diag(B_0, B_1, \cdots, B_{d^{\lfloor n'/2 \rfloor}-1}), && (4.3) \\ B_j \;&=\; \begin{cases} diag(R_y(\phi_{js+1}), R_y(\phi_{js+2}), \cdots, R_y(\phi_{js+s}), 1), & \text{for odd } d; \\ diag(R_y(\phi_{js+1}), R_y(\phi_{js+2}), \cdots, R_y(\phi_{js+s})), & \text{for even } d, \end{cases} && (4.4) \end{aligned}$$

where $s = \lfloor d^{n-t}/2 \rfloor$ and $j = 0, 1, 2, \cdots, d^{\lfloor n'/2 \rfloor} - 1$. Then, D corresponds to a $\lfloor n/2 \rfloor$-fold uniformly controlled $\lceil n/2 \rceil$ qudit operation when d is odd, and to a $(n-1)$-fold uniformly controlled one-qudit operation when d is even.

Let the input matrix be $d^n \times d^n$, where n is the number of qudits. In the first step, Step 2 in Algorithm 2 is proceeded. The, next $\lceil \log_2(d^{\lceil n/2 \rceil}) \rceil - 1$ repetitions, Step 3 in Algorithm 2 is proceeded. Then, all matrices corresponding to the U



(a) For odd d

(b) For even d

Figure 4.7. Structure of the quantum circuits (a) when d is odd and (b) when d is even, after $h_1 = \lceil \log_2 d^{n-t} \rceil$ times of repetitions.

and V parts of the CSD are block-diagonal matrices, where the diagonals are composed of $d^{\lfloor n/2 \rfloor} \times d^{\lfloor n/2 \rfloor}$ unitary matrices. After these repetitions, each sub-block corresponds to an operation on $\lfloor n/2 \rfloor$ qudits or an operation on $\lceil n/2 \rceil$ qudits, as shown in Figure 4.7.

Similarly, the above CSD is applied to operations on $\lfloor n/2 \rfloor$ qudits, *i.e.*, $G_j^{(1)}$ in Figure 4.7. As described above, when d is odd, then the middle part D in the CSD is a $\lfloor n/2 \rfloor$-fold uniformly controlled $\lceil n/2 \rceil$-qudit operation. Thus, the CSD is also recursively applied to operations on $\lceil n/2 \rceil$ qudits (denoted as $B_j^{(p)}$ for $j = 1, 2, 3, \cdots, 2^p - 1$ in Figure 4.7). When d is even, the recursive decompositions for $B_j^{(p)}$'s are not necessary because these are already one-qudit operations ($d \times d$ matrices). This is the reason the size of the quantum circuit produced by the proposed method depends on the properties of d.

Let $n_1 = \lfloor n/2 \rfloor$. In the first step, Step 2 in Algorithm 2 is proceeded, and the partition size $\ell = \lfloor n_1/2 \rfloor$. Then, the next $\lceil \log_2 d^{\lfloor n/4 \rfloor} \rceil - 1$ repetitions, Step 3 in Algorithm 2 is proceeded. After these repetitions, each sub-block corresponds to an operation on $\lfloor n/4 \rfloor$ qudits or an operation on $\lceil n/4 \rceil$ qudits.

Similarly, by repeating these procedures, the size of target operations decrease half of that in the previous steps and the procedure will repeat until the size of target operations are one. Let h_j be the number of repetitions between the j-th applications of Step 1 in Algorithm 2 and the $(j+1)$-th applications of Step 1 in Algorithm 2, *i.e.*, $h_1 = \lceil \log_2 (d^{\lceil n/2 \rceil}) \rceil$ and $h_2 = \lceil \log_2 d^{\lfloor n/4 \rfloor} \rceil$ as mentioned above.

Therefore, when the input $d^n \times d^n$ unitary matrix is decomposed, the number of repetition h is as follows:

$$\begin{aligned} h &= h_1 + h_2 + \cdots + h_{\lfloor \log_2 d^n \rfloor} \\ &= \lceil \log_2 d^{\lfloor n/2 \rfloor} \rceil + \lceil \log_2 d^{\lfloor n/4 \rfloor} \rceil + \lceil \log_2 d^{\lfloor n/8 \rfloor} \rceil + \cdots + \lceil \log_2 d \rceil, \\ &< \lceil \log_2 d^{\lfloor n/2 \rfloor + \lfloor n/4 \rfloor + \lfloor n/8 \rfloor + \cdots + 1} \rceil, \\ &\leq \lceil \log_2 d^{n + \log_2 n} \rceil. \end{aligned}$$

As discussed in Section 3.1, if the input $d^n \times d^n$ unitary matrix is decomposed, $(d-1)(n-1)$ times of repetitions of the CSD is required. Compared to the above result, the proposed method can reduce the number of repetitions required to decompose a $d^n \times d^n$ unitary matrix into a sequence of uniformly controlled one-qudit operations.

An input $d^n \times d^n$ unitary matrix can be translated into a sequence of uniformly controlled one-qudit operations by using the author's synthesis method. Then, the known translation between a uniformly controlled one-qudit operation and a sequence of elementary gates (cf. Section 2.2) can be applied. Here, the number of uniformly controlled one-qudit gates appearing in the quantum circuit produced by the proposed method is estimated as follows:

Theorem 2 *By using the CSD based on the balanced partitioning (Algorithm 1), an arbitrary $d^n \times d^n$ unitary matrix can be translated into a sequence of uniformly controlled one-qudit gates. The number of those gates can be bounded by*

(a) $O(n^{2+\log_2 d} d^n)$ if d is odd,

(b) $O(d^n)$ if d is a power of two, and

(c) $O(nd^n)$ otherwise.

Proof. Note that the partition size according to Algorithm 1 is chosen in each step. As shown in Figure 4.7, let $G_j^{(i)}$ for $j = 1, 2, 3, \cdots, 2^i$ be a matrix corresponding to the U (or V) part of the CSD that appears after the i repetitions. Let $p = \lceil \log_2 d^{n-t} \rceil$, then the size of each diagonal sub-block of $G_j^{(p)}$ is $d^t \times d^t$, where $t = \lfloor n/2 \rfloor$. Here, the produced matrix products correspond to a quantum circuit as in Figure 4.7. The total number of $\vee_n^t G^{(p)}$ gates, where $G^{(p)} = G_1^{(p)}, G^{(p)}_{,} \cdots, G_{2^p}^{(p)}$, is $2^p = 2^{\lceil \log_2 d^{n-t} \rceil}$. Next, the CSD is recursively applied to each $d^t \times d^t$ sub-block of $G_j^{(p)}$.

Let $f(n)$ be the number of uniformly controlled one-qudit operations used to implement an arbitrary operation on n qudits. Let a_n be the number of $\vee_n^t G^{(p)}$ gates appearing in Figure 4.7. The number of $\vee_n^{n-t} B^{(p)}$ gates is $a_n - 1$. Then,

$$f(n) \quad = \quad \begin{cases} a_n \cdot f\left(\lfloor n/2 \rfloor\right) + (a_n - 1) \cdot f\left(\lceil n/2 \rceil\right), & \text{for odd } d; \\ a_n \cdot f\left(\lfloor n/2 \rfloor\right) + (a_n - 1), & \text{for even } d, \end{cases} \tag{4.5}$$

where

$$a_n \quad = \quad 2^{\lceil \log_2 d^{n-t} \rceil} \begin{cases} = d^{\lceil n/2 \rceil}, & \text{when } d \text{ is a power of two;} \\ < 2d^{\lceil n/2 \rceil}, & \text{otherwise.} \end{cases}$$

(a) When d is odd, $f(n)$ can be estimated as follows:

$$f(n) \ < \ 2a_n f\left(\lceil n/2 \rceil\right) \ < \ 4d^{\lceil n/2 \rceil} f\left(\lceil n/2 \rceil\right) \ < \ 4^{\log_2 n} d^{n+\log_2 n} \ = \ n^{2+\log_2 d} d^n.$$

Here, note that $a_n = (\log_d n)(\log_2 d)$. Theorem 2(a) is therefore true.

(b) When d is a power of two, $a_n = d^{\lceil n/2 \rceil}$. In (4.5), let $g(n)$ be $f(n)+1$, then

$$\begin{aligned} g(1) &= f(1)+1 \ = \ 1; \\ g(n) &= d^{\lceil n/2 \rceil} g\left(\lfloor n/2 \rfloor\right), \qquad \text{for } n > 1. \end{aligned}$$

Here, the following equation follows from mathematical induction.

$$g(n) \ = \ d^{n-1}, \qquad \text{for } n \geq 1. \tag{4.6}$$

The basis is trivial, since $g(1) = d^0 = 1$. For the induction step, assume that (4.6) holds for any integer $m < n$. Then,

$$g(n) \ = \ d^{\lceil n/2 \rceil} g\left(\lfloor n/2 \rfloor\right) d^{\lceil n/2 \rceil} d^{\lfloor n/2 \rfloor - 1} \ = \ d^{n-1}.$$

Thus, (4.6) holds for n as well. Since $f(n) = g(n) - 1 \ = \ d^{n-1} - 1$, $f(n)$ is bounded by $O(d^n)$. Theorem 2(b) is therefore true.

(c) When d is even and is not a power of two, let $g(n)$ be $f(n)+1$, then

$$\begin{aligned} g(1) &= f(1)+1 \ = \ 2; \\ g(n) &\leq 2d^{\lceil n/2 \rceil} g\left(\lfloor n/2 \rfloor\right), \qquad \text{for } n > 1. \end{aligned}$$

Again, mathematical induction is used to prove the following inequality.

$$g(n) \ \leq \ nd^n, \qquad \text{for } n \geq 1. \tag{4.7}$$

The basis is trivial, since $g(1) \leq 1 \cdot d^1 = d$. For the induction step, assume that (4.7) holds for any integer $m < n$. Then,

$$g(n) \ \leq \ d^{\lceil n/2 \rceil} g\left(\lfloor n/2 \rfloor\right) \ \leq \ d^{\lceil n/2 \rceil} \cdot \lfloor n/2 \rfloor \cdot d^{\lfloor n/2 \rfloor} \ = \ \lfloor n/2 \rfloor d^n \ < \ nd^n.$$

Thus, (4.7) holds for n as well. Since $f(n) = g(n) - 1 \ \leq \ nd^n - 1$, $f(n)$ is bounded by $O(nd^n)$. Theorem 2(c) is therefore true. $\square$.

Since a uniformly controlled one-qudit operation can be composed of $O(d^n)$ elementary gates, the size of the quantum circuit produced by the proposed method can be bounded by $O(n^{2+\log_2 d} d^{2n})$ when d is odd, $O(d^{2n})$ when d is a power of two, and $O(nd^{2n})$ otherwise.

Table 4.1. The total number of CINC gates and CINC^{-1} gates appearing in the produced quantum circuit.

d n	2	3	4	5	6	7	8
2	36 $\triangle$4	156 $\triangle$36	440 $\triangle$72	990 $\triangle$280	1932 $\triangle$420	3416 $\triangle$588	5616 $\triangle$784
3	346 $\triangle$112	3969 $\triangle$3660	21248 $\triangle$4464	78125 $\triangle$69720	226800 $\triangle$60960	559433 $\triangle$381780	1224704 $\triangle$142240
4	2208 $\triangle$308	45198 $\triangle$20088	396288 $\triangle$40824	2175000 $\triangle$670320	8841312 $\triangle$1563660	29143338 $\triangle$4928616	82280448 $\triangle$4750256
5	10432 $\triangle$1560	434484 1382952	6533120 $\triangle$685440	55187500 252347440	320760000 $\triangle$38074200	1.4×10^{9} 1.0×10^{10}	5.3×10^{9} $\triangle 3.1 \times 10^{8}$
6	44416 $\triangle$14364	3787884 8254764	99074048 $\triangle$22254984	1303750000 2476305000	1.1×10^{10} $\triangle 3.7 \times 10^{9}$	6.7×10^{10} $\triangle 1.4 \times 10^{11}$	3.2×10^{11} $\triangle 3.9 \times 10^{10}$
7	176896 $\triangle$60960	32693463 127837404	1529020416 $\triangle$357389712	3.2×10^{10} 1.4×10^{11}	3.9×10^{11} $\triangle 1.8 \times 10^{11}$	3.2×10^{12} 1.1×10^{13}	2.0×10^{13} $\triangle 2.5 \times 10^{12}$
8	677376 $\triangle$125476	28022687 465572880	2.4×10^{10} $\triangle 2.9 \times 10^{9}$	7.8×10^{11} 8.8×10^{11}	1.4×10^{13} $\triangle 4.2 \times 10^{12}$	1.6×10^{14} $\triangle 9.8 \times 10^{13}$	1.3×10^{15} $\triangle 8.0 \times 10^{13}$
9	2579456 $\triangle$512040	2.4×10^{9} 3.1×10^{10}	3.7×10^{11} $\triangle 4.6 \times 10^{10}$	1.9×10^{13} 6.6×10^{14}	4.9×10^{14} $\triangle 1.0 \times 10^{14}$	7.6×10^{15} 8.1×10^{17}	8.2×10^{16} $\triangle 5.2 \times 10^{15}$
10	9811968 $\triangle$2070572	2.1×10^{10} 1.8×10^{11}	5.9×10^{12} $\triangle 7.3 \times 10^{11}$	4.8×10^{14} 6.5×10^{15}	1.8×10^{16} $\triangle 2.4 \times 10^{15}$	3.7×10^{17} 1.1×10^{19}	5.3×10^{18} $\triangle 3.3 \times 10^{17}$
11	37580800 $\triangle$8331312	1.9×10^{11} 3.2×10^{12}	9.4×10^{13} $\triangle 1.2 \times 10^{13}$	1.2×10^{16} 1.9×10^{17}	6.3×10^{17} $\triangle 1.2 \times 10^{17}$	1.8×10^{19} 4.7×10^{20}	3.4×10^{20} $\triangle 2.1 \times 10^{19}$
12	145154048 $\triangle$66879540	1.7×10^{12} 1.3×10^{13}	1.5×10^{15} $\triangle 3.8 \times 10^{14}$	3.0×10^{17} 1.3×10^{18}	2.3×10^{19} $\triangle 1.1 \times 10^{19}$	8.9×10^{20} 4.4×10^{21}	2.2×10^{22} $\triangle 2.7 \times 10^{21}$

5. Comparison

The result presented here is asymptotically worse than the non-CSD approaches proposed by Brennen *et al.* [10, 15] except when d is a power of two. However, for a small number of qudits, the proposed method can produce more efficient quantum circuits than those produced by Brennen *et al.*'s methods. Table 4.1 shows total number of CINC gates and CINC^{-1} gates appearing in the produced quantum circuits. Here, in each cell, the upper line denotes the count obtained using the most efficient methods from Brennen *et al.*'s methods, *i.e.*, the triangle algorithm or the spectral decomposition. The bottom line denotes the count obtained using the author's method. Here, $\triangle$ means the count in the author's method is better than in the others.

As shown in Table 4.1, when d is odd, the proposed method is better than the othersonly when $n \leq 4$. However, when d is even, the proposed method is better than the other methods at least when $n \leq 12$. For example, the proposed method is advantageous when $n < 67$ for $d = 6$ and $n < 105$ for $d = 10$. Since matrix decomposition cannot treat large operations (Note that the matrix size grows exponentially with the number of inputs of qudits), it is useful for small number of qudits n. In such a case, the proposed method can produce the smallest circuits compared to the others.

In addition, the CSD-based methods have an advantage in producing polynomial-size quantum circuits for some efficient quantum computation. See, Section 6.

6. Application to the Synthesis of Quantum Circuits for the Radix-d QFT

This section presents that the CSD is useful for producing a polynomial-size quantum circuit for the QFT. The QFT that acts on n qudits, F_n, is a $d^n \times d^n$ unitary matrix such that

$$F_n|x\rangle \;=\; \frac{1}{\sqrt{d^n}} \sum_{y\in\{0,1,\cdots,d-1\}^n} \omega_n^{jk}|y\rangle, \qquad \omega_n \;=\; \exp\left(\frac{-2\pi i}{d^n}\right).$$

Here, the bit reversal versions of F_n is denoted as $\widetilde{F}_n$. Let χ_j^k be the SWAP operation that acts on the j-th and the k-th qudits and let $P_n = \chi_{n-1}^n \chi_{n-2}^n \cdots \chi_1^n$. Then, $\widetilde{F}_n = P_n^\dagger F_n P_n$.

Let D_Ω be a $d^n \times d^n$ diagonal matrix of the form

$$D_\Omega \;=\; diag(\Omega_n^{d-1}, \Omega_n^{d-2}, \cdots, \Omega_n, I_{d^{n-1}}), \tag{4.8}$$

where $\Omega_n = diag(\omega^{d^{n-1}-1}, \omega^{d^{n-1}-2}, \cdots, \omega, 1)$, and I_k denotes the $k \times k$ identity operation. Then, $\widetilde{F}_n$ can be written as follows:

$$\widetilde{F}_n \;=\; (I_{d^{n-1}} \otimes F_1) D_\Omega (\widetilde{F}_{n-1} \otimes I) P_n.$$

Claim 1 *Let $I_{m-\ell}$ is the $(m-\ell)\times(m-\ell)$ identity matrix, σ is the $\ell \times \ell$ diagonal matrix of the form $diag(1, -1, 1, -1, \cdots)$ with alternative positive and negative* *signs, and $\rho = \sigma \otimes I_{m-\ell}$. Then, the CSD of $\widetilde{F}_n$ with the balanced partitioning presented in this chapter is as follows:*

$$\widetilde{F}_n \;=\; UDV,$$

where,

$$\begin{aligned} U &= (I_{d^{n-1}} \otimes F_1) D_\Omega P_n \sqrt{\rho\widetilde{\rho}}^{\dagger}, \\ D &= \sqrt{\widetilde{\rho}\rho}, \\ V &= P_n^\dagger (\widetilde{F}_{n-1} \otimes I) P_n. \end{aligned}$$

Proof. This can be shown as an extension of the result for decomposing the radix-2 QFT in Chapter 3. Here, a Lie theoretic interpretation of the CSD is applied. As discussed in [12], the CSD is an example of the KAK decomposition in the Lie group theory. Suppose that the input matrix $G \in SU(m)$ is decomposed. Then, the CSD can be given as follows:

$$G \;=\; UDV, \qquad \text{where U, V} \in \mathrm{G_K} \text{ and D} \in \mathrm{G_A}.$$

Here, G_K and G_A are subgroups of $SU(m)$ such that $G_A \subset G_M$ and

$$\begin{aligned} G_K &= \{X \mid X \in SU(m) \text{ and } \Theta(X) = X\}, \\ G_M &= \{X \mid X \in SU(m) \text{ and } \Theta(X) = X^\dagger\}. \end{aligned}$$

These relations are derived from the relation between the KAK decomposition and the Cartan decomposition. Here, $\Theta(X)$ is called the global Cartan involution, which induces the Cartan involution θ of the Lie algebra $\mathfrak{su}(\mathfrak{m})$ with associated Cartan decomposition.

In the proposed CSD, let ℓ be the partition size, then $\Theta(x)$ can be given as follows:

$$\Theta(x) = \rho x \rho, \qquad \rho = \sigma \otimes I_{m-\ell}, \tag{4.9}$$

where $I_{m-\ell}$ is the $(m-\ell) \times (m-\ell)$ identity matrix and σ is the $\ell \times \ell$ diagonal matrix of the form $diag(1, -1, 1, -1, \cdots)$ with alternative positive and negative signs.

Then, for $\widetilde{F}_n$, the following relations in terms of Θ are hold.

$$\Theta(I_{d^{n-1}} \otimes F_1) \;=\; I_{d^{n-1}} \otimes F_1, \qquad \Theta(D_\Omega) \;=\; D_\Omega.$$

Let $\widetilde{\rho}$ be $P_n^\dagger \rho P_n$.

The CSD can be computed in terms of the global Cartan involution. Details of the algorithm are presented in Chapter 3 for the two-level quantum system. According to the algorithm, a decomposition of $\widetilde{F}_n$ can be computed as follows:

1. Compute $M^2 = \Theta(\widetilde{F}_n^\dagger)\widetilde{F}_n$.

$$M^2 \;=\; P_n^\dagger(\widetilde{F}_{n-1}^\dagger \otimes I)\widetilde{\rho}\rho(\widetilde{F}_{n-1} \otimes I)P_n.$$

2. Decompose $M^2 = K_1 A_1 K_1^\dagger$ such that $\Theta(K_1) = K_1$ and $\Theta(B_1) = B_1^\dagger$. Then,

$$\begin{aligned} K_1 &= P_n^\dagger(\widetilde{F}_{n-1}^\dagger \otimes I)P_n, \\ A_1 &= P_n^\dagger \widetilde{\rho}\rho P_n \;=\; \widetilde{\rho}\rho. \end{aligned}$$

3. Compute D such that $D^2 = A_1$ and $\Theta(D) = D^\dagger$. Thus, $D \;=\; \sqrt{\widetilde{\rho}\rho}$.

4. Compute $M = K_1 D K_1^\dagger$.

5. Compute $K_2 = F_n M^\dagger$. Thus, $K_2 \;=\; (I_{d^{n-1}} \otimes F_1)D_\Omega \sqrt{\rho\widetilde{\rho}}^\dagger (\widetilde{F}_{n-1}^\dagger \otimes I)P_n$.

6. Let $U = K_2 K_1$, $V = K_1^\dagger$, then $F_n = UDV$ gives the KAK decomposition.

Then, the CSD of $\widetilde{F}_n$ can be written as $\widetilde{F}_n = UDV$, where

$$\begin{aligned} U &= (I_{d^{n-1}} \otimes F_1)D_\Omega \sqrt{\widetilde{\rho}\rho}^\dagger P_n \;= (I_{d^{n-1}} \otimes F_1)D_\Omega P_n \sqrt{\widetilde{\rho}\rho}^\dagger, \\ D &= \sqrt{\widetilde{\rho}\rho}, \\ V &= P_n^\dagger(\widetilde{F}_{n-1} \otimes I)P_n. \end{aligned}$$

Note that the circuit of F_n is obtained by turning the circuit of $\widetilde{F}_n$ upside down, *i.e.*, by reversing the order of qudits.

For a general $d^n \times d^n$ unitary matrix, both sides of the CSD, *i.e.*, U and V in (4.1), are uniformly controlled operations. However, in decomposition of the QFT, these are operations that act on one-qudit F_1 and on $(n-1)$-qudit

F_{n-1}, respectively. Figure 4.8 shows a quantum circuit for $\widetilde{F}_4$, when $d = 3$ and $n = 4$. Therefore, the proposed method can reduce the number of repetitions of the recursive steps. Furthermore, the produced matrices contain some identity operations and a sequence of conjugate operations, such as $D_\rho^\dagger$ and D_ρ.

Next, a decomposition of D_Ω is considered. As shown in (4.8), D_Ω can be seen as a block-diagonal matrix, where the diagonals composed of $d^{n-1} \times d^{n-1}$ diagonal matrices.

Consider when Ω_n, which is the second diagonal block of D_Ω, is decomposed. Here, Ω_n can be seen as a block-diagonal matrix, with the diagonals composed of $d \times d$ unitary matrices. Let $\beta_1 = diag(1, \omega, \omega^2, \omega^3, \cdots, \omega^{d-1})$, then

$$\begin{aligned} \Omega_n &= diag(\beta_1, \omega^d \beta_1, \omega^{2d} \beta_1, \omega^{3d} \beta_1, \cdots, \omega^{d^{n-2}} \beta_1), \\ &= diag(1, \omega^d, \omega^{2d}, \omega^{3d}, \cdots, \omega^{d^{n-2}}) \otimes \beta_1. \end{aligned}$$

Similarly, the diagonal matrix $diag(1, \omega^d, \omega^{2d}, \omega^{3d}, \cdots, \omega^{d^{n-2}})$ can be seen as a block-diagonal matrix, with the diagonals composed of $d \times d$ unitary matrices. Repeating this process, Ω_n can be written as follows:

$$\begin{aligned} \Omega_n &= \beta_{n-1} \otimes \beta_{n-2} \otimes \cdots \otimes \beta_1, \\ \beta_j &= diag(1, \omega^{d^{j-1}}, \omega^{2d^{j-1}}, \omega^{3d^{j-1}}, \cdots, \omega^{(d-1)d^{j-1}}). \end{aligned}$$

Thus, the sub-block Ω_n can be composed of $n - 1$ one-qudit operations.

The rest of the diagonal blocks of D_Ω are a power of Ω_n. Thus,

$$\Omega_n^k = \beta_{n-1}^k \otimes \beta_{n-2}^k \otimes \cdots \otimes \beta_1^k.$$

Then, D_Ω is written as a sequence of controlled unitary operations. Figure 4.9 shows a quantum circuit for D_Ω when $d = 3$ and $n = 4$. In general, the number of controlled unitary operations appearing in the circuit is $(d - 1)(n - 1)$.

Let $f(n)$ be the number of one- or two-qudit operations used to implement F_n. Then,

$$\begin{aligned} f(n) &= f(n-1) + (d-1)(n-1) + 1 \\ &= \frac{1}{2}(d-1)(n^2 - n) + n. \end{aligned}$$

Since the number of elementary gates needed to implement a two-qudit operation is a constant, the size of the quantum circuit for F_n is $O(n^2)$.

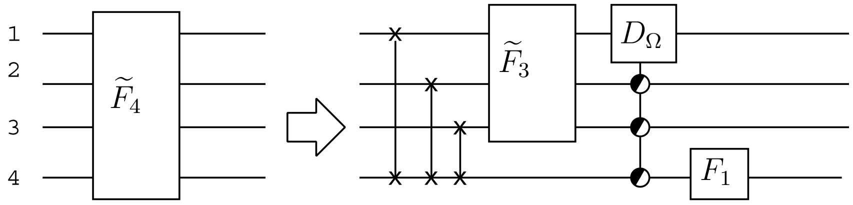


Figure 4.8. Quantum circuit for $\widetilde{F}_4$, when $d = 3$ and $n = 4$.

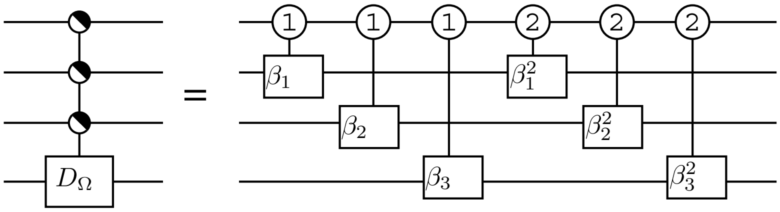


Figure 4.9. Quantum circuit for D_Ω in Figure 4.8.

7. Summary

The size of the quantum circuit produced by the non-CSD based approach proposed by Brennen *et al.*, is asymptotically optimal. However, the number of two-qudit operations appearing in the produced quantum circuit by the method is not optimal. In this chapter, a new method, which is based on the CSD, for synthesizing quantum circuits for the d-level quantum system is proposed. To produce efficient quantum circuits, the author introduces the new CSD based on balanced partitioning. The proposed method can save considerable number of two-qudit operations appearing in the produced quantum circuit, when the number of qudits n is small. For example, when $n = 2$ and $d = 3$, the proposed method can reduce the number of CINC gates appearing in the produced quantum circuit from 156 to 36.

As described in section 4, the CSD for the two-level quantum system is suitable for designing a polynomial-size quantum circuit for the radix-2 QFT. Similarly, the CSD for the d-level quantum system provided in this chapter has the same advantage in producing polynomial-size quantum circuits for the general radix-d QFT.

Future directions will include improving the size of the circuit to $O(d^{2n})$. In the two-level quantum system, the quantum multiplexor decomposition [42] is useful for reducing the number of CNOT gates. A generalization of this multiplexor decomposition to the d-level quantum system will provide the same effect. In addition, the CSD has some redundancy in decomposed matrices. For example, consider the decomposition of the QFT described in Section 6. The matrices $U = (F_1 \otimes I_{n-1})D_\Omega D_\rho^\dagger$ and $D = D_\rho^\dagger$ have redundant (conjugate) elements $D_\rho^\dagger$ and D_ρ. Those elements do not appear in the produced quantum circuit. Thus, it is important to provide a simplification technique that eliminates the redundancy. To find a novel polynomial-size quantum circuit by using the proposed synthesis method will also be included in the future directions.

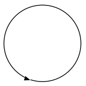
Chapter 5

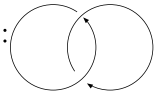
Quantum Circuit for the AJL Algorithm

1. Introduction

After Shor's discovery of his celebrated quantum algorithm, new efficient quantum algorithms that would be exponentially faster than any known conventional algorithm have been sought. A natural strategy to find such quantum algorithms is to select a concrete problem believed to be impossible to solve in polynomial time using a conventional computer and then explore a quantum algorithm that solves the problem in polynomial time. Although NP-complete problems were considered to be hopeful candidates in the early stage, no efficient quantum algorithm for the NP-complete problems seems to exist. Instead of NP-complete problems, BQP-complete problems [9], which are the hardest problems in the class of languages accepted by bounded-error quantum Turing machines in polynomial time, have been focused on recently.

A natural (non-artificial) BQP-complete problem was found by Freedman *et al* [19,20]. They proved the equivalence between quantum field theory and quantum computation and consequently showed that a problem of approximating the Jones polynomial at the fifth root of unity is BQP-complete. The Jones polynomial [2,27,28], $V_L(t)$, is an invariant of an oriented link L and is given as a Laurent polynomial in the variable $\sqrt{t}$ with integer coefficients. For example,

L_1: 

L_2: 

$$V_{L_1}(t) = 1, \qquad V_{L_2}(t) = \sqrt{t}(1 + t^2).$$

It is known that the problem of computing the Jones polynomial is #P-hard [18, 26] in conventional complexity theory. The procedure for computing the Jones polynomial works inductively on the number m of crossings in the given oriented link and seems to grow exponentially as m gets large. The known algorithm for computing the Jones polynomial on a conventional computer is $O(2^{O(\sqrt{m})})$ time when the input knot is given by a link diagram (the input length depends on m) [41]. Another algorithm for computing the 2-variable polynomial, which contains the Jones polynomial as its special case, for knots requires $O(n!m^3)$ time [35], where the input knot is given by a closure of an n-strand braid word of length m (thus, the input size is $m \log(n)$). To approximate the Jones polynomial at a certain value except for some trivial values is also considered to be intractable on conventional computers [3, 18, 21].

A concrete quantum algorithm that approximates the Jones polynomial at the kth root of unity was given by Aharonov, Jones, and Landau [3]. Their algorithm will be called the AJL algorithm in this chapter. In the AJL algorithm, an oriented link is assumed to be given as the trace or the plat closures of an n-strand braid word of length m. Then, the complexity of the AJL algorithm is poly(n, m, k). Although the precise order is not given in the original paper, it can be checked that the algorithm runs in time $O(mn \log^2 k)$ on an $O(n + \log 2k)$-qubit quantum computer. If m and k are bounded by polynomials in n, then the algorithm is a polynomial-time quantum algorithm in the input size (in this case, the input size is polynomial in n). The problem is still intractable on conventional computers even under this assumption. Later, Wocjan and Yard showed that the running time of the AJL algorithm is $O(poly(m))$, *i.e.*, it depends only on the number of crossings m [49].

In this chapter, the author focuses on the quantum circuit design that efficiently performs the AJL algorithm rather than the complexity-theoretic result. A new quantum circuit that implements the AJL algorithm in time $O(mn)$ for $k \geq n/2 + 1$ and in time $O(mk)$ for $k < n/2 + 1$ will be presented, where $O(n^2)$ qubits are necessary to perform this method. The main idea is to change the encoding scheme that assigns qubit states to every path in the path model. The original quantum circuit comprises operations that have many control qubits, while the proposed quantum circuit consists of operations on at most four qubits. This means that the proposed method has an advantage in implementation. The proposed method is useful in a practical sense, because the Jones polynomial is known as a useful knot invariant.

This research direction is also interesting in the theoretical sense. In [29], Jozsa stated a conjecture that "Any polynomial time quantum algorithm can be implemented with only $O(\log n)$ quantum layers interspersed with polynomial time classical computations." Although the conjecture remains unproven in general, Cleve and Watrous have shown that it holds for Shor's algorithm [16]. Jozsa's conjecture is affirmatively answered, if there exists a quantum circuit with logarithmic depth (in the input length of $poly(n)$) for the AJL algorithm.

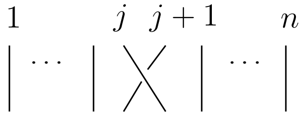
This chapter is organized as follows: Section 2 describes notations and give some brief background about the AJL algorithm. Section 3 presents two explicit quantum circuits for Q_j based on Ref. [3] and on the author's method. Section 4 concludes with a brief summary.

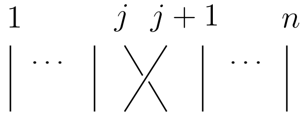
2. Preliminary

Let b be an n-strand braid. In the AJL algorithm, a knot (or a link) is given as the trace closure or as the plat closure of b. Fig. 5.1 shows an example of the trace closure (a) and the plat closure (b). The trace closure of the braid b is a closed braid obtained by connecting the jth point on the top bar to the jth point on the bottom bar, where $j = 1, 2, \cdots, n$. The plat closure of b is a closed braid obtained by connecting the $(2j-1)$th to the $(2j)$th points on the top bar and the bottom bar, respectively. The Jones polynomial is evaluated at the kth root of unity, $t = \exp(2\pi i/k)$. In this case, b is presented as a unitary operation $\rho(b)$, which acts on the Hilbert space $\mathcal{H}_{n,k}$. Here, $\mathcal{H}_{n,k}$ is a subspace of the Hilbert space of dimension 2^n.

The structure of $\mathcal{H}_{n,k}$ is described by paths on the graph G_k, which is a straight line graph with $k-1$ vertices ordered from bottom to top beginning with one, as shown in Fig. 5.2(a). Then, all possible n-step paths over the graph G_k starting from the vertex 1 and never leave G_k are considered. These paths can be viewed as a two-dimensional diagram, as shown in Fig. 5.2(b). All possible paths move right along the grid line in Fig. 5.2(b) starting from $(1, 1)$ and ending at (n, ℓ), where $1 \leq \ell \leq k - 1$. Each n-step path is associated with an n-qubit state by encoding a movement in the upper direction (the walk from vertices j to $j + 1$) by $|1\rangle$ and a movement in the lower direction (the walk from vertices j to $j - 1$) by $|0\rangle$. For example, the paths shown in Fig. 5.2(b), which is an eight-step path that walks $1 \rightarrow 2 \rightarrow 3 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 3 \rightarrow 4 \rightarrow 5$, is encoded as $|11011011\rangle$. Then, $\mathcal{H}_{n,k}$ is the space spanned by all the legitimate n-step paths. A unitary representation $\rho(b)$ can be defined on the path model.

Let σ_j be a crossing between the jth and the $(j + 1)$th strands as

σ_j =


, σ_j^{-1} =


.

Now, let

$$a = -i \exp\left(\frac{\pi i}{2k}\right), \qquad \lambda_\ell = \begin{cases} \sin(\pi\ell/k), & 1 \leq \ell \leq k-1; \\ 0, & \text{otherwise,} \end{cases} \tag{5.1}$$

then, ρ_j can be defined as follows:

Definition 2 *Let q be an encoded path on G_k, i.e., $|q\rangle$ is a basis state of $\mathcal{H}_{n,k}$. Let $q|_j$ be the first $j - 1$ bit string of q, $q|^j$ be the last $n - j - 1$ bit string of q, and $z_j = z_j(q)$ be a label of the vertex reached after $q|_j$. Then,*

$$\begin{aligned}
\rho_j\, |q|_j\, 00\, q|^j\rangle &= a^{-1}\, |q|_j\, 00\, q|^j\rangle, \\
\rho_j\, |q|_j\, 01\, q|^j\rangle &= \alpha_{(z_j,-)}\, |q|_j\, 01\, q|^j\rangle + \beta_{(z_j)}\, |q|_j\, 10\, q|^j\rangle, \\
\rho_j\, |q|_j\, 10\, q|^j\rangle &= \alpha_{(z_j,+)}\, |q|_j\, 10\, q|^j\rangle + \beta_{(z_j)}\, |q|_j\, 01\, q|^j\rangle, \\
\rho_j\, |q|_j\, 11\, q|^j\rangle &= a^{-1}\, |q|_j\, 11\, q|^j\rangle,
\end{aligned}$$

where $\alpha_{(\ell,\pm)} = a\lambda_{\ell\pm1}/\lambda_\ell + a^{-1}$ and $\beta_{(\ell)} = a\sqrt{\lambda_{\ell-1}\lambda_{\ell+1}}/\lambda_\ell$.

Definition 3 *A unitary operation Q_j, which acts on the 2^n-dimensional Hilbert space, can be defined as $Q_j = I \oplus \rho_j$, where I is the identity operation, which acts on the orthogonal complement of the subspace $\mathcal{H}_{n,k}$.*

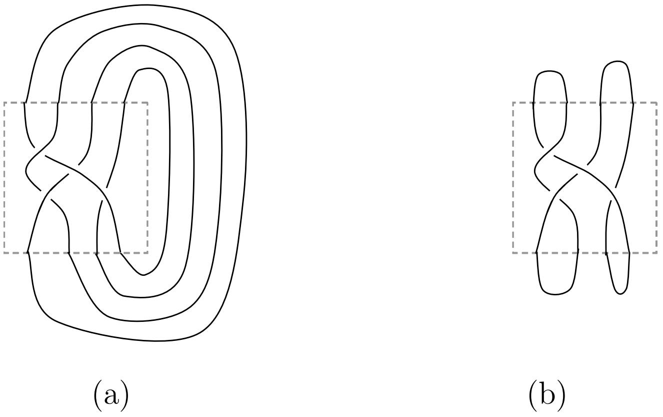


Figure 5.1. An example of (a) the trace closure and (b) the plat closure of a braid.

An n-strand braid b can be constructed by iteratively applying the σ_j ($j = 1, 2, \cdots, n-1$) operator. For example, the braid shown in Fig. 5.1(a) can be denoted as $\sigma_1^\dagger \sigma_2 \sigma_1^\dagger \sigma_3$. Thus, $Q(b)$ can be given as products of the Q_j operators, such as $Q(b) = Q_1^\dagger Q_2 Q_1^\dagger Q_3$. Let b^{tr} and b^{pl} be closed braids obtained as the trace closure of b and as the plat closure of b. Let m be the number of crossings in b. The AJL algorithm is provided for the plat closure and for the trace closure, respectively. The AJL algorithm for the plat closure can be described as follows:

```
Step 1. For $j = 1$ to poly$(n, m, k)$, repeat the following procedure.
        i.  Generate a state $|\psi\rangle = |1010\cdots10\rangle$.
        ii. Output a random variable $x_j$ whose expectation value is
            $Re\langle\psi|Q(b)|\psi\rangle$ using the Hadamard test.
Step 2. Do the same but for random variable $y_j$ whose expectation value
        is $Im\langle\psi|Q(b)|\psi\rangle$ using the variant of the Hadamard test.
Step 3. Let $r$ be the average over all $x_j + iy_j$ achieved in steps 1 and 2.
        Compute an approximation value of $(-a)^{-3w(b^{pl})} d^{3n/2-1} \lambda_1 r/\eta$.
```

In the algorithm, steps 1 and 2 are done on a quantum computer and the final step is done on a conventional computer. Here, $d = -a^2 - a^{-2} = 2\cos(\pi/k)$ and $\eta = \sum_\ell \lambda_\ell \dim(\mathcal{H}_{n,k,\ell})$, where $\mathcal{H}_{n,k,\ell}$ is the subspace spanned by all the

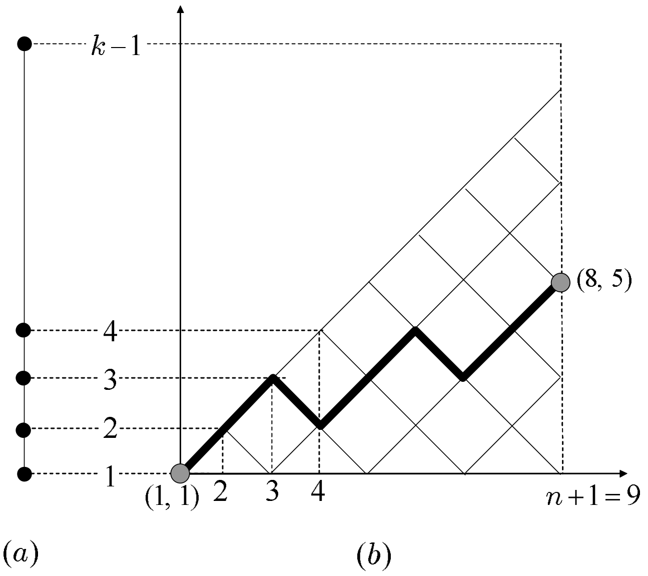


Figure 5.2. (a) The graph G_k and (b) the path model diagram corresponding to G_k.

legitimate n-step paths starting from $(1, 1)$ and ending at (n, ℓ). The *writhe* of a link L, denoted by $w(L)$, is defined as the total number of positive crossings minus the total number of negative crossings in the given link. For the trace closure, a path $|p\rangle \in \mathcal{H}_{n,k}$ is chosen, where the location the path ends on (k) is chosen according to some specific probabilities, instead of $|\psi\rangle$ and compute the expectation values. More precisely, the input states are not chosen uniformly at random from all possible paths, but rather the location the path ends on is chosen according to some specific probabilities. Then, an approximation value of the Jones polynomial can be computed by $(-a)^{-3w(b^{tr})}d^{n-1}r$.

Aharonov, Jones, and Landau showed that an n-qubit unitary operation $Q(b)$ can be implemented using poly(n, m, k) elementary gates. Although the AJL algorithm applies the Hadamard test (steps 1 and 2) recursively up to poly(n, m, k) times, these recursive quantum computations can be parallelized; thus, the AJL algorithm can be performed in time polynomial in n, m and k. For details of the algorithm and the Hadamard test, see *e.g.*, Ref. [3, 11].

3. Quantum Circuit for Implementing the AJL Algorithm

In this section, the implementation of a unitary operation $Q(b)$, which is an important operation in the AJL algorithm, is discussed. Here, the author studies the implementation of Q_j, which is a unitary operation corresponding to a crossing between the jth and the $(j+1)$th strands. Designing a quantum circuit for Q_j is sufficient for designing a quantum circuit for $Q(b)$ because $Q(b)$ is given as products of elements of $\{Q_1, Q_2, \cdots, Q_{n-1}\}$. To compare the performance between the original quantum circuit and the proposed quantum circuit, Section 3.1 will presents the details of implementation of Q_j based on the method provided in Ref. [3]. Then, Section 3.2 provides a new quantum circuit for implementing Q_j. An example will be given in Section 3.3. Throughout the rest of this chapter, the following notations are used.

Definition 4 *Let U be a 2×2 unitary matrix. For $x_1, x_2, \cdots, x_s, x_{s+1} \in \{0,1\}$, an s-fold controlled unitary operation, denoted by $\wedge_s(U)$, is defined as follows:*

$$\wedge_s(U) \;=\; \begin{cases} |x_1, x_2, \cdots, x_s\rangle \otimes U|x_{s+1}\rangle, & \text{if } |\mathrm{x_1, x_2, \cdots x_s}\rangle = |\mathrm{2^s - 1}\rangle; \\ |x_1, x_2, \cdots, x_s, x_{s+1}\rangle, & \text{otherwise.} \end{cases}$$

Definition 5 *Let U_j's ($j = 0, 1, 2 \cdots, 2^s - 1$) be 2×2 unitary matrices. A s-fold uniformly controlled gate, denoted by $\vee_s(U(2))$, is defined as follows:*

$$\vee_s(U(2)) \;=\; \sum_{j=0}^{2^s-1} |j\rangle\langle j| \otimes U_j \;=\; diag(U_0, U_1, U_2, U_3, \cdots, U_{2^s-1}).$$

A uniformly controlled unitary operation was first introduced in Ref. [37] and is regarded as a sequence of controlled unitary operations that applies U_j to the target qubits when the target qubits are in the state $|j\rangle$.

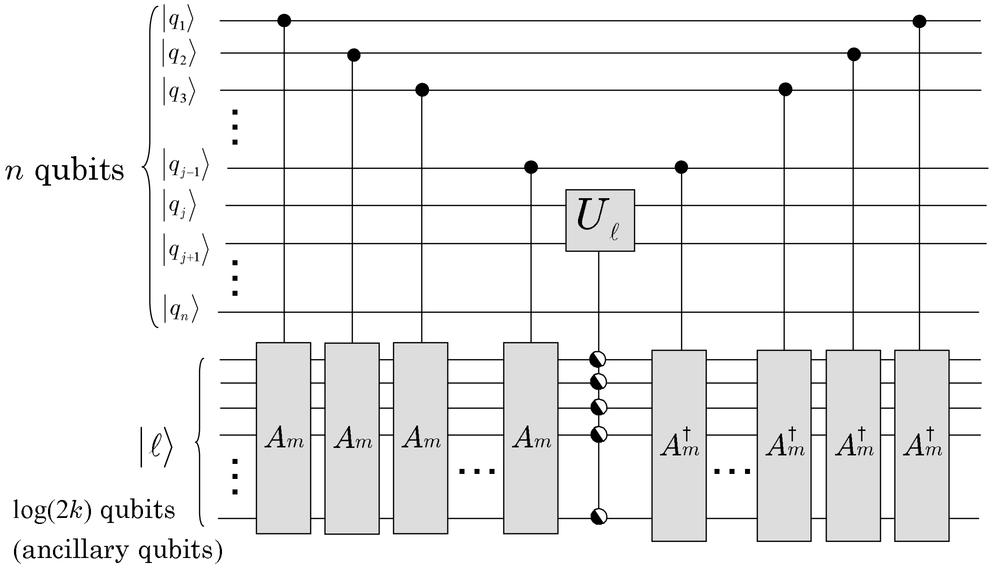


Figure 5.3. A quantum circuit for Q_j.

3.1 Previous Quantum Circuit Design

In this section, the author investigates an explicit quantum circuit for Q_j, which is based on the design proposed in Ref. [3] (Claim 3.2). Let

$$\begin{aligned} U_\ell &= \begin{pmatrix} a^{-1} & 0 & 0 & 0 \\ 0 & \alpha_{(\ell,-)} & \beta_{(\ell)} & 0 \\ 0 & \beta_{(\ell)} & \alpha_{(\ell,+)} & 0 \\ 0 & 0 & 0 & a^{-1} \end{pmatrix} = diag(a^{-1}, R_\ell, a^{-1}), \qquad (5.2) \\ R_\ell &= \begin{pmatrix} \alpha_{(z_j,-)} & \beta_{(z_j)} \\ \beta_{(z_j)} & \alpha_{(z_j,+)} \end{pmatrix}. \end{aligned}$$

Claim 2 *The unitary operation Q_j in Definition 3 can be implemented using $O(n \log^2 k)$ elementary gates.*

Proof. Fig. 5.3 shows a quantum circuit for implementing Q_j. Here, $O(\log 2k)$ ancillary qubits are used as a counter. The counter $|\ell\rangle$ is initially set to $|1\rangle$ and the $\wedge_1(A_m)$ gate acts as follows: $\wedge_1(A_m)|q_j\rangle|\ell\rangle = |q_j\rangle|\ell-(-1)^{q_j} \mod 2k\rangle$. Thus,

$|\ell\rangle$ increases by one when the control qubit $|q_j\rangle = |1\rangle$ (a movement in the upper direction in Fig. 5.2), and decreases by one when the control qubit $|q_j\rangle = |0\rangle$ (a movement in the opposite direction in Fig. 5.2). After applying the sequence of $\wedge_1(A_m)$ gates, the counter $|\ell\rangle$ equals z_j.

Then, the two qubit operation U_ℓ is applied on the jth and the $(j+1)$th qubits. Since various U_ℓ's are applied depending on the value of the counter $|\ell\rangle$, this operation can be regarded as $\vee_{O(\log 2k)}(U_\ell)$ gates, which is a sequence of $\wedge_{O(\log 2k)}(U_\ell)$ gates as shown in Fig. 5.4(a). In practice, not all the $\wedge_{O(\log 2k)}(U_\ell)$ gates, where $\ell = 0, 1, \cdots, 2^{O(\log 2k)} - 1$, are used because the number of $\ell\,(= z_j)$'s appearing in Q_j, *i.e.*, the number of vertices reached after the first $j-1$ steps paths in the path model diagram, is $\min(\lceil j/2 \rceil, k-1)$. Finally, a sequence of $\wedge_1(A_m^\dagger)$ gates is used to initialize the counter.

Here, the number of elementary gates used in the quantum circuit can be estimated as follows: A sequence of $\wedge_1(A_m)$ and $\wedge_1(A_m^\dagger)$ gates can be implemented using $O(j \log^2 k)$ elementary gates because $\wedge_1(A_m)$ can be implemented using $O(\log^2 k)$ elementary gates [8, 48] and the number of $\wedge_1(A_m)$ and $\wedge_1(A_m^\dagger)$ gates used in the circuit is $2(j-1)$, where $1 \leq j \leq n-1$. Thus, the number of elementary gates needed to implement a sequence of $\wedge_1(A_m)$ gates and a sequence of $\wedge_1(A_m^\dagger)$ gates is bounded by $n \log^2 k$.

The two qubit unitary operation U_ℓ can be implemented using the two CNOT gates and one controlled unitary gate $\wedge_1(a \cdot R_\ell)$ as shown in Fig. 5.4(b), so the $\wedge_{O(\log 2k)}(U_\ell)$ gate is transformed into a sequence of $\wedge_{O(\log 2k)+1}(X)$ and $\wedge_{O(\log 2k)+1}(a \cdot R_\ell)$ gates, where X is the NOT gate. Thus, the $\wedge_{O(\log 2k)}(U_\ell)$ gate can be implemented using $O(\log^2 k)$ elementary gates using the result in Ref. [6]. The number of $\wedge_{O(\log 2k)}(U_\ell)$ gates appearing in Q_j is $j/2$ for $k \geq n/2+1$ and $k-1$ for $k < n/2+1$. Therefore, the number of elementary gates used to implement Q_j can be bounded by $n \log^2 k$. □

3.2 Proposed Quantum Circuit Design

In the previous method, an n-step path is encoded as an n-qubit state. In the proposed new design, qubits are arranged at each vertex in the path model diagram. The number of qubits required to encode n-step paths is $\sum_{j=1}^{n+1} \min(\lceil j/2 \rceil, \lceil k/2 \rceil)$. If $n \leq k+1$, then the number of qubits is $(n/2+1)^2$ for even n and $n^2/4+3n/2+2$

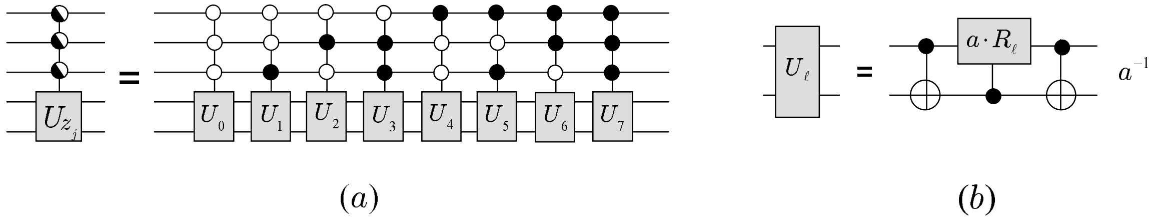


Figure 5.4. Quantum circuits for (a) the uniformly controlled operation $\vee_3(U_j)$ and (b) U_ℓ. The scalar phase factor indicated at the end of the quantum circuit is the total phase.

for odd n. The qubits on the path are encoded as $|\overline{1}\rangle$ and the rest of qubits are encoded as $|\overline{0}\rangle$. For example, for $n = 8$ and $k > 7$, $|\psi\rangle = |10101010\rangle$ is encoded as $|\widetilde{\psi}\rangle = |\overline{1101010010010001000010000}\rangle$. Here, qubits are indexed from the lower left to the upper right beginning with one. Fig. 5.5(a) shows the initial arrangement of qubits of state $|\psi\rangle$. Here, each circle represents a qubit. A black circle denotes a qubit of state $|\overline{1}\rangle$ and a white circle denotes a qubit of state $|\overline{0}\rangle$.

Let q_j be the j-th qubit in the original qubit layout and $|q_j\rangle$ denotes the state of q_j. Let $p_{(j,\ell)}$ be the qubit at (j,ℓ) in the proposed qubit layout. Consider the case that $|q_j q_{j+1}\rangle$ is $|\overline{00}\rangle$, *i.e.*, $|q|_j 00q|^j\rangle$ in the original encoding scheme. In the proposed qubit layout, $|p_{(j,z_j)} p_{(j+1,z_j-1)} p_{(j+2,z_j-2)}\rangle$ are set to $|\overline{111}\rangle$ because these qubits are on the path $|q|_j 00q|^j\rangle$ [see Fig. 5.5(b)]. Similarly, depending on the states of q_j and q_{j+1}, the states of some qubits around $p_{(j,z_j)}$ on the proposed qubit layout is determined. Fig. 5.6 shows correspondence between the original path encodings and the state of qubits in the proposed qubit layout. Then, Q_j can be redefined as follows:

Definition 6 *Let $p_{(j,\ell)}$ be the qubit at (j,ℓ) in the proposed qubit layout. Then, Q_j, which is defined in Definition 3, can be rewritten as an operation that acts on the four qubits, $p_{(j,z_j)}$, $p_{(j+2,z_j)}$, $p_{(j+1,z_j-1)}$, and $p_{(j+1,z_j+1)}$. The state of the qubits is denoted as $|p_{(j,z_j)}, p_{(j+2,z_j)}\rangle \otimes |p_{(j+1,z_j-1)}, p_{(j+1,z_j+1)}\rangle$, where the former*

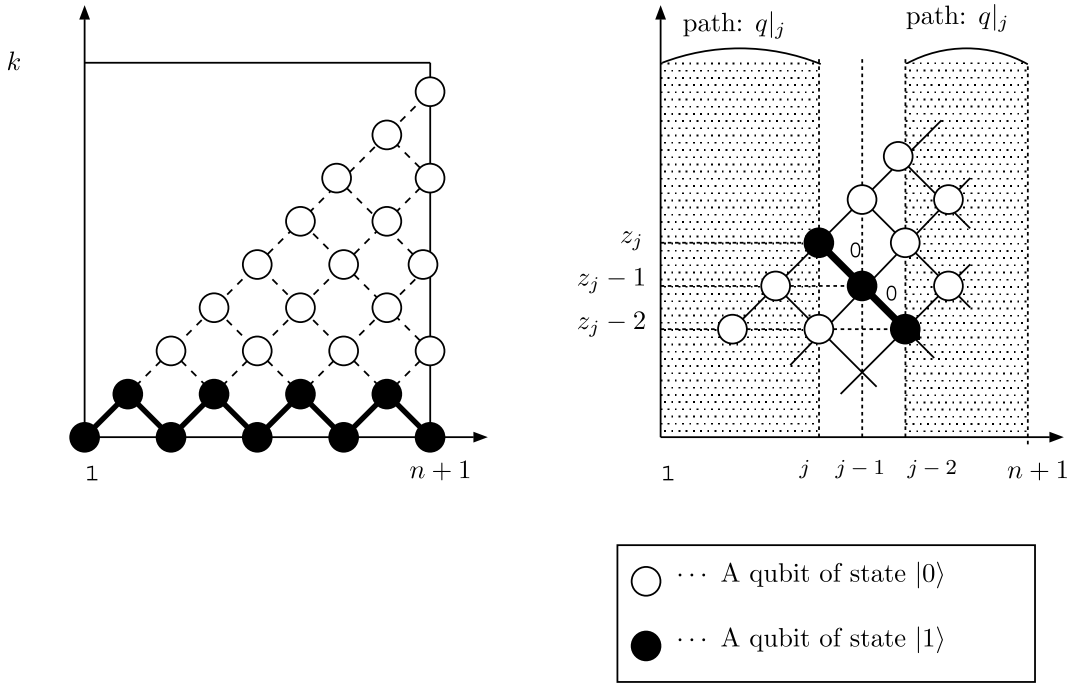


Figure 5.5. (a) Initial arrangement of qubits of state $|\psi\rangle$ and (b) the state of qubits corresponding to $|p|_j 00 p|^j\rangle$ in the qubit layout.

two qubits are the control qubits and the latter two qubits are the target qubits.

$$\widetilde{Q}_j\,|\overline{10}\rangle \otimes |\overline{10}\rangle = |\overline{10}\rangle \otimes\ a^{-1}\,|\overline{10}\rangle, \tag{5.3}$$

$$\widetilde{Q}_j\,|\overline{11}\rangle \otimes |\overline{10}\rangle = |\overline{11}\rangle \otimes \left(\alpha_{(z_j,-)}\,|\overline{10}\rangle + \beta_{(z_j)}\,|\overline{01}\rangle\right), \tag{5.4}$$

$$\widetilde{Q}_j\,|\overline{11}\rangle \otimes |\overline{01}\rangle = |\overline{11}\rangle \otimes \left(\alpha_{(z_j,+)}\,|\overline{01}\rangle + \beta_{(z_j)}\,|\overline{10}\rangle\right), \tag{5.5}$$

$$\widetilde{Q}_j\,|\overline{10}\rangle \otimes |\overline{01}\rangle = |10\rangle \otimes a^{-1}|\overline{01}\rangle, \tag{5.6}$$

For the rest of states of the target qubits, $\widetilde{Q}_j$ acts as the identity operation.

Note that if $z_j = 1$, ρ_1 can be defined as follows:

$$\begin{aligned} \widetilde{Q}_j\,|\overline{10}\rangle \otimes |\overline{0}\rangle &= |\overline{10}\rangle \otimes |\overline{0}\rangle, & \widetilde{Q}_j\,|\overline{11}\rangle \otimes |\overline{0}\rangle &= |\overline{11}\rangle \otimes |\overline{0}\rangle, \\ \widetilde{Q}_j\,|\overline{10}\rangle \otimes |\overline{1}\rangle &= |\overline{10}\rangle \otimes a^{-1}|\overline{1}\rangle, & \widetilde{Q}_j\,|\overline{11}\rangle \otimes |\overline{1}\rangle &= |\overline{11}\rangle \otimes \alpha_{(1,+)}\,|\overline{1}\rangle, \end{aligned}$$

where the target qubit is $p_{(j+1,z_j+1)}$.

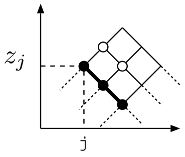
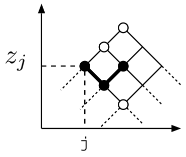
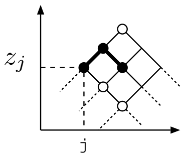
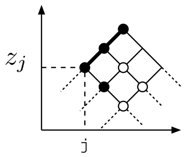
Original path encoding	State of qubits in the proposed model	Original path encoding	State of qubits in the proposed model
$q\|_j 00 q\|^j$	 Control = $\|\overline{10}\rangle$ Target = $\|\overline{10}\rangle$	$q\|_j 01 q\|^j$	 Control = $\|\overline{11}\rangle$ Target = $\|\overline{10}\rangle$
$q\|_j 10 q\|^j$	 Control = $\|\overline{11}\rangle$ Target = $\|\overline{01}\rangle$	$q\|_j 11 q\|^j$	 Control = $\|\overline{10}\rangle$ Target = $\|\overline{01}\rangle$

Figure 5.6. Correspondence between t¡he original path encoding and the state of qubits in the proposed model.

Claim 3 *The unitary operation $\widetilde{Q}_j$ in Definition 6 can be implemented using $O(n)$ elementary gates for $k \geq n/2 + 1$ and using $O(k)$ elementary gates for $k < n/2 + 1$.*

Proof. Let $T = diag(1, a^{-1}, a^{-1}, 1)$ and $V_\ell = diag(1, \widetilde{R}_\ell, 1)$, where $\widetilde{R}_\ell = XR_\ell X$ (X is the NOT gate). The operation defined in (5.3) and (5.6) can be implemented as a controlled unitary operation that applies T to the target qubits $(q_{(j,z_j-1)},\ q_{(j,z_j+1)})$ when the control qubits $(q_{(j,z_j)}$ and $q_{(j+2,z_j)})$ are in the state $|\overline{10}\rangle$. Similarly, the operation defined in (5.5) and (5.4) can be implemented as a controlled unitary operation that applies V_{z_j} to the target qubits $(q_{(j,z_j-1)}$ and $q_{(j,z_j+1)})$ when the control qubits $(q_{(j,z_j)}$ and $q_{(j+2,z_j)})$ are in the state $|\overline{11}\rangle$. Let a sequence of $\wedge_2(T)$ and $\wedge_2(V_{z_j})$ gates that applied on the four vertices $\{(j, z_j), (j+1, z_j-1), (j+1, z_j+1), (j+2, z_j)\}$ be $B_{(j,z_j)}$. Then, the operation $\widetilde{Q}_j$ is replaced by a sequence of $B_{(j,z_j)}$'s on the subspace of the qubits spanned by legitimate paths.

For example, consider when $j = 4$. Fig. 5.7 shows an image of operation $B_{(j,z_j)}$ in the proposed qubit layout and the quantum circuit for $\widetilde{Q}_j$ for $j = 4$. In the qubit layout, vertices (qubits) are indexed from the bottom-left to the top-right. There are two qubits of length $j = 4$, *i.e.*, the fifth and the sixth qubits. The operation $B_{(4,2)}$ is applied on four qubits, where the controlled qubits are indexed by 5 and 10 and the target qubits are indexed by 7 and 8. Similarly, the operation $B_{(4,4)}$ is applied on the controlled qubits of indices 6 and 11 and the target qubits of indices 8 and 9.

In general, the number of $B_{(j,z_j)}$ gates used to implement $\widetilde{Q}_j$ is $\min(\lceil j/2 \rceil, k-1)$, which is equal to the number of vertices of length j in the proposed qubit layout, *i.e.*, the number of vertices reached after the first $j-1$ steps paths in the path model diagram. Here, Fig. 5.8(a) shows a quantum circuit for $B_{(j,\ell)}$. Here, $P = diag(1, a^{-1})$ and R_ℓ is the single-qubit operation defined in (5.2). The two qubit gates T and V_ℓ can be implemented using the two CNOT gates and one single-qubit operation P (or one controlled unitary operation $\wedge_1(R_\ell)$). After simplifying the quantum circuit, $B_{(j,\ell)}$ can be implemented using three $\wedge_3(W)$ gates and two $\wedge_2(W)$ gates, where W is a single-qubit operation, *e.g.*, P, R_ℓ, and X. It is known that $\wedge_{n-1}(W)$ gate can be implemented using $O(n)$ elementary gates with one ancillary qubit. Therefore, the number of elementary gates needed to implement $B_{(j,z_j)}$ can be regarded as a constant. Thus, the number of elementary gates used to implement $\widetilde{Q}_j$ can be bounded by $O(n)$ for $k \geq n/2+1$ and by $O(k)$ for $k < n/2+1$. □

3.3 Example

As an example, consider a 4-strand braid ($n = 4$). Assume that $k > 5$, then legitimate paths in G_k are as follows:

$$\begin{aligned}
\{|1010\rangle, |1100\rangle\} &\in \mathcal{H}_{n,k,1}, \\
\{|1011\rangle, |1101\rangle, |1110\rangle\} &\in \mathcal{H}_{n,k,3}, \\
\{|1111\rangle\} &\in \mathcal{H}_{n,k,5}.
\end{aligned}$$

Thus, $\eta = 2\lambda_1 + 3\lambda_3 + \lambda_5$. The unitary operations ρ_1, ρ_2 and ρ_3, which work in the basis ordered as above, are as follows.

$$\begin{aligned}
\rho_1 &= diag\left(\alpha_{(1,+)}, \ a^{-1}, \ \alpha_{(1,+)}, \ a^{-1}, \ a^{-1}, \ a^{-1}\right), \\
\rho_2 &= diag\left(R_2, \ R_2, \ a^{-1}, \ a^{-1}\right), \\
\rho_3 &= \left(\alpha_{(1,+)}, \ a^{-1}, \ a^{-1}, \ R_3, \ a^{-1}\right).
\end{aligned}$$

Now, let $\widetilde{Q}_1$, $\widetilde{Q}_2$, and $\widetilde{Q}_3$ be extensions of ρ_1, ρ_2, and ρ_3 in the proposed qubit layout. Fig. 5.9 shows quantum circuits for $\widetilde{Q}_1$, $\widetilde{Q}_2$, and $\widetilde{Q}_3$.

Consider when a unitary representation of the braid is as shown in Fig. 5.1(a). Since $\rho(b)$ is obtained by $\rho_1^{-1}\rho_2\rho_{-1}\rho_3$, a quantum circuit for $\widetilde{Q}(b)$ is obtained by concatenating a quantum circuit for $\widetilde{Q}_3$, $\widetilde{Q}_1^\dagger$, $\widetilde{Q}_2$, and $\widetilde{Q}_1^\dagger$ in this order.

4. Summary

In this chapter, the author presents a new method for implementing $Q(b)$, which is a unitary operation used to compute the Jones polynomial approximation using the AJL algorithm. If the given n-strand braid b has m crossings, then the original method by Aharonov, Jones, and Landau can implement $Q(b)$ using $O(mn \log^2 k)$ elementary gates. In the proposed method, $\widetilde{Q}(b)$ is used instead of $Q(b)$, and $\widetilde{Q}(b)$ can be implemented using $O(mn)$ or $O(mk)$ elementary gates for $k \geq n/2 + 1$ or $k < n/2 + 1$, respectively, by arranging qubits as a two-dimensional array. Thus, the proposed method requires $O(n^2)$ qubits, whereas the original one uses $n + O(\log 2k)$ qubits. The proposed method has an advantage in speed if k is a large number.

In general, the number of elementary gates used to implement $\widetilde{Q}(b)$ can be bounded by $O(mn)$ because $O(mk)$ holds only when $k < n/2 + 1$, *i.e.*, $O(k) < O(n)$. Thus, the performance does not depend on k in the proposed method. This means that the AJL algorithm can be implemented in polynomial time when m is bounded by a polynomial in n.

In addition, the cost of transforming a unitary operation $Q(b)$ into a sequence of elementary gates in the proposed method is smaller than that in the original method. In the proposed method, $\widetilde{Q}_j$ can be written as a sequence of $B_{(j,z_j)}$ gates, which is a sequence of controlled unitary operations, where the number of controlled qubits is two or three. But, in the original method, Q_j can be written as a sequence of unitary operations, where the number of controlled qubits is $O(\log 2k)$ (for the $\vee_{O(\log 2k)}(U_\ell)$ gate). Thus, as the number of controlled qubits increases, it is costly to determine each rotation angle of one-qubit rotations appearing in the decomposed quantum circuit.

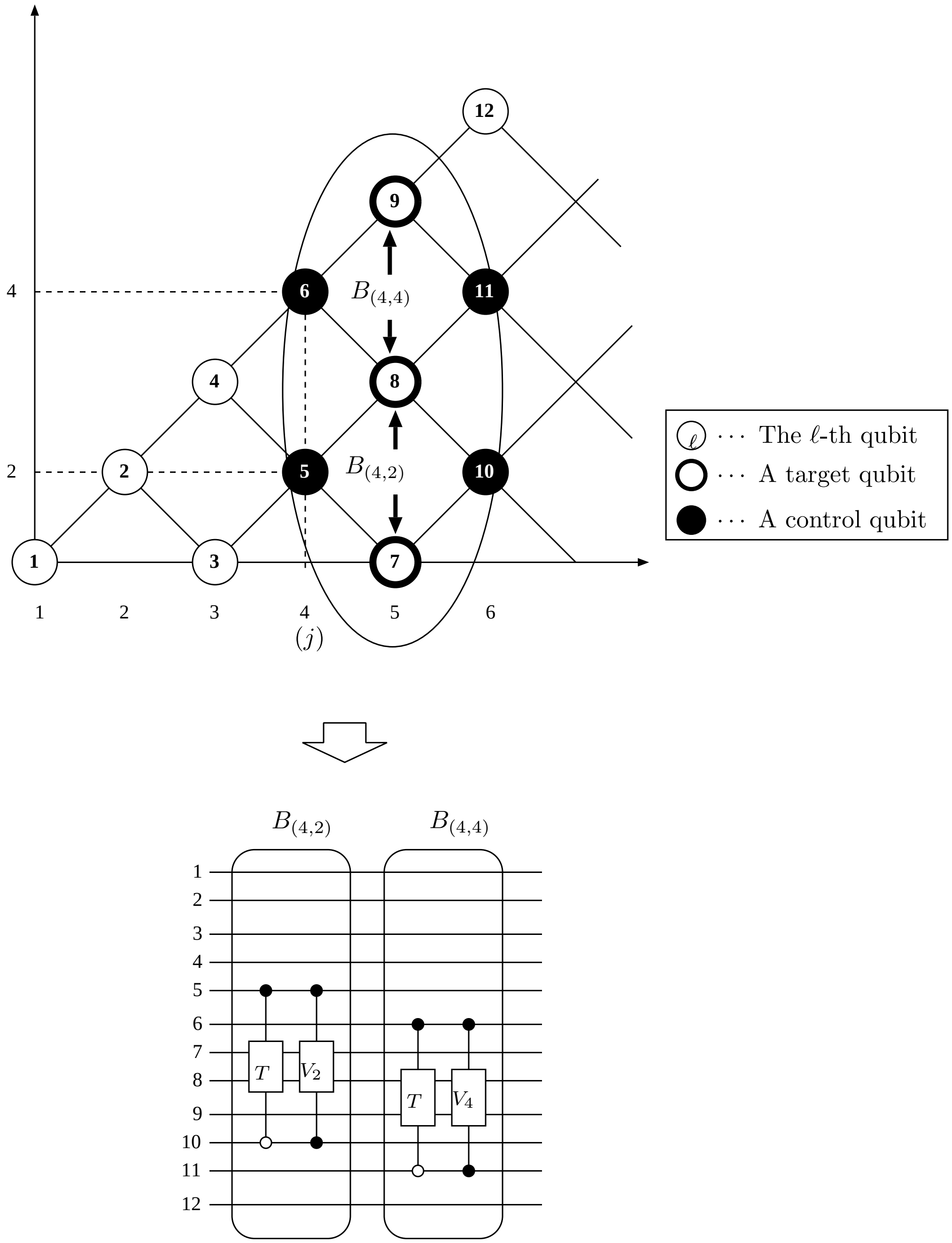


Figure 5.7. An image of operation $B_{(j,z_j)}$ for $j = 4$ and its quantum circuit.

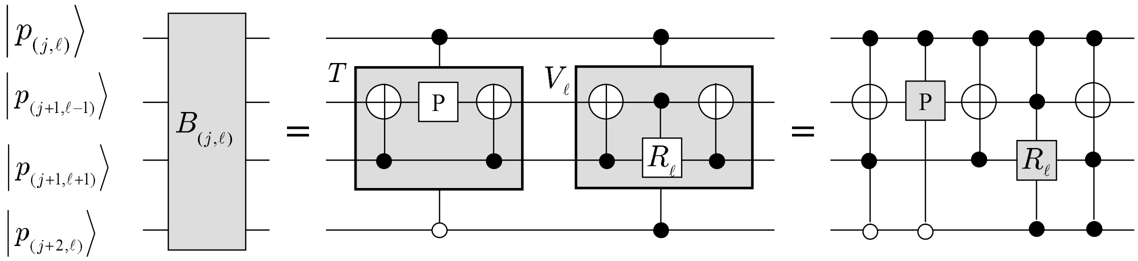


Figure 5.8. A quantum circuit for $B_{(j,\ell)}$.

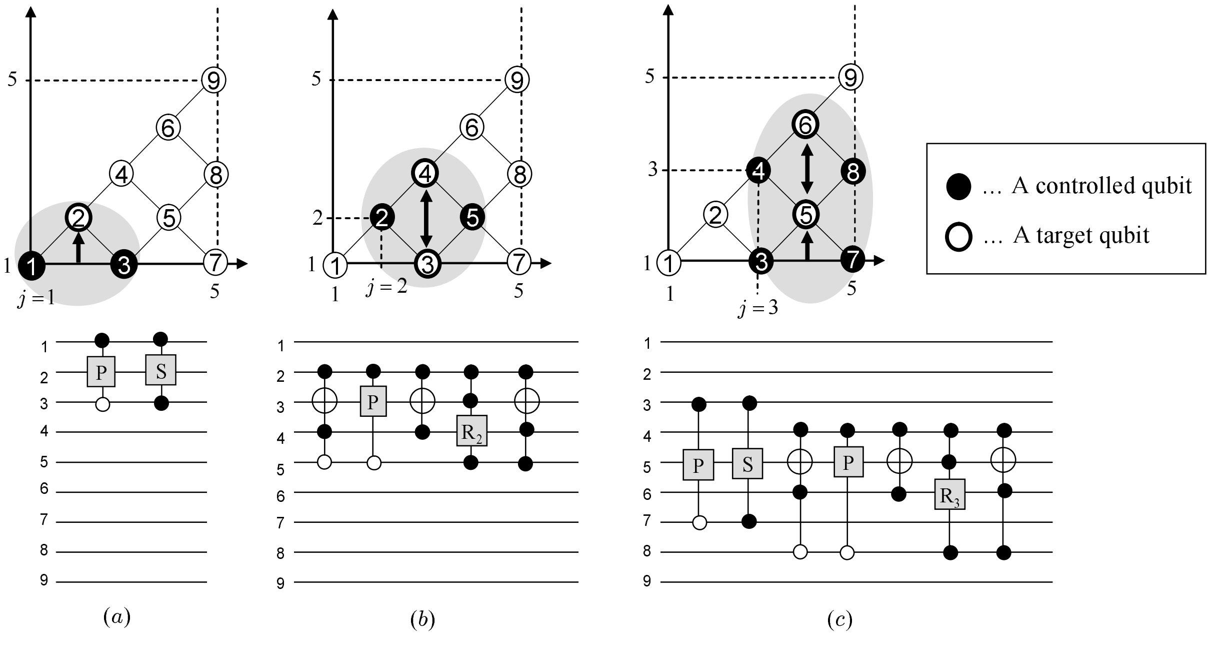


Figure 5.9. Examples of quantum circuits for (a) $\widetilde{Q}_1$, (b) $\widetilde{Q}_2$, and (c) $\widetilde{Q}_3$ in the proposed method when $n = 4$ and $k \geq 5$.

Chapter 6

Conclusion and Future Directions

In this thesis, the author describes two methods for the synthesis of quantum circuits by matrix decomposition. One method for the synthesis of the two-level quantum system (Chapter 3) involves the KAK matrix decomposition. Here, a generic algorithm for computing the KAK decomposition is presented. The KAK decomposition involves variants of decomposition, and the quantum circuits synthesized by the author's synthesis methods differ depending on which variant (Cartan involution) is chosen for the KAK decomposition.

The second method is used for the synthesis of the d-level quantum system (Chapter 4). Here, a new partitioning method termed balanced partitioning was introduced. This method also involves the KAK decomposition, and it is based on a divide-and-conquer strategy. The KAK decomposition is preferred in the two-level quantum system, because it is roughly the SVD of each partitioned matrix, where the input matrix is partitioned into four sub-blocks. However, to apply the KAK decomposition to a $d^n \times d^n$ unitary matrix, the size of partitioning is an important factor for achieving computational efficiency and a suitable size of quantum circuits synthesized by the above methods.

Balanced partitioning results in the most appropriate partition sizes in each step of the KAK decomposition. This method can reduce the considerable number of repetitions of the KAK decomposition in order to effectively synthesize quantum circuit. Therefore, the method reduces the number of elementary gates appearing in the quantum circuit synthesized by the proposed method.

The author also shows that the proposed method is useful for synthesizing polynomial-size quantum circuits for the radix-d QFT, where d is any integer greater than or equal to two. In general, to synthesize an efficient quantum circuit, some simplification or optimization techniques are required after the proposed synthesis method. In future, the author intends to develop such techniques. However, it is difficult to develop efficient and effective simplification methods for the synthesis of arbitrary quantum circuits. To synthesize quantum circuit for the radix-d QFT, no such techniques are required, and a polynomial-size quantum circuit is automatically synthesized. This is an advantage of the author's synthesis method compared to those of the other synthesis methods that involve another matrix decomposition, such as the QR decomposition and spectral decomposition.

An interesting problem is to determine the families of unitary operators that can be computed effectively, *i.e.*, in polynomial time in the quantum circuit model. Such a family of unitary operators includes the Clifford group $\mathcal{C}_n = \{U \in SU(2^n) | UPU^\dagger \in \mathcal{P}_n, \forall P \in \mathcal{P}_n\}$, where $\mathcal{P}_n$ is the Pauli group $\mathcal{P}_n = \{\pm 1, \pm i\} \cdot \{I, X, Y, Z\}^{\otimes n}$ for the identity matrix I and the Pauli matrices X, Y, and Z. Any Clifford operation can be implemented by combining Hadamard gate, phase gate, and the CNOT gate. The size of the quantum circuit is at most $O(n^2/\log n)$ [1, 23, 39]. As already mentioned, the radix-d QFT is also an example of the unitary operation contained in such a family. However, other families of unitary operators that can be computed effectively is not known, and to determine or describe the properties of such families in quantum circuit model is still an open problem.

Since the KAK decomposition have reference to Lie group theory, the matrices (gates) produced by the decomposition have some reference to Lie group theory. Therefore, the proposed synthesis method can be a useful tool for describing the properties of families of unitary operators (input unitary matrices) that can be translated into polynomial-size quantum circuits. This is a future applicability of the proposed synthesis method.

In Chapter 5, the author synthesizes a novel quantum circuit that executes the AJL algorithm. The AJL algorithm approximates the problem of evaluating the Jones polynomial at the k-th root of unity. The problem is considered to be intractable on conventional computers. The author presents a new quantum circuit that executed the AJL algorithm in $O(mn)$, whereas the previous quantum circuit required $O(mn \log^2 k)$ elementary gates and $O(n + \log 2k)$ qubits, where m is the number of crossings and n is the number of strands of the input knot. Thus, the result shows that the efficiency of the AJL algorithm does not depend on k. If a logarithmic-depth quantum circuit that implements the AJL algorithm is synthesized, the important conjecture, which states that any polynomial time quantum algorithm can be implemented with only $O(\log n)$ (where n is the input length) quantum layers interspersed with polynomial time conventional computations [29], can be affirmatively proved. It is an interesting future direction to synthesize such a quantum circuit by developing the idea described in Chapter 5.

Acknowledgements

The author would like to gratefully acknowledge Professor Yasuhiko Nakashima of Nara Institute of Science and Technology (NAIST) for supervising this dissertation. The author would like to sincerely appreciate his continuous support, guidance and encouragement.

The author wishes to express her gratitude to Professor Hiroyuki Seki of NAIST for proof reading this dissertation. The author would like to sincerely appreciate for his valuable comments and critical comments.

The author would like to especially thank Associate Professor Shigeru Yamashita of NAIST (at present, who is a professor of Ritsumeikan University) for his continuous guidance, support, and for proof reading this dissertation. The author would also like to thank Assistant Professor Masaki Nakanishi of NAIST (at present, who is a associate professor of Yamagata University) for his continuous guidance, support, and for proof reading this dissertation. This dissertation owes much to their thoughtful and helpful comments.

The author also like to express her appreciation to Dr. Hiroshi Nishimura and Dr. Takayuki Yasuno of Nippon Telegraph and Telephone Corporation (NTT) for their support and encouragement. This dissertation could not be accomplished without their understanding.

This dissertation is based on the studies that was carried out at NTT Communication Science Laboratories (NTT CS Labs.). The author would like to express her appreciation to Dr. Yoshinobu Tonomura, Dr. Tatsuya Hirahara, Dr. Takehiro Moriya, Dr. Eisaku Maeda, Dr. Kiyoshi Shirayanagi, Dr. Toshiharu Sugawara, and Dr. Yoshifumi Manabe for giving her the opportunity to complete this work at NTT CS Labs.

The author would like to express her appreciation to the members of the

Computing Theory Research Group of NTT CS Labs. The author would like to especially thank Dr. Hiroshi Sekigawa and Dr. Yasuhito Kawano for their fruitful collaborations. All this work could not be carried out without their continuous guidance and valuable advices. The author would also like to thank Dr. Seiichiro Tani, Dr. Yasuhiro Takahashi, Dr. Go Kato, and Dr. Ken Mano for fruitful discussions and valuable advices. The author is also indebted to Dr. Fumiaki Morikoshi of NTT Basic Research Laboratories for his valuable comments.

The author would like to appreciate Professor Akio Hosoya of Tokyo Institute of Technology, Professor Mikio Nakahara of Kinki University, Dr. Juha J. Vartiainen, Dr. Mikko Möttönen, and Dr. Ville Bergholm of Helsinki University of Technology for their helpful discussions and suggestions related to Chapter 3. Also, the author would like to appreciate the helpful discussions with Associate Professor Igor Markov of University of Michigan on several points in Chapter 4.

The author also would like to thank Assistant Professor Takashi Nakada and the other members of NAIST Nakashima Laboratory for fruitful discussions and helpful advices.

Finally, the author would like to thank her husband and her parents from the bottom of her heart for continuous support and encouragement.

References

[1] Aaronson, S. and Gottesman, D.: Inproved simulation of stabilizer circuits, *Phys. Rev. A*, Vol. 70, p. 052328 (2004).

[2] Adams, C. C.: *The Knot Book: An Elementary Introduction to the Mathematical Theory of Knots*, W. H. Freeman & Co., New York (1994).

[3] Aharonov, D., Jones, V. and Landau, Z.: A Polynomial Quantum Algorithm for Approximating the Jones Polynomial, *Proc. of the 38th annual ACM symposium on Theory of Computing (STOC'06)*, Seattle, WA, USA, pp. 427–436 (2006). quant-ph/0511096.

[4] Bacon, D., Chuang, I. L. and Harrow, A. W.: The quantum Schur transform: I. Efficient qudit circuits. quant-ph/0601001.

[5] Bacon, D., Chuang, I. L. and Harrow, A. W.: Efficient quantum circuits for Schur and Clebsh-Gordan transforms, *Phys. Rev. Lett.*, Vol. 97, p. 170502 (2006).

[6] Barenco, A., Bennett, C. H., Cleve, R., DiVincenzo, D. P., Margolus, N., Shor, P., Sleator, T., Smolin, J. A. and Weinfurter, H.: Elementary gates for quantum computation, *Phys. Rev. A*, Vol. 52, No. 5, pp. 3457–3467 (1995).

[7] Bartlett, S. D., de Guise, H. and Sanders, B. C.: Quantum encodings in spin systems and harmonic oscillators, *Phys. Rev. A.*, Vol. 65, p. 052316 (2002).

[8] Beauregard, S., Brassard, G. and Fernandez, J. M.: Quantum arithmetic on Galois Fields. quant-ph/0301163.

[9] Bernstein, E. and Vazirani, U.: Quantum Complexity Theory, *SIAM Journal of Computation*, Vol. 26, No. 5, pp. 1411–1473 (1997).

[10] Brennen, G. K., Bullock, S. S. and O'Leary, D. P.: Efficient circuits for exact-universal computation with qudits, *Quantum Inf. Comput.*, Vol. 6, No. 4 & 5, pp. 436–454 (2006).

[11] Buhrman, H., Cleve, R., Watrous, J. and de Wolf, R.: Quantum fingerprinting, *Phys. Rev. Lett.*, Vol. 87, No. 16, p. 167902 (2001).

[12] Bullock, S. S.: Note on the Khaneja Glaser decomposition, *Quantum Inf. Comput.*, Vol. 4, No. 5, pp. 396–400 (2004).

[13] Bullock, S. S., Brennen, G. K. and O'Leary, D. P.: Time reversal and n-qubit canonical decomposition, *J. Math. Phys.*, Vol. 46, p. 062105 (2005).

[14] Bullock, S. S. and Markov, I.: Arbitrary two-qubit computation in 23 elementary gates, *Phys. Rev. A*, Vol. 68, p. 012318 (2003).

[15] Bullock, S. S., O'Leary, D. P. and Brennen, G. K.: Asymptotically optimal quantum circuits for d-level systems, *Phys. Rev. Lett.*, Vol. 94, p. 230502 (2005).

[16] Cleve, R. and Watrous, J.: Fast parallel circuits for the quantum Fourier transform, *Proc. of the 41st annual symposium on Foundations of Computer Science*, Redondo Beach, CA, USA, pp. 526–536 (2000). quant-ph/0006004.

[17] Deutsch, D.: Quantum computational networks, *Proc. Roy. Soc. London Ser. A*, Vol. 425, pp. 73–90 (1989).

[18] Freedman, M. H.: P/NP, and the quantum field computer, *Proc. Natl. Acad. Sci.*, Vol. 95, pp. 98–101 (1998).

[19] Freedman, M. H., Kitaev, A. and Wang, Z.: Simulation of topological field theories by quantum computers, *Comm. Math. Phys*, Vol. 227, No. 3, pp. 587–603 (2002).

[20] Freedman, M. H., Larsen, M. and Wang, Z.: A modular functor which is universal for quantum computation, *Comm. Math. Phys.*, Vol. 227, pp. 605–622 (2002).

[21] Garnerone, S., Marzuoli, A. and Rasetti, M.: Quantum knitting, *Laser Physics*, Vol. 16, No. 11, pp. 1582–1594 (2006). quant-ph/0606137.

[22] Golub, G. H. and VanLoan, C. F.: *Matrix Computations*, Johns Hopkins Studies In The Mathematical Sciences, Baltimore, MD, USA, 3rd edition (1996).

[23] Gottesman, D.: Stabilizer Codes and Quantum Error Correction, PhD Thesis, California Institute of technology, Pasadena, CA (1997).

[24] Helgason, S.: *Differential Geometry, Lie Groups and Symmetric Spaces*, Academic Press (1978).

[25] Høyer, P.: Efficient quantum transforms. quant-ph/9702028.

[26] Jaeger, F., Vertigan, D. L. and Welsh, D. J. A.: On the computational complexity of the Jones and Tutte polynomials, *Math. Proc. Camb. Phil. Soc.*, Vol. 108, pp. 35–53 (1990).

[27] Jones, V. F. R.: Jones Polynomial. http://math.berkeley.edu/˜vfr/.

[28] Jones, V. F. R.: A Polynomial Invariant for Knots via von Neumann Algebras, *Bull. Am. Math. Soc.*, Vol. 12, pp. 103–111 (1985).

[29] Jozsa, R.: An introduction to measurement based quantum computing, *NATO Science Series, III: Computer and Systems Sciences*, Vol. 199, pp. 137–158 (2006). quant-ph/0508124.

[30] Khan, F. S. and Perkowski, M. M.: Synthesis of ternary quantum logic circuits by decomposition. quant-ph/0511041.

[31] Khan, F. S. and Perkowski, M. M.: Synthesis of multi-qudit hybrid and *d*-valued quantum logic circuits by decomposition, *Theoretical Computer Science*, Vol. 367, No. 3, pp. 336–356 (2006).

[32] Khaneja, N., Blockett, R. and Glaser, S.: Time optimal control in spin systems, *Phys. Rev. A*, Vol. 63, p. 032308 (2001).

[33] Khaneja, N. and Glaser, S.: Cartan decomposition of $SU(2^n)$, constructive controllability of spin systems and universal quantum computing, *Chem. Phys.*, Vol. 267, pp. 11–23 (2002).

[34] Knapp, A. W.: *Lie Groups Beyond an Introduction*, Brikhäuser, 2nd edition (2005).

[35] Morton, H. R. and Short, H. B.: Calculating the 2-variable polynomial for knots presented as closed braids, *Journal of Algorithms*, Vol. 11, pp. 117–131 (1990).

[36] Möttönen, M. and Vartiainen, J. J.: Decompositions of general quantum gates. quant-ph/0504100.

[37] Möttönen, M., Vartiainen, J. J., Bergholm, V. and Salomaa, M.: Quantum circuits for general multiqubit gates, *Phys. Rev. Lett.*, Vol. 93, No. 13, p. 130502 (2004).

[38] Möttönen, M., Vartiainen, J. J., Bergholm, V. and Salomaa, M.: Transformation of quantum states using uniformly controlled rotations, *Quantum. Inf. Comput*, Vol. 5, pp. 467–473 (2005).

[39] Nielsen, M. A. and Chuang, I. L.: *Quantum Computation and Quantum Information*, Cambridge University Press, New York, NY, USA, 2nd edition (2000).

[40] Savage, C.: A survey of combinatorial Gray codes, *SIAM Review*, Vol. 39, No. 4, pp. 605–629 (1997).

[41] Sekine, K., Imai, H. and Imai, K.: Computation of Jones polynomial (In Japanese), *Transactions of the JSIAM*, Vol. 8, No. 3, pp. 341–354 (1998).

[42] Shende, V. V., Markov, I. L. and Bullock, S. S.: Synthesis of quantum logic circuits, *IEEE Trans. on Computer-Aided Design*, Vol. 25, No. 6, pp. 1000–1010 (2006).

[43] Shor, P. W.: Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer, *SIMAM J. Comput*, Vol. 26, No. 5, pp. 1484–1509 (1994).

[44] Svore, K. M.: Compiling quantum circuits into elementary unitary operations (2004). QIP2004, Waterloo, Canada.

[45] Tucci, R.: Quantum fast Fourier transform viewed as a special case of recursive application of Cosine-Sine decomposition. quant-ph/0407010.

[46] Tucci, R.: A rudimentary quantum computer. quant-ph/9902062.

[47] Vartiainen, J. J.: Unitary Transformations for Quantum Computing, PhD Thesis, Helsinki University of Technology, Espoo, Finland (2005).

[48] Vedral, V., Barenco, A. and Ekert, A.: Quantum networks for elementary arithmetic operations, *Phys. Rev. A*, Vol. 54, No. 1, pp. 147–153 (1996).

[49] Wocjan, P. and Yard, J.: The Jones polynomial: quantum algorithms and applications in quantum complexity theory (2008). quant-ph/0603069.

[50] Zilic, Z. and Radecka, K.: Scaling and better approximating quantum Fourier transform by higher radices, *IEEE Transactions on Computers*, Vol. 56, No. 2, pp. 202–207 (2007).

List of Publications

Journal Papers

1. Yumi Nakajima, Yasuhito Kawano, and Hiroshi Sekigawa: A new algorithm for producing quantum circuits using KAK decompositions, Quantum Information & Computation, Vol. 6, No. 1, pp. 67–80 (2006). quant-ph/0509196

2. Yumi Nakajima, Yasuhito Kawano, and Hiroshi Sekigawa: Efficient quantum circuits for approximating the Jones polynomial, Quantum Information & Computation, Vol. 8, No. 5, pp. 489–500 (2008).

3. Yumi Nakajima, Yasuhito Kawano, Hiroshi Sekigawa, Masaki Nakanishi, Shigeru Yamashita, and Yasuhiko Nakashima: Synthesis of quantum circuits for d-level systems by using cosine-sine decomposition, Quantum Information & Computation, Vol. 9, No. 5 & 6, pp. 423–443 (2009).

Conferences and Workshops (Refereed)

1. Yumi Nakajima, Yasuhito Kawano, and Hiroshi Sekigawa: An algorithm for generating quantum circuits using the Khaneja-Glaser decomposition, The Eighth Workshop on Quantum Information Processing (QIP2005), Cambridge, MA, USA (2005).

2. Yumi Nakajima, Yasuhito Kawano, and Hiroshi Sekigawa: An algorithm for decomposing unitary matrices using Cartan decomposition, Conference on Applications of Computer Algebra (ACA2005), Nara, Japan (2005).

3. Yumi Nakajima, Yasuhito Kawano, and Hiroshi Sekigawa: A new algorithm for producing quantum circuits using KAK decompositions, The Nineth Workshop on Quantum Information Processing (QIP2006), Paris, France (2006).

4. Yumi Nakajima, Yasuhito Kawano, and Hiroshi Sekigawa: A quantum circuit for approximating the Jones polynomial of the plat closure, The Tenth Workshop on Quantum Information Processing (QIP2007). Brisbane, Australia (2007).

5. Yumi Nakajima, Yasuhito Kawano, and Hiroshi Sekigawa: Synthesis of quantum circuits for d-level systems, Asian Conference on Quantum Information Science (AQIS2007), pp. 135–136, Kyoto, Japan (2007).

6. Yumi Nakajima, Yasuhito Kawano, Hiroshi Sekigawa, Masaki Nakanishi, Shigeru Yamashita, and Yasuhiko Nakashima: Synthesis of quantum circuits for d-level systems using KAK decomposition, The Eleventh Workshop on Quantum Information Processing (QIP2008), NewDelhi, India (2007).

Conferences and Workshops (Not Refereed)

1. Yumi Murakami, Yasuhito Kawano, and Hiroshi Sekigawa: Generating Quantum Circuits using Cartan Decomposition, The Tenth Quantum Information Technology Symposium (QIT10), pp. 123–126 (2004). (In Japanese)

2. Yumi Nakajima, Yasuhito Kawano, and Hiroshi Sekigawa: Design automation for quantum circuit using Cartan decomposition, Workshoop on Quantum Computing 2005 – Algorithms, Physical Realizations and Beyond, Kinki University –, Osaka, Japan (2005).

3. Yumi Nakajima, Yasuhito Kawano, and Hiroshi Sekigawa: A quantum circuit for approximating Jones polynomial, The Fifteenth Quantum Information Technology Symposium (QIT15), pp. 170–173 (2006). (In Japanese)

Review Articles

1. Yumi Nakajima, Yasuhito Kawano, and Hiroshi Sekigawa: Quantum Circuit Design, Information Processing Society of Japan (IPSJ) Magazine, Vol. 47, No. 12, pp. 1335–1340 (Dec. 2006). (In Japanese)