Quantum Fourier Transform over Symmetric Groups

Yasuhito Kawano
NTT Communication Science Laboratories
3-1, Morinosato Wakamiya, Atsugi-shi
Kanagawa, 243-0198, Japan
kawano.yasuhito@lab.ntt.co.jp

Hiroshi Sekigawa*
Department of Math, Tokai University
4-1-1, Kitakaname, Hiratsuka-shi
Kanagawa, 259-1292, Japan
sekigawa@tokai-u.jp

ABSTRACT

This paper proposes an $O(n^4)$ quantum Fourier transform (QFT) algorithm over symmetric group S_n, the fastest QFT algorithm of its kind. We propose a fast Fourier transform algorithm over symmetric group S_n, which consists of $O(n^3)$ multiplications of unitary matrices, and then transform it into a quantum circuit form. The QFT algorithm can be applied to constructing the standard algorithm of the hidden subgroup problem.

Categories and Subject Descriptors

H.4 [**Information Systems Applications**]: Miscellaneous; D.2.8 [**Software Engineering**]: Metrics—*complexity measures, performance measures*

Keywords

Quantum Fourier Transform; Fast Fourier Transform; Symmetric Group; Representation Theory

1. INTRODUCTION

The quantum Fourier transform (QFT) plays an important role in many quantum algorithms exponentially faster than the classical counterparts. Shor's quantum algorithm [14] that efficiently solves the factoring problem applies the QFT over the cyclic groups. Quantum circuits for the QFT over the cyclic groups have been studied in detail, and many efficient QFT circuits over the cyclic groups have been proposed [5, 7, 4, 11].

Shor's algorithm can be naturally generalized to the standard algorithm for the hidden subgroup problem [12, §5.4.3]. An especially interesting application of the hidden subgroup problem is the graph isomorphism problem: Given two graphs, decide whether there exists an isomorphism map from one to the other. It is an open question whether there exists an efficient quantum algorithm that solves the graph isomorphism problem. The standard algorithm for the hidden subgroup problem has been thought to be one of the candidates that can efficiently solve the graph isomorphism problem or its subproblems [12, §5.4.4]. A standard algorithm that solves Simon's problem over non-abelian groups faster than classical algorithms has been found [1]. On the other hand, it is suggested that the graph isomorphism problem would not be solved using the standard algorithm (cf. [10]).

The standard algorithm for the graph isomorphism problem uses the QFT over symmetric groups instead of the QFT over cyclic groups. Efficient quantum circuits that perform the QFT over symmetric groups have thus been studied [2, 9]. An efficient QFT circuit over symmetric groups was first proposed by Beals [2]. Later, Moore, Rockmore, and Russell applied recent progress in the fast Fourier transform (FFT) algorithm, and proposed efficient QFT circuits over non-abelian groups, including symmetric groups [9]. However, their QFT circuits for symmetric groups sum amplitudes over cosets serially, where the sum-of-amplitudes is the most complex calculation in the QFT. On the other hand, since Coppersmith's well-known circuit [5, 12] for the QFT over the cyclic groups sums amplitudes in parallel, the time complexity of Coppersmith's circuit is much lower than the time complexities of circuits proposed in [2, 9]. It is known that Coppersmith's circuit is a quantum counterpart of the FFT algorithm over the cyclic groups.

The purpose of this paper is to propose an algorithm that performs QFT over symmetric groups more efficiently by calculating the sum-of-amplitudes in parallel. For this purpose, we propose an FFT (classical) algorithm over symmetric groups, which consists of a multiplication of sparse unitary matrices, and then transform it to a quantum circuit form. As a byproduct, we obtain an FFT algorithm over symmetric groups. A detailed discussion of applications can be found in [6, page 326].

This paper is constructed as follows. In section 2, we explain background notions and symbols of representation theory for symmetric groups. In section 3, the FFT algorithm is proposed. In section 4, the QFT algorithm is described. Finally, section 5 concludes the paper.

2. REPRESENTATION THEORY

Background notions and symbols of representation theory for symmetric groups are given in this section. More detailed explanations can be found in [15, 16, 8, 13].

*The current address of the author is: Department of Mathematical Information Science, Tokyo University of Science, 1-3 Kagurazaka, Shinjuku-ku, Tokyo, 162-8601, Japan. sekigawa@rs.tus.ac.jp

ISSAC'13, June 26–29, 2013, Boston, Massachusetts, USA.

2.1 Basic Notions

Let n be a positive integer. A permutation $i_1 \mapsto j_1, i_2 \mapsto j_2, \cdots, i_n \mapsto j_n$ over $\{1, 2, \cdots, n\}$ is denoted $\begin{pmatrix} i_1 & \cdots & i_n \\ j_1 & \cdots & j_n \end{pmatrix}$. When $i_k = j_k$ in the above permutation, the column of i_k and j_k is often abbreviated. A *multiplication* of permutations is defined by

$$\begin{pmatrix} j_1 & \cdots & j_n \\ k_1 & \cdots & k_n \end{pmatrix} \cdot \begin{pmatrix} i_1 & \cdots & i_n \\ j_1 & \cdots & j_n \end{pmatrix} = \begin{pmatrix} i_1 & \cdots & i_n \\ k_1 & \cdots & k_n \end{pmatrix}.$$

Define S_n as the group of the set of all permutations over $\{1, 2, \cdots, n\}$ with the multiplication. The S_n is called the *symmetric group* of order n. The number of elements in S_n, denoted $|S_n|$, is $n!$.

A permutation $\begin{pmatrix} i_1 & i_2 & \cdots & i_k \\ i_2 & i_3 & \cdots & i_1 \end{pmatrix}$ is called a *cyclic permutation*. It is denoted $c_{(i_1, i_2, \cdots, i_k)}$ or $i_1 \mapsto i_2 \mapsto \cdots \mapsto i_k \mapsto i_1$ in this paper. When $k = 2$, a permutation is called a *transposition*. An element in S_n can be decomposed into a multiplication of transpositions.

The quotient ring $\mathbb{Z}/n\mathbb{Z}$ is denoted $\mathbb{Z}_n$. For any element g in S_n, there is a unique $(i_1, i_2, \cdots, i_{n-1}) \in \mathbb{Z}_2 \times \mathbb{Z}_3 \times \cdots \times \mathbb{Z}_{n-1}$ such that

$$g = c_{(1,2,\cdots,n)}^{i_{n-1}} \cdots c_{(1,2,3)}^{i_2} c_{(1,2)}^{i_1}.$$

The map $g \mapsto (i_1, i_2, \cdots, i_{n-1})$ is called the *canonical coding* in this paper. The canonical coding will be used for coding an element in S_n on quantum states, i.e., an element g in S_n will be encoded as $|i_1, i_2, \cdots, i_{n-1}\rangle$ using qubit, qutrit, $\cdots$, and qu$n$it.

Let $g \mapsto (i_1, i_2, \cdots, i_{n-1})$ be the canonical coding. Define g_i, where $i = \sum_{j=1}^{n-1} i_j \cdot \frac{n!}{(j+1)!}$, as g. Then, $\{g_i | i = 0, 1, \cdots, n! - 1\}$ is an enumeration of elements in S_n, i.e., $S_n = \{g_0, g_1, \cdots, g_{n!-1}\}$. For example, $g_0 = id$ and $g_{n!/2} = c_{(1,2)}$. The S_3 is enumerated as id, $c_{(1,2,3)}$, $c_{(1,3,2)}$, $c_{(1,2)}$, $c_{(2,3)}$, and $c_{(1,3)}$. This order will be used for the column number of the Fourier transform matrix.

2.2 Irreducible Representations

Let V be a finite dimensional vector space over the field $\mathbb{C}$. Let $U(V)$ be the group defined by the set of unitary transforms from V to V. Given an orthonormal basis of V, each element in $U(V)$ is represented as an $n \times n$ unitary matrix, where n is the dimension of V.

If a function $\rho : S_n \to U(V)$ is homomorphic (i.e., $\rho(g_1 g_2) = \rho(g_1) \cdot \rho(g_2)$ is satisfied for all $g_1, g_2 \in S_n$) for a vector space V, then ρ is called a *representation* and V is called a *representation space*. If a subspace W of a representation space V satisfies $\rho(g)(W) \subseteq W$ for all $g \in S_n$, W is called an invariant subspace. Since V and $\{0\}$ are always invariant, they are called *trivial* invariant subspaces. If there is no non-trivial invariant space, then ρ is called an *irreducible* representation. The set of all irreducible representations of S_n is denoted Λ_n.

2.3 Young Diagrams

A Young diagram is a diagram with left-aligned and top-aligned square boxes. By enumerating the numbers of the boxes in the first, second, $\cdots$, and k-th rows, a Young diagram with n boxes is encoded as an ordered set of n numbers $(\lambda_1, \lambda_2, \cdots, \lambda_k, 0, \cdots, 0)$, where $n = \sum_{i=1}^{k} \lambda_i$ and $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_k > 0$. $(\lambda_1, \lambda_2, \cdots, \lambda_k, 0, \cdots, 0)$ is often written as $(\lambda_1, \lambda_2, \cdots, \lambda_k)$ by omitting zeros. Since a Young diagram of S_n is a non-increasing sequence of numbers such that the sum of them is n, it is sometimes called "a partition of n."

It is known that any irreducible representation for S_n corresponds to a Young diagram with n boxes, which is a partition of n. The set of all Young diagrams with n boxes (= the set of all partitions of n) can then be seen as the set of all irreducible representations of S_n. We will identify irreducible representations of S_n, the Young diagrams with n boxes, and partitions of n hereafter, and denote the set of them as Λ_n. For example, $(2, 1)$ is a partition of 3, a Young diagram with three boxes, and an irreducible representation of S_3. Hence, there is a vector space V such that $(2, 1) : S_3 \to U(V)$ is an irreducible representation.

2.4 Standard Young Tableaus

To calculate the dimension of the representation space, the notion of the standard Young tableau is introduced.

A *Young tableau* with n boxes is a diagram obtained from a Young diagram with n boxes by writing numbers from 1 to n into the boxes of the Young diagram. Here, different boxes must have different numbers. A Young tableau is standard if the number in each box is greater than both the number in the box above and number in the box to the left. It is known that the number of standard Young tableaus is equal to the dimension of the representation space.

The dimension of the representation space for $\lambda \in \Lambda_n$ is denoted by d_λ. Let $\rho_n = \bigoplus_{\lambda \in \Lambda_n} (I_{d_\lambda} \otimes \lambda)$ be the representation defined by $\rho_n(g) = \bigoplus_{\lambda \in \Lambda_n} (I_{d_\lambda} \otimes \lambda(g))$. Each $\rho_n(g)$ is then written as an $n! \times n!$ unitary matrix of $d_\lambda \times d_\lambda$ block matrices with duplication of d_λ for all $\lambda \in \Lambda_n$.

2.5 Bratteli diagram

While the column number of the Fourier transform is determined by the enumeration $\{g_0, g_1, \cdots, g_{n!-1}\}$ of the group elements in S_n, the row number is determined using the Bratteli diagram. The Bratteli diagram (Figure 1) is a directed acyclic graph with a root node such that

1. a Young diagram with n boxes is assigned on a node in the n-th row, and
2. μ, which is a Young diagram with $n+1$ boxes, is a child node of λ, which is a Young diagram with n boxes, if and only if μ is obtained by adding a box to λ.

We introduce a lexicographic order for nodes on each row of the Bratteli diagram, i.e., define that $(\lambda_1, \lambda_2, \cdots, \lambda_k) > (\lambda'_1, \lambda'_2, \cdots, \lambda'_{k'})$ if and only if $\lambda_i > \lambda'_i$ for the smallest i such that $\lambda_i \neq \lambda'_i$. The Bratteli diagram can then be drawn on a plane by drawing λ on the left-hand side of λ' if $\lambda > \lambda'$. Figure 1 shows the Bratteli diagram for $n \leq 6$. As already explained, an irreducible representation for S_n corresponds to a Young diagram with n boxes. Thus, all irreducible representations for S_n are enumerated in the n-th row of the Bratteli diagram. We will identify an irreducible representation of S_n, a partition of n, a Young diagram with n boxes, and a node in the n-th row of the Bratteli diagram hereafter. The symbol $\lambda \in \Lambda_n$, which is an irreducible representation of S_n, is often used to show the node of the Bratteli diagram corresponding to λ. Then, the number of paths from the root node to a node λ is equal to d_λ. Then, the following relation holds: $n! = \sum_{\lambda \in \Lambda_n} d_\lambda^2$.

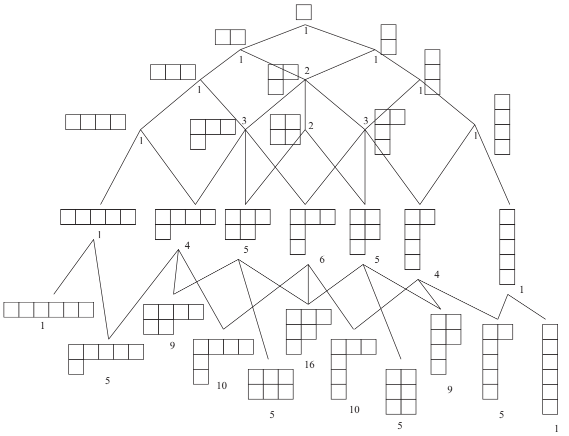

Figure 1: The Bratteli diagram of S_6. Each node of the nth column represents a Young diagram for S_n, which corresponds to an irreducible representation of S_n. The number written at each node shows the dimension of the representation space of the corresponding representation. Note that $n! = \sum_{\lambda \in \Lambda_n} d_\lambda^2$. For example, $4! = 1^2 + 3^2 + 2^2 + 3^2 + 1^2$.

Let $\mathcal{P}(\lambda)$ be the set of paths from the root node to node λ. Then, $|\mathcal{P}(\lambda)| = d_\lambda$. An element $p \in \mathcal{P}(\lambda)$ for $\lambda \in \Lambda_n$ can be encoded by an ordered set of $n-1$ numbers $(p_1, p_2, \cdots, p_{n-1})$ such that a new box is added to the $(p_k + 1)$st row of the Young diagram λ at the kth stage. For example, $\mathcal{P}(2,1,1) = \{(0,1,2),(1,0,2),(1,2,0)\}$. It is easily seen that

$$(p_1, p_2, \cdots, p_{n-1}) \in \mathbb{Z}_2 \times \mathbb{Z}_3 \times \cdots \times \mathbb{Z}_n.$$

Define $<$ by $(p_1, p_2, \cdots, p_{n-1}) < (p'_1, p'_2, \cdots, p'_{n-1})$ if and only if $p_i < p'_i$ for the smallest i such that $p_i \neq p'_i$.

Since $n! = \sum_{\lambda \in \Lambda_n} d_\lambda^2$, $|\{|\lambda, p, q\rangle | \lambda \in \Lambda_n, p, q \in \mathcal{P}(\lambda)\}| = n!$. Define the order by $|\lambda, p, q\rangle < |\lambda', p', q'\rangle \Leftrightarrow \lambda > \lambda'$ or $\lambda = \lambda' \wedge p < p'$ or $\lambda = \lambda' \wedge p = p' \wedge q < q'$. (Notice that the direction of the inequality sign for λ is reversed) For each element in $\{|\lambda, p, q\rangle | \lambda \in \Lambda_n, p, q \in \mathcal{P}(\lambda)\}$, a number less than $n!$ is given according to the above order. It will be used as the row number of the Fourier transform.

2.6 Adapted Gel'fand-Tsetlin Bases

The $\lambda(g)$ ($g \in S_n$, $\lambda \in \Lambda_n$) can be represented by a $d_\lambda \times d_\lambda$ matrix, but it depends on the selection of an orthonormal basis of the representation space. *Adapted Gel'fand-Tsetlin bases* are very useful for making an efficient Fourier transform algorithm on S_n ($n = 2, 3, 4, \cdots$). Here, adapted Gel'fand-Tsetlin bases are series of orthonormal bases $\{v_{\lambda,p,q} | \lambda \in \Lambda_n, p, q \in \mathcal{P}(\lambda)\}_n$ on the representation spaces of irreducible representations for S_n ($n = 2, 3, 4, \cdots$) that satisfies the following conditions.

Let λ be an irreducible representation of S_n. Then, λ corresponds to a Young diagram. (Hence, it is a node in the n-th row of the Bratteli diagram.) Let $\{\mu | \mu \searrow \lambda\}$ be irreducible representations which are parents of λ in the Bratteli diagram. From the properties of the Bratteli diagram, $d_\lambda = \sum_{\mu \searrow \lambda} d_\mu$.

For $\lambda \in \Lambda_n$, $\lambda(g)$ ($g \in S_n$) is represented by a $d_\lambda \times d_\lambda$ matrix when we select an orthonormal basis of the representation space of λ. Denote this matrix as $\lambda(g)_{\mathcal{B}_\lambda}$, where $\mathcal{B}_\lambda$ is the basis. On the other hand, since μ's ($\mu \searrow \lambda$) are irreducible representations for S_{n-1}, any $g \in S_{n-1}$ is represented by the direct sum of $d_\mu \times d_\mu$ matrices when we select orthonormal bases of the representation spaces of μ's ($\mu \searrow \lambda$). Denote this matrix as $\oplus_{\mu \searrow \lambda} \mu(g)_{\mathcal{B}_\mu}$, where $\mathcal{B}_\mu$ ($\mu \searrow \lambda$) are the bases. Generally, $\lambda(g)_{\mathcal{B}_\lambda}$ and $\oplus_{\mu \searrow \lambda} \mu(g)_{\mathcal{B}_\mu}$ are different because they depend on the selections of the bases. However, by selecting good bases,

$$\lambda(g)_{\mathcal{B}_\lambda} = \bigoplus_{\mu \searrow \lambda} \mu(g)_{\mathcal{B}_\mu} \qquad (1)$$

holds for all $g \in S_{n-1}$. If (1) holds for all irreducible representations λ, the set of bases is called the adapted Gel'fand-Tsetlin bases. Elements in S_{n-1} are represented by block diagonal matrices in the $d_\lambda \times d_\lambda$ matrix that represents elements of S_n if we use the adapted Gel'fand-Tsetlin bases.

The adapted Gel'fand-Tsetlin bases are not unique. In this paper, we select a set of adapted Gel'fand-Tsetlin bases and use it consistently. Therefore, $\lambda(g)_{\mathcal{B}_\lambda}$ will be simply denoted $\lambda(g)$ if there's no confusion.

2.7 Specht Polynomial

Vandermonde determinant $\Delta(x_1, \ldots, x_n)$ is defined as

$$\Delta(x_1, \ldots, x_n) = \begin{pmatrix} x_1^{n-1} & x_2^{n-1} & \ldots & x_n^{n-1} \\ x_1^{n-2} & x_2^{n-2} & \ldots & x_n^{n-2} \\ \vdots & \vdots & \ddots & \vdots \\ x_1 & x_2 & \ldots & x_n \\ 1 & 1 & \ldots & 1 \end{pmatrix}.$$

Let $\mathcal{T}$ be an m-row 1-column Young tableau. Specht polynomial $\Delta(\mathcal{T})$ is defined by

$$\Delta(\mathcal{T}) = \Delta(x_1, \ldots, x_m).$$

For any Young tableau $\mathcal{T}$ with i columns, Specht polynomial $\Delta(\mathcal{T})$ is defined by

$$\Delta(\mathcal{T}) = \Delta(\mathcal{T}_1) \ldots \Delta(\mathcal{T}_i),$$

where $\mathcal{T}_j$ is the j-th column of $\mathcal{T}$. For example, $\mathcal{T}$ is a Young tableau such that it has three columns, the numbers of the first column are $1, 4, 5$, the second are $2, 3$, and the third are $6, 7$. Then,

$$\Delta(\mathcal{T}) = \Delta(x_1, x_4, x_5)\Delta(x_2, x_3)\Delta(x_6, x_7).$$

Let g be an element of S_n and $\mathcal{T}$ be a Young tableau. $g(\Delta(\mathcal{T}))$ is defined as $\Delta(g(\mathcal{T}))$, where $g(\mathcal{T})$ is the Young tableau obtained by replacing x_i by $x_{g(i)}$ for all $i = 1, 2, \cdots, n$. Let V be the complex-number coefficient vector space on $\{g(\Delta(\mathcal{T})) | g \in S_n\}$. Generally, $g(\Delta(\mathcal{T}))$'s ($g \in S_n$) are not linearly independent. It is known that Specht polynomials of standard Young tableaus are linearly independent. Since any $g \in S_n$ that maps $\Delta(\mathcal{T})$ to $g(\Delta(\mathcal{T}))$ defines a linear operation from V to V, a map from S_n to $U(V)$ is defined. This map is homomorphic, so it is a representation of S_n. In addition, the map is irreducible, and any irreducible representations of S_n can be enumerated by this method. However, the set of Specht polynomials for standard Young tableaus is not a set of adapted Gel'fand-Tsetlin bases. A method for making adapted Gel'fand-Tsetlin bases is proposed in what follows.

2.8 Construction of Adapted Gel'fand-Tsetlin Bases

In this subsection, we will give an algorithm that calculates unitary matrices that represent $\{\lambda(g) | \lambda \in \Lambda_n, g \in S_n\}$. The representation is not unique, but to make efficient QFT circuits later, those matrices should be chosen under the selection of special bases.

The following order between standard Young tableaus with n boxes is introduced.

Definition 1. *For Young tableaus $\mathcal{T}_1$ and $\mathcal{T}_2$, $\mathcal{T}_1 < \mathcal{T}_2$ is defined as follows:*

> *There exists i $(1 \leq i \leq n)$ such that (1) for all j $(i < j \leq n)$ j belongs to the same columns of $\mathcal{T}_1$ and $\mathcal{T}_2$ and (2) the column that contains i in $\mathcal{T}_1$ is on the left-hand side of the column that contains i in $\mathcal{T}_2$.*

Then, "<" is the total order on the set of standard Young tableaus with n boxes.

Next, we introduce the Hermitian product on the vector space V defined from Specht polynomials that correspond to standard Young tableaus with n boxes. Since V is a subspace of the space W of linear combinations of monomials of $x_1, \ldots, x_n$, we can write f, $g \in W$ as

$$f = \sum_\alpha a_\alpha x^\alpha, \qquad g = \sum_\alpha b_\alpha x^\alpha,$$

where α is a multi-index. The Hermitian product on V is defined as the restriction of the Hermitian product

$$(f, g) = \sum_\alpha a_\alpha \overline{b_\alpha}$$

on W.

Adapted Gel'fand-Tsetlin bases on S_n are calculated as follows.

Algorithm 1 (Adapted Gel'fand-Tsetlin bases).

1. *Enumerate all Young diagrams with n boxes. The order is left-to-right of the n-th row of the Bratteli diagram.*
2. *Perform the following operations for each Young diagram according to the order determined in 1.*
   - *(a) Enumerate all standard Young tableaus in the order of $<$ of Definition 1. (Start from the smallest order and end at the largest).*
   - *(b) From the end of the order of standard Young tableaus, orthonormalize Specht polynomials one by one. For example, on the scheme of the Gram-Schmidt orthonomalization, orthogonalize all Specht polynomials first, and then normalize the obtained polynomials at the end. (In this way, orthogonalizing can be performed by only four arithmetic operations.)*

The polynomials obtained by the above algorithm are adapted Gel'fand-Tsetlin bases. This comes from the following facts.

- Specht polynomials of standard Young tableaus for different Young diagrams are orthogonal.
- For a Young diagram λ with n boxes, divide the set of all standard Young tableaus obtained from λ into $\mathcal{A}_1 \cup \mathcal{A}_2 \cup \cdots \cup \mathcal{A}_q$ $(\mathcal{A}_i \neq \emptyset)$ such that $m < m'$ if and only if $\mu < \mu'$, where m and m' are the numbers of the columns that contain the boxes into which n is written for $\mathcal{T} \in \mathcal{A}_\mu$ and $\mathcal{T}' \in \mathcal{A}_{\mu'}$, respectively. Let V_i $(1 \leq i \leq q)$ be the subspace generated by the Specht polynomials $\Delta(\mathcal{T})$ $(\mathcal{T} \in \mathcal{A}_q \cup \mathcal{A}_{q-1} \cup \cdots \cup \mathcal{A}_{q-i+1})$. Then,

  $$V_1 \subset V_2 \subset \cdots \subset V_q$$

  is a sequence of invariant subspaces of S_{n-1}.

  In addition, inside V_{i+1}, V_i and $V_i^\perp$ (orthogonal complement of V_i) are invariant subspaces of S_{n-1}.
- An irreducible representation of S_n that corresponds to a Young diagram λ with n boxes is the direct sum of irreducible representations of Young diagrams obtained by subtracting a box from λ.

2.9 Fourier Transform

Let $f : S_n \to \mathbb{C}$ be a function. The Fourier transform $\hat{f}$ of f is defined as

$$\hat{f}(\lambda) = \sqrt{\frac{d_\lambda}{|S_n|}} \sum_{g \in S_n} f(g)\lambda(g)$$

for each $\lambda \in \Lambda_n$ (cf. [12, page 615]). The Fourier transform can be expressed as a matrix form

$$\mathfrak{F}_n = \sum_{\lambda \in \Lambda_n} \sum_{p,q \in \mathcal{P}(\lambda)} \sum_{g \in S_n} \sqrt{\frac{d_\lambda}{|S_n|}} [\lambda(g)]_{q,p} |\lambda, p, q\rangle\langle g|. \qquad (2)$$

Here, $[\lambda(g)]_{q,p}$ is the (q, p) element of matrix representation of $\lambda(g)$. Notice that the index is not (p, q) but (q, p). Thanks to this definition, the algorithm proposed later becomes a little easier than that for the $\mathfrak{F}_n$ with $[\lambda(g)]_{p,q}$ as the coefficient.

The following are the concrete matrices of $\mathfrak{F}_2$ and $\mathfrak{F}_3$ calculated from the adapted Gel'fand-Tsetlin bases of the Specht polynomial defined above.

$$\mathfrak{F}_2 = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$$

$$\mathfrak{F}_3 = \frac{1}{\sqrt{6}} \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 \\ \sqrt{2} & -\frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} & \sqrt{2} & -\frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \\ 0 & \frac{\sqrt{3}}{\sqrt{2}} & -\frac{\sqrt{3}}{\sqrt{2}} & 0 & \frac{\sqrt{3}}{\sqrt{2}} & -\frac{\sqrt{3}}{\sqrt{2}} \\ 0 & -\frac{\sqrt{3}}{\sqrt{2}} & \frac{\sqrt{3}}{\sqrt{2}} & 0 & \frac{\sqrt{3}}{\sqrt{2}} & -\frac{\sqrt{3}}{\sqrt{2}} \\ \sqrt{2} & -\frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} & -\sqrt{2} & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ 1 & 1 & 1 & -1 & -1 & -1 \end{pmatrix}$$

It is easily seen that $\mathfrak{F}_2$ is the Hadamard matrix.

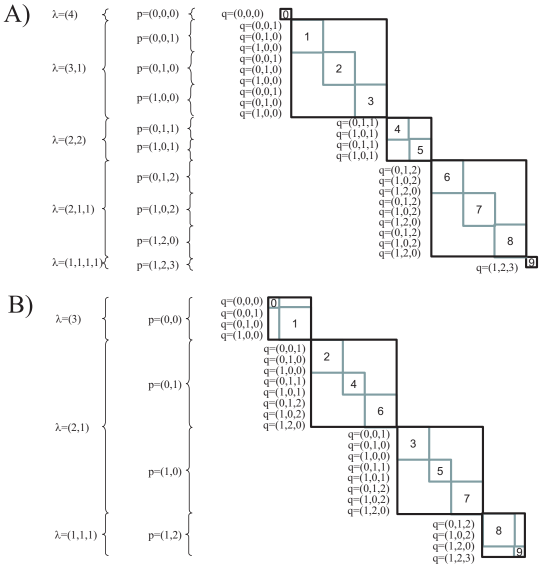

Figure 2: A) The form of $\rho_4(g)$, where $\rho_4 = \bigoplus_{\lambda \in \Lambda_4}(I_{d_\lambda} \otimes \lambda)$ and $g \in S_4$. It consists of one 1×1, three 3×3, two 2×2, three 3×3, and one 1×1 block diagonal matrices. The numbers 0-9 written in the submatrices are matrix numbers. Matrices 1, 2, and 3 are the same. Similarly, matrices 4 and 5 are the same, and matrices 6, 7, and 8 are the same. B) The form of $T_4^\dagger \rho_4(g) T_4$, which consists of a replacement of submatrices in panel A. The same matrix numbers in A and B mean the matrices are the same.

3. FFT ALGORITHM

3.1 Basis Transform T_n

A matrix for a basis transform, denoted T_n, is introduced before decomposing $\mathfrak{F}_n$ to simpler matrices. The T_n is an $n! \times n!$ matrix for a basis transform from the basis

$$\mathcal{B}_n = \{|\lambda, p, q\rangle | \lambda \in \Lambda_{n-1}, p \in \mathcal{P}(\lambda), \exists \mu(\lambda \searrow \mu \wedge q \in \mathcal{P}(\mu))\} \quad (3)$$

to the basis

$$\mathcal{B}'_n = \{|\lambda', p', q'\rangle | \lambda' \in \Lambda_n, p', q' \in \mathcal{P}(\lambda')\}.$$

It is defined as $T_n^\dagger |\lambda', p', q'\rangle = |\lambda, p, q\rangle$, where $q = q'$, $p = p'|_{n-2}$, and $p \in \mathcal{P}(\lambda)$. Here, $p'|_i$ is the restriction to the first i elements, i.e., $p'|_{n-2} = (p_1, p_2, \cdots, p_{n-2})$ when $p' = (p_1, p_2, \cdots, p_{n-1})$. It is easy to show that $T_n^\dagger$ is a one-to-one onto map from $\mathcal{B}'_n$ to $\mathcal{B}_n$. Hence, T_n is an $n! \times n!$ matrix with elements; zero or one.

Since Λ_n is the set of all irreducible representations of S_n, an element $g \in S_n$ can be written as $\bigoplus_{\lambda \in \Lambda_n} (I_{d_\lambda} \otimes \lambda(g))$. By the definition of T_n,

$$T_n^\dagger \left(\bigoplus_{\lambda \in \Lambda_n} (I_{d_\lambda} \otimes \lambda(g)) \right) T_n = \bigoplus_{\mu \in \Lambda_{n-1}} \left(I_{d_\mu} \otimes (\oplus_{\mu \searrow \lambda} \lambda(g)) \right). \quad (4)$$

An example for $n = 4$ is given in Figure 2.

3.2 Inductive Decomposition of $\mathfrak{F}_n$

The H_n is defined by

$$H_n = T_n^\dagger \mathfrak{F}_n (\mathfrak{F}_{n-1} \otimes I_n)^\dagger. \quad (5)$$

Then, since $\mathfrak{F}_n = T_n H_n (\mathfrak{F}_{n-1} \otimes I_n)$,

$$\begin{aligned} \mathfrak{F}_n &= T_n H_n (\mathfrak{F}_{n-1} \otimes I_n) \\ &= T_n H_n (T_{n-1} H_{n-1} (\mathfrak{F}_{n-2} \otimes I_{n-1}) \otimes I_n) \\ &= \cdots . \end{aligned}$$

Hence, $\mathfrak{F}_n$ can be calculated from T_m, H_m ($m \leq n$). This is the existing FFT algorithm [3, 6].

It is proved that H_n is an $n! \times n!$ matrix consisting of $nd_\lambda \times nd_\lambda$ matrices with duplication of d_λ for $\lambda \in \Lambda_{n-1}$ on the basis $\mathcal{B}_n$. For example, for $n = 4$, the form of H_4 is the area inside the thick frame of Figure 2, panel B.

The sizes of submatrices of H_n, which are $nd_\lambda \times nd_\lambda$ for $\lambda \in \Lambda_n$, are smaller than the size of $\mathfrak{F}_n$, which is $n! \times n!$; however, nd_λ grows exponentially when n gets large. We will decompose H_n into a multiplication of $O(n^2)$ sparse matrices.

3.3 Inducing H'_{n-1} from H_{n-1}

We will introduce $n! \times n!$ matrices K_n, P_n, and A_n such that H_n is calculated by a multiplication of $\{K_n, P_n, A_n, T_n, H_{n-1}\}$. However, since H_n is an $n! \times n!$ matrix, the matrix sizes of H_n and H_{n-1} are different. To fix the problem, an $n! \times n!$ matrix H'_{n-1} is induced from H_{n-1}.

By (5), the input and output of H_{n-1} are defined on the bases $\mathcal{B}'_{n-2} \times \mathbb{Z}_{n-1}$ and $\mathcal{B}_{n-1}$, respectively. Since $\mathcal{B}_{n-1} \cong \mathcal{B}'_{n-2} \times \mathbb{Z}_{n-1}$, which is defined by the order of encoding, we can consider both the input and output of H_{n-1} to be defined on the basis $\mathcal{B}'_{n-2} \times \mathbb{Z}_{n-1}$, i.e., $H_{n-1} = \bigoplus_{\lambda \in \Lambda_{n-2}} (I_{d_\lambda} \otimes H_{n-1,\lambda})$, where $H_{n-1,\lambda}$ is an $(n-1)d_\lambda \times (n-1)d_\lambda$ matrix defined on the basis $\{|\lambda, p, q, i\rangle | q \in \mathcal{P}(\lambda), i < n-1\}$ for some $p \in \mathcal{P}(\lambda)$. Hence, there is a set of numbers $\{h_{\lambda,q,r,i,j} | q, r \in \mathcal{P}(\lambda), i, j < n-1\}$ such that

$$H_{n-1,\lambda} = \sum_{q,r \in \mathcal{P}(\lambda)} \sum_{i,j < n-1} h_{\lambda,q,r,i,j} |\lambda, p, q, i\rangle\langle\lambda, p, r, j|.$$

Define $H'_{n-1,\lambda}$ as

$$\sum_{q \in \mathcal{P}(\lambda)} |\lambda, p, q, 0\rangle\langle\lambda, p, q, 0| \\ + \sum_{q,r \in \mathcal{P}(\lambda)} \sum_{i,j < n-1} h_{\lambda,q,r,i,j} |\lambda, p, q, i+1\rangle\langle\lambda, p, r, j+1|.$$

The $H'_{n-1,\lambda}$ is an $nd_\lambda \times nd_\lambda$ matrix, where $\lambda \in \Lambda_{n-2}$.

The H'_{n-1} is then defined as

$$\bigoplus_{\mu \in \Lambda_{n-1}} \left(I_{d_\mu} \otimes \left(\oplus_{\lambda \searrow \mu} H'_{n-1,\lambda} \right) \right).$$

3.4 Basis Transform P_n

The matrix sizes of H_n and H'_{n-1} are the same; however, the output bases are different. i.e., the output bases of H_n and H'_{n-1} are defined on $\mathcal{B}_n$ and $\mathcal{B}'_{n-1} \times \mathbb{Z}_n$, respectively. To fix the difference, we introduce a basis transform P_n that changes the basis $\mathcal{B}'_{n-1} \times \mathbb{Z}_n$ to the basis $\mathcal{B}_n$.

The basis transform P_n is defined as a multiplication of two matrices $P_{n,1}$ and $P_{n,2}$, i.e., $P_n = P_{n,2} P_{n,1}$.

$P_{n,1}$ is $\bigoplus_{\lambda \in \Lambda_{n-1}} (I_{d_\lambda} \otimes P_{n,1,\lambda})$, where $P_{n,1,\lambda} = \sum_{x,y=0}^{nd_\lambda - 1} a_{x,y} |x\rangle\langle y|$ is the $nd_\lambda \times nd_\lambda$ matrix defined by

$$a_{x,y} = \begin{cases} 1 & \text{if } x < d_\lambda \wedge y = nx \\ 1 & \text{if } x \geq d_\lambda \wedge y = x - \lfloor \frac{nd_\lambda - x - 1}{n-1} \rfloor \\ 0 & \text{o.w.} \end{cases}.$$

$P_{n,2}$ is $\bigoplus_{\lambda \in \Lambda_{n-1}} (I_{d_\lambda} \otimes P_{n,2,\lambda})$ and $P_{n,2,\lambda}$ is determined as follows. Fix $\lambda \in \Lambda_{n-1}$ in the following argument. Enumerate the elements of $\{\mu \in \Lambda_n | \lambda \searrow \mu\}$ in the decreasing lexicographic order, i.e., from left to right in Figure 1. We denote the sequence as $\lambda_\downarrow$. For each element $\mu \in \lambda_\downarrow$, enumerate $\{\lambda' \in \Lambda_{n-1} | \lambda' \searrow \mu\}$ in the decreasing lexicographic order and denote it $\mu_\uparrow$. Then, we define $\lambda_{\downarrow\uparrow}$ as the sequence with repetition by concatenating $\mu_\uparrow$ for $\mu \in \lambda_\downarrow$. Similarly, we define $\lambda_{\uparrow\downarrow}$ as the sequence with repetition by concatenating $\nu_\downarrow$ for $\nu \in \lambda_\uparrow$. When we compare $\lambda_{\downarrow\uparrow}$ and $\lambda_{\uparrow\downarrow}$ as sets with duplication, it is easily proved that $\lambda_{\downarrow\uparrow}$ is $\lambda_{\uparrow\downarrow}$ plus λ. Hence, there are one-to-one maps from $\{\lambda_{\downarrow\uparrow}\}$ to $\{\lambda, \lambda_{\uparrow\downarrow}\}$. Select one of them and denote it $\mathfrak{f}_\lambda$, i.e., $\mathfrak{f}_\lambda(i) = j$ means that the ith element of $\{\lambda_{\downarrow\uparrow}\}$ is equal to the jth element of $\{\lambda, \lambda_{\uparrow\downarrow}\}$ for all $i = 0, 1, \cdots, |\lambda_{\downarrow\uparrow}| - 1$. Then, define $P_{n,2,\lambda} = (b_{i,j})_{i,j=0}^{|\lambda_{\downarrow\uparrow}|-1}$ by

$$b_{i,j} = \begin{cases} I_{d_{\lambda_i}} & \text{if } \mathfrak{f}_\lambda(i) = j \\ 0 & \text{o.w.} \end{cases},$$

where $b_{i,j}$ is a $d_{\lambda_i} \times d_{\lambda_j}$ matrix such that λ_i is the ith element of $\lambda_{\downarrow\uparrow}$ and λ_j is the jth element of $\lambda_{\uparrow\downarrow}$.

P_n depends on the selection of $\mathfrak{f}_\lambda$; however, the key lemma (Lemma 1) holds for any selection of $\mathfrak{f}_\lambda$.

3.5 Controlled Cyclic Permutation K_n

The H_n and $P_n H'_{n-1}$ then have the same input and output bases. We can prove that for any $g \in S_{n-1}$, both $H_n^\dagger \cdot T_n^\dagger \rho_n(g) T_n \cdot H_n$ and $(P_n H'_{n-1})^\dagger \cdot T_n^\dagger \rho_n(g) T_n \cdot (P_n H'_{n-1})$ have non-zero elements at almost the same positions of the matrices. However, the values are different. To fix the difference, we introduce the following K_n defined as

$$K_n = \sum_{i=0}^{n-1} \left(\rho_{n-1}(c_{(i,i+1,\cdots,n-1)}) \otimes |i\rangle\langle i| \right). \tag{6}$$

Here, $c_{(i,i+1,\cdots,n-1)}$ is the cyclic permutation $i \mapsto i+1 \mapsto \cdots \mapsto n-1 \mapsto i$ when $0 < i < n-1$ and is the identity if $i = 0$ or $n-1$. Then, $H_n^\dagger \cdot T_n^\dagger \rho_n(g) T_n \cdot H_n$ and $(P_n H'_{n-1} K_n)^\dagger \cdot T_n^\dagger \rho_n(g) T_n \cdot (P_n H'_{n-1} K_n)$ are the same for any $g \in S_{n-1}$.

3.6 Residue A_n

Finally, A_n is defined as $H_n K_n^\dagger H'^\dagger_{n-1} P_n^\dagger$. Obviously, $H_n = A_n P_n H'_{n-1} K_n$. The following is the key lemma.

Lemma 1. *For any $g \in S_{n-1}$, $T_n^\dagger \rho_n(g) T_n$ and A_n are commutative.*

Roughly speaking, Lemma 1 means that A_n is close to the identity matrix. The following theorem shows that the $n!$-dimensional Hilbert space can be separated into many small-dimensional invariant subspaces of A_n. Let e_λ be the number of children of λ in the Bratteli diagram. Obviously, $e_\lambda \leq n$ for $n \geq 2$.

Theorem 1. *For $\lambda \in \Lambda_{n-1}$ and $p, q \in \mathcal{P}(\lambda)$, let $W_{\lambda,p,q}$ be the e_λ-dimensional subspace spanned by $\{|\lambda, p, (q, x)\rangle \in \mathcal{B}_n | \lambda_p = \lambda_q\}$, where λ_p is the last node of p and (q, x) is $(q_1, \cdots, q_{n-2}, x)$ when $q = (q_1, \cdots, q_{n-2})$. Then, $A_n W_{\lambda,p,q} \subseteq W_{\lambda,p,q}$ for any $\lambda \in \Lambda_{n-1}$ and $p, q \in \mathcal{P}(\lambda)$. Furthermore, for each $\lambda \in \Lambda_{n-1}$, there exists an $e_\lambda \times e_\lambda$ matrix $A'_{n,\lambda}$, independently of p and q, such that $A'_{n,\lambda} = A_n$ on $W_{\lambda,p,q}$. In addition, A_n performs as the identity on the subspace $\bigcap\{W^\perp_{\lambda,p,q} | \lambda \in \Lambda_{n-1}, p, q \in \mathcal{P}(\lambda)\}$.*

The A_n was defined as $H_n K_n^\dagger H'^\dagger_{n-1} P_n^\dagger$; however, for calculating A_n, it is not necessary to multiply H_n, $K_n^\dagger$, $H'^\dagger_{n-1}$, and $P_n^\dagger$. By Theorem 1, A_n can be written as a direct sum of I_1's (the identity operator on a one-dimensional Hilbert space) and $e_\lambda \times e_\lambda$ matrices $A'_{n,\lambda}$ with duplication of d_λ^2 for all $\lambda \in \Lambda_{n-1}$. Theorem 2 shows that $A'_{n,\lambda}$ can be calculated easily from the diagonal elements of $T_n^\dagger \rho_n(c_{(n-1,n)}) T_n$ and $\{d_\lambda\}_\lambda$. Note that $T_n^\dagger \rho_n(c_{(n-1,n)}) T_n$ is
$\bigoplus_{\lambda \in \Lambda_{n-1}} \left(I_{d_\lambda} \otimes \left(\oplus_{\lambda \searrow \mu} \mu(c_{(n-1,n)}) \right) \right)$ and $\oplus_{\lambda \searrow \mu} \mu(c_{(n-1,n)})$ is

$$\sum_{\lambda \searrow \mu} \sum_{q,r \in \mathcal{P}(\mu)} [\mu(c_{(n-1,n)})]_{q,r} |\lambda, p, q\rangle\langle\lambda, p, r|$$

for some $p \in \mathcal{P}(\lambda)$.

Theorem 2. *For (μ, ν) such that $\nu \searrow \lambda \searrow \mu$, define $q_{\mu,\nu} \in \mathcal{P}(\mu)$ as follows: $q_{\mu,\nu}|_{n-3} \in \mathcal{P}(\nu)$ and $q_{\mu,\nu}|_{n-2} \in \mathcal{P}(\lambda)$. Then, $A'_{n,\lambda}$ is equal to*

$$\sum_{\lambda \searrow \mu} \sum_{\nu \searrow \lambda} \sqrt{\frac{(n-1) d_\mu d_\nu}{n d_\lambda^2}} [\mu(c_{(n-1,n)})]_{q_{\mu,\nu}, q_{\mu,\nu}} |\lambda, \mu\rangle\langle\lambda, \nu|$$
$$+ \sum_{\lambda \searrow \mu} \sqrt{\frac{d_\mu}{n d_\lambda}} |\lambda, \mu\rangle\langle\lambda, \lambda|.$$

3.7 Algorithm and Complexity

We have the following relations.

(a) $H_2 = \mathfrak{F}_2 = H$ (the Hadamard matrix)

(b) $H_n = A_n P_n H'_{n-1} K_n$ for $n \geq 3$

(c) $\mathfrak{F}_n = T_n H_n (\mathfrak{F}_{n-1} \otimes I_n)$ for $n \geq 3$

$\mathfrak{F}_n$ for $n \geq 2$ can then be expressed as a multiplication of induced matrices from $\{A_m, P_m, K_m | m \leq n\}$ and H as follows. By (b), $H_n = A_n P_n H'_{n-1} K_n = A_n P_n (A_{n-1} P_{n-1} H'_{n-2} K_{n-1})' K_n = \cdots = A_n P_n A^{(1)}_{n-1} P^{(1)}_{n-1} \cdots H^{(n-2)} \cdots K^{(1)}_{n-1} K_n$, where, e.g., $H^{(i)} = H''^{\cdots}$ (i times $'$). Substitute this H_n into (c), then H_n can be eliminated from (c). Since the obtained relation calculates $\mathfrak{F}_n$ from $\mathfrak{F}_{n-1}$, $\mathfrak{F}_n$ can be calculated inductively.

Suppose we are given $f : S_n \to \mathbb{C}$. Let $|f\rangle$ be the $n!$-dimensional vector corresponding to f. The Fourier transform $\mathfrak{F}_n |f\rangle$ is calculated by applying the matrices in the above matrix decomposition of $\mathfrak{F}_n$ to $|f\rangle$ one by one.

We evaluate the complexity of the algorithm. The complexity is counted as the number of multiplications and additions, which is the same rule as [6]. The total complexity will be shown as $O(n!n^3)$. The key is that all $A_m^{(n-m)}$'s, $P_m^{(n-m)}$'s, $K_m^{(n-m)}$'s, and $H^{(n-2)}$ are sparse matrices.

It suffices to show that the complexity for calculating $H_n|f\rangle$ from $|f\rangle$ is $O(n!n^2)$ because T_n is just an order change of elements of the vector, which can be performed by copying $n!$ values.

Since $c_{(i,i+1,\cdots,n-1)} = c_{(n-2,n-1)} \cdots c_{(i+1,i+2)} c_{(i,i+1)}$, K_n is calculated by a multiplication of $n-1$ matrices of adjacent transpositions. Each adjacent transposition is expressed as a direct sum of 1×1 and 2×2 matrices in Young's orthogonal representation. Hence, the complexity for calculating $K_n|f\rangle$ from $|f\rangle$ is $O(n!n)$. Similarly, the complexity for calculating $K_m^{(n-m)}|f\rangle$ ($m \leq n$) from $|f\rangle$ is $O((n-1)!m^2)$. $H^{(n-2)}$ can

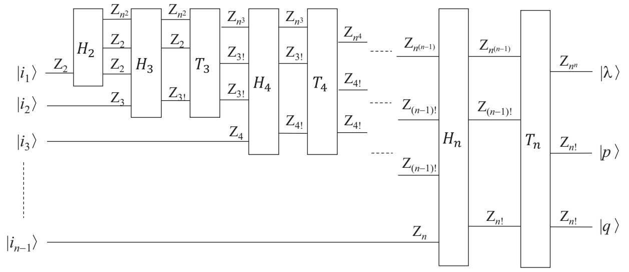

Figure 3: Circuit for $\mathfrak{F}_n$, where H_2, T_m, and H_m are given in Figures 4, 5, and 6, respectively. The input is $|i_1, \cdots, i_{n-1}\rangle$ such that $g = c_{(1,2,\cdots,n)}^{i_{n-1}} \cdots c_{(1,2,3)}^{i_2} c_{(1,2)}^{i_1}$ for $g \in S_n$. The output is $|\lambda, p, q\rangle$ such that $\lambda \in \mathbb{Z}_{n^n}$ and $p, q \in \mathbb{Z}_{n!}$, which are codes of $\lambda \in \Lambda_n$ and $p, q \in \mathcal{P}(\lambda)$, respectively.

be calculated with $O((n-1)!)$ operations. $P_m^{(n-m)}$ is an order change of elements of the vector, similar to $T_m^{(n-m)}$. Since $A_m^{(n-m)}$ is a direct sum of 1×1 and $e_\lambda \times e_\lambda$ matrices, where $\lambda \in \Lambda_m$ and $e_\lambda \leq m$, the complexity is $O(n!m)$. The complexity for calculating $H_n|f\rangle$ from $|f\rangle$ is then $O(n!n^2)$.

4. QFT ALGORITHM

The quantum counterpart of $\mathfrak{F}_n$ is called the quantum Fourier transform (QFT), which is defined the same as $\mathfrak{F}_n$ [See (2)]. We will use the same symbol $\mathfrak{F}_n$ since there's no confusion.

We propose a quantum circuit that performs $\mathfrak{F}_n$, shown in Figure 3. The circuit is $O(n^4)$-depth on $O(n \log n)$ qubits, where gates represented as a direct sum of 1×1 and 2×2 matrices whose elements are easily calculated in a classical computer are defined as elementary gates. Basic notions and symbols of quantum circuits are found in [12]. The words "qu$m$it" and "qu$n$it" will be used in the meaning of m-state and n-state quantum resources, respectively.

4.1 Algorithm

By the induction relation (c) in Subsection 3.7, $\mathfrak{F}_n$ can be calculated by $\mathfrak{F}_2$ $(= H_2)$, H_3, T_3, H_4, T_4, $\cdots$, H_n, and T_n. This can be depicted as the circuit in Figure 3.

The input of the $\mathfrak{F}_n$ circuit is $(i_1, i_2, \cdots, i_{n-1}) \in \mathbb{Z}_2 \times \mathbb{Z}_3 \times \cdots \times \mathbb{Z}_n$ such that $g = c_{(1,2,\cdots,n)}^{i_{n-1}} \cdots c_{(1,2,3)}^{i_2} c_{(1,2)}^{i_1}$. The output is (λ, p, q) such that λ is a partition of n, where p and q are paths from the root node to the node λ in the Bratteli diagram. A partition of n is encoded using n numbers less than n. Hence, λ is encoded as $(\lambda_1, \lambda_2, \cdots, \lambda_n) \in \mathbb{Z}_n \times \mathbb{Z}_n \times \cdots \times \mathbb{Z}_n$. In addition, p is encoded as $(p_1, p_2, \cdots, p_{n-1}) \in \mathbb{Z}_2 \times \mathbb{Z}_3 \times \cdots \times \mathbb{Z}_n$ and q is encoded as $(q_1, q_2, \cdots, q_{n-1}) \in \mathbb{Z}_2 \times \mathbb{Z}_3 \times \cdots \times \mathbb{Z}_n$. Therefore, $(\lambda, p, q) \in \mathbb{Z}_{n^n} \times \mathbb{Z}_{n!} \times \mathbb{Z}_{n!}$.

To compensate for the difference between the input and output, fresh quantum resources are added in the circuits for H_2, T_3, T_4, $\cdots$, T_n. The circuits for H_2, H_m, and T_m $(m \geq 3)$ are given below. The ordered set, e.g., (λ, p, q) will be denoted using the ket symbol $|\lambda, p, q\rangle$ hereafter.

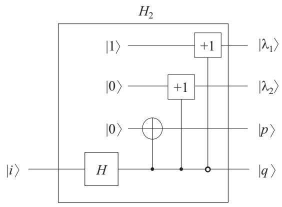

Figure 4: A quantum circuit for performing H_2. The input $|i\rangle$ is $|0\rangle$ or $|1\rangle$, which means the group element $c_{(1,2)}^0$ (the identity) or $c_{(1,2)}^1$ (the transposition $1 \leftrightarrow 2$) in S_2, respectively. The output is $\frac{1}{\sqrt{2}}(|(2,0),(0),(0)\rangle + |(1,1),(1),(1)\rangle$ or $\frac{1}{\sqrt{2}}(|(2,0),(0),(0)\rangle - |(1,1),(1),(1)\rangle$ according to the input $|i\rangle = |0\rangle$ or $|1\rangle$, respectively. The $+1$ gate performs an operation $|x\rangle \mapsto |x+1 \mod n\rangle$ on a qu$n$it.

4.2 Quantum Circuit for H_2

The H_2 is the unitary transform that performs

$$\begin{aligned} |0\rangle &\mapsto \frac{1}{\sqrt{2}}(|(2,0),(0),(0)\rangle + |(1,1),(1),(1)\rangle, \\ |1\rangle &\mapsto \frac{1}{\sqrt{2}}(|(2,0),(0),(0)\rangle - |(1,1),(1),(1)\rangle. \end{aligned}$$

It can be performed by the circuit shown in Figure 4, where the first and second horizontal lines are qu$n$its and the third and fourth lines are qubits. Hence, two qu$n$its and one qubit are added as fresh quantum resources in the circuit.

4.3 Quantum Circuit for T_m

A quantum circuit that performs T_m consists of two parts. (See Figure 5.)

In the first part, a qu$m$it with the initial state $|0\rangle$ is added as p_{m-1} and changed to $q \backslash p$, where $q \backslash p$ means the setminus $\{q_1, q_2, \cdots, q_{m-1}\} \setminus \{p_1, p_2, \cdots, p_{m-2}\}$. By this operation, $(p_1, p_2, \cdots, p_{m-1})$ is in $\mathcal{P}(\lambda)$. The $q \setminus p$ is easily calculated by an addition circuit for $\sum_{i=1}^{m-1} q_i - \sum_{i=1}^{m-2} p_i \mod m$.

In the second part, a qu$n$it with the initial state $|0\rangle$ is added as λ_m and the number of $\lambda_{q \backslash p}$ is incremented by one. It can be performed using controlled addition circuits; $|\lambda_i\rangle$ is changed to $|\lambda_i + 1 \mod n\rangle$ when $p_{m-1} = i$.

4.4 Quantum Circuit for H_m

The circuit that performs H_m has an input $|\lambda, p, q, i\rangle$, where $\lambda = (\lambda_1, \cdots, \lambda_{m-1}) \in \mathbb{Z}_{n^{m-1}}$, $p = (p_1, \cdots, p_{m-2}) \in \mathbb{Z}_{(m-1)!}$, $q = (q_1, \cdots, q_{m-2}) \in \mathbb{Z}_{(m-1)!}$, $|i\rangle \in \mathbb{Z}_m$, and an output $|\lambda', p', q'\rangle$, where $\lambda' = (\lambda'_1, \cdots, \lambda'_{m-1}) \in \mathbb{Z}_{n^{m-1}}$, $p' = (p'_1, \cdots, p'_{m-2}) \in \mathbb{Z}_{(m-1)!}$, $q' = (q'_1, \cdots, q'_{m-1}) \in \mathbb{Z}_{m!}$.

Since $H_m = A_m P_m H'_{m-1} K_m$, the circuit is constructed as shown in Figure 6. Some notes are listed below.

K_m is the set of $\lambda(c_{(i,\cdots,m-1)})$ gates according to the value i of the fourth register. Hence, they are put in line in the first part of H_m.

Before performing H_{m-1}, the basis must be changed; $\lambda \mapsto (\lambda_1, \cdots, \lambda_{q_{m-2}} - 1, \cdots, \lambda_{m-1})$, $p \mapsto p|_{m-3}$, $q \mapsto q|_{m-3}$, and $i \mapsto i - 1 \mod m$. The change of λ is performed by the reverse operation of the second stage of T_{m-1}. The change of i is performed by an addition circuit. After performing H_{m-1}, the basis is restored to the previous point of the change.

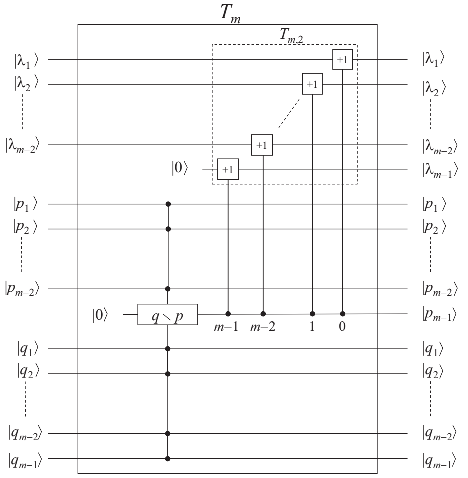

Figure 5: Quantum circuit for performing T_m. The $q \setminus p$ is the setminus $\{q_1, q_2, \cdots, q_{m-1}\} \setminus \{p_1, p_2, \cdots, p_{m-2}\}$, which can be calculated as $\sum_{i=1}^{m-1} q_i - \sum_{j=1}^{m-2} p_j \mod m$. The numbers on the p_{m-1} qu$m$it mean the conditions. The operation in the dashed line is named $T_{m,2}$.

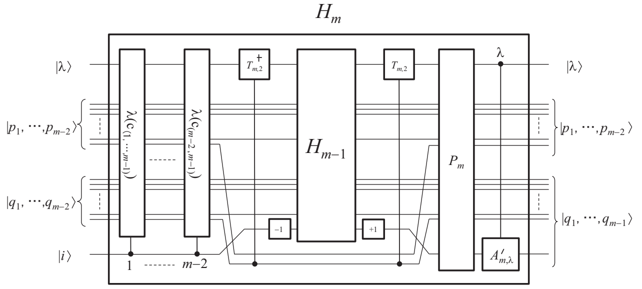

Figure 6: Quantum circuit for performing H_m, which contains H_{m-1} as a nested part. The gate $T_{m,2}$ is the operation in Figure 5. The -1 and $+1$ gates are $|i\rangle \mapsto |i-1 \mod m\rangle$ and $|i\rangle \mapsto |i+1 \mod m\rangle$, respectively.

P_n is a simple basis change, which can be performed in $O(n^2)$. A_m is the $A'_{m,\lambda}$ operation according to the value of $\lambda \in \Lambda_m$. Each $A'_{m,\lambda}$ is an $e_\lambda \times e_\lambda$ matrix, where $e_\lambda \leq m$. It is known that a $2^n \times 2^n$ unitary matrix can be performed using 4^n elementary gates. Thus, an $m \times m$ unitary matrix can be performed using m^2 elementary gates, which is $O(n^2)$.

4.5 Complexity

The circuit for H_n can be performed in $O(n^3)$. By Figure 3, the complexity of $\mathfrak{F}_n$ is then $O(n^4)$.

5. CONCLUSION

This paper proposed a QFT algorithm over symmetric groups. The time complexity of the algorithm is $O(n^4)$ for S_n. We also proposed an $O(n!n^3)$ FFT (classical) algorithm over symmetric groups. Estimating the time complexity of a QFT circuit using only elementary gates, such as single-qubit rotations and controlled-not gates, remains as further research. Another interesting problem is whether the new approach can be generalized to other groups.

6. ACKNOWLEDGEMENTS

This work was supported by JSPS KAKENHI Grant Number 25330021.

7. REFERENCES

[1] G. Alagic, C. Moore, and A. Russell. Quantum algorithms for Simon's problem over nonabelian groups. *ACM Transactions on Algorithms*, 6(1):No. 19, December 2009.

[2] R. Beals. Quantum computation of Fourier transforms over symmetric groups. In *Proceedings of the twenty-ninth annual ACM Symposium on the Theory of Computing (STOC)*, pages 48–53. ACM, May 1997.

[3] M. Clausen. Fast generalized Fourier transforms. *Theoret. Comput. Sci.*, 67:55–63, 1989.

[4] R. Cleve and J. Watrous. Fast parallel circuits for the quantum Fourier transform. In *Proceedings of the 41st Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 526–536. IEEE, 2000.

[5] D. Coppersmith. An approximate Fourier transform useful in quantum factoring. In *Technical Report RC19642*. IBM, 1994.

[6] P. Diaconis and D. Rockmore. Efficient computation of the Fourier transform on finite groups. *J. AMS*, 3(2):297–332, 1990.

[7] L. Hales and S. Hallgren. An improved quantum Fourier transform algorithm and applications. In *Proceedings of the 41st Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 515–525. IEEE, 2000.

[8] M. Hall. *The Theory of Groups*. Macmillan, New York, 1959.

[9] C. Moore, D. Rockmore, and A. Russell. Generic quantum Fourier transforms. *ACM Transactions on Algorithms*, 2(4):707–723, October 2006.

[10] C. Moore, A. Russell, and P. Sniady. On the impossibility of a quantum sieve algorithm for graph isomorphism: unconditional results. In *Proceedings of the thirty-ninth annual ACM symposium on Theory of computing (STOC)*, pages 536–545. ACM, 2007.

[11] M. Mosca and C. Zalka. Exact quantum Fourier transforms and discrete logarithm algorithms. *Int. J. Quant. Inf.*, 2(1):91–100, 2004.

[12] M. A. Nielsen and I. L. Chuang. *Quantum Computation and Quantum Information.* Cambridge Univ. Press, Cambridge, UK, 2000.

[13] J.-P. Serre. *Représentations Linéaires des Groupes Finis.* Hermann, Paris, 1971.

[14] P. W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM J. Comput.*, 26(5):1484–1509, October 1997.

[15] H. Weyl. *The Classical Groups, their invariants and representations (2nd ed.)*. Princeton Univ. Press, Princeton, 1946.

[16] H. Weyl. *The Theory of Groups and Quantum Mechanics.* Dover, New York, 1950.