Shor's Algorithm and Its Impact On Present-Day Cryptography

Research Capstone (Math 4020)

Alexandra Veliche

December 6, 2018

Faculty Consultant: Professor Christopher King

In this project, I will be exploring the mathematics used in the field of quantum computation, as well as its applications to cryptography. I will be focusing on the quantum Fourier transform and its application in Shor's Algorithm. This algorithm can be used to factor large integers - a classically infeasible problem upon which many present-day cryptosystems rely. I will analyze the roots of unity used in Shor's algorithm and visualize the qubit transformations involved throughout.

Contents

1 Motivation

As a mathematics major and computer science minor, I am interested in the connection between mathematics and theoretical computer science, particularly in the field of cryptography. I have been fascinated with cryptography since middle school, and have studied both private-key and public-key encryption for various projects as part of coursework, as well as on my own. Since I have studied both private and public key cryptography in the past, and have had experience with their applications on my cybersecurity co-op, learning about quantum computing would complement what I know by providing me with a better understanding of the modern-day challenges and fears for the future involved in keeping information secure and communication private. While I have encountered encryption algorithms that have been broken or are still in use, I am not familiar with algorithms in the quantum field, which concerns itself with the future of cryptography. For this reason, I have chosen this topic in order to broaden and deepen my understanding of present-day cryptography. Quantum computation relies heavily on linear algebra, physics, and some complexity theory, and while I have taken Linear Algebra, Physics, Algorithms, and Theory of Computation, I was required to study these subjects more in depth.

2 Approach

In order to create a foundation for further exploration of the field, I have been studying the basics of quantum computation, as well as some topics of linear algebra, number theory, and physical and algorithmic concepts underlying it. I have studied the quantum Fourier transform and its implementation in Shor's algorithm. The purpose of this paper is to clearly explain these concepts and make them accessible to those who have never before encountered this field. As part of this, I have focused on providing simple examples, making connections to more familiar concepts (such as group theory) and visualizing the processes involved throughout. A secondary goal is to provide a sense of why this field is important, by describing how it impacts modern cryptography.

3 The Factoring Problem

Factoring a given number may not seem like such a difficult task at first glance, but when the number is a few hundred digits long and is the product of two very large primes, the problem becomes quite difficult. Even using the fastest algorithms on the fastest computers, the search for factors becomes infeasible, because as the number becomes larger, the number of operations required to factor it increases exponentially. The fastest classical algorithm to date is the General Number Field Sieve (GNFS) algorithm [9], which has an asymptotic running time of $O(e^{c(log\ n)^{1/3}(log\ log\ n)^{2/3}})$, exponential in the length of the given number n. As a more practical example, in 2010 researchers managed to factor a 232-digit (768-bit) RSA key using hundreds of machines in the span of two years. They estimated that a 1024-bit number would be a thousand times more difficult, even with technological advances [7]. At the time of this writing, no algorithm has been found that can factor any integer in polynomial time, and it is predominantly thought that no such algorithm exists [2]. [1] In the quantum world, however, this is another matter entirely. In 1995, Peter Shor formulated an algorithm that could solve the factoring problem in polynomial time [3]. But until now, no quantum

[1] If a polynomial-time algorithm were to be found, this would solve the Millennium Prize problem of "P vs. NP", because it is known that factoring is an NP-problem.

computer powerful enough to run it has been built, so the factoring problem remains infeasible for the time being.

4 Introduction to Quantum Computing

In order to understand the way Shor's algorithm works, we first describe some basic concepts of quantum computing. In classical computers, the basic unit of information is the binary digit, or bit, which can have a value of either 0 or 1. In quantum computing, the analog of the bit is the quantum bit, or qubit; but instead of having only two possible states, a qubit can be in the basis states $|0\rangle, |1\rangle$, or any linear combination of the two $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$, where $\alpha, \beta \in \mathbb{C}$. This linear combination is known as a *superposition* of quantum states, and has the normalization condition $|\alpha|^2 + |\beta|^2 = 1$ [2].

An n-qubit system would have 2^n computational basis states $|0\rangle, |1\rangle, ...|2^n - 1\rangle$, so that any quantum state would be of the form

$$|\psi\rangle = \alpha_0|0\rangle + \alpha_1|1\rangle ... + \alpha_{2^n-1}|2^n - 1\rangle = \sum_{j=0}^{2^n-1} \alpha_j|j\rangle, \text{ where } \alpha_j \in \mathbb{C}, \ \forall\ 0 \leq j \leq 2^n - 1.$$

For example, a 2-qubit system would have basis states $|0\rangle, |1\rangle, |2\rangle, |3\rangle$. Since each qubit in the system can be in one of two states, these basis states can be rewritten in terms of the states of each qubit as $|00\rangle, |01\rangle, |10\rangle, |11\rangle$.

Because any arbitrary state $|\psi\rangle$ is a linear combination of the basis states, it can be thought of as a vector. The *bra-ket* notation for the states is known as the *Dirac notation* (see terminology section) [4]. With this correspondence to vectors, we have the following definition, which will be used from now on [4, 5]:

Definition: Let V and W be complex vector spaces of dimensions n and m, respectively. Their *tensor product* $V \otimes W$ is a complex vector space of dimension nm, whose elements are spanned by the tensor products of vectors from V and W. Let $|v\rangle \in \mathbb{C}^m, |w\rangle \in \mathbb{C}^n$ be two arbitrary states. Then their tensor product is given by $|v\rangle \otimes |w\rangle$, also written as $|v\rangle|w\rangle$ or $|vw\rangle$.

(The formal definition and properties of the tensor product is beyond the scope of this paper, as only tensors of vectors are used.) To get more of an intuition about how this works, consider the 2-qubit system as an example: here the basis states are $|0\rangle$ and $|1\rangle$. These can be represented by vectors as follows

$$|0\rangle = 1|0\rangle + 0|1\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \text{ and } |1\rangle = 0|0\rangle + 1|1\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix}.$$

The tensor product of these two basis states can be found in the following manner:

$$|0\rangle \otimes |1\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \otimes \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \begin{bmatrix} 0 \\ 1 \end{bmatrix} \\ 0 \begin{bmatrix} 0 \\ 1 \end{bmatrix} \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} = |01\rangle.$$

Notice that in this case we obtained a vector of length 4 from two vectors of length 2, and that the resulting

state $|01\rangle$ corresponds to the state second basis state $|1\rangle$ of the 4-qubit system with basis states $|0\rangle, |1\rangle, |2\rangle, |3\rangle$. Hence we have a transformation from $(\mathbb{C}^2)^{\otimes 2}$ to $\mathbb{C}^4$ [1].

Now we introduce a useful representation for a single qubit: Since a qubit can be in any of an infinite number of possible states, it can be visually represented by a point on the *Bloch sphere* shown below [2]:

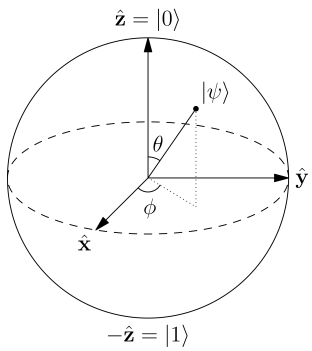

Figure 1: Bloch sphere

Here the poles are the basis states $|0\rangle$ and $|1\rangle$. Since the phase of the qubit lies on the sphere with these poles, it can be written in terms of spherical coordinates as follows [1]:

$$|\psi\rangle = cos\Big(\frac{\theta}{2}\Big)|0\rangle + e^{i\phi}sin\Big(\frac{\theta}{2}\Big)|1\rangle, \text{ where } \theta, \phi \in \mathbb{R}.$$

Here the amplitudes of the basis states for this qubit state are in terms of the angles defining the position on the sphere. This alternate representation is important to keep in mind for the discussion of roots of unity in later sections.

5 Quantum Fourier Transform

The quantum Fourier transform (QFT) is the key component of Shor's algorithm and other useful quantum algorithms. Its classical analog is the discrete Fourier transform (DFT), a linear transformation that converts a sequence of complex numbers, often denoted as the input vector $x = (x_0, x_1, ...x_{n-1})$, into another sequence of complex numbers, denoted by the output vector $y = (y_0, y_1, ...y_{n-1})$, by the following formula [2, 10]:

$$y_k = \frac{1}{\sqrt{n}} \sum_{j=0}^{n-1} x_j e^{2\pi i j k/n}, \text{ where } 0 \leq k \leq n-1.$$

The quantum version is also a linear transformation, but instead of acting on a simple vector $x \in \mathbb{C}^n$, it transforms a quantum state $|x\rangle$ into another state $|y\rangle$. Given an input state that is a superposition of the 2^n basis states of an n-qubit system, we have that

$$|x\rangle = \sum_{j=0}^{2^n-1} x_j |j\rangle, \text{ where } 0 \leq j \leq 2^n - 1.$$

The QFT maps the input state to the state $|y\rangle = \sum_{j=0}^{2^n-1} y_j|j\rangle$ by the same formula as above [2]:

$$y_k = \frac{1}{\sqrt{2^n}} \sum_{j=0}^{2^n-1} x_j e^{2\pi ijk/2^n}, \text{ where } 0 \leq k \leq 2^n - 1.$$

The inverse quantum Fourier transform (QFT^{-1}), is defined similarly, but with the output basis states given by the *adjoint* of the previous transformation:

$$y_k = \frac{1}{\sqrt{2^n}} \sum_{j=0}^{2^n-1} x_j e^{-2\pi ijk/2^n}, \text{ where } 0 \leq k \leq 2^n - 1.$$

Recall that the adjoint of a complex matrix is the complex conjugate composed with the transpose (see terminology section for details). Hence, because a complex number can be considered as a single-entry matrix, its transpose is itself, so that its adjoint is its complex conjugate. In polar form, the complex number $a + bi = re^{i\phi}$ has the complex conjugate $a - bi = re^{-i\phi}$ [1].

The transformation of a quantum state under the QFT can be decomposed into the transformation of the computational basis states of the system: for an n-qubit system, the basis states are $|0\rangle, |1\rangle, ...|2^n - 1\rangle$. Thus given an arbitrary basis state $|j\rangle$, where $0 \leq j \leq 2^n - 1$, the QFT transforms it by the following:

$$|j\rangle \to \frac{1}{\sqrt{2^n}} \sum_{k=0}^{2^n-1} e^{2\pi ijk/2^n}|k\rangle.$$

Here, the state $|k\rangle$ can be written in its binary form $k = k_1 \cdot 2^{n-1} + k_2 \cdot 2^{n-2} + ... + k_n \cdot 2^0$, while its binary fraction is given by $k \cdot 2^{-n} = k_1 \cdot 2^{-1} + k_2 \cdot 2^{-2} + ... + k_n 2^n = 0.k_1k_2...k_n$ [2]. By the properties of the Dirac notation, the binary form of the state $|k\rangle = |k_1k_2...k_n\rangle$ is the same as the tensor product $|k_1\rangle \otimes |k_2\rangle \otimes ... \otimes |k_n\rangle$. Using these facts, the QFT transformation of a basis state can be rewritten as the following product [2]:

$$\begin{aligned}
|j\rangle \to \frac{1}{\sqrt{2^n}} \sum_{k=0}^{2^n-1} e^{2\pi ijk/2^n}|k\rangle &= \frac{1}{\sqrt{2^n}} \sum_{k_1=0}^{1} \sum_{k_2=0}^{1} ... \sum_{k_n=0}^{1} e^{2\pi ij0.k_1k_2...k_n}|k\rangle \\
&= \frac{1}{\sqrt{2^n}} \sum_{k_1=0}^{1} \sum_{k_2=0}^{1} ... \sum_{k_n=0}^{1} e^{2\pi ij(\sum_{l=1}^{n} k_l \cdot 2^{-l})}|k_1k_2...k_n\rangle \\
&= \frac{1}{\sqrt{2^n}} \sum_{k_1=0}^{1} \sum_{k_2=0}^{1} ... \sum_{k_n=0}^{1} \bigotimes_{l=1}^{n} e^{2\pi ijk_l2^{-l}}|k_l\rangle \\
&= \frac{1}{\sqrt{2^n}} \bigotimes_{l=1}^{n} \Big(\sum_{k_l=0}^{1} e^{2\pi ijk_l2^{-l}}|k_l\rangle \Big) \\
&= \frac{1}{\sqrt{2^n}} \bigotimes_{l=1}^{n} \Big(|0\rangle + e^{2\pi ij2^{-l}}|1\rangle \Big) \\
&= \frac{1}{\sqrt{2^n}} \Big(|0\rangle + e^{2\pi i0.j_n}|1\rangle \Big) \Big(|0\rangle + e^{2\pi i0.j_{n-1}j_n}|1\rangle \Big) ... \Big(|0\rangle + e^{2\pi i0.j_1j_2...j_n}|1\rangle \Big)
\end{aligned}$$

This product is used as often as the formal definition of the QFT, as it clearly shows the transformation of each qubit in the system as a superposition of states. The significance of this product is explained in the following section.

6 QFT Circuit Representation

In preparation for representing the QFT as a circuit, we introduce some important quantum logic gates:

A simple quantum gate is the *Pauli-X* gate, also known as the *NOT* gate [5]. It operates on a single qubit in the following manner: given a quantum state $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$, the gate switches the $|0\rangle$ and $|1\rangle$ basis states to give the resulting state $X(|\psi\rangle)=\alpha|1\rangle + \beta|0\rangle$. For a single qubit system, this gate can be represented by the matrix

$$X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}.$$

In terms of the Bloch sphere representation, this is equivalent to a rotation by π around the x-axis.

Unlike the *Pauli-X* gate, the *SWAP* gate acts on two qubits at once, swapping one with the other [5]. Since the basis states for a 2-qubit system are represented by $|00\rangle, |01\rangle, |10\rangle, |11\rangle$, this gate can be represented by the matrix

$$SWAP = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

.

Another important gate is the *Hadamard gate* represented by the matrix

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

This sends $|0\rangle \mapsto \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$ and $|1\rangle \mapsto \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$, and is equivalent to a rotation around the y-axis by $\frac{\pi}{2}$ followed by a reflection in the x-y plane [5]. We will focus on the transformation of the basis state $|0\rangle$, as this is used in the QFT circuit. For a 2-qubit system with basis states $|00\rangle, |01\rangle, |10\rangle, |11\rangle$, the Hadamard gate acts on each qubit by $H \otimes H$ and sends $|00\rangle = |0\rangle \otimes |0\rangle$ to the following:

$$\begin{aligned}(H \otimes H)(|0\rangle \otimes |0\rangle) &= H|0\rangle \otimes H|0\rangle \\ &= \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \otimes \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) \\ &= \frac{1}{2}(|00\rangle + |01\rangle + |10\rangle + |11\rangle) \\ &= \frac{1}{2}(|0\rangle + |1\rangle + |2\rangle + |3\rangle) \text{ in terms of the alternate basis.}\end{aligned}$$

In general, for an n-qubit system, the Hadamard gate acts on the basis state $|00...0\rangle$ by the following:

$$H^{\otimes n}(|0\rangle^{\otimes n}) = (\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle))^{\otimes n} = \frac{1}{\sqrt{2^n}} \sum_{j=0}^{2^n-1} |j\rangle.$$

A third important gate, is the *controlled-phase gate* represented by the matrix

$$R_k = \begin{bmatrix} 1 & 0 \\ 0 & e^{\frac{2\pi i}{2^k}} \end{bmatrix},$$

where $\frac{2\pi i}{2^k}$ is the phase shift [5]. This acts on a single qubit, and only changes the amplitude of the $|1\rangle$ component of the phase state.

Using these basic quantum gates, the QFT on a given basis state $|j\rangle$ can be represented with the following circuit, in which each binary digit $|j_i\rangle$ of the basis state $|j\rangle = |j_1 j_2 ... j_n\rangle = |j_1\rangle \otimes |j_2\rangle \otimes ... \otimes |j_n\rangle$ is passed through a Hadamard gate and a series of controlled-phase gates. The controlled-phase gates for a single digit is determined by the states of the digits after it in the binary form for $|j\rangle$. After each digit passes through its series of gates, the resulting states for each pair $|j_i\rangle$ and $|j_{n-i}\rangle$ are swapped so that the state $|j_i\rangle$ corresponds to the transformed state $\frac{1}{\sqrt{2}}\Big(|0\rangle + e^{2\pi i 0.j_i ... j_{n-1} j_n}|1\rangle\Big)$. In the product form of the QFT, the swapping has no overall effect, but in the circuit, the swapping matches each initial digit state with its resulting state. This swapping is explained further in the example below. The QFT circuit for an n-qubit system is given by the following figure:

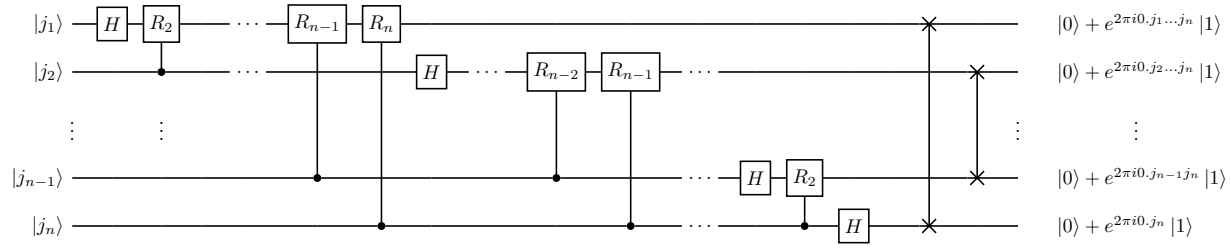

Figure 2: General QFT Circuit

Example:

To better understand how the QFT works, consider the 2-qubit system, with the basis state $|j\rangle = |j_1 j_2\rangle$ as the input. Then the QFT circuit is given by the following figure:

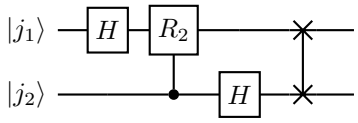

Figure 3: QFT Circuit for 2-Qubit System

As seen previously, the basis states for this system are $|0\rangle, |1\rangle, |2\rangle, |3\rangle$. Since each qubit in the system has the state $|0\rangle$ or $|1\rangle$, the basis states can be rewritten as $|00\rangle, |01\rangle, |10\rangle, |11\rangle$. Thus, any arbitrary state is of the form $|\psi\rangle = \alpha_0|00\rangle + \alpha_1|01\rangle + \alpha_2|10\rangle + \alpha_3|11\rangle$.

Using the definition of the QFT, the input state $|j\rangle$ is transformed into the following:

$$QFT(|j\rangle) = \frac{1}{\sqrt{2^2}} \sum_{k=0}^{2^2-1} e^{\frac{2\pi ijk}{2^2}} |k\rangle = \frac{1}{2} \sum_{k=0}^{3} e^{\frac{\pi ijk}{2}} |k\rangle = \frac{1}{2} \Big(|0\rangle + e^{2\pi i 0.j_2}|1\rangle\Big) \Big(|0\rangle + e^{2\pi i 0.j_1 j_2}|1\rangle\Big).$$

Using the QFT circuit, the following process is performed: First the Hadamard gate is applied to the first qubit state $|j_1\rangle$. For the cases where $|j_1\rangle = |0\rangle$ and $|j_1\rangle = |1\rangle$, then we obtain the following:

$$H(|0\rangle) = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \text{ and } H(|1\rangle) = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle).$$

These possible results can be combined into the following form, using binary representation:

$$H(|j_1\rangle) = \frac{1}{\sqrt{2}}(|0\rangle + (-1)^{j_1}|1\rangle) = \frac{1}{\sqrt{2}}(|0\rangle + e^{\frac{2\pi i j_1}{2}}|1\rangle) = \frac{1}{\sqrt{2}}(|0\rangle + e^{2\pi i 0.j_1}|1\rangle).$$

Next, the controlled-phase gate is applied to the transformed qubit state:

$$R_2(H|j_1\rangle) = \begin{bmatrix} 1 & 0 \\ 0 & e^{\frac{2\pi i j_2}{2^2}} \end{bmatrix} \Big(\frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ e^{2\pi i 0.j_1} \end{bmatrix} \Big) = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ e^{2\pi i 0.j_1 + \frac{2\pi i j_2}{4}} \end{bmatrix} = \frac{1}{\sqrt{2}} \Big(|0\rangle + e^{2\pi i 0.j_1 j_2}|1\rangle\Big).$$

Notice that here the controlled-phase gate includes the state of $|j_2\rangle$, because it is controlled by $|j_2\rangle$ in the circuit. Combining the individual qubit transformations, we obtain the following result:

$$\begin{aligned} QFT(|j_1 j_2\rangle) &= R_2(H|j_1\rangle) \otimes H|j_2\rangle \\ &= \frac{1}{2} \Big(|0\rangle + e^{2\pi i 0.j_1 j_2}|1\rangle\Big) \Big(|0\rangle + e^{2\pi i 0.j_2}|1\rangle\Big) \end{aligned}$$

Comparing this result from the QFT circuit to the result obtained from the QFT equation, we notice that the resulting state for each individual qubit is switched. In order to match the circuit result exactly with the definition, the swapping is implemented in the circuit at the very end.

Note that the QFT equation can be written in terms of the primitive nth root of unity: Let $\omega := e^{\frac{\pi i}{2}} = i$. Then the 4^{th} roots of unity are given by $e^{\frac{\pi ik}{2}}$, for $1 \leq k \leq 4$. These are $\omega^2 = -1, \omega^3 = -i, \omega^4 = 1$. Using these roots of unity, the QFT for this 2-qubit system can be written in the following matrix form:

$$QFT_2 = \frac{1}{2} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & \omega & \omega^2 & \omega^3 \\ 1 & \omega^2 & 1 & \omega^2 \\ 1 & \omega^3 & \omega^2 & \omega \end{bmatrix} = \frac{1}{2} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & i & -1 & -i \\ 1 & -1 & 1 & -1 \\ 1 & -i & -1 & i \end{bmatrix}.$$

Here the columns of the matrix correspond to the possible cases for the state of the qubits in the system: $|00\rangle, |01\rangle, |10\rangle, |11\rangle$, respectively. Notice that these roots of unity form a group under multiplication, of order 4: $\langle\omega\rangle = \{1, \omega, \omega^2, \omega^3\}$. Since $gcd(1, 4) = 1$ and $gcd(3, 4) = 1$, both ω and ω^3 are generators of this group. A visual representation of this group is shown below:

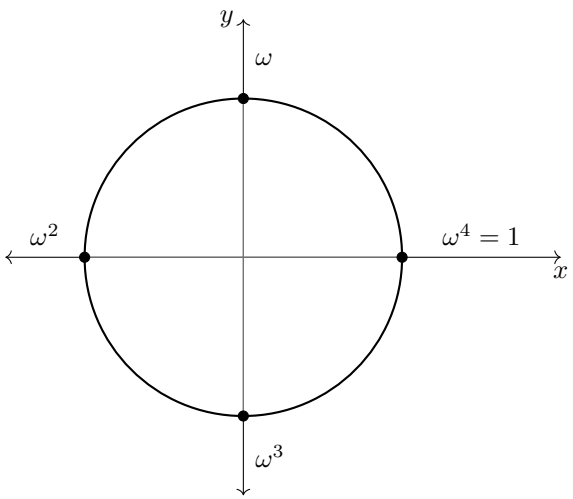

Figure 4: Roots of Unity for 2-Qubit System

Similarly, the QFT for a 3-qubit system can be represented as the following matrix:

$$QFT_3 = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & \omega & \omega^2 & \omega^3 & \omega^4 & \omega^5 & \omega^6 & \omega^7 \\ 1 & \omega^2 & \omega^4 & \omega^6 & 1 & \omega^2 & \omega^4 & \omega^6 \\ 1 & \omega^3 & \omega^6 & \omega & \omega^4 & \omega^7 & \omega^2 & \omega^5 \\ 1 & \omega^4 & 1 & \omega^4 & 1 & \omega^4 & 1 & \omega^4 \\ 1 & \omega^5 & \omega^2 & \omega^7 & \omega^4 & \omega & \omega^6 & \omega^3 \\ 1 & \omega^6 & \omega^4 & \omega^2 & 1 & \omega^6 & \omega^4 & \omega^2 \\ 1 & \omega^7 & \omega^6 & \omega^5 & \omega^4 & \omega^3 & \omega^2 & \omega \end{bmatrix}$$

In this case, the group of roots of unity is generated by $\omega := e^{\frac{\pi i}{4}}$, which has order 8. The visual representation of this group is given below:

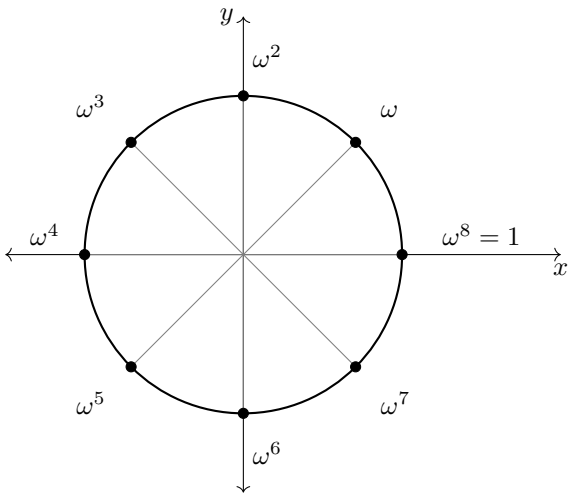

Figure 5: Roots of Unity for 3-Qubit System

7 Shor's Algorithm

Now that we have discussed the basic components, we can put them together into Shor's algorithm. This algorithm consists of two parts: (I) a classical algorithm for reducing the factoring problem to order-finding, and (II) a quantum order-finding subroutine that uses the QFT, and is called by part (I) [5, 11, 12]. First we define the order-finding problem:

Problem: Given integers $x, N \in \mathbb{Z}_+$ such that $x < N$ and $gcd(x, N) \neq 1$, find the smallest $r \in \mathbb{Z}_+$ such that $x^r = 1(mod\ N)$. This r is the *order* of the integer x modulo N (see Terminology).

Before introducing the algorithm, we first state some number theory definitions and theorems used in part (I) of the algorithm (without proof, as this is beyond the scope of this paper) [2]:

(1) **Theorem:** Let $N \in \mathbb{Z}_+$ be a composite number and $x \in \mathbb{Z}_+$ in the range $1 \leq x \leq N$ be a non-trivial solution to the equation $x^2 = 1(mod\ N)$ (i.e. $x \neq \pm 1(mod\ N)$). Then at least one of $gcd(x+1, N)$ and $gcd(x-1, N)$ is a non-trivial factor of N.

(2) **Theorem:** Suppose $N \in \mathbb{Z}_+$ is an odd composite with prime factorization $N = p_1^{\alpha_1} \cdot p_2^{\alpha_2} \cdot ... \cdot p_k^{\alpha_k}$. Let $x \in \mathbb{Z}_N^*$ be a random, uniformly-chosen element with order r. Then we have the following probability:

$$P(r \text{ odd or } x^{\frac{r}{2}} = -1(mod\ N)) \leq \frac{1}{2^k}$$

.

(3) **Euclidean Algorithm:** Given two integers $a, b \in \mathbb{Z}_+$, their greatest common denominator can be found by the following recursive algorithm [12]:

$$gcd(a, b) = \begin{cases} b, & \text{if } a \equiv 0\ (mod\ b) \\ gcd(a, a\ mod\ b), & \text{otherwise} \end{cases}$$

Definition: Given a set of integers $a_0, a_1, ...a_n \in \mathbb{Z}_+$, the *finite simple continued fraction* they define is given by:

$$a_0 + \cfrac{1}{a_1 + \cfrac{1}{a_2 + \cfrac{1}{... + \cfrac{1}{a_n}}}} =: [a_0\ a_1\ a_2\ ...\ a_n].$$

These integers are known as the *partial denominators* of this fraction [8]. The *continued fractions algorithm* is a process of finding these partial denominators from a given fraction, and is illustrated by the example below:

Example:

Consider the fraction $\frac{43}{30}$. The partial denominators of this fraction can be found by repeatedly rewriting each fraction as a partial fraction, as shown below:

$$\frac{43}{30} = 1 + \frac{13}{30} = 1 + \frac{1}{\frac{30}{13}} = 1 + \frac{1}{2 + \frac{4}{13}} = 1 + \frac{1}{2 + \frac{1}{\frac{13}{4}}} = 1 + \frac{1}{2 + \frac{1}{3 + \frac{1}{4}}} = [1\ 2\ 3\ 4].$$

Hence, the partial denominators of this fraction are 1, 2, 3, and 4.

Now we introduce the main process of Shor's algorithm given by part (I):

I. Classical algorithm for the reduction of factoring to order-finding:

Input: odd composite $N \in \mathbb{Z}_+$
Output: non-trivial factors of N

1. Choose a random $a \in \mathbb{Z}_+$ such that $a < N$.
2. Compute $gcd(a, N)$ using the Euclidean Algorithm (4) above.
3. If $gcd(a, N) \neq 1$, then return the non-trivial factor $gcd(a, N)$.

   Else, use the subroutine (II) to find r, the order of a modulo N.
4. If r odd or $a^{\frac{r}{2}} \equiv -1 \ (mod\ N)$, then go back to 1. and choose a different a.

   Else, return the non-trivial factor(s) given by $gcd(a^{\frac{r}{2}} + 1, N)$ and/or $gcd(a^{\frac{r}{2}} - 1, N)$.

In this algorithm, the problem of factoring a given integer N is reduced to the problem of finding the order of an element modulo N, and using this to find the non-trivial factors of N. This is possible with the properties given by the theorems stated above. Here, Theorem (1) implies that if the conditions for r in step (4) do not hold, there exist the non-trivial factors described. By Theorem (2), the probability that r meets the conditions in the "if" clause is quite small, depending on the number of prime factors of N. Overall, there is a high probability that the algorithm will find a suitable a eventually, so the algorithm halts (the formal complexity analysis of this algorithm is beyond the scope of this paper, however).

The quantum order-finding subroutine called in step (3) is given below. Here we use *registers* to group the input qubits into two sets, each treated as a composite system [4]. Let R_1 denote the collection of t qubits given as input (ii) and R_2 denote the collection of l given as input (iii). The circuit representation for this subroutine algorithm is given below [2]:

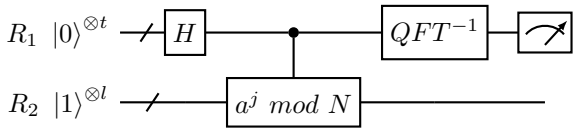

Figure 6: Quantum Order-Finding Subroutine Circuit

II. Quantum subroutine algorithm for order-finding:

Input:

(i) black box transformation $U_{a,N} : |j\rangle|k\rangle \to |j\rangle|a^j k \ (mod\ N)\rangle$ for $a \in \mathbb{Z}_+$ where $gcd(a, N) = 1$

(ii) t qubits intialized to $|0\rangle$, where $t := 2l + 1 + \lceil log(2 + \frac{1}{2\epsilon})\rceil$ and $l :=$ number of bits in N

(iii) l qubits intialized to $|1\rangle$

Output: order of a modulo N

1. Apply the Hadamard gate to each of the t qubits in R_1:

$$H^{\otimes t}(|0\rangle^{\otimes t}) = \Big(\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)\Big)^{\otimes t} = \frac{1}{\sqrt{2^t}} \sum_{j=0}^{2^t-1} |j\rangle.$$

2. Apply the black box transformation $U_{a,N}$ to each of the l qubits in R_2:

$$U_{a,N}(\frac{1}{\sqrt{2^t}} \sum_{j=0}^{2^t-1} |j\rangle|1\rangle) = \frac{1}{\sqrt{2^t}} \sum_{j=0}^{2^t-1} |j\rangle|a^j mod\ N\rangle = \frac{1}{\sqrt{r2^t}} \sum_{s=0}^{r-1} \sum_{j=0}^{2^t-1} e^{\frac{2\pi ijs}{r}} |j\rangle|u_s\rangle =: |\psi\rangle,$$

where $|u_s\rangle = \frac{1}{\sqrt{r}} \sum_{s=0}^{r-1} e^{\frac{-2\pi ijs}{r}} |a^j mod\ N\rangle$.

3. Apply the inverse QFT to R_1:

$$\begin{aligned} QFT^{-1}(|\psi\rangle) &= \frac{1}{\sqrt{r2^t}} \sum_{s=0}^{r-1} \sum_{j=0}^{2^t-1} e^{\frac{2\pi ijs}{r}} \Big(\frac{1}{\sqrt{2^t}} \sum_{k=0}^{2^t-1} e^{\frac{-2\pi ijk}{2^t}} |k\rangle\Big)|u_s\rangle \\ &= \frac{1}{2^t\sqrt{r}} \sum_{s=0}^{r-1} \Big(\sum_{j,k=0}^{2^t-1} e^{2\pi ij(\frac{s}{r} - \frac{k}{2^t})}\Big)|k\rangle|u_s\rangle \\ &= \frac{1}{\sqrt{r}} \sum_{s=0}^{r-1} \Big(\frac{1}{2^t} \sum_{j,k=0}^{2^t-1} e^{2\pi ij(\frac{s}{r} - \frac{k}{2^t})}|k\rangle\Big)|u_s\rangle \qquad (*) \\ &= \frac{1}{\sqrt{r}} \sum_{s=0}^{r-1} \sum_{k=0}^{2^t-1} \alpha_{k,s}|k\rangle|u_s\rangle \end{aligned}$$

where $\alpha_{k,s} := \frac{1}{2^t} \sum_{j=0}^{2^t-1} e^{\frac{2\pi ij}{2^t}(\frac{s2^t}{r} - k)}$.

4. Measure R_2 to choose an s, then measure R_1 to obtain a value $\widetilde{k}$ for this s.

5. Apply the continued fractions algorithm to $\frac{\widetilde{k}}{2^t}$ to find the partial denominators $r_0, r_1, ...r_l$, and test r_i at each step to find the order r.

In this algorithm, the black box $U_{a,N}$ given as input (i) is a unitary transformation that changes the second state of the tensor product input, depending on the $a \in \mathbb{Z}_+$ chosen in part (I) of the algorithm [2]. Because the output of this function can no longer be written as a product of separate states, the two states have become *entangled*. The term *black box* indicates that the inner workings of this subroutine are hidden, and that the algorithm is only interested in its input and output. In practice, however, this can be implemented with an additional subroutine algorithm. This transformation is significant because this

subroutine is a specific case of the *phase estimation* procedure. This procedure performs the following: given a unitary matrix U and an eigenvector $|u\rangle$, it estimates the phase ϕ of the eigenvalue $e^{2\pi i\phi}$. In this subroutine algorithm, the desired phase is $\frac{s}{r}$, where r is the order to be determined. The $2l+1$ qubits given as input (ii) are just enough to represent the value of N^2, as the length of N in binary is given by $l = \lceil log(N) \rceil$ [2]. The remaining $\lceil log(2 + \frac{1}{2\epsilon}) \rceil$ are used to estimate the phase depending on the desired number of digits of accuracy and the desired probability of success (the derivation of this value is beyond the scope of this paper this paper). The l qubits given as input (iii) are used for computation purposes in modifying the state of the qubits in the first register.

In step (1) of the subroutine algorithm, the Hadamard gate is applied to the collection of t qubits set to $|0\rangle$, in the same manner as shown in the previous section.

In step (2), the black box $U_{a,N}$ is applied to each of the tensor products of the transformed t qubits and l qubits. Since the black box only affects the state of the second qubit in the product, it is described as being applied only to the second register R_2. However, it modifies the second qubit based on the state of the first, so that these become entangled; so in effect, it is acting on both registers at once. Hence, the transformations in steps (1) and (2) can be summarized in the following way:

$$U_{a,N}((H^{\otimes t} \otimes I)(|0\rangle^{\otimes t} \otimes |1\rangle)) = U_{a,N}\Big(\frac{1}{\sqrt{2^t}} \sum_{j=0}^{2^t-1} |j\rangle \otimes |1\rangle\Big).$$

For ease of reference, the result of this transformation is named $|\psi\rangle$. In step (2), the state of the tensor product is rewritten from being in terms of the output of $U_{a,N}$ to being in terms of the states $|u_s\rangle$. The states $|u_s\rangle$ are the eigenstates of the transformation matrix U defined by $U|x\rangle = |ax \ mod \ N\rangle$, and has corresponding eigenvalues $e^{\frac{-2\pi is}{r}}$ [1, 2]. This transformation changes the summation from being in the basis $\big\langle |1\rangle, |a \ mod \ N\rangle, ...|a^{r-1} \ mod \ N\rangle\big\rangle$ to being in terms of the eigenvectors $\big\langle |u_o\rangle, |u_1\rangle, ...|u_{r-1}\rangle\big\rangle$.

In step (3), the inverse QFT is applied to the previous result $|\psi\rangle$. Since this only affects the state of the first set of qubits in the tensor product, it is in effect being applied only to the first register R_1. The result of this step is in terms of the eigenstate $|u_s\rangle$ and the possible values for the order r. In reality, the quantum computer implementing this algorithm would not perform the formal linear algebra analysis done to obtain the state $|\widetilde{k}\rangle$ [1]. Instead it would use a series of implicit measurements like the following [1, 2]:

Let $\alpha_{k,s} := \dfrac{1}{2^t} \displaystyle\sum_{j=0}^{2^t-1} e^{\frac{2\pi ij}{2^t}(\frac{s2^t}{r} - k)}$

Then the QFT^{-1} equation at $(*)$ can be rewritten as $(*) = \dfrac{1}{\sqrt{r}} \displaystyle\sum_{s=0}^{r-1} \sum_{k=0}^{2^t-1} \alpha_{k,s}|k\rangle|u_s\rangle$

Measure R_2 to choose an s and obtain $: \displaystyle\sum_{k=0}^{2^t-1} \alpha_{k,s}|k\rangle$

Measure R_1 to choose a $0 \leq k \leq 2^t - 1$ as done before.

The probability of choosing the value k in measuring R_1 is determined by $P(k) := |\alpha_{k,s}|^2$,which is represented in the plot below. Here the peak occurs when k is closest to $\frac{s}{r}2^t$, so with a high probability, the machine will choose a value k close to this.

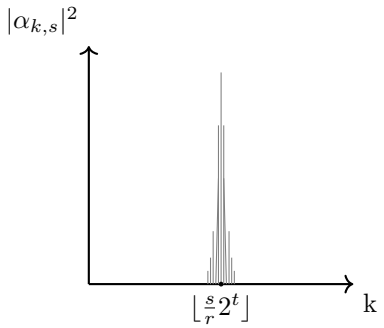

Figure 7: Probability Distribution for Measurement of R_1 After Measurement of R_2

This measurement can also be understood in terms of the 2^t roots of unity of $\alpha_{k,s}$: Let $b + \delta := \frac{s2^t}{r} - k$ be the error in measuring the value k, where b is the integral part and δ is the decimal part of the difference, and denote the root of unity by $\omega := e^{\frac{2\pi i}{2^t}}$. Then $\alpha_{k,s}$ can be rewritten as follows:

$$\alpha_{k,s} = \frac{1}{2^t} \sum_{j=0}^{2^t-1} \omega^{j(b+\delta)}.$$

As an example, consider the case where $b = 1$. Here the roots would be of the form $\omega^{j(b+\delta)}$ for $0 \leq j \leq 2^t - 1$. A visualization of these roots of unity is shown below:

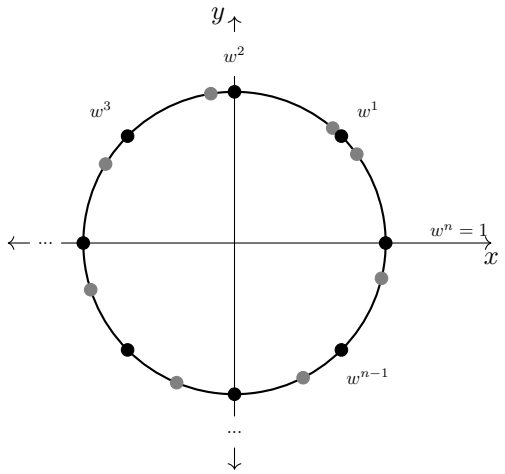

Figure 8: Roots of Unity for Shor's Algorithm

As shown by the black points in the figure above, if $b = 1$ and $\delta = 0$, then the roots of unity are those determined by the exponent j. If $b > 1$ and $\delta = 0$, then the roots are shifted by a factor of ω^b, but are still in the original set of roots. If $\delta \neq 0$, then each root is shifted relative to the exponent j of ω^j, as shown by the gray points in the figure above. In the cases where $\delta = 0$, the sum of the roots will be $\alpha_{k,s} = 1$ if $b = 0$ and 0 otherwise. In the alternative case, there will be some error: $\alpha_{k,s} \simeq 1$ if $b = 0$ and $\alpha_{k,s} \simeq 0$ otherwise.

Finally, in step (4) the algorithm measures R_1 to obtain $|\widetilde{k}\rangle$, the state which best approximates $\lfloor \frac{s}{r} 2^t \rfloor$ [1]. This is used to form the fraction $\frac{\widetilde{k}}{2^t}$, and the continued fractions algorithm is applied to find its partial denominators $[r_0, r_1, ...r_l]$. For each partial denominator r_i, the algorithm tests to see if $a^{r_i} \equiv 1 \; mod \; N$. If

so, the order $r = r_i$ is returned.

To better understand how Shor's algorithm works, consider the following example, in which the non-trivial factors of $N = 21$ are found:

Example:

1. Suppose that the random integer chosen is $a = 13$.
2. Then using the Euclidean algorithm, we find that $gcd(13, 21) = 1$.
3. Since 13 and 21 are relatively prime, we call the quantum subroutine to find the order r such that $13^r \equiv 1(mod\ 21)$.

inputs:
- $U_{13,21}(|j\rangle|k\rangle) = |j\rangle|13^j(mod\ 21)\rangle$.
- Since 21 is 10101 in binary, $l = 5$, and so R_2 contains 5 qubits initialized to $|1\rangle$.
- Suppose that the desired accuracy is $\epsilon = 0.2$. Then $t = 2(5) + 1 + \lceil log(2 + \frac{1}{2(0.2)})\rceil = 12$, and so R_1 contains 12 qubits initialized to $|0\rangle$.

1) Applying the Hadamard gate to R_1, we obtain: $H^{\otimes 12}(|0\rangle^{\otimes 12}) = \frac{1}{2^6}(|0\rangle + |1\rangle + ... + |2^{12} - 1\rangle)$.

2) Applying the black box operation, we obtain the state:

$$U_{13,21}\Big(\frac{1}{2^6}\sum_{j=0}^{2^{12}-1}|j\rangle|1\rangle\Big) = \frac{1}{2^6}(|0\rangle|1\rangle + |1\rangle|13\rangle + |2\rangle|1\rangle + |3\rangle|13\rangle + ...).$$

3) Apply QFT^{-1} to the current state and measure R_2.

4) Then measure R_1. Suppose that the measurement of R_1 results in 2048.

5) Performing the continued fractions algorithm on $\frac{2048}{4096}$, we obtain the only partial denominator 2. Since $13^2 \equiv 1\ (mod\ 21)$, we return the order $r = 2$.

4. Since $r = 2$ is not odd and $13^{\frac{2}{2}} = 13 \not\equiv -1(mod\ 21)$, we return the factors:
$gcd(13 + 1, 21) = 7$ and $gcd(13 - 1, 21) = 3$.

8 Conclusion

Now that we have seen how Shor's algorithm factors a given number in quantum polynomial time, we discuss the impact this could have on modern cryptography. The commonly-used RSA cryptosystem relies on the difficulty of factoring large integers of the form $N = p \cdot q$, where p and q are two large primes [14]. This cryptosystem is used as part of more complex encryption schemes to securely transmit data in anything from communication to signing electronic documents. Shor's algorithm is not the only quantum algorithm that can solve an infeasible problem - others have been created that can solve the discrete logarithm problem, for example, upon which Elliptic Curve cryptography relies. Because of this, Shor's algorithm and other quantum algorithms pose a potential threat to most modern encryption schemes. According to the National Institute of Standards and Technology (NIST), quantum computers will bring an end to modern cryptography as we know it [14]. This being the case, considerable research is being done in the field of post-quantum cryptography, in an effort to develop cryptosystems that are resistant to both classical and quantum attacks. In the future, I plan to study both classical and post-quantum cryptography with a focus on the mathematics involved.

9 Terminology

- *Qubit* - Shortened form of "quantum bit", the basic unit of quantum information.
- *Superposition* - A qubit's property of being able to exist in multiple states at the same time. A superposition of states $|\alpha\rangle$ and $|\beta\rangle$ is a linear combination of these states $|phi\rangle = c_1|\alpha\rangle + c_2|\beta\rangle$, where the amplitudes are $c_1, c_2 \in \mathbb{C}$.
- *Dirac notation* - the standard notation for qubit states used in quantum mechanics, consisting of the "bra" $\langle$ | and "ket" | $\rangle$.
- *Tensor Product* - given two vector spaces V and W of dimensions m and n, respectively, their tensor product $V \otimes W$ is an mn-dimensional vector space, whose elements are spanned by the tensor products $|v\rangle \otimes |w\rangle$, where $|v\rangle \in V, |w\rangle \in W$; alternate notations include $|v\rangle|w\rangle$ and $|vw\rangle$. (see definition in Section 7)
- *Bloch sphere* - a visualization of the state of a single qubit using spherical coordinates; here the north and south poles of the z-axis represent the $|0\rangle$ and $|1\rangle$ states, respectively, while the represented state is any point on the sphere.
- *Complex Conjugate* - the complex conjugate of $a + bi \in \mathbb{C}$ is $\overline{a + bi} = a - bi$, also denoted by $(a + bi)^*$. In polar form, the complex conjugate of $re^{i\phi}$ is $re^{-i\phi}$
- *Linear Operator* - a function between vector spaces $A : V \to W$, such that $A(\alpha|u\rangle + \beta|v\rangle) = \alpha(A|u\rangle) + \beta(A|v\rangle)$; if V and W have dimensions n and m respectively, then A can be represented by an $n \times m$ matrix.
- *Unitary Operator* - a linear operator A that satisfies $AA^\dagger = A^\dagger A = I$, where I is the identity operator; A is normal, diagonalizable, norm-preserving, and invertible; all eigenvalues of A have modulus 1.
- *Adjoint* - given a matrix A, its adjoint is given by $A^\dagger = (A*)^T$, the composition of the complex conjugate with the transpose; as a linear operator, $A^\dagger$ satisfies $\langle v|Aw\rangle = \langle A^\dagger v|w\rangle$.
- *Root of Unity* - A complex number $\omega \in \mathbb{C}$ with the property $\omega^k = 1$ for some $k \in \mathbb{Z}_+$. The n^{th} roots of unity for any $n \in \mathbb{Z}_+$ are given by the group $U_n = \{e^{\frac{2\pi ik}{n}} \mid k \in \mathbb{Z}_+, k \leq n\}$.
- *Order* - Given an element $x \in \mathbb{Z}_n^*$ of the group modulo n under multiplication, the order of x is the smallest $r \in \mathbb{Z}_+$ such that $x^r \equiv 1 \ (mod\ N)$.
- *Continued Fraction* - (defined in section 7)
- *Register* - A set of qubits treated as a composite system [4]. Specifically, in computer science, a register is a type of memory storage used to store data.
- *Black Box* - A function or system that is viewed only in terms of its input and output, and whose inner mechanism is unknown.
- *Entanglement* - Two qubits are entangled if the state $|\phi\rangle$ describing their relationship cannot be written in terms of the states $|\alpha\rangle$ and $|\beta\rangle$ such that $|phi\rangle = |\alpha\rangle|\beta\rangle$. This is a mysterious physical property of qubits, but crucial in quantum computation [2].
- *Phase Estimation Algorithm* - An algorithm that performs the following procedure: given a unitary operator U with eigenvector $|u\rangle$ and corresponding eigenvalue $e^{2\pi i\phi}$, estimates the phase ϕ.

Bibliography

[1] Professor Christopher King, *Meetings and Conversations*

[2] Nielsen, M.A. & Chuang I.L., *Quantum Computation and Quantum Information*, 2000
This is the main textbook I will be using to learn the basics of quantum computation and the necessary background.

[3] Shor, Peter W., *Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer*, August 30, 1995,
Shor's original paper describing his quantum algorithm.

[4] CERN, *Linear Algebra for Quantum Computation*, Appendix A,
https://cds.cern.ch/record/1522001/files/978-1-4614-6336-8_BookBackMatter.pdf
This is a reference to supplement the corresponding linear algebra section in the textbook.

[5] Lavor, Mansure, Portugal, *Shor's Algorithm for Factoring Large Integers*
https://arxiv.org/pdf/quant-ph/0303175.pdf
This article will be used as a supplement to the corresponding textbook section on Shor's Algorithm. It is a bit convoluted in terms of organization, but gives an interesting perspective.

[6] Lomonaco, Samuel J., *A Lecture on Shor's Quantum Factoring Algorithm*
https://arxiv.org/pdf/quant-ph/0010034.pdf
This lecture gives a clear overview of the steps in Shor's algorithm.

[7] Leinjung, T., *Factorization of a 768-bit RSA modulus*, February 18, 2010,
https://eprint.iacr.org/2010/006.pdf
This study exemplifies the difficulty of factoring numbers.

[8] Weisstein, Eric W., *Continued Fractions*, Wolfram Mathworld,
http://mathworld.wolfram.com/ContinuedFraction.html
Reference for the continued fractions terminology used.

[9] Weisstein, Eric W., *Number Field Sieve*, Wolfram MathWorld,
http://mathworld.wolfram.com/NumberFieldSieve.html
This article is for reference when comparing algorithm runtimes and discussing complexity analysis.

[10] Styles, Iain B., *Lecture 6: The Quantum Fourier Transform*,
http://www.cs.bham.ac.uk/internal/courses/intromqc/current/lecture06_handout.pdf
These slides contains some examples and a minimal overview of the DFT and QFT, useful for getting a basic idea of the subject.

[11] *Chapter 5: QFT, Period Finding & Shor's Algorithm*,
https://courses.edx.org/c4x/BerkeleyX/CS191x/asset/chap5.pdf
This book chapter is useful for more detailed aspects of the QFT and Shor's algorithm, contains some good examples, and discusses roots of unity.

[12] Bäumer, Elisa, Sobez, Jan-Grimo, Tessarini, Stefan *Shor's Algorithm*,
https://qudev.phys.ethz.ch/content/QSIT15/Shors%20Algorithm.pdf
These slides give a nice overview of the components and subroutines of Shor's algorithm.

[13] Rupert, Steven,, Cabell-Kluch, Zach, Pigg, Jonathan, *Shor's Algorithm Simulator*,
Colorado School of Mines, 2011, http://blendmaster.github.io/ShorJS/
A nice simulator for factoring a number using Shor's algorithm that explains the steps involved. This was used to obtain the values used in the example of Shor's algorithm.

[14] *Impact of Quantum Computing on Present Cryptography*, International Journal of Advanced Computer Science and Applications, Vol. 9, No. 3, 2018,
https://arxiv.org/pdf/1804.00200.pdf
I will be using this article to understand the applications of quantum computation on modern-day cryptography.

[15] Kay, Alastair, *Tutorial on the Quantikz Package*,
http://mirror.hmc.edu/ctan/graphics/pgf/contrib/quantikz/manual.pdf
Latex package used to generate the circuit diagrams throughout the paper.

[16] Aryal, Supreme, *Example: Unit Circle*, March 16, 2010
http://www.texample.net/tikz/examples/unit-circle/
Latex code for drawing a unit circle with common angle coordinates.

[17] RockyRock, *Plot 8th roots of unity on complex plane with PGFPlots*, May 16, 2018
https://tex.stackexchange.com/questions/431736/plot8throotsofunity oncomplexplanewithpgfplots
Inspiration for code used to plot roots of unity diagrams.