The Quantum Fourier Transform

Adrian Valencia[#1] Angel Martinez[*2] Jacob Ursenbach[#3]

Department of Computer Science

California State University, Chico 400 W First St Chico, CA, United States 95929-0722

[1]adrian.valen2002@gmail.com, avalenciasantos@csuchico.edu

[2]martinezangel3698@gmail.com, ammartinez20@csuchico.edu

[3]jacob.ursenbach@gmail.com, jlursenbach@csu.fullerton.edu

***Abstract*—**

The Quantum Fourier Transform (QFT) is a critical algorithm in Quantum Computing. In this paper, we investigate the mathematical roots of the QFT, explain the gate representation, and use circle notation to show how the phases of the quantum wave functions are shifted similarly to the Discrete Fourier Transform. We describe how the QFT has applications in many other algorithms and use Shor's Algorithm as a specific example, illustrating how the QFT is one of the key algorithms quantum computing relies on to provide an exponential speedup over other algorithms. Finally, we discuss the difficulties in implementing the QFT with modern Quantum computers and what fields and different types of research the QFT can be used for. This overview will hopefully promote a better understanding of the QFT for the reader, as it did for us when writing it, and provide a doorway towards additional quantum exploration and development.

I. Introduction

A. Gentle Start

Quantum computing is a burgeoning field at the intersection of physics, computer science, and mathematics. Traditional classical computing operates on bits that exist in either state 0 or state 1. Until recently, all of modern computer science, including the vast capabilities of modern AI models like GPT 4 and Dali 2, have used this binary framework to build the world we know. Quantum computing, however, works with quantum bits, or qubits, which can exist in a state of superposition. They can exist in the states of both 0 and 1 simultaneously. This crucial difference and the properties of entanglement and quantum interference allow quantum computers to process information fundamentally differently than classical computers and have the potential to allow us to achieve feats that would take traditional classical computers years, if not centuries, to complete.

B. Motivation

In their 2013 book, "Quantum Mechanics with Applications to Nanotechnology and Information Science," Yehuda B. Band and Yshai Avishai stated: "The quantum Fourier transform is the only known tool in quantum computation that yields an exponential advantage over classical computational methods" [1]. As the field of quantum computing grows, so does the need for efficient quantum algorithms. Quantum computing will be specifically useful for its potential to run algorithms that outperform their classical counterparts in solving certain problems. The Quantum Fourier Transform (QFT) is the core of what makes these algorithms so exciting and is the engine that allows them to do what they do. It is one of the few tools in quantum computation that yields an exponential advantage over classical computational methods. The QFT transforms a quantum state into its frequency representation. This capability can be used to solve problems that would be infeasible for classical computers.

C. What applications can benefit

The most famous example of how the QFT can be used to perform tasks exponentially faster than any other method is Shor's algorithm. Shor's algorithm uses the QFT to factor huge prime numbers efficiently, which is forcing an industry-wide rethinking of encryption due to classical cryptography's reliance on this task being unrealistic for even traditional supercomputers [2]. In fields like artificial intelligence and machine learning, algorithms utilizing the QFT could potentially sift through massive search spaces exponentially more efficiently [3]. Algorithms utilizing the QFT also allow for the simulation of quantum systems like complex molecules in a way that can open the door to medical and chemical advances that we could never achieve otherwise [4]. As a fundamental quantum algorithm, the QFT underpins these possibilities and is key to realizing quantum computing's full potential. This is the secret sauce of quantum computing.

II. Problem Statement

Traditional computers are functionally incapable of solving some problems within a reasonable time. Quantum algorithms can be a solution for this. Still, for that to occur, computer scientists must be capable of leveraging the Quantum Fourier Transform effectively for various custom purposes. This remains a challenge due to the difficulty of gaining a nuanced understanding of the QFT.

III. Goal

As Computer Scientists, who are working towards fully comprehending and exploiting the potential of Quantum Computing and taking full advantage of quantum algorithms, a comprehensive understanding of the Quantum Fourier Transform is non-negotiable. This paper will attempt to dissect and illuminate the algorithm, providing sufficient clarity and understanding so that readers can effectively incorporate the QFT in their own custom quantum algorithms.

IV. Background

Background for the Quantum Fourier Transform:

A. *What is the Quantum Fourier Transform?*

The Quantum Fourier Transform algorithm is a quantum algorithm that performs mathematical operations on quantum states and transforms them into the frequency domain representation [5]. This frequency domain representation yields information about the frequencies found in the input, a series of quantum states represented as qubits that are in superposition. The QFT then uses the properties of superposition with these qubits and performs mathematical calculations that measure the phases and amplitudes of each state. This results in finding the underlying frequencies present in the input quantum states.

B. *What does the Quantum Fourier Transform do?*

The traditional Fourier Transform algorithm is an algorithm that breaks down wave-based signals into their individual frequency components. This enables us to identify the different frequencies present in the signals and their intensities. For example, in the realm of audio processing, it enables us to understand the spectral components of a sound signal.

In quantum computing, we have similar needs. Rather than focusing on audio or electrical waves, quantum computing requires analyzing multiple qubits simultaneously. This allows us to leverage superposition to understand their phases and frequencies. This capability provides the ability to perform various functions on a set of entangled, superimposed qubits and use the properties of constructive and destructive interference to collapse the wave functions into the solutions for any number of various mathematical questions. By analyzing the 'n' number of qubits in superposition simultaneously, we can discern the interacting phases for all the qubits. This allows us to decode the frequencies of the quantum states.

V. Algorithm Analysis

The Quantum Fourier algorithm begins by applying a series of rotations. Generally, Hadamard gates set the entire system into a new quantum state. However, this is not always true. Phase shift operations provide a systematic way to create an interference pattern among the states so that once the qubits are measured, a string of bits represents the sampled frequencies from the initial state. Note that the QFT is generally used on bit strings representing integers [6]. The final product of the QFT is a quantum state, which is a complex superposition where the amplitudes and phases represent the Fourier transform of the input state. The amplitudes and phases of the individual quantum states in the superposition correspond to the Fourier coefficients.

C. *Classical Fourier Transform*

The classical Discrete Fourier Transform (DFT) is defined as such [7]:

$$X_k = \sum_{n=0}^{N-1} x_n \cdot e^{-\frac{i2\pi}{N} kn}.$$

Eq. 1 Discrete Fourier Transform

This function takes a time-domain wave-based signal and decomposes it into the frequency-domain components. For example, this example shows 2 pure wave signals and their combined waveform [8].

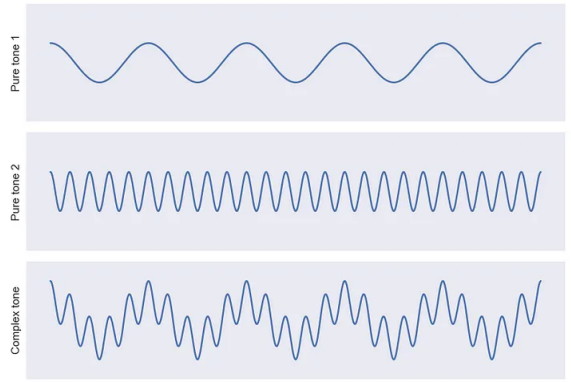

Fig. 1 Three different sinusoidal functions. The third is a combination of the top two.

The DFT is then leveraged to decompose the frequency components of the wave functions. This provides a much more intuitive method of understanding the signals found within the wave function. The following figure illustrates the results of the decompositions of the wave functions:

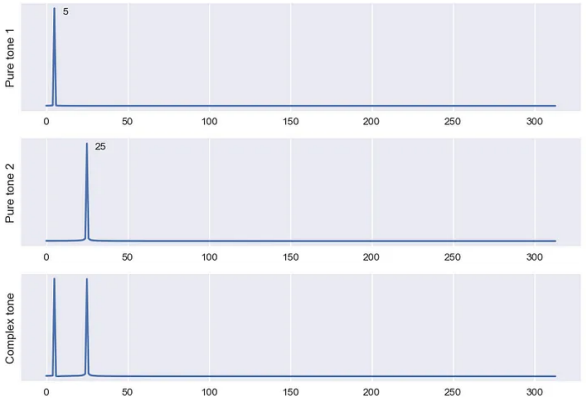

Fig. 2 Frequency Domain representation of wave functions in Figure 1

One of the essential qualities of the Discrete Fourier Transform is its status as a Unitary Operator. This is a requirement for use in Quantum Circuits. We will return to this point later; this allows the function to operate bidirectionally so that no information is lost in the transformation process. Also, as a result of its unitary status, the function can also be represented as a linear matrix transformation, allowing for easier mathematical computation.

If it is hard to understand how the DFT decomposes a time-domain wave function into its frequency-domain components, the alternate version of the formula, the Fourier Series (where Eq. 1 comes from) might help clarify.

$$g(t) = a_0 + \sum_{m=1}^{\infty} a_m \cos\left(\frac{2\pi mt}{T}\right) + \sum_{n=1}^{\infty} b_n \sin\left(\frac{2\pi nt}{T}\right)$$

Eq. 2 Fourier Series

Equation 2 breaks g(t), a periodic function, into the sums of a constant, cosine, and sine functions. This illustrates how the DFT breaks up the time-domain wave into the frequency domain.

Simon Xu, a computer science educator, provides a video with a fantastic demonstration of the decomposition of white noise using this function [9]. This paper has used the phrase: "time domain to frequency domain" more than once, and the below screenshot of his video provides a fantastic illustration of what that means. Notice the x-axis of the representations shift from time to frequency. As Simon points out in the video, since white noise contains all frequencies within the human spectrum at equal levels, there is a straight line on the frequency domain graph rather than a spike, as illustrated in Figure 2.

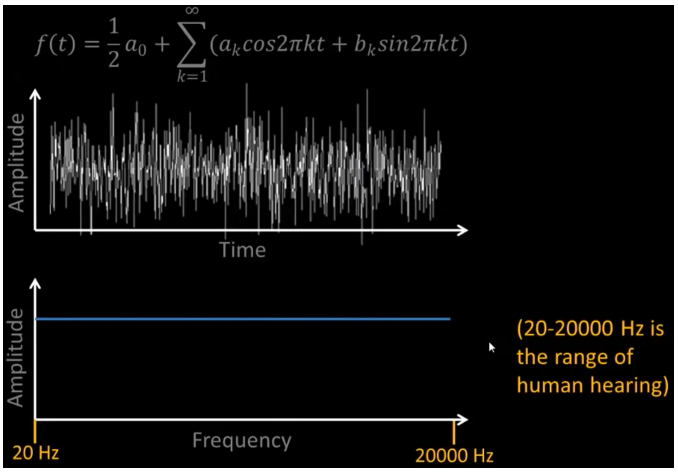

Fig. 3 Screenshot from Discrete Fourier Transform - Simple Step by Step by Simon Xu. Timestamp 1:15

D. The Quantum Fourier Transform

The QFT has a slightly different mathematical representation than the DFT due to the difference in qubit phases and traditional waves. The mathematical representation is as follows [7]:

$$\text{QFT: } |x\rangle \mapsto \sum_{n=0}^{N-1} e^{-\frac{2\pi i}{N}nx} |n\rangle$$

Eq. 3 Quantum Fourier Transform

As you can see, equations 1 and 3 are almost identical. In the QFT, x represents the basis state, while n is the data size.

E. Step-by-Step description of the QFT (for 3 QB circuit):

The following is a step-by-step description of how one would run the QFT for a 3 qubit circuit.

E.A.1 *Step 1: Initialization*

Let's assume we're implementing Shor's algorithm. One of the reasons the QFT can help speed up Shor's algorithm is that we can define a periodic function where the period is unknown [10]. That function defines the number we want to factor in a way that if you're able to find the period of the function, you can factor the number.

E.A.2 Step 2: Apply Hadamard Gate to Qubit 1

We will be setting each qubit into superposition

E.A.3 Step 3: Apply Controlled Phase Gates to Qubit 1

This is where the QFT starts to mirror the DFT. Since there are 3 qubits, we will apply 2 controlled phase gates, each controlled by a different qubit. The phase rotation is dependent on the number of qubits away from the first.

E.A.4 Step 4:Apply Hadamard Gate to Qubit 2

We continue down the line, putting each qubit into superposition

E.A.5 Step 5: Apply Controlled Phase Gate to Qubit 2

Now we only use the third qubit as a control and apply 1 phase gate. The algorithm applies phase gates to every qubit, with all untouched qubits defined as controls for the phase gates.

E.A.6 Step 6: Apply Hadamard Gate to Qubit 3

We set the last qubit into superposition. Notice there was no phase change. We want the qubits to create an interference pattern, and the goal was for them all to be offset. Since all of the other qubits have shifted their phases, this one can stay at its original phase.

E.A.7 Step 7: Swap Qubits

We swap qubits because the QFT is a "Little Endian" function, wherein the least significant qubits are processed first. This contrasts with most traditional computing, which is "Big Endian", processing the most significant bit first. The translation can happen in the quantum circuit but is sometimes left out, as it can also be done in the traditional computer as well [11].

We now have a quantum state that is generally defined as $|\varphi\rangle$, which should define the time-domain wave functions in the frequency domain.:

E.A.8 Example of a 3 Qubit Circuit

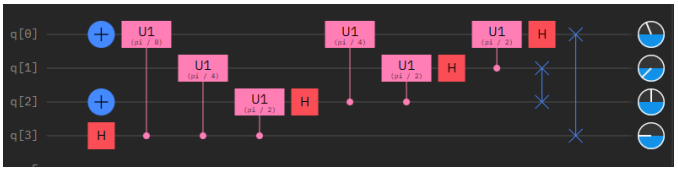

Fig. 4 3 Qubit Circuit

E.A.9 3 qubit QFT Code Block (OPENQASM 2.0)

```
OPENQASM 2.0;
include "qelib1.inc";

qreg q[4];
creg c[5];

x q[0];
x q[2];
h q[3];
cu1(pi / 8) q[3], q[0];
cu1(pi / 4) q[3], q[1];
cu1(pi / 2) q[3], q[2];
h q[2];
cu1(pi / 4) q[2], q[0];
cu1(pi / 2) q[2], q[1];
h q[1];
cu1(pi / 2) q[1], q[0];
h q[0];
swap q[1], q[2];
swap q[0], q[3];
```

F. Math for QFT on 2 qubits

F.A.1 2 Qubit Circuit:

Here's the most basic application of the QFT, on a 2 qubit system. You can identify the few processes that occur in the QFT.

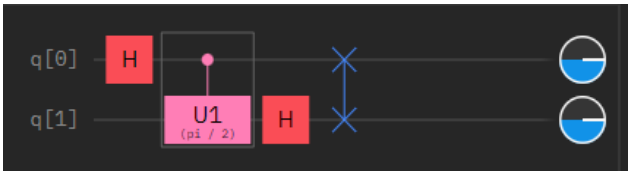

Fig. 4 QFT on 2 qubits

F.A.2 Matrix for QFT on 2 qubits:

$$QFT_2 \frac{1}{2}\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & i & -1 & -i \\ 1 & -1 & 1 & -1 \\ 1 & -i & -1 & i \end{bmatrix}$$

Eq. 4 Quantum Fourier Transform 2 QB matrix

This matrix is the quantum Fourier Transform for 2 qubits. It is the final result of applying all of the gates in the circuit. The gate representations are included below. If you multiply all of the below gates in order, you will end up with the above matrix. All qubits have been entangled, and the probability amplitudes of the states in the superposition are determined by the Fourier transform of the amplitudes of the input states

$$swap: \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Eq. 4 Swap Gate

$$I \otimes H: \frac{1}{2}\begin{bmatrix} 1 & 1 & 0 & 0 \\ 1 & -1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & -1 \end{bmatrix}$$

Eq. 5 $I_2 \otimes$ Hadamard

$$ctrl-Z: \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & i \end{bmatrix}$$

Eq. 5 Phase shift (ctrl-sqrt{z})

$$H \otimes I: \frac{1}{2}\begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & -1 & 0 \\ 0 & 1 & 0 & -1 \end{bmatrix}$$

Eq. 5 Hadamard $\otimes$ I_2

G. QFT is a Unitary Circuit

As this is a quantum circuit, This may go without saying, but the properties of such are integral to the proper working and application of the circuit.

G.A.1 Reversibility shown in proof [12]:

This proof shows that the QFT is unitary by multiplying it with its dagger (the complex conjugate of its transpose). When you multiply the two matrices together, you get the identity matrix.

$$\hat{F} = \left(\sum_{j',k'} \frac{e^{2\pi i j' k'/N}}{\sqrt{N}} |k'\rangle\langle j'| \right), \quad \hat{F}^\dagger = \left(\sum_{j,k} \frac{e^{-2\pi i j k/N}}{\sqrt{N}} |j\rangle\langle k| \right)$$

$$\begin{aligned} \hat{F}^\dagger \hat{F} &= \frac{1}{N} \sum_{j,k,j',k'} e^{2\pi i (j'k' - jk)/N} |j\rangle\langle j'| \delta_{kk'} \\ &= \frac{1}{N} \sum_{j,k,j'} e^{2\pi i (j'-j)k/N} |j\rangle\langle j'| \\ &= \sum_{j,j'} |j\rangle\langle j'| \delta_{jj'} = \sum_{j} |j\rangle\langle j| = \hat{I}. \end{aligned}$$

Eq. 6 Proof of QFT Unitary Status

H. Circle Notation

For the circle notation you'll notice the general results don't change for the phase shift gates, it mirrors the Hadamard gate exactly. The only change you'll see is a shift of the phase within the circles.

H.A.1 QFT

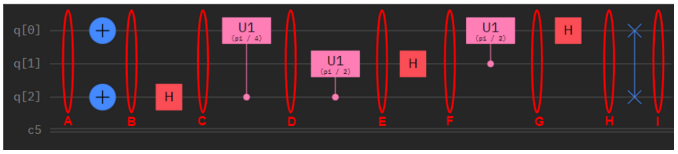

Fig. 5 3 Qubit Quantum Fourier Transform Circuit

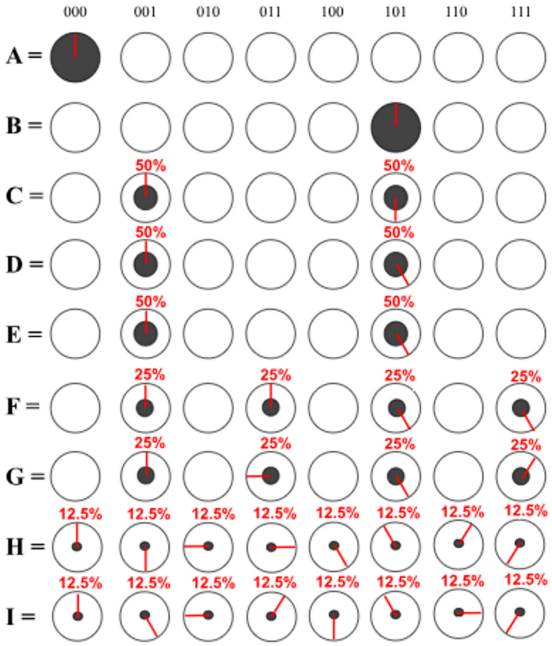

Fig. 6 3 QB QFT circle notation

H.A.2 Inverse QFT

The Inverse QFT is just as useful as the QFT, and the two functions do not always have to be used together; however, you'll see that the inverse QFT completely reverses everything that occurs in the QFT.

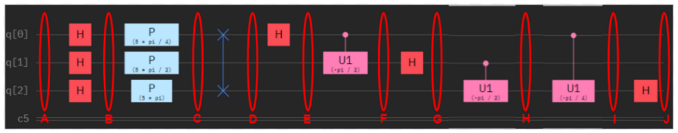

Fig. 7 3 Qubit Quantum Inverse Fourier Transform Circuit

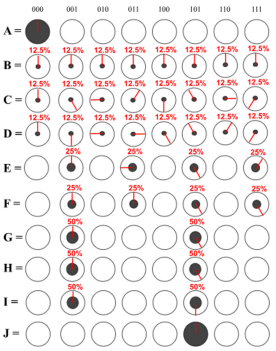

Fig. 8 3 Qubit Inverse QFT Circle Notation

I. How does the QFT interact with other algorithms:

I.A.1 Shor's Algorithm (Period finding)

Shor's algorithm uses the fact that you can define factors of a number by a function that introduces a form of periodicity [13]. Traditionally we would use Euclid's Algorithm [13]

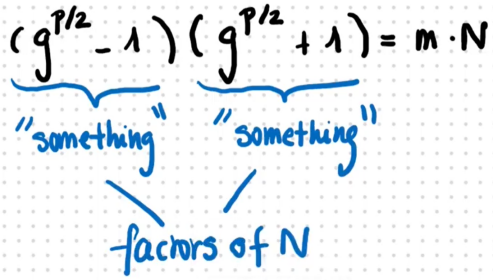

Fig. 9 Shor's Algorithm periodicity function

I.A.1.1 Complexity in Shore's Algorithm using classical vs quantum computers

The complexity Shor's algorithm provides over classical computing is known to be an exponential speedup. The closest competitor in classical computing is the number field sieve which has a time complexity of $O(e^{cd^{1/3}})$. Shor's, however speeds this up by removing the exponential, reducing the time complexity to $O((logN)^3)$.

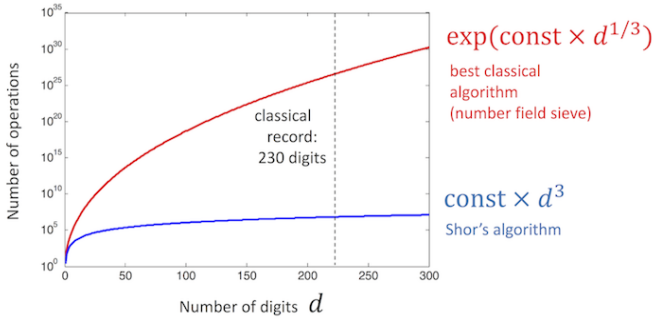

Fig. 10 Shor's vs. traditional quantum computing time complexity

You can see that Shor's algorithm cuts the computational complexity in Figure 10 exponentially.

VI. QFT IN OTHER ALGORITHMS

Shor's - The Quantum Fourier algorithm fits into other algorithms, such as Shor's algorithm, by taking advantage of efficient period-finding, which is important for factoring large prime numbers. With that said, Shor's algorithm utilizes the Quantum Fourier transform algorithm to find the period of a function. By factoring the large prime numbers into their prime factors, Shor's algorithm breaks the problem down into finding the period and applying the classical computations to extract the factors. To begin, Shor's algorithm starts by initializing the quantum computer by setting up two registers, an input register that holds the prime number to be factored and the output register which holds the final result. Next, the registers are put into superposition so that the quantum computer can consider multiple possible values simultaneously on all qubits. Afterward, the Quantum Fourier transform algorithm is applied to the input register, where it operates on all qubits that are in superposition and transforms them into their new representation, the frequency domain, which allows for encoding of the periodicity of the input number. Furthermore, the output register is then measured, which collapses the quantum states into a specific value which is the measurement result of the periodic value related to the factors of the prime number used in the input register. Lastly, classical computations are used to perform mathematical calculations on the output register to extract the factors of the input number. In summary, Shor's algorithm utilizes the Quantum Fourier algorithm to efficiently find the period of a function which allows for factoring large numbers, and then relies on classical computing to extract the factors from the output register at the end of the algorithm.

VII. Challenges in using QFT

With the current state of quantum computers and their algorithms, there is room for error when implementing the QFT circuit. A major problem with the QFT algorithm is that it is a "coherent" protocol where all of the qubits in the circuit need to be in a superposition of states simultaneously. Therefore, it assumes that the qubits are perfect. This is an issue because nothing is 100% perfect in quantum computing. It is all based on probability which means that there is always a chance that the state of a qubit will change. These unwanted changes can be due to noise, dephasing, and/or loss. This will result in the output of the QFT algorithm to be incorrect.

VIII. Potential future uses

The Quantum Fourier Transform can be used for future medical discoveries. Currently, the QFT is still being researched to be implemented in such ways; however, the algorithm's fundamental properties are needed to make such discovery possible. The QFT can be used to break down a molecule into its possible frequencies by calculating the Fourier transform of the molecule. Therefore from a high-level point of view, the QFT will receive a molecule as its input and decompose it to its possible frequencies (vibrational patterns). This will allow researchers to compare different molecules to solvents or proteins, which in turn can simulate how different properties will interact with each other. Then the inverse-QFT can be used to reconstruct the simulated molecule mixture into a new drug target or discovery.

IX. Acknowledgements

All quantum circuits were built using the IBM quantum composer. found at:
https://quantum-computing.ibm.com/

Our Circle notation was completed using the O'reilly circle notation generator:
https://oreilly-qc.github.io/

Latex formulas were created using the free formula editor:
http://www.sciweavers.org/free-online-latex-equation-editor

References

[1] S. M. Metev and V. P. Veiko, *Laser Assisted Microtechnology*, 2nd ed., R. M. Osgood, Jr., Ed. Berlin, Germany: Springer-Verlag, 1998.

[2] P. W. Shor, "Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer," in SIAM Journal on Computing, vol. 26, no. 5, pp. 1484-1509, 1997.

[3] M. Schuld, I. Sinayskiy, and F. Petruccione, "An introduction to quantum machine learning," in Contemporary Physics, vol. 56, no. 2, pp. 172-185, 2015.

[4] S. Lloyd, "Universal quantum simulators," in Science, vol. 273, no. 5278, pp. 1073-1078, 1996.

[5] M. A. Nielsen and I. L. Chuang, Quantum Computation and Quantum Information, Cambridge University Press, 2010.

[6] A. Ekert and P. Hayden, "Quantum bits and quantum secrets," in Contemporary Physics, vol. 50, no. 2, pp. 103-123, 2009.

[7] "Fourier-Transform,"DeepAI, https://deepai.org/machine-learning-glossary-and-terms/fourier-transform (accessed Jul. 6, 2023).

[8] P. B. Bryan, "Fourier transform, applied (1): Introduction to the frequency domain," Medium, https://towardsdatascience.com/the-fourier-transform-1-ca31adbfb9ef (accessed Jul. 6, 2023).

[9] "Discrete fourier transform - simple step by step," YouTube, https://www.youtube.com/watch?v=mkGsMWi_j4Q (accessed Jul. 6, 2023).

[10] R. Cleve, A. Ekert, C. Macchiavello, and M. Mosca, "Quantum algorithms revisited," Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences, vol. 454, no. 1969, pp. 339–354, 1998.

[11] D. Mermin, Quantum Computer Science: An Introduction. Cambridge University Press, 2007.

[12] J.Hui, "QC-quantum Fourier transform," Medium, https://jonathan-hui.medium.com/qc-quantum-fourier-transform-45436f90a43 (accessed Jul. 6, 2023).

[13] P. W. Shor, "Algorithms for quantum computation: Discrete logarithms and factoring," in Proceedings 35th Annual Symposium on Foundations of Computer Science, Santa Fe, NM, USA, 1994, pp. 124-134, doi: 10.1109/SFCS.1994.365700.