Quantum Circuit for the Fast Fourier Transform

Ryo Asaka, Kazumitsu Sakai and Ryoko Yahagi

Department of Physics, Tokyo University of Science, Kagurazaka 1-3, Shinjuku-ku, Tokyo, 162-8601, Japan

November 8, 2019

Abstract

We propose an implementation of the algorithm for the fast Fourier transform (FFT) as a quantum circuit consisting of a combination of some quantum gates. In our implementation, a data sequence is expressed by a tensor product of vector spaces. Namely, our FFT is defined as a transformation of the tensor product of quantum states. It is essentially different from the so-called quantum Fourier transform (QFT) defined to be a linear transformation of the amplitudes for the superposition of quantum states. The quantum circuit for the FFT consists of several circuits for elementary arithmetic operations such as a quantum adder, subtractor and shift operations, which are implemented as effectively as possible. Namely our circuit does not generate any garbage bits. The advantages of our method compared to the QFT are its high versatility, and data storage efficiency in terms, for instance, of the quantum image processing.

1 Introduction

Quantum computing, which utilizes quantum entanglement and quantum superpositions inherent to quantum mechanics, is rapidly gaining ground to overcome the limitations of classical computing. Shor’s algorithm solving the integer factorization problem in a polynomial time, and Grover’s algorithm making it possible to substantially speed up the search in unstructured databases are one of the best-known examples of the astounding properties of quantum computing (see, for example, for various applications of quantum computing).

An implementation of the Fourier transform as a quantum circuit sometimes plays a crucial role on quantum computing. Indeed, the quantum Fourier transform (QFT) is a key ingredient of Shor’s and Grover’s algorithms. Here, the QFT is the Fourier transform for

*E-mail: 1219502@ed.tus.ac.jp
†E-mail: k.sakai@rs.tus.ac.jp
‡E-mail: yahagi@rs.tus.ac.jp
the amplitudes of a quantum state:

\[ \sum_{j=0}^{N-1} x_j |j\rangle \mapsto \sum_{k=0}^{N-1} X_k |k\rangle, \quad (1.1) \]

where we set \( N = 2^n \), and the amplitudes \( \{X_k\} \) are the classical discrete Fourier transform of the amplitudes \( \{x_j\} \)

\[ X_k = \sum_{j=0}^{N-1} W_N^{jk} x_j, \quad x_j = \frac{1}{N} \sum_{k=0}^{N-1} W_N^{-jk} X_k, \quad (1.2) \]

where \( W_N := \exp(-2\pi i/N) \). Due to the superposition of the state (1.1) and quantum parallelism, the QFT can be implemented in a quantum circuit consisting of \( O(n^2) \) quantum gates, which is much more efficient than the fast Fourier transform (FFT) whose complexity of the computation is \( O(n2^n) \).

The Fourier transform that we consider in this paper is somewhat different from the QFT: we propose a quantum implementation of the algorithm of the FFT rather than the QFT. In our procedure, a data sequence is expressed in terms of a tensor product of vector spaces:

\[ \bigotimes_{j=0}^{N-1} |x_j\rangle \]

We define the Fourier transform as

\[ \bigotimes_{j=0}^{N-1} |x_j\rangle \mapsto \bigotimes_{k=0}^{N-1} |X_k\rangle, \quad (1.3) \]

where the data sequence \( \{X_k\} \) is the Fourier transform of \( \{x_j\} \) as expressed in (1.2). We adopt the reversible FFT [6] as an algorithm of the above Fourier transform, and implement it as a quantum circuit whose computational complexity is \( O(n2^n) \). In this point of view, the processing speed is the same as the classical one, as long as we consider only a single data sequence. Nevertheless, there are following advantages compared to the classical FFT, and even compared to the QFT. The first is due to quantum parallelism: Namely, utilizing quantum superposition of multiple data sets, we can simultaneously process them. The second is due to its high versatility: the method is always applicable to data sets that can be processed by the conventional FFT. The third advantage is its data storage efficiency in terms, for instance, of the quantum image (see [2, 8, 9, 10] for some applications of the QFT to quantum data sets).

Let us illustrate the third advantage above with a simple example: an \( L \times L \) pixel image with a grayscale value ranging from 0 to \( M - 1 \) (\( M = 2^m \)) (see Fig. 1 for \( L = 2 \)). (This problem is equivalent to a lattice quantum many-body problem on an \( L \times L \) square lattice with each site occupied by a particle with \( M \) degrees of freedom.) This quantum image \( |\psi^{(\alpha)}\rangle \) (\( \alpha \) denotes the label of the image) can be represented by a tensor product of vector spaces [11, 12, 13, 14]:

\[ |\psi^{(\alpha)}\rangle = \bigotimes_{(j,k)=(0,0)}^{(L-1,L-1)} |x_{j,k}^{(\alpha)}\rangle, \quad 0 \leq x_{j,k}^{(\alpha)} \leq M - 1. \quad (1.4) \]

Since \( |\psi^{(\alpha)}\rangle \in (\mathbb{C}^2)^{\otimes mL^2} \), it uses \( mL^2 \) qubits. By use of the quantum superposition, the quantum FFT (let us denote it as QFFT) can simultaneously process at most \( 2^mL^2 \) quantum
0 ≤ x_{j,k} ≤ M − 1 (M = 2^m)

Figure 1: A 2 × 2 pixel image with a grayscale value ranging from 0 to M − 1 (M = 2^m) is depicted in the left panel. x_{j,k} in the right panel denotes the value of the grayscale on the (j,k)-site. The quantum image can be represented as |x_{0,0}\rangle \otimes |x_{0,1}\rangle \otimes |x_{1,0}\rangle \otimes |x_{1,1}\rangle ∈ (\mathbb{C}^2)^{\otimes 4m}.

Images:

\[ |\Psi\rangle = \sum_{\alpha=1}^{2^{mL^2}} c_{\alpha} |\psi^{(\alpha)}\rangle ∈ (\mathbb{C}^2)^{\otimes mL^2} \quad (c_{\alpha} ∈ \mathbb{C}). \] (1.5)

On the other hand, to apply the QFT to the above image processing, we need to prepare the quantum image in the form of

\[ |\tilde{\psi}^{(\alpha)}\rangle = \sum_{(j,k)=(0,0)}^{(L-1,L-1)} x_{j,k}^{(\alpha)} |j,k\rangle, \] (1.6)

where |\tilde{\psi}^{(\alpha)}\rangle ∈ (\mathbb{C}^2)^{\otimes 2\log_2 L} which uses only 2\log_2 L qubits (cf. (1.4) for the QFFT) [15, 16, 17]. However, since the Fourier coefficients (see (1.1) and (1.2)) are expressed as the amplitudes of the superposition, it takes exponentially long time to extract all of them completely. Furthermore, to properly perform the Fourier transform for the multiple 2^{mL^2} quantum images, they must be represented as

\[ |\tilde{\Psi}\rangle = \bigotimes_{\alpha=1}^{2^{mL^2}} |\tilde{\psi}^{(\alpha)}\rangle ∈ (\mathbb{C}^2)^{\otimes (2^{mL^2+1}) \log_2 L}. \] (1.7)

Namely, for the QFT, (2^{mL^2+1}) \log_2 L qubits are required to process the 2^{mL^2} quantum images, which are much larger than mL^2 qubits for the QFFT. Moreover, the QFT must be applied to each image individually. As a result, the total processing time for the QFFT is shorter than that for the QFT, when the number of the quantum images is sufficiently large.

In this paper, we construct a quantum circuit of the above explained QFFT, by implementing some elementary arithmetic operations such as a quantum adder [18, 19, 20, 21, 22, 23], subtractor [24, 25, 26, 27] and newly developed shift-type operations, as efficiently as possible: our quantum circuit does not generate any garbage bits.

\footnote{The multiple quantum images can also be represented as \( \bigoplus_{\alpha=1}^{2^{mL^2}} |\psi^{(\alpha)}\rangle ∈ (\mathbb{C}^2)^{2\log_2 L + mL^2} \) which uses \( 2\log_2 L + mL^2 \) qubits. However, the resulting Fourier image obtained by the QFT are no longer the same as any of the Fourier images of the individual quantum images.}
The outline of the paper is as follows. In the subsequent section, introducing the algorithm of the FFT, we show the elementary arithmetic operations required for the implementation of the QFFT as a quantum circuit. In section 3, we actually implement these elementary arithmetic operations into quantum circuits. In section 4, combining these elementary circuits efficiently, we construct a quantum circuit for the FFT. Section 5 is devoted to a summary and discussion. Some technical details are deferred to the appendices.

2 Elementary operations required for the QFFT

In this section, we briefly introduce the algorithm of the FFT and pictorially represent several arithmetic operations required for the implementation of the FFT as a quantum circuit. (See [28], for instance, for the detailed algorithm of the FFT.)

2.1 Algorithm of the FFT

Let us start the formula (1.2) and (1.3) of the Fourier transform. Setting \( W_N = \exp(-2\pi i/N) \) \((N = 2^n)\) and decomposing the summation in (1.2) into the odd and even parts, we have

\[
|X_k\rangle = \left| G_k^{(n-1,0)} + W_N^k G_k^{(n-1,1)} \right>, \quad |X_{k+N/2}\rangle = \left| G_k^{(n-1,0)} - W_N^k G_k^{(n-1,1)} \right>, \tag{2.1}
\]

where \(0 \leq k \leq N/2 - 1\), and \(G_k^{(n-1,p)} (p = 0, 1)\) is the Fourier coefficients for \(\{x_{2r+p}\}\) \((0 \leq r \leq N/2 - 1)\):

\[
G_k^{(n-1,p)} = \sum_{r=0}^{N/2-1} W_N^{2rk} x_{2r+p} \quad (p = 0, 1). \tag{2.2}
\]

Note that \(G_k^{(n-1,p)} = G_k^{(n-1,p)}\) and \(W_N^{N/2} = -1\). Pictorially, (2.1) can be represented as so-called a butterfly diagram:

\[
\begin{bmatrix}
|G_k^{(n-1,0)}\rangle \\
|G_k^{(n-1,1)}\rangle
\end{bmatrix} \rightarrow
\begin{bmatrix}
|X_k\rangle \\
|X_{k+N/2}\rangle
\end{bmatrix} =
\begin{bmatrix}
|G_k^{(n-1,0)} + W_N^k G_k^{(n-1,1)}\rangle \\
|G_k^{(n-1,0)} - W_N^k G_k^{(n-1,1)}\rangle
\end{bmatrix},
\tag{2.3}
\]

where \(0 \leq k \leq N/2 - 1\). Here the broken line means the multiplication by \(-1\). For convenience, we also denote it as a matrix-like notation:

\[
\begin{bmatrix}
|X_k\rangle \\
|X_{k+N/2}\rangle
\end{bmatrix} =
\begin{bmatrix}
1 & 1 \\
1 & -1
\end{bmatrix}
\begin{bmatrix}
1 & 0 \\
0 & W_N^k
\end{bmatrix}
\begin{bmatrix}
|G_k^{(n-1,0)}\rangle \\
|G_k^{(n-1,1)}\rangle
\end{bmatrix} =
\begin{bmatrix}
|G_k^{(n-1,0)} + W_N^k G_k^{(n-1,1)}\rangle \\
|G_k^{(n-1,0)} - W_N^k G_k^{(n-1,1)}\rangle
\end{bmatrix}. \tag{2.4}
\]

Here the matrix-like operation is defined as

\[
\begin{bmatrix}
A & B \\
C & D
\end{bmatrix}
\begin{bmatrix}
|a\rangle \\
|b\rangle
\end{bmatrix} =
\begin{bmatrix}
|Aa + Bb\rangle \\
|Ca + Db\rangle
\end{bmatrix} \quad (A, B, C, D \in \mathbb{C}). \tag{2.5}
\]
As seen in (2.4), to implement the FFT in a quantum circuit, the multiplication of the matrices

\[
\begin{bmatrix}
1 & 1 \\
1 & -1
\end{bmatrix}, \quad \begin{bmatrix}
1 & 0 \\
0 & W_N^k
\end{bmatrix}
\]  

(2.10)

Figure 2: A pictorial representation of the FFT for \( N = 8 \).

Do not confuse the above manipulation with conventional matrix operations: the results are not linear combinations of \(|a\rangle\) and \(|b\rangle\).

Again we decompose the Fourier transform for \( \{x_{2r}\} \) (resp. \( \{x_{2r+1}\} \)) into that for \( \{x_{4s}\} \) and \( \{x_{4s+2}\} \) (resp. \( \{x_{4s+1}\} \) and \( \{x_{4s+3}\} \)) (0 \( \leq s \leq N/4 - 1 \)). The result reads

\[
\begin{bmatrix}
G_k^{(n-1,p)} \\
G_k^{(n-1,p)}
\end{bmatrix} = \begin{bmatrix}
G_k^{(n-2,p)} + W_N^{k} G_k^{(n-2,p+2)} \\
G_k^{(n-2,p)} - W_N^{k} G_k^{(n-2,p+2)}
\end{bmatrix} (p = 0, 1; 0 \leq k \leq N/4 - 1),
\]  

(2.6)

Repeating this procedure, one obtains the following recursion relation:

\[
\begin{bmatrix}
G_k^{(n-m,p)} \\
G_k^{(n-m,p)}
\end{bmatrix} = \begin{bmatrix}
G_k^{(n-m,1,p)} + W_N^{k} G_k^{(n-m-1,p+2^m)} \\
G_k^{(n-m,1,p)} - W_N^{k} G_k^{(n-m-1,p+2^m)}
\end{bmatrix},
\]  

(2.8)

where 0 \( \leq p \leq 2^n - 1 \), 0 \( \leq k \leq N/2^m - 1 \). The initial states are given by

\[
|G_0^{(0,p)}\rangle = |x_p\rangle \ (0 \leq p \leq N - 1).
\]  

(2.9)

Most importantly, in each step given by (2.8), the computational complexity is \( O(2^n) \). Namely the total computational complexity of the Fourier transform is reduced from \( O(2^n n) \) to \( O(2^n n) \) for the above procedure. This is the algorithm of the FFT. In Fig. 2, we pictorially depict the FFT for \( N = 8 \).

### 2.2 Elementary operations in the FFT

As seen in (2.4), to implement the FFT in a quantum circuit, the multiplication of the matrices

\[
\begin{bmatrix}
1 & 1 \\
1 & -1
\end{bmatrix}, \quad \begin{bmatrix}
1 & 0 \\
0 & W_N^k
\end{bmatrix}
\]  

(2.10)
should be carried out in terms of quantum computation. The first one is separated into an adder, a subtractor and shift operators by the LDU decomposition

\[
\begin{bmatrix}
1 & 1 \\
1 & -1
\end{bmatrix}
= \begin{bmatrix}
1 & 0 \\
1 & -1
\end{bmatrix}
\begin{bmatrix}
1 & 0 \\
0 & 2
\end{bmatrix}
\begin{bmatrix}
1 & 1 \\
0 & 1
\end{bmatrix}.
\] (2.11)

Utilizing the matrix-like notation as in (2.4), the action of the first matrix defined in (2.4) on states \(|a\rangle\) and \(|b\rangle\) can be graphically interpreted as

\[
|a\rangle \quad |b\rangle \quad |a + b\rangle
\]

\[
|a\rangle \quad |b\rangle \quad |a - b\rangle
\]

\[
\begin{bmatrix}
1 & 1 \\
1 & -1
\end{bmatrix}
\begin{bmatrix}
|a\rangle \\
|b\rangle
\end{bmatrix}
= \begin{bmatrix}
|a + b\rangle \\
|a - b\rangle
\end{bmatrix}.
\] (2.12)

Note again that the above operation differs from the conventional matrix operations. On the other hand, the second matrix is simply expressed as

\[
|a\rangle \quad |a\rangle \\
|b\rangle \quad W^k_N |b\rangle
\]

\[
\begin{bmatrix}
1 & 0 \\
0 & W^k_N
\end{bmatrix}
\begin{bmatrix}
|a\rangle \\
|b\rangle
\end{bmatrix}
= \begin{bmatrix}
|a\rangle \\
|W^k_N b\rangle
\end{bmatrix}.
\] (2.13)

Thus the butterfly diagram as in (2.3) or (2.4) can be written as

\[
|a\rangle \quad |a + W^k_N b\rangle \\
|b\rangle \quad |a - W^k_N b\rangle
\]

\[
|a\rangle \quad |a\rangle \\
|b\rangle \quad |b\rangle
\]

\[
\begin{bmatrix}
1 & 0 \\
0 & W^k_N
\end{bmatrix}
\begin{bmatrix}
|a\rangle \\
|b\rangle
\end{bmatrix}
= \begin{bmatrix}
|a + W^k_N b\rangle \\
|a - W^k_N b\rangle
\end{bmatrix}.
\] (2.14)

Consequently, the QFFT can be implemented into a quantum circuit consisting of adders, subtractors and shift operators. In the next section, we explain these arithmetic operators. An actual implementation of these operators into the butterfly operations (2.14) is deferred to section 4.

3 Quantum circuits for arithmetic operations

In this section, we pictorially present a concept of some quantum arithmetic operations such as a quantum adder, subtractor and shift operators, which are required to implement the QFFT as a quantum circuit. Some detailed implementations for these operators are deferred to appendices.

Here we adopt two's complement notation to represent a negative number. Let us write a state \(|a\rangle\) \((a \geq 0)\) using the binary representation \(|a\rangle = |a_{n-1} \cdots a_0\rangle := |a_{n-1}\rangle \otimes \cdots \otimes |a_0\rangle\)
\[
|a_0\rangle \\
|a_1\rangle \\
|a_2\rangle \\
|a_{\pm}\rangle \\
|0\rangle \\
|a_{\pm}\rangle
\]

\[
|a_0\rangle \\
|a_1\rangle \\
|a_2\rangle \\
|a_{\pm}\rangle \\
|0\rangle \\
|a_{\pm}\rangle
\]

Figure 3: An operation to increase the number of digits from 4-qubit to 6-qubit.

\[
\{a_j \in 0, 1\}\). Let \(m (m > n)\) be a total number of qubits of the system. Let us express \(|a\rangle\) as

\[
|a\rangle = \left[ a_+ a_+ \cdots a_+ a_{n-1} \cdots a_0 \right]_{m-n},
\]  

where \(a_+ = 0\). Then, a negative number \(|b\rangle (= |a - 1\rangle\) can be represented by the complement of \(|a\rangle\):

\[
|b\rangle = \left[ a_- a_- \cdots a_- \bar{a}_{n-1} \cdots \bar{a}_0 \right]_{m-n},
\]

where \(a_- = \bar{a}_+ = 1, \bar{0} = 1\) and \(\bar{1} = 0\). Namely, for the \(m\)-qubit system, the number \(\{-2^{m-1}, -2^{m-1} + 1, \ldots, 2^{m-1} - 1\}\) can be expressed by this notation. For instance, \(m = 3\)

\[
|0\rangle = |000\rangle, \quad |1\rangle = |001\rangle, \quad |2\rangle = |010\rangle, \quad |3\rangle = |011\rangle, \quad |-4\rangle = |100\rangle, \quad |-3\rangle = |101\rangle, \quad |-2\rangle = |110\rangle, \quad |-1\rangle = |111\rangle.
\]  

3.1 Sign extension

In the actual computation, to avoid overflow, we sometimes need to increase the number of bits (a so-called sign extension). This operation can be achieved by just inserting \(a_{\pm}\)'s to the representation: For instance, the representation for the \(m\)-qubit system can be extended to that for the \(l\)-qubit system \((l > m)\):

\[
\left[ a_{\pm} \cdots a_{\pm} a_{n-1} \cdots a_0 \right]_{m-n} \rightarrow \left[ a_{\pm} \cdots a_{\pm} a_{l-1} \cdots a_0 \right]_{l-n}.
\]  

In Fig. 3, we show a quantum circuit to increase the number of digits from 4-qubit to 6-qubit.

3.2 Adding and subtracting operations

Let us consider an adder and a subtractor, by slightly modifying the arguments developed in [23, 27]. A quantum adder is defined as a transformation of the state:

\[
|a\rangle \otimes |b\rangle \rightarrow |a\rangle \otimes |a + b\rangle.
\]  

7
Graphically, it reads

\[
\begin{align*}
|a\rangle & \quad \longrightarrow \quad |a\rangle \\
|b\rangle & \quad \longrightarrow \quad |a + b\rangle \\
|a\rangle & \quad \longrightarrow \quad |a\rangle \\
|b\rangle & \quad \longrightarrow \quad |-a + b\rangle \\
\end{align*}
\]

\begin{bmatrix}
1 & 0 \\
1 & 1 \\
\end{bmatrix}
\begin{bmatrix}
|a\rangle \\
|b\rangle \\
\end{bmatrix}
= 
\begin{bmatrix}
|a\rangle \\
|-a + b\rangle \\
\end{bmatrix}.

\tag{3.6}
\]

In Appendix A, we implement this adder as a quantum circuit (see Fig. 8).

On the other hand, using the identity \(\overline{a + b} = a - b\), we define a quantum subtractor as

\[
\begin{cases}
|a\rangle \otimes |\overline{a + b}\rangle = |a\rangle \otimes |a - b\rangle \\
|a\rangle \otimes |a + b\rangle = |a\rangle \otimes |-a + b\rangle
\end{cases},
\tag{3.7}
\]

which can be implemented by just inserting NOT gates (denote it by the symbol \(\oplus\)) to the above defined adder \(3.6\) \[27\]:

\[
\begin{align*}
|a\rangle & \quad \oplus \quad |a\rangle \\
|b\rangle & \quad \oplus \quad |\overline{a + b}\rangle \\
\end{align*}
\]

\begin{bmatrix}
1 & 0 \\
1 & -1 \\
\end{bmatrix}
\begin{bmatrix}
|a\rangle \\
|b\rangle \\
\end{bmatrix}
= 
\begin{bmatrix}
|a\rangle \\
|-a + b\rangle \\
\end{bmatrix}.

\tag{3.8}
\]

### 3.3 Sign changing operation

Due to the identity

\[-a = (-1) \times a = \overline{a} + 1, \tag{3.9}\]

we can change the sign of the input number by an adder with NOT gate:

\[
\begin{align*}
|1\rangle & \quad \longrightarrow \quad |1\rangle \\
|a\rangle & \quad \oplus \quad |\overline{a}\rangle \quad \longrightarrow \quad \oplus \quad |\overline{a}\rangle \\
\end{align*}
\]

\begin{bmatrix}
1 & 0 \\
1 & -1 \\
\end{bmatrix}
\begin{bmatrix}
|a\rangle \\
|\overline{a}\rangle \\
\end{bmatrix}
= 
\begin{bmatrix}
|a\rangle \\
|\overline{a}\rangle \\
\end{bmatrix}.

\tag{3.10}
\]

### 3.4 Arithmetic shift operations

Let us implement an operation to multiply by \(2^p\) \((p \in \mathbb{N})\):

\[
|a\rangle \mapsto |2^p a\rangle.
\tag{3.11}
\]

This operation is carried out by shifting the digits to the left (arithmetic left shift):

\[
|a\rangle = |a_{m-n} \ldots a_{n-1} \ldots a_0\rangle \mapsto |\underbrace{a_{m-n} \ldots a_{n-1} \ldots a_0}_{m-n-p} \underbrace{0 \ldots 0}_{p}\rangle = |2^p a\rangle.
\tag{3.12}
\]

8
Figure 4: The left (right) panel denotes a quantum circuit for an arithmetic left shift (right shift). The least significant digit $a_0$ is lost for this right shift operation (right panel).

Let us pictorially express this operation as

$$ |a⟩ \xrightarrow{2^p} |2^p a⟩ $$

(3.13)

In a similar manner, we can define an arithmetic right shift which is an operation to multiply by $2^{-p}$:

$$ |a⟩ = |a_{-}\ldots a_{-m-n}a_{n-j-1}\ldots a_{0}a_{-1}\ldots a_{-j}⟩ $$

$$ \mapsto |a_{-}\ldots a_{-m-n}a_{n-j-1}\ldots a_{0}a_{-1}\ldots a_{-j+p}⟩ = |2^{-p} a⟩ $$

(3.14)

where $a_{-k}$ ($1 \leq k \leq j - p$) are the fractional part of $a$. Note that, in the above operation, $p$ significant digits are lost. We also graphically denote this operation

$$ |a⟩ \xrightarrow{2^{-p}} |2^{-p} a⟩ $$

(3.15)

The actual implementation of these shift operations into quantum circuits can be accomplished by certain combinations of SWAP and CNOT gates. In Fig. 4, we show a quantum circuit for the left (right) shift operation for $m = 3$, $n = 3$ and $p = 1$.

Combining the adder, the subtractor and the shift operations, we can arithmetically manipulate an arbitrary number.

4 Decomposition of the butterfly operation

Now we decompose the butterfly operation (2.14) (see also (2.3)), which plays a central role on the QFFT, into the elementary arithmetic operations shown in previous section. First,
Figure 5: A quantum circuit of an adder with a right shift operation \((4.6)\) for \(p = 2\). Though two additional qubits are required for this right shift operation, these two qubits are reusable. Moreover this quantum circuit does not generate any garbage bits. Some elements of the circuit is defined in Fig. 6.

we decompose the butterfly operation into elementary operations:

\[
\begin{align*}
|a\rangle & \quad |a + b\rangle \\
|b\rangle & \quad |a - b\rangle \\
& \quad \equiv \\
& \quad |a\rangle \quad |a + b\rangle
\end{align*}
\]  

(4.1)

In the butterfly operation, the input states consist of \(|(W_N^k)\rangle\). Let us abbreviate the component \(W_N^k = \exp(-i\frac{2\pi}{N}k)\) to \(\exp(i\theta)\) for simplicity. The calculation of \(|\exp(i\theta)\rangle\) is decomposed into

\[
|\exp(i\theta)\rangle = |(\cos \theta + i \sin \theta)(a_r + ia_i)\rangle = \begin{bmatrix} 1 & i \\ \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix} \begin{bmatrix} a_r \\ a_i \end{bmatrix},
\]  

(4.2)

where \(a_r\) and \(a_i\) are, respectively, the real and imaginary part of \(a\). The rotation matrix is further decomposed into adding (resp. subtracting) operators \((3.6)\) (resp. \((3.8)\)) with arithmetic right shift operations \((3.15)\) [6, 28, 29]:

\[
\begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix} = \begin{cases} 
\begin{bmatrix} 1 & \cos \theta - 1 \\ 0 & \sin \theta \end{bmatrix} & \begin{bmatrix} 1 & 0 \\ \sin \theta & 1 \end{bmatrix} & \begin{bmatrix} 1 & \cos \theta - 1 \\ 0 & \sin \theta \end{bmatrix} \\
-1 & \cos \theta + 1 \\ 0 & \sin \theta \\ 1 & \sin \theta & -1 \end{cases},
\]  

(4.3)

(4.4)

Because

\[
\frac{\cos \theta - 1}{\sin \theta} \leq 1 \text{ for } \theta \in \left[-\frac{\pi}{2}, \frac{\pi}{2}\right], \quad \frac{\cos \theta + 1}{\sin \theta} < 1 \text{ for } \theta \in \left[-\pi, -\frac{\pi}{2}\right) \cup \left(\frac{\pi}{2}, \pi\right],
\]  

(4.5)
Figure 6: The Toffoli gate (left panel) and the Peres gate (right panel). $V$ and $V^\dagger$ are defined by (4.7).

to apply the right shift operator, we use (4.3) for $\theta \in [-\frac{\pi}{2}, \frac{\pi}{2})$, while, for $\theta \in [-\pi, -\frac{\pi}{2}] \cup (\frac{\pi}{2}, \pi]$, we use (4.4). The each matrix in the RHS of the above decomposition is schematically given by

$$V = \frac{1 + i}{2} \begin{pmatrix} 1 & -i \\ -i & 1 \end{pmatrix}, \quad V^\dagger = \frac{1 - i}{2} \begin{pmatrix} 1 & i \\ i & 1 \end{pmatrix}. \quad (4.7)$$

In fact, this decomposition makes it possible to efficiently implement the elementary arithmetic operations required for the QFFT. Namely, we develop an implementation of the above procedure into a quantum circuit so as not to generate any garbage bits. For $p = 2$, the quantum circuit is shown in Fig. 5. The circuit consists of the Toffoli gate [30] and the Peres gate [31] defined as in Fig. 6, where $V$ and $V^\dagger$ are, respectively, given by

$$V = \begin{pmatrix} 1 + i \\ -i \end{pmatrix} \begin{pmatrix} 1 \\ -i \end{pmatrix}, \quad V^\dagger = \begin{pmatrix} 1 - i \\ i \end{pmatrix} \begin{pmatrix} 1 \\ i \end{pmatrix}. \quad (4.8)$$

Thanks to the circuit (4.6), we can construct quantum circuits of adding and subtracting for an arbitrary number:
Thus, the decompositions (4.3) and (4.4) are summarized graphically:

\[ |a\rangle \rightarrow W_N^k |a\rangle \rightarrow |W_N^k a\rangle, \]

(4.9)

Considering that quantum circuit is reversible, the calculation of inverse FFT requires two matrices

\[
\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & (W_N^k) \end{bmatrix}^{-1} = \begin{bmatrix} 1/2 & 1/2 \\ 0 & (W_N^{-k}) \end{bmatrix},
\]

(4.10)

which are also decomposed into an adder, a subtractor and shift operators.

5 Summary and Discussion

In this paper, we have discussed the implementation of the FFT as a quantum circuit. The quantum version of the FFT (QFFT) is defined as a transformation of a tensor product of quantum states. The QFFT has been constructed by a combination of several fundamental arithmetic operators such as an adder, subtractor and shift operators which have been implemented into the quantum circuit of the QFFT without generating any garbage bits.

One of the advantages of the QFFT is due to its high versatility: the QFFT are applicable to all the problems that can be solved by the conventional FFT. The frequency domain filtering of digital images is one of the possible applications of the QFFT. The major advantage of using the QFFT lies in its quantum superposition: multiple images are processed simultaneously (see Fig. 7 as a conceptual image of quantum parallelism). It is even superior to the QFT when the number of images is sufficiently large. Thus, the QFFT will be a great help in the industries including security, healthcare, and marketing research, where numerous images are involved. In particular we can create a high pass filter for edge detection (Fig. 7).

There are several issues remaining to be solved. The first one is how to effectively generate a quantum superposition of multiple data sets as in (1.5). Though, in principle, one can generate it using the Hadamard gate and the Pauli-X gate, it is still a difficult problem to superpose arbitrary data sets with arbitrary probability amplitudes. The second is how to use the resultant multiple data sets obtained by performing the QFFT. The QFFT sustains all the information of Fourier coefficients until the moment the quantum state is measured. If the quantum state that contains the Fourier coefficients of multiple data sets was passed on to some quantum device directly and there were some proper techniques to handle it, it would play a key role in the field of quantum machine learning.
Acknowledgment

The present work was partially supported by Grant-in-Aid for Scientific Research (C) No. 16K05468 from the Japan Society for the Promotion of Science.

A Implementation of the adder circuit

Let us consider the addition of two $n$-bit numbers $|a\rangle$ and $|b\rangle$ with the binary representation $|a\rangle = |a_{n-1} \cdots a_0\rangle$, $|b\rangle = |b_{n-1} \cdots b_0\rangle$ ($\{a_j, b_j \in \{0, 1\}\}$). It is calculated as

\[ c_\pm \cdots c_\pm c_n \ c_{n-1} \cdots c_2 \ c_1 \]
\[ a_\pm \cdots a_\pm a_\pm a_{n-1} \cdots a_2 \ a_1 \ a_0 \]
\[ + \ b_\pm \cdots b_\pm b_\pm b_{n-1} \cdots b_2 \ b_1 \ b_0 \]
\[ s_\pm \cdots s_\pm s_n \ s_{n-1} \cdots s_2 \ s_1 \ s_0 \]

(A.1)

where the carry bit $c_j$ and the sum bit $s_j$ ($j = 1, \cdots n$) are defined by

\[ c_j = \begin{cases} 
  a_0 b_0 & (j = 0) \\
  a_{j-1} b_{j-1} \oplus b_{j-1} c_{j-1} \oplus c_{j-1} a_{j-1} & (2 \leq j \leq n) \\
  a_j b_j \oplus c_j & (1 \leq j \leq n-1) \\
  c_j & (j = n) 
\end{cases} \]

(A.2)

Note that the symbol $\oplus$ denotes exclusive disjunction. In terms of a quantum circuit, this addition is implemented in the transformation of the state

\[ |a\rangle \otimes |b\rangle \longrightarrow |a\rangle \otimes |a+b\rangle, \]

(A.3)
Figure 8: An adder circuit. The circuit consists of the Toffoli and the Peres gates defined as in Fig. 6.

and Fig. 8 shows the actual circuit which is a slightly modified version of a quantum adder originally developed in [23].

B Number of extra qubits $a_{\pm}$ required for the QFFT

The key structure of the QFFT is the butterfly operation combined with the multiplication of $W_k^N$. In the whole QFFT process, we have $\log_2 N$ layers of this structure. By counting the ratio of the possible value ranges of the input and output of this structure, and multiplying it by $\log_2 N$, we can estimate the number of qubits $a_{\pm}$ required for the QFFT.

Let $I_r$ and $I_i$ be the real and imaginary part of the input $I$ and impose the following conditions on $I$:

$$\begin{cases} -2^n \leq I_r, I_i \leq 2^n - 1 \\ I_r^2 + I_i^2 \leq 2^{2n} \end{cases} . \quad (B.1)$$

Because the initial input is a real number ($I_i = 0$), it satisfies the above conditions. The multiplication of $W_k^N$ is essentially a rotation in the complex space. Namely, it preserves the distance:

$$[WI_r]^2 + [WI_i]^2 \leq 2^{2n}, \quad (B.2)$$

where the suffixes $i$ and $r$ represent the real and imaginary part, respectively. The output $O$ of the butterfly structure then satisfies

$$\begin{cases} -2^{n+1} \leq O_r, O_i \leq 2^{n+1} - 1 \\ O_r^2 + O_i^2 \leq 2^{2(n+1)} \end{cases} . \quad (B.3)$$

The absolute value of the output is doubled compared to that of the input. Therefore, one extra qubit $a_{\pm}$ is required for one layer and $\log_2 N a_{\pm}$’s for the total QFFT.
References

[1] Shor, P.W.: Algorithms for quantum computation: discrete logarithms and factoring. Proc. Annu. Symp. Found. Comput., 124-134 (1994)

[2] Grover, L.K.: A fast quantum mechanical algorithm for database search. In Proc. ACM STOC, 212-219 (1996)

[3] Nielsen, M.A., Chuang, I.L.: Quantum Computation and Quantum Information. Cambridge University Press, Cambridge (2000)

[4] Jozsa, R.: Quantum algorithms and the Fourier transform. Proc. R. Soc. London Ser. A 454, 323-337 (1998)

[5] Cooley, J.W., Tukey, J.W.: An algorithm for the machine calculation of complex Fourier series. Math. Comput 19(90), 297-301 (1965)

[6] Oraintara, S., Chen, Y.J., Nguyen, T.Q.: Integer fast Fourier transform. IEEE Trans. Signal Process. 50, 607-618 (2002)

[7] Li, H.S., Fan, P., Xia, H., Song, S., He, X.: The quantum Fourier transform based on quantum vision representation. Quantum Inf. Process. 17, 333 (2018)

[8] Gong, L.H., He, X.T., Tan, R.C., Zhou, Z.H.: Single channel quantum color image encryption algorithm based on HSI model and quantum Fourier transform. Int. J. Theor. Phys. 57, 5973 (2018)

[9] Li, P., Xiao, H.: An improved filtering method for quantum color image in frequency domain. Int. J. Theor. Phys. 57(1), 258278 (2018)

[10] Zhou, S.S., Loke, T., Izaac, J.A., Wang, J. B.: Quantum Fourier transform in computational basis. Quantum Inf. Process. 16(3), 82 (2017)

[11] Zhang, Y., Lu, K., Gao, Y.H., et al.: NEQR: a novel enhanced quantum representation of digital images. Quantum Inf. Process. 12(12), 2833-2860 (2013)

[12] Jiang, N., Wu, W.Y., Wang, L., Zhao, N.: Quantum image pseudocolor coding based on the density-stratified method. Quantum Inf. Process. 14(5), 1735-1755 (2015)

[13] Wang, J.: QRDA: quantum representation of digital audio. Int. J. Theor. Phys. 55, 1622-1641 (2015)

[14] Li, H.S., Zhu, Q., Song, L., et al.: Image storage, retrieval, compression and segmentation in a quantum system. Quantum Inf. Process. 12(9), 22692290 (2013)

[15] Sun B., Le P.Q., Iliyasu A.M., Adrian Garcia J., Yan F., Dong J., F., Hirota, K.: A multi-channel representation for images on quantum computers using the RGB α color space. Proceedings of the IEEE 7th International Symposium on Intelligent Signal Processing, pp. 160-165 (2011)

[16] Yuan, S., Mao, X., Xue, Y., Chen, L., Xiong, Q., Compare, A.: SQR: a simple quantum representation of infrared images. Quantum Inf. Process. 13(6), 127 (2014)
[17] Le, P., Dong, F., Arai, Y., Hirota, K.: Flexible representation of quantum images and its computational complexity analysis. In: Proceedings of the 10th Symposium on Advanced Intelligent Systems (ISIS 2009), pp. 146149 (2009)

[18] Vedral, V., Berenco, A., Ekert, A.: Quantum networks for elementary arithmetic operations. Phys. Rev. A 54(1), 147153 (1996)

[19] Draper, T.G.: Addition on a Quantum Computer. arXiv:quant-ph/0008033 (2000)

[20] Kaye, P.: Reversible Addition Circuit Using One Ancillary Bit with Application to Quantum Computing. arXiv:quant-ph/0408173v2 (2004)

[21] Cuccaro, S.A., Draper, T.G., Kutin, S.A. et al.: A new quantum ripple-carry addition circuit. arXiv preprint arXiv:quant-ph/0410184 (2004)

[22] Takahasi, Y., Kunihiro, N.: A fast quantum circuit for addition with few qubits. Quantum. Inform. Comput. 8, 636649 (2008)

[23] Thapliyal, H., Ranganathan, N.: Design of efficient reversible logic-based binary and BCD adder circuits. ACM J. Emerg. Technol. Comput. Syst. (JETC) 9(3), 17 (2013)

[24] Cheng, K., Tseng, C.: Quantum full adder and subtractor. Electron. Lett. 38 (22), 1343-1344 (2002)

[25] Thapliyal, H., Ranganathan, N.: Design of efficient reversible binary subtractors based on a new reversible gate. IEEE Compu. Soc. Symp. VLSI. 229-234 (2009)

[26] Thapliyal, H., Ranganathan, N.: A new design of the reversible subtractor circuit. In: Proceedings of the IEEE Conference on Nanotechnology, pp. 1430-1435 (2011)

[27] Thapliyal, H.: Mapping of subtractor and adder-subtractor circuits on reversible quantum gates. Transaction on Computer Science XXVII. LNCS, 9570, pp. 10-34. Springer, Heidelberg (2016)

[28] Rao, K.R., Kim, D.N., Hwang, J.J.: Fast Fourier transform-algorithms and applications. 1st edn. Springer Publishing Company. (NY), (2010)

[29] Daubechies, I., Sweldens, W.: Factoring wavelet transforms into lifting steps. J Fourier. Anal. Appl. 4(3), 247-269 (1998)

[30] Toffoli, T.: Reversible computing. In: Tech memo MIT/LCS/TM-151, MIT Lab for Computer Science (1980).

[31] Peres A.: Reversible logic and quantum computers. Phys. Rev. A 32, 3266-3276 (1985)