ΠΟΛΥΤΕΧΝΕΙΟ ΚΡΗΤΗΣ
TECHNICAL UNIVERSITY OF CRETE

Shor's Algorithm

Stratakis Andreas

GitHub repository:
https://github.com/astratakis/shors-algorithm

TECHNICAL UNIVERSITY OF CRETE
ELECTRICAL AND COMPUTER ENGINEERING

JULY 2022

CHAPTER 1 – ABSTRACT

Shor's algorithm is a quantum algorithm for finding prime factors of an integer. The goal of this paper is to explain the problem, the algorithm and to give a generic implementation of the quantum circuit that executes this algorithm.

In number theory integer factorization is the decomposition of a composite number into a product of smaller integers. If these factors are further restricted to prime numbers, the process is called prime factorization.

There is not yet any 'classical' algorithm that can factor all integers in polynomial time. That is to factor a b bit number in $O(b^k)$ for some constant k. Neither the existence nor the non existence of such algorithm has been proven. This problem is currently unsolved in computer science. It is suspected though that no polynomial algorithm exists and thus the problem is in class NP (although it is not yet prooven). The current best implementation runs in $O(N)$ using

This problem is very important because number factorization is used in cryptography. Since it is very easy to multiply two numbers and evaluate the initial number and it is very hard to find those two factors of the initial number, Ron Rivest, Adi Shamir and Leonard Adleman came up with an algorithm called RSA that exploits this property in order to encrypt messages and secure Internet communication.

Breaking this property of exponential complexity to calculate the factors of a number means that someone with a working quantum computer with a couple of thousand fully controllable qubits would be able to decrypt any message that uses this RSA protocol in a couple of hours. This would essentially break the Internet, because RSA is widely used today in pretty much all the Internet.

Luckily there are many other encryption protocols that cannot be broken (yet) in polynomial time by a quantum computer. Also the technology of quantum computers is currently in its infancy. Scientists are currently trying to figure out how to fully control a couple of qubits.

CHAPTER 2 – A BETTER SOLUTION

Shor's algorithm takes as input a bad guess g and the number N (the number that we want to factor), where g does not share any factors with N and creates a much better guess that probably does share factors with N.

For any number g where $\gcd(g,N) = 1$:

$$g^r \equiv 1\ (mod\ N)$$

The value r is called the $order_N(g)$ and when known it can be used to possibly derive factors of N.

Given that $r = order_N(g)$:

$$g^r \equiv 1\ (mod\ N)$$

$$g^r - 1 \equiv 0\ (mod\ N)$$

$$\left(g^{r/2} - 1\right) \cdot \left(g^{r/2} + 1\right) \equiv 0\ (mod\ N)$$

Now the expression looks like a product of two integer numbers and it is divisible by some constant times N. Note that in order for the values $\left(g^{r/2} - 1\right)$ and $\left(g^{r/2} + 1\right)$ to be integers, r must be even.

Assuming that r is even $\left(g^{r/2} - 1\right) \cdot \left(g^{r/2} + 1\right) \equiv 0\ (mod\ N)$ can be written as $\left(g^{r/2} - 1\right) \cdot \left(g^{r/2} + 1\right) = k \cdot N$ where $k \in \mathbb{Z}$. One of the integers $\left(g^{r/2} - 1\right)$ or $\left(g^{r/2} + 1\right)$ could be a multiple of k or N in which case the problem would not be solved. It turns out that the probability of one of these factors sharing factors with N is order of half ($\Pr \sim \frac{3}{8}$). This means that as long as Shor's algorithm runs in polynomial time, this process is worth repeating until a solution is found.

CHAPTER 3 – QUANTUM FOURIER TRANSFORM (QFT)

3.1 QFT

In quantum computing the Quantum Fourier Transform (QFT) is a linear transformation on qubits, and is the quantum analogue of the discrete Fourier transform. The QFT applies on the amplitudes of a wavefunction and produces a different wavefunction as shown below.

The discrete Fourier transform acts on a vector $[x_0 \quad x_1 \quad \dots \quad x_{N-1}]$ and maps it to another vector $[y_0 \quad y_1 \quad \dots \quad y_{N-1}]$ according to the formula:

$$y_k = \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} x_j \cdot e^{2\pi i \frac{jk}{N}}$$

Similarly, the quantum Fourier transform acts on a quantum state $|X\rangle$ and maps it to the quantum state $|Y\rangle$ where:

$$|X\rangle = \sum_{j=0}^{N-1} x_j \cdot |j\rangle \qquad |Y\rangle = \sum_{k=0}^{N-1} x_k \cdot |k\rangle \qquad y_k = \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} x_j \cdot e^{2\pi i \frac{jk}{N}}$$

Note that only the amplitudes of the state were affected by this transformation. The QFT can also be expressed as the unitary matrix:

$$U_{QFT} = \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} \sum_{k=0}^{N-1} e^{2\pi i \frac{jk}{N}} \cdot |k\rangle\langle j|$$

$$U_{QFT} = \frac{1}{\sqrt{N}} \begin{bmatrix} 1 & 1 & 1 & 1 & \cdots & 1 \\ 1 & \omega & \omega^2 & \omega^3 & \cdots & \omega^{N-1} \\ 1 & \omega^2 & \omega^4 & \omega^6 & \cdots & \omega^{2(N-1)} \\ 1 & \omega^3 & \omega^6 & \omega^9 & \cdots & \omega^{3(N-1)} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \omega^{N-1} & \omega^{2(N-1)} & \omega^{3(N-1)} & \cdots & \omega^{(N-1)(N-1)} \end{bmatrix}, \text{ where } \omega = e^{\frac{2\pi i}{N}}$$

The QFT transforms between two basis, the computational (Z) basis and the Fourier basis. Every possible multi-qubit state in the computational basis has a distinct corresponding state in the Fourier basis. Note that N refers to the number of dimensions in the Hilbert space of n qubits, thus $N = 2^n$, where n is the number of qubits.

The inverse quantum Fourier transform is the conjugate transpose of the QFT.

$$|State\ in\ Computational\ Basis\rangle \xrightarrow{QFT} |State\ in\ Fourier\ Basis\rangle$$

$$|State\ in\ Fourier\ Basis\rangle \xrightarrow{QFT^\dagger} |SState\ in\ Computational\ Basis\rangle$$

A similar notation that is used in many textbooks is:

$$QFT|x\rangle = |\tilde{x}\rangle$$

$$QFT^\dagger|\tilde{x}\rangle = |x$$

A more intuitive way to understand the quantum Fourier transform is the following. Let's assume a basis state $|x\rangle$ in the Z axis of the Hilbert space of N qubits. The transformation will be:

$$U_{QFT}|x\rangle = \frac{1}{\sqrt{N}}\begin{bmatrix} 1 & 1 & 1 & 1 & \cdots & 1 & \cdots & 1 \\ 1 & \omega & \omega^2 & \omega^3 & \cdots & \omega^x & \cdots & \omega^{N-1} \\ 1 & \omega^2 & \omega^4 & \omega^6 & \cdots & \omega^{2x} & \cdots & \omega^{2(N-1)} \\ 1 & \omega^3 & \omega^6 & \omega^9 & \cdots & \omega^{3x} & \cdots & \omega^{3(N-1)} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ 1 & \omega^{N-1} & \omega^{2(N-1)} & \omega^{3(N-1)} & \cdots & \omega^{x(N-1)} & \cdots & \omega^{(N-1)(N-1)} \end{bmatrix} \cdot \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 1 \to |x\rangle \\ \vdots \\ 0 \end{bmatrix}$$

$$U_{QFT}|x\rangle = \frac{1}{\sqrt{N}} \cdot \begin{bmatrix} 1 \\ \omega^x \\ \omega^{2x} \\ \omega^{3x} \\ \vdots \\ \omega^{x(N-1)} \end{bmatrix}, \text{ where } \omega = e^{\frac{2\pi i}{N}}$$

Notice that ω^x is a phase rotation on a particular state. The probability of measuring each state does not change because:

$$Pr(|x\rangle) = \left|\frac{1}{\sqrt{N}} \cdot e^{\frac{2\pi i \cdot jx}{N}}\right|^2 = \frac{1}{N} \cdot \left|e^{\frac{2\pi i \cdot jx}{N}}\right|^2 = \frac{1}{N}$$

Example 1:

Let's assume that we have 1 qubit. This means that $N = 2$ and $\omega = e^{i\pi}$

$$U_{QFT} = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

Notice that the QFT on 1 qubit is equal to the Hadamard transformation.

3.2 Example of QFT on 3 qubits.

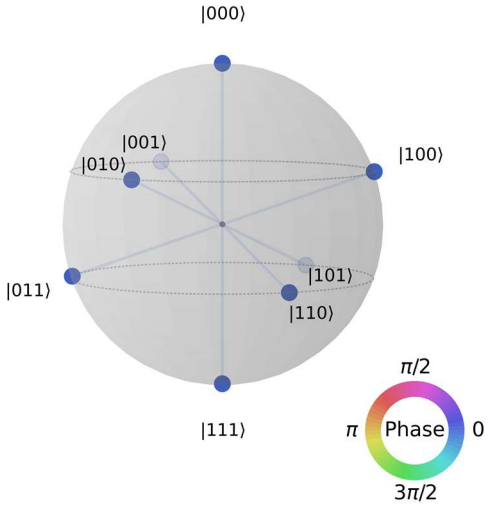
This is another example where $n = 3$ qubits. This means that $N = 8$ and $\omega = e^{\frac{i\pi}{4}}$. In this case the unitary QFT matrix is:

$$U_{QFT} = \frac{1}{\sqrt{8}}\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & e^{\frac{i\pi}{4}} & i & e^{\frac{3i}{4}} & -1 & e^{\frac{5i\pi}{4}} & -i & e^{\frac{7i}{4}} \\ 1 & i & -1 & -i & 1 & i & -1 & -i \\ 1 & e^{\frac{3i\pi}{4}} & -i & e^{\frac{i\pi}{4}} & -1 & e^{\frac{7i\pi}{4}} & i & e^{\frac{5i\pi}{4}} \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & e^{\frac{5i\pi}{4}} & i & e^{\frac{7i\pi}{4}} & -1 & e^{\frac{i\pi}{4}} & -i & e^{\frac{3i}{4}} \\ 1 & -i & -1 & i & 1 & -i & -1 & i \\ 1 & e^{\frac{7i\pi}{4}} & -i & e^{\frac{5i\pi}{4}} & -1 & e^{\frac{3i\pi}{4}} & i & e^{\frac{i\pi}{4}} \end{bmatrix}$$

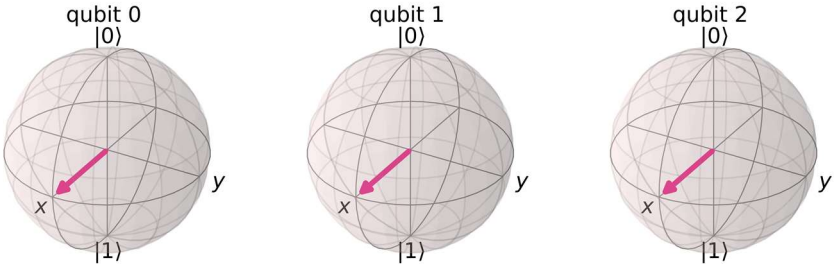
The matrix above can be visualized as vectors as shown to the table below:

	$\|000\rangle$	$\|001\rangle$	$\|010\rangle$	$\|011\rangle$	$\|100\rangle$	$\|101\rangle$	$\|110\rangle$	$\|111\rangle$
$\widetilde{\|000\rangle}$	→	→	→	→	→	→	→	→
$\widetilde{\|001\rangle}$	→	↗	↑	↖	←	↙	↓	↘
$\widetilde{\|010\rangle}$	→	↑	←	↓	→	↑	←	↓
$\widetilde{\|011\rangle}$	→	↖	↓	↗	←	↘	↑	↙
$\widetilde{\|100\rangle}$	→	←	→	←	→	←	→	←
$\widetilde{\|101\rangle}$	→	↙	↑	↘	←	↗	↓	↖
$\widetilde{\|110\rangle}$	→	↓	←	↑	→	↓	←	↑
$\widetilde{\|111\rangle}$	→	↘	↓	↙	←	↖	↑	↗

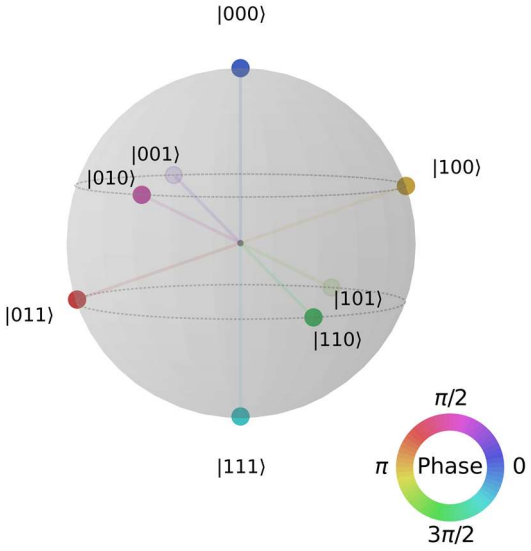
Note that the fourier of the state $|0\rangle$ is equal the Hadamard transform of all qubits.



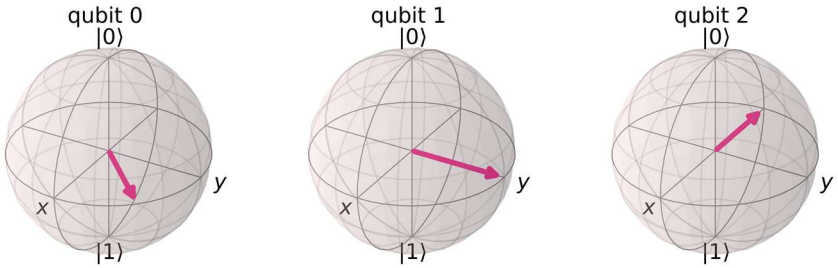
Q sphere of the QFT(0)



Bloch Multivector of QFT(0)



Q sphere of the QFT(1)



Bloch Multivector of QFT(1)

The quantum Fourier transform of a sum of states is equal to the sum of the QFT of each state. Suppose the state $|\psi\rangle = |x\rangle + |y\rangle$. This state is obviously not normalized for the sake of simplicity.

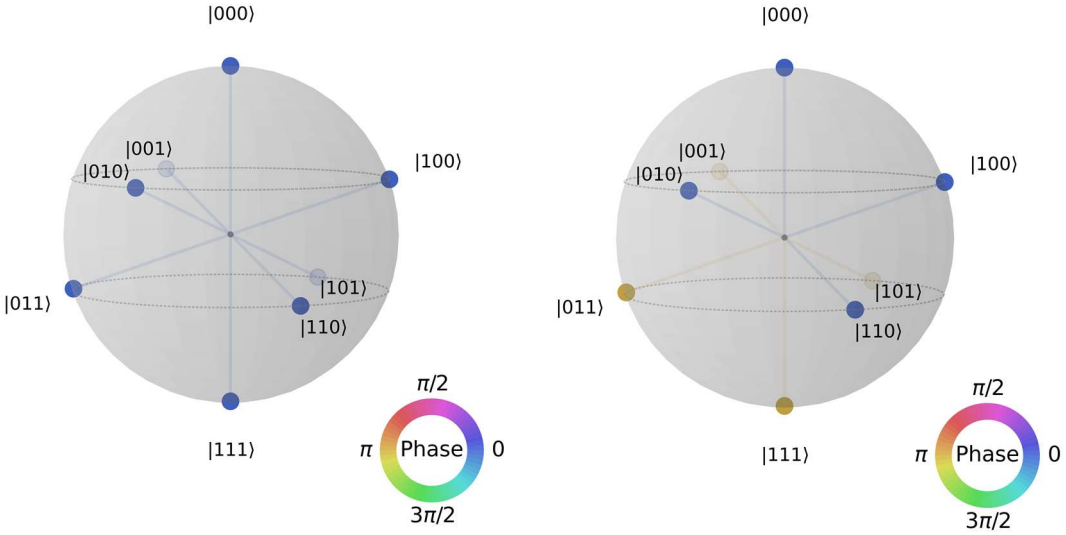
$$U_{QFT} = \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} \sum_{k=0}^{N-1} e^{2\pi i \frac{jk}{N}} \cdot |k\rangle\langle j|$$

$$U_{QFT} = \frac{1}{\sqrt{N}} \left(\sum_{j=0}^{N-1} e^{2\pi i \frac{jk}{N}} \cdot |x\rangle\langle j| + \sum_{j=0}^{N-1} e^{2\pi i \frac{jk}{N}} \cdot |y\rangle\langle j| \right)$$

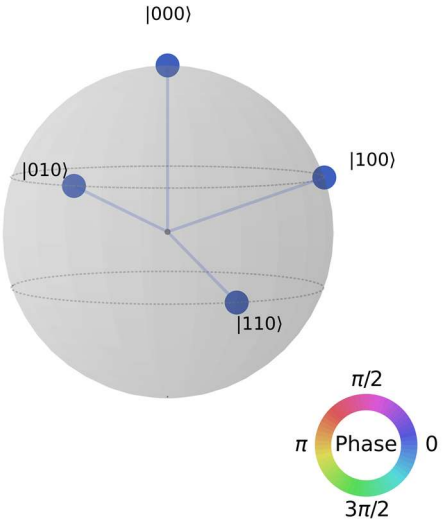
Using the formula $y_k = \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} x_j \cdot e^{2\pi i \frac{jk}{N}}$ this can be interpreted as:

$$QFT(|x\rangle + |y\rangle) = QFT|x\rangle + QFT|y\rangle$$

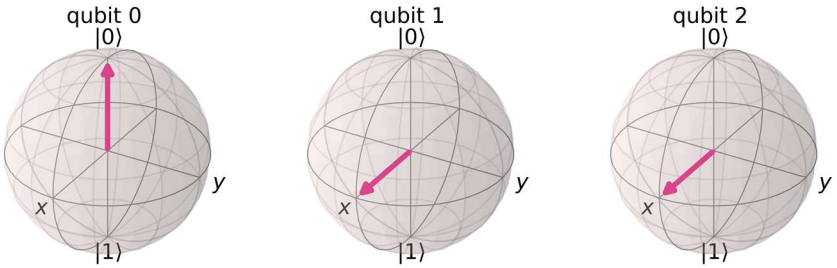
For example $|\psi\rangle = \frac{|0\rangle+|4\rangle}{\sqrt{2}}$. The QFT of $|\psi\rangle$ is $|\tilde{\psi}\rangle = \frac{1}{\sqrt{2}}QFT|0\rangle + \frac{1}{\sqrt{2}}QFT|4\rangle$



The odd states $|1\rangle, |3\rangle, |5\rangle, |7\rangle$ have opposite phases and the even states $|0\rangle, |2\rangle, |4\rangle, |6\rangle$ have the same phases. The odd states should destructively interfere and the even states should constructively interfere with each other. The final state is the following:



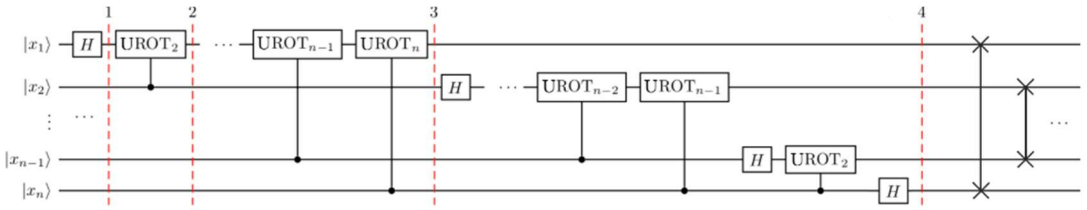
Q sphere of the QFT on superposition of 0 and 4



Bloch multivector of the QFT on superposition of 0 and 4

3.3 The circuit

The circuit of the Quantum Fourier transform consists of polynomial elementary gates.



At first we have the state $|x_1 x_2 \dots x_n\rangle$.

After the first Hadamard transformation on the first qubit the state is:

$$H_1|x_1 x_2 \dots x_n\rangle = \frac{1}{\sqrt{2}}\left[|0\rangle + e^{\frac{2i\pi}{2}x_1}|1\rangle\right] \otimes |x_2 x_3 \dots x_n\rangle$$

Next is the controlled phase shift on the first qubit which creates the state:

$$\frac{1}{\sqrt{2}}\left[|0\rangle + e^{\frac{2i\pi}{2}x_1 + \frac{2i\pi}{2^2}x_2}|1\rangle\right] \otimes |x_2 x_3 \dots x_n\rangle$$

After the final controlled phase shift on the first qubit the state will be:

$$\frac{1}{\sqrt{2}}\left[|0\rangle + e^{\frac{2i\pi}{2}x_1 + \frac{2i\pi}{2^2}x_2 + \dots + \frac{2i\pi}{2^n}x_n}|1\rangle\right] \otimes |x_2 x_3 \dots x_n\rangle$$

Notice that:

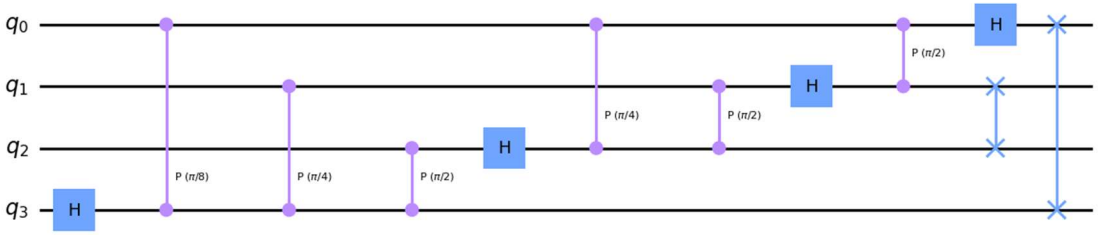
$$x = 2^{n-1}x_1 + 2^{n-2}x_2 + \dots + 2^1 x_{n-1} + 2^0 x_n$$

The expression can be written as:

$$\frac{1}{\sqrt{2}}\left[|0\rangle + e^{\frac{2i\pi}{2^n}x}|1\rangle\right] \otimes |x_2 x_3 \dots x_n\rangle$$

Applying this process again to the rest of the qubits $x_2 \dots x_n$ the final state will be:

$$\frac{1}{\sqrt{2}}\left[|0\rangle + e^{\frac{2i\pi}{2^n}x}|1\rangle\right] \otimes \frac{1}{\sqrt{2}}\left[|0\rangle + e^{\frac{2i\pi}{2^{n-1}}x}|1\rangle\right] \otimes \dots \otimes \frac{1}{\sqrt{2}}\left[|0\rangle + e^{\frac{2i\pi}{2^2}x}|1\rangle\right] \otimes \frac{1}{\sqrt{2}}\left[|0\rangle + e^{\frac{2i\pi}{2^1}x}|1\rangle\right]$$



Example of QFT on 4 qubits

CHAPTER 4 – SHORS ALGORITHM

4.1 The algorithm

1) Pick a random number $a \in [2, N-1]$
2) Compute $K = gcd(a, N)$ using Euclidean algorithm.
3) If $K \neq 1$ then K is a factor of N and we are done.
4) Use the quantum period finding subroutine to find $r = order_N(a)$
5) If r is odd, then repeat the process from step 1.
6) If $a^{r/2} + 1 \equiv 0 \, (mod \, N)$ then repeat again the process from step 1.
7) The factors of N are $gcd\left(a^{\frac{r}{2}} + 1, N\right)$ and $gcd\left(a^{\frac{r}{2}} - 1, N\right)$

4.2 Quantum period finding subroutine

First a superposition of $2n$ qubits is created. Then using the controlled U_a operation an entangled state $|x\rangle|a^x \, mod \, N\rangle$ is generated. Notice that the state $|x\rangle$ is the superposition $\frac{1}{\sqrt{2^{2n}}} \cdot \sum_{n=0}^{2^{2n}-1} |n\rangle$. Measuring the remainder $|a^x \, mod \, N\rangle$ collapses the entangled state's waveform into a superposition of only the states $|x\rangle$ that would create the remainder $|a^x \, mod \, N\rangle$. Since $|a^x \, mod \, N\rangle$ is a periodic function the superposition after the measurement would contain states that are r states apart, where $r = order_N(a)$.

Proof that the function $f(x) = a^x \, mod \, N$ is periodic with period $order_N(a)$:

For any given random x_1 $f(x_1) = a^{x_1} \, mod \, N = s$

Since a and N do not share any factors $gcd(a, N) = 1$:

The order of a modulo N is defined as:

$$r = order_N(a) \text{ such that } a^r \equiv 1 \, (mod \, N)$$

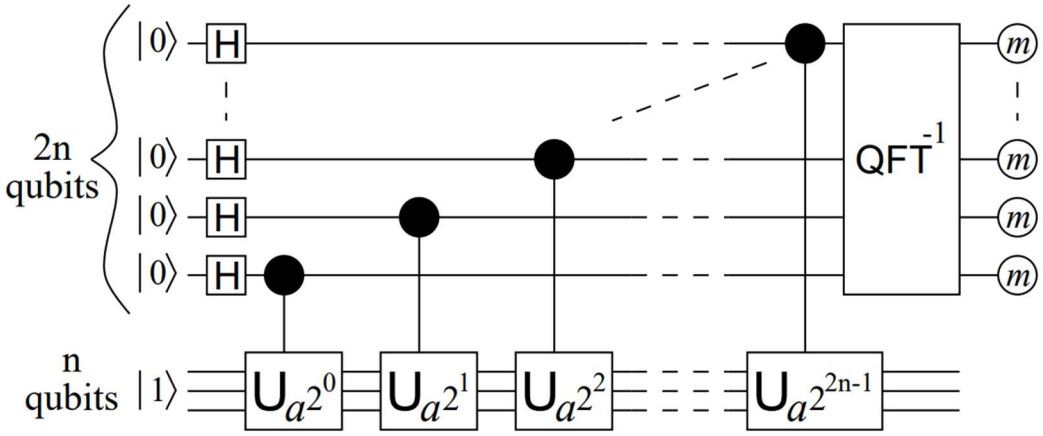
$$\left.\begin{matrix} a^r \equiv 1 \, (mod \, N) \\ a^{x_1} \equiv s \, (mod \, N) \end{matrix}\right\} \xrightarrow{multiply} a^{x_1 + r} \equiv s$$

Since x_1 was picked at random and s is a random integer remainder then the function $f(x)$ is periodic with period $r = order_N(a)$.

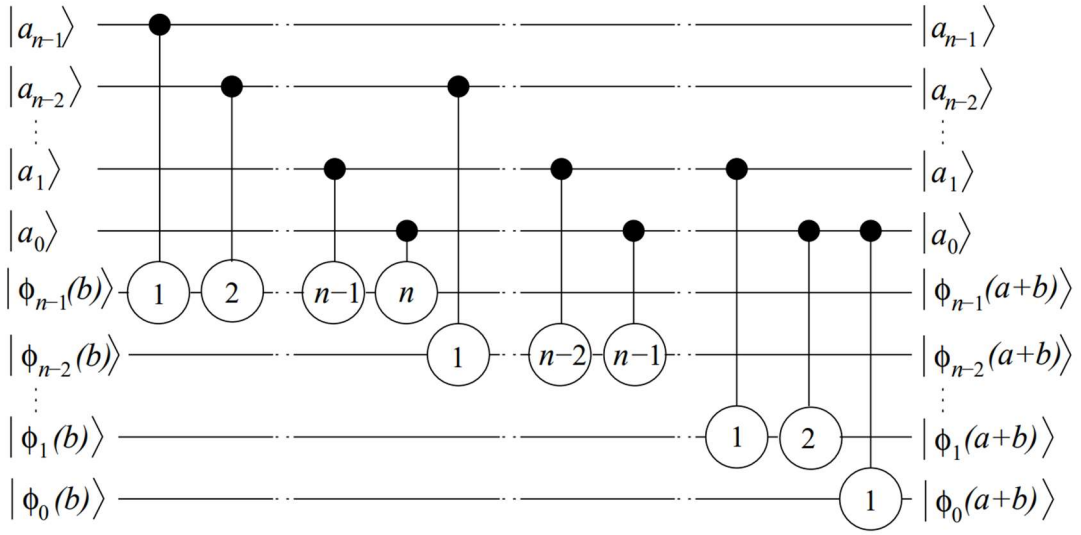
At this point the inverse quantum Fourier transform can be applied on the first $2n$ qubits in order to find the period r. After this transformation the first $2n$ qubits are measured and final measurement divided by 2^{2n} will be a multiple of $\frac{1}{r}$.

This measurement must be performed many times in order to get a reliable answer for the period r. This is because the multiple of $\frac{1}{r}$ may share factors with r which will lead to a bad guess.

4.3 The circuit

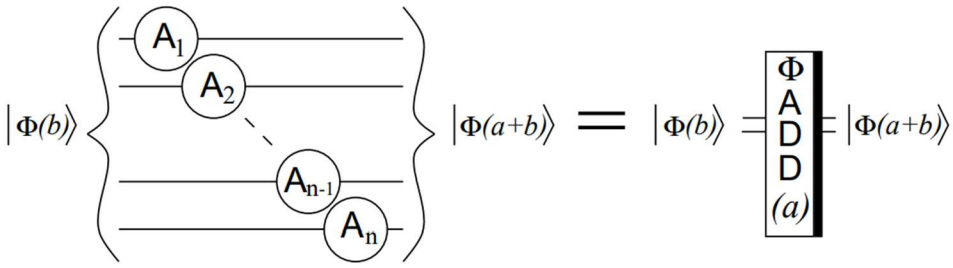


Shor's algorithm quantum circuit



Addition described by Draper

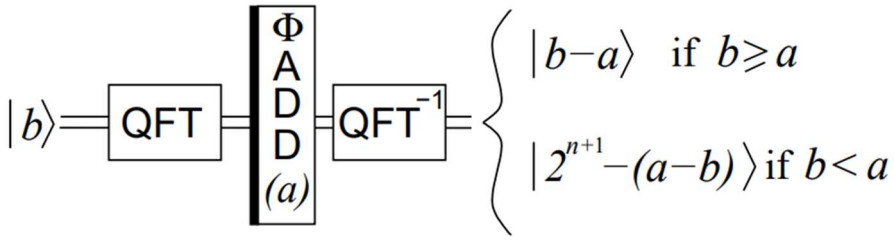
In order to add two numbers a and b we apply quantum Fourier transform and apply the addition in the Fourier basis. The Draper method can be implemented classically. The number a is represented in classical bits. This means that the controlled operations can be reduced to single qubit operations



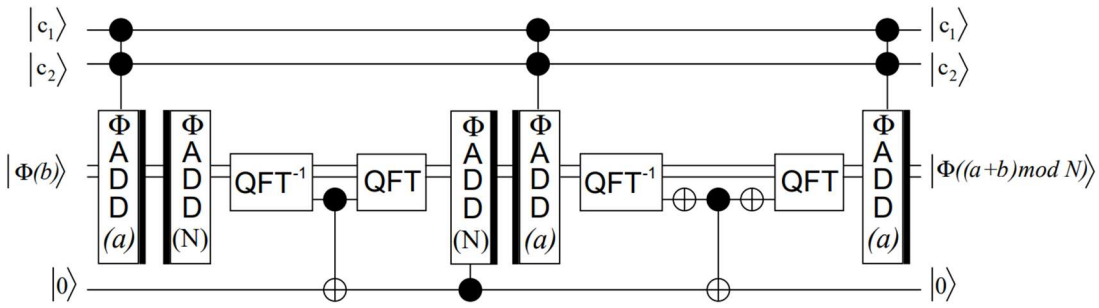
Addition using Draper method where A transformations are classically computed.

4.3.1 The modular adder gate

In order to prevent overflow 1 more qubit is used to store the carry of the addition. Notice that the thick black bar on the right is used to distinguish an operation by the respective unitary inverse. The overflow qubit is very important because it will decide if the modular operation will be performed or not.



The effect of the quantum subtraction



The modular adder circuit

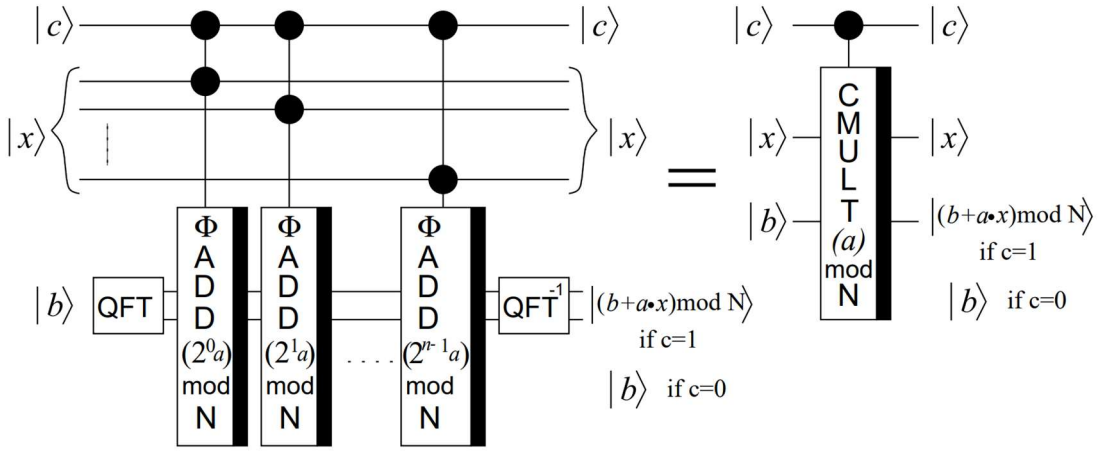
The first step is to add a to b. If the operation overflows the overflow qubit will be flipped. Then N is subtracted from $a+b$ in the Fourier basis. In order to check the overflow qubit we first change from the Fourier basis to the computational basis using the inverse quantum Fourier transform. One more qubit is used to store the overflow information. If the overflow qubit is '1' then flip the state of the flag qubit. Then change into Fourier basis using QFT and if the helper qubit is '1' add N in order to perform the modular addition. However the state of the flag qubit is now changed and it needs to go back to '0' to correctly perform the operation. Notice that the state of the other qubits must not change. In order to perform this, subtract a from the last state and check the overflow qubit in the computational basis. Then flip the helper qubit back to its original state and add again a in the Fourier basis.

4.3.2 The modular multiplier gate

In this step we must calculate $(ax)\,mod\,N$

$$(ax)mod\,N = (\ldots((2^0 a x_0)\text{mod N} + 2^1 a x_1)\text{mod N} + \ldots + 2^{n-1} a x_{n-1})\text{mod N}$$

This makes this problem a lot easier. Each value x_i corresponds each qubit of the state $|x\rangle$. It can be replaced by a controlled operation. Also $2^i a$ can be calculated very efficiently using the repeating squaring algorithm. The circuit of the modular multiplier is the following:

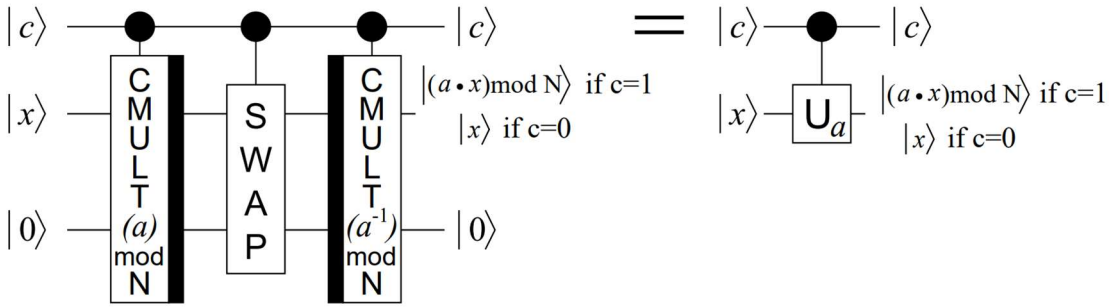


The modular multiplier

The last step is to implement the final controlled U operation. Notice that using the modular multiplier only the state $|b\rangle$ is changed. In order to apply the change on the state $|x\rangle$:

$$|c\rangle|x\rangle|0\rangle \rightarrow |c\rangle|x\rangle|(ax)mod\,N\rangle \rightarrow |c\rangle|(ax)mod\,N\rangle|x\rangle$$
$$\rightarrow$$
$$|c\rangle|(ax)mod\,N\rangle|(x - a^{-1}ax)modN\rangle \rightarrow |c\rangle|(ax)mod\,N\rangle|0\rangle$$

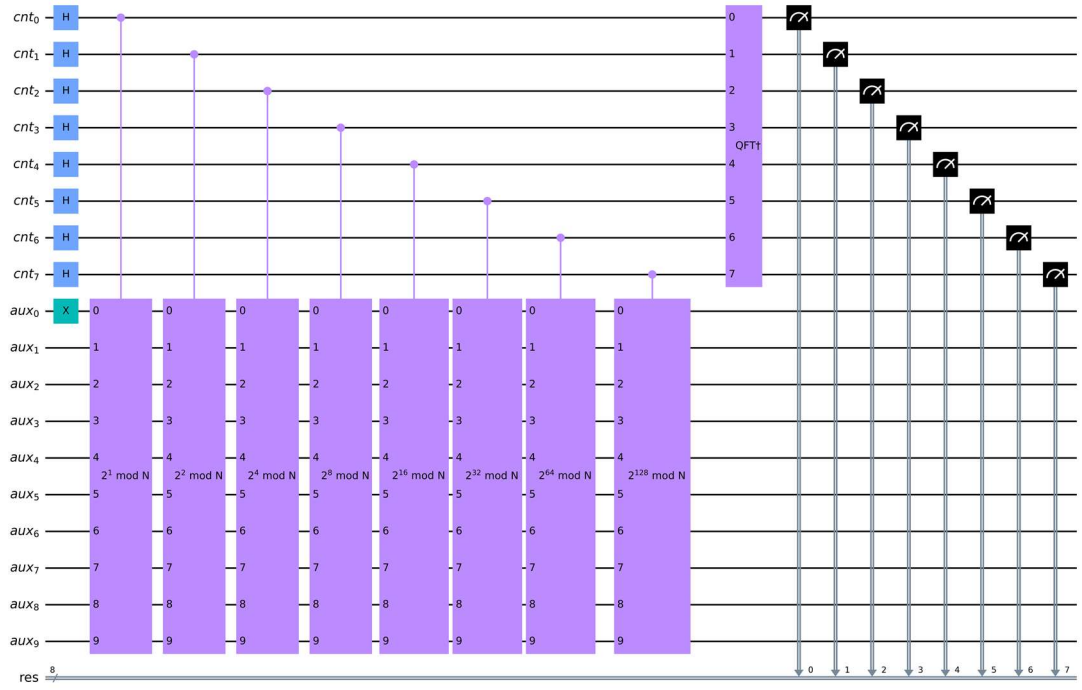
These transformations correspond to the following circuit:



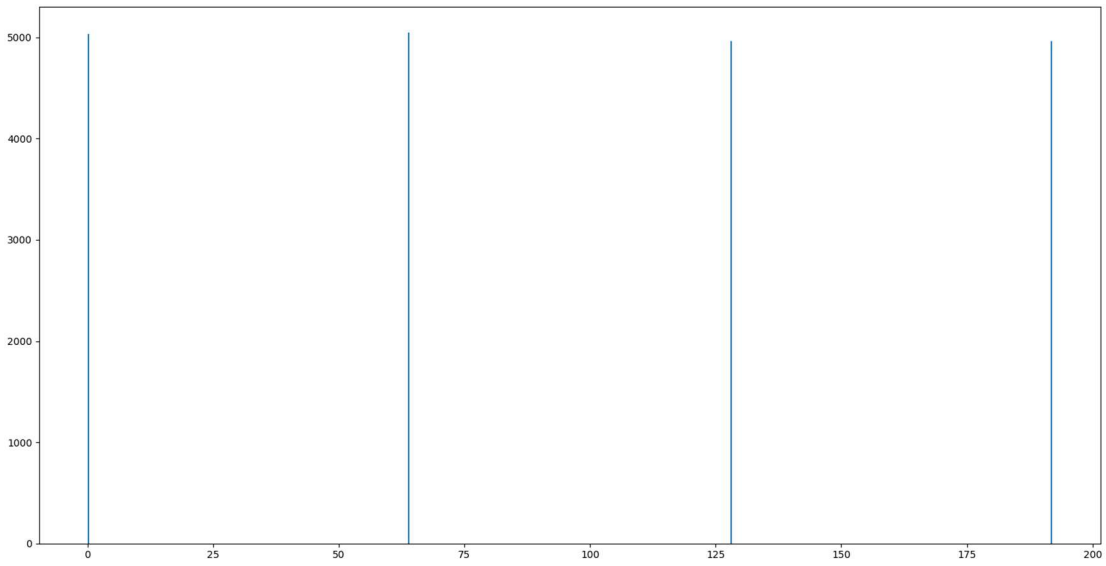
The controlled U operation

CHAPTER 5 – SIMULATION

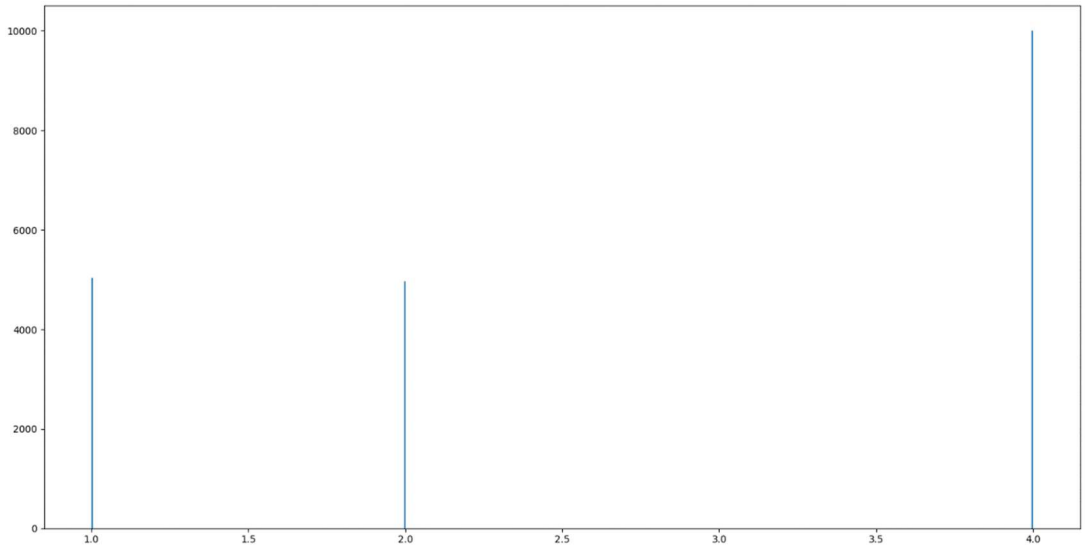
5.1 Shor's circuit for $N = 15$ and $a = 2$



Quantum circuit



Measurement results



Plot of best guesses

The solutions are evaluated from the most probable to the least probable. In this case the most probable solution is $r = 4$. This means that:

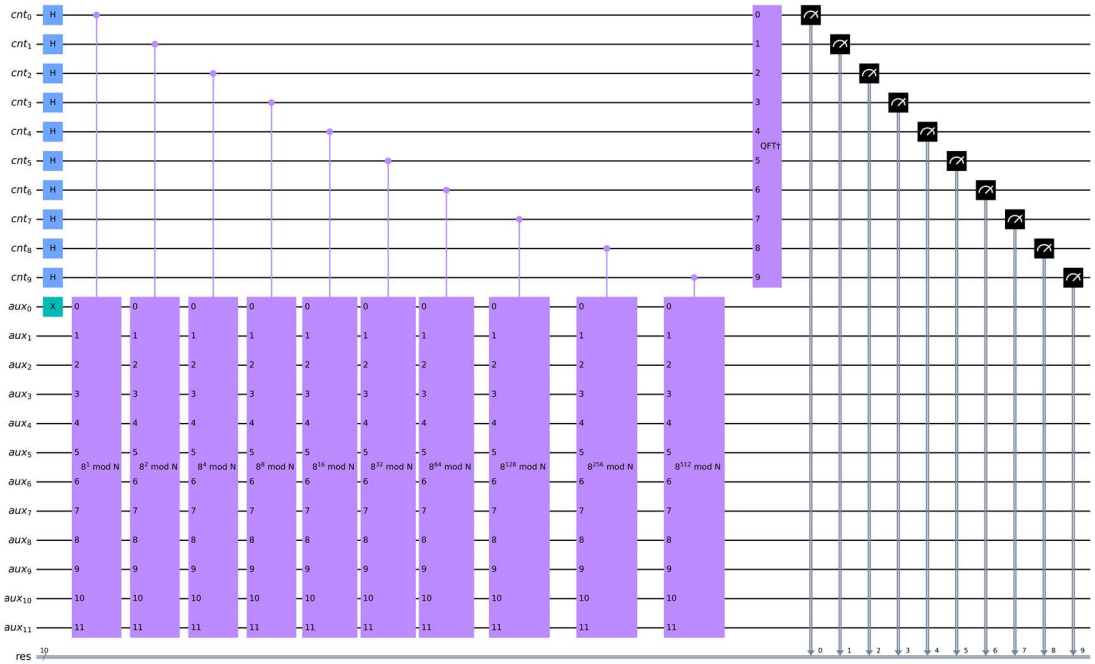
$$(2^2 - 1) \cdot (2^2 + 1) = k \cdot 15$$

$$\gcd(3, 15) = 3$$

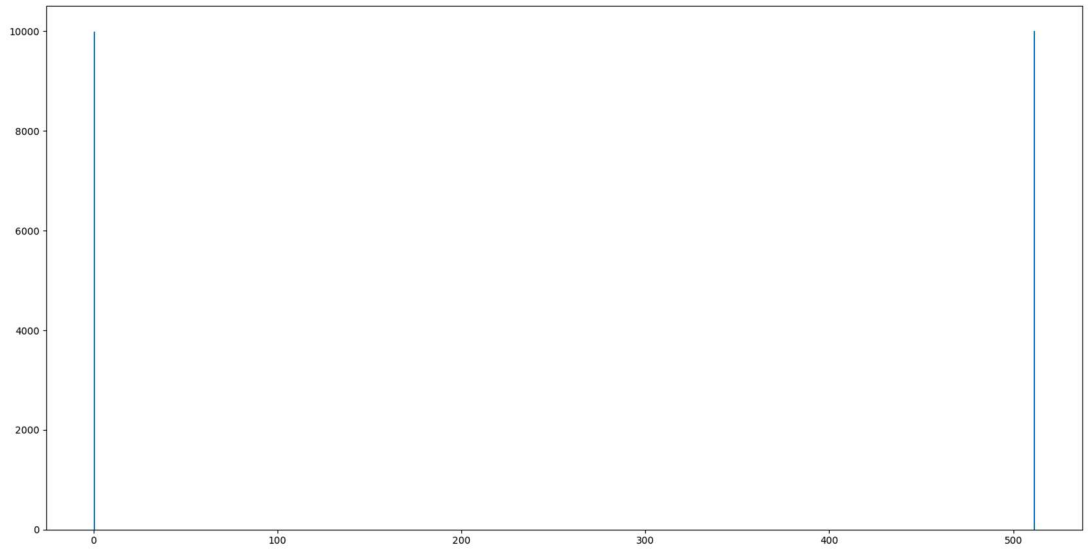
$$\gcd(5, 15) = 5$$

Therefore the factors of $N = 15$ are $p = 3$ and $q = 5$

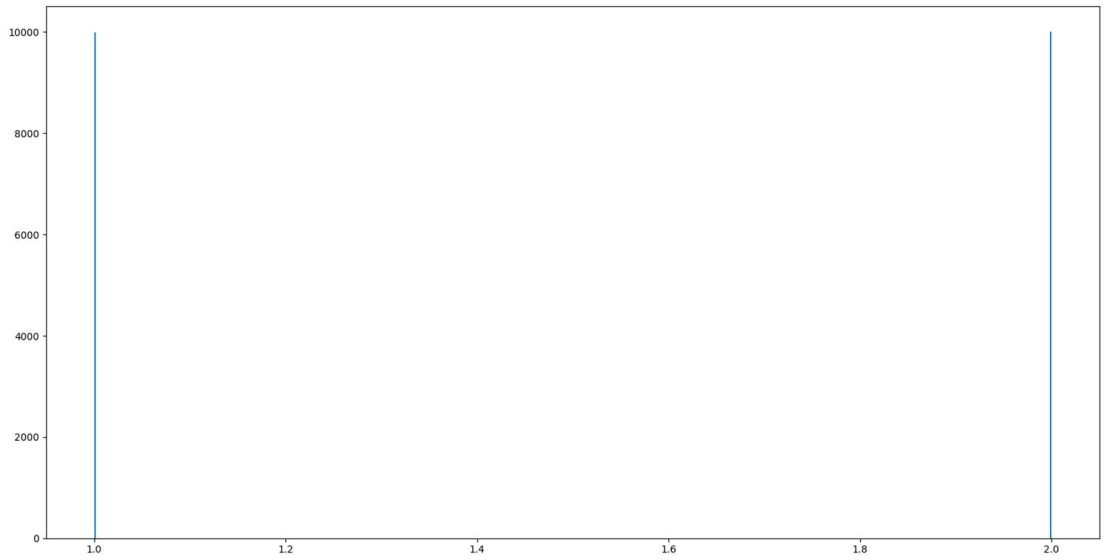
5.2 Shor's circuit for $N = 21$ and $a = 8$



Quantum circuit



Measurement results



Plot of best guesses

In this case the most probable solution is $r = 1$. This contradicts the initial assumption that r is even. This means that the next most probable solution is evaluated. For $r = 2$ we have:

$$(8^1 + 1) \cdot (8^1 - 1) = k \cdot 21$$

The first factor is $8^1 + 1 = 9$

$$\gcd(9, 21) = 3$$

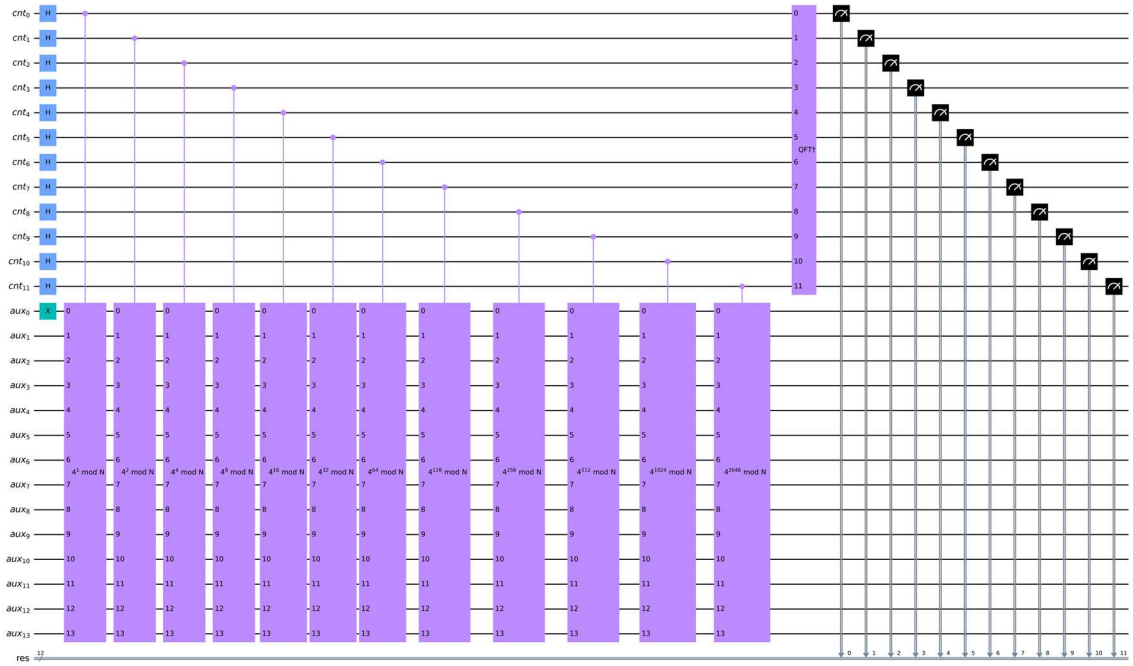
Note that the number may share factors with the initial number. The Euclidian algorithm is very efficient and the actual factor of the number can be calculated easily.

The second factor is:

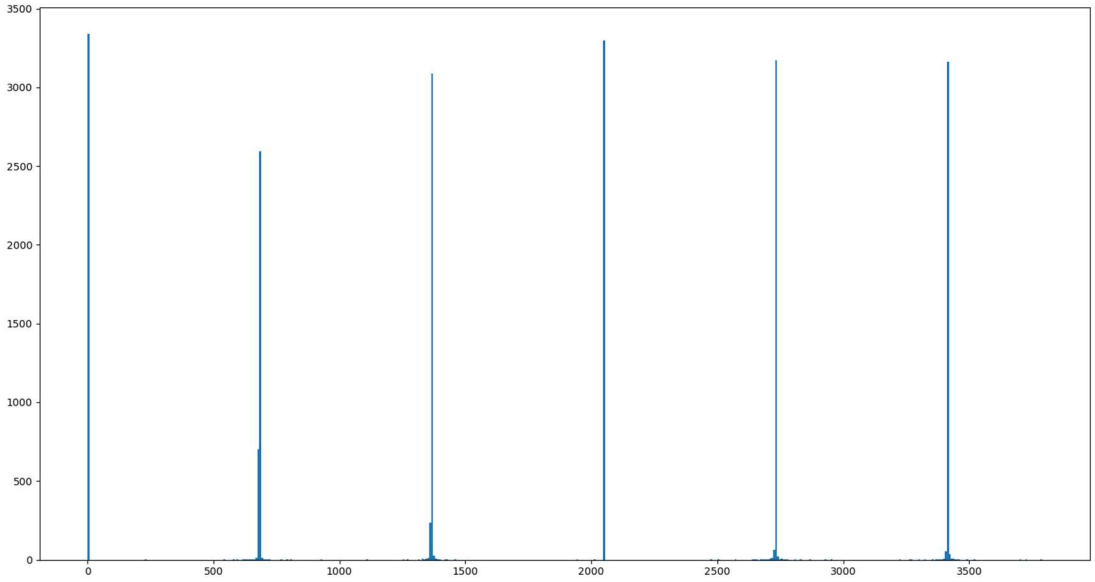
$$\gcd(3, 21) = 7$$

Therefore the factors of $N = 21$ are $p = 3$ and $q = 7$

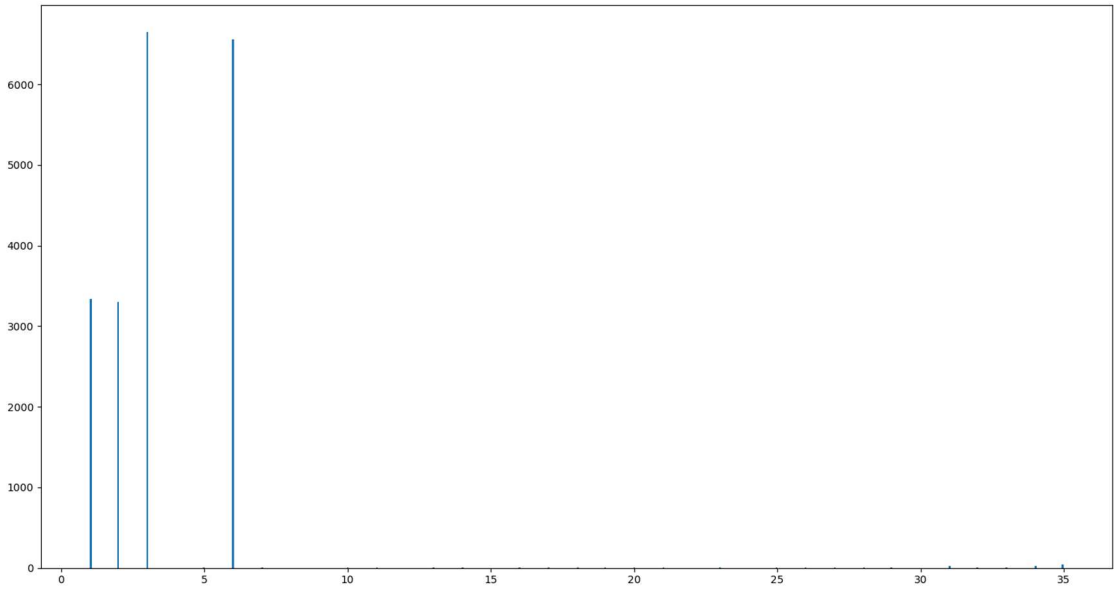
5.3 Shor's circuit for $N = 35$ and $a = 4$



Quantum circuit



Measurement results



Plot of best guesses

In this case the first solution to be evaluated is $r = 3$. This is an odd number and it is skipped. The second number will be $r = 6$.

$$(4^3 + 1) \cdot (4^3 - 1) = k \cdot 35$$

$$\gcd(65, 35) = 5$$

$$\gcd(63, 35) = 7$$

Therefore the factors of $N = 35$ are $p = 5$ and $q = 7$

Note that the measurement results appear to have a slight deviation from the theoretical values. This error also translates to the plot of best guesses where there are guesses (with low probability) of $r = 35$. The Aer simulator that was used for this experiment is ideal and does not simulate errors. That means that the error must have been created form inaccurate floating point expression when calculating the theta angle on Fourier addition.

CHAPTER 6 – COMPLEXITY ANALYSIS

Each controlled U operation can be classically calculated in polynomial time using the repeating square algorithm. This means that the preprocessing of a^{2^n} can be calculated in $O(loglog(a^{2^n}))$. This is somewhat equal to $O(n)$. Notice that in this case is the number of counting qubits.

Each controlled U operation also contains two controlled multiplication gates and a controlled swap gate. The controlled swap gate swaps all qubits of the state $|x\rangle$. This means that the controlled swap runs in $O(n)$ time where n is the numbers of qubits that are needed to represent N in binary.

Each controlled multiplication gate contains n controlled modular adder gates which means that the controlled multiplier gate has time complexity $O(n)$.

Finally each controlled modular adder gate consists of several quantum Fourier transform gates and some adder gates. The adder gates are classically calculated in $O(n^2)$ but their runtime on a quantum device is $O(1)$. The QFT though has a runtime of $O(n^2)$.

Notice that the quantum Fourier transform needs $O(n^2)$ elementary gates to be implemented. The classical Discrete Fourier transform needs $O(n2^n)$ classical gates to be implemented. That's an exponational improvement.

The total runtime complexity of the quantum circuit is $O(n \cdot (n^2 + n + n^2)) = O(n^3)$. The circuit is also classically computable in polynomial time.

The space complexity is $O(n)$ since $4n + 2$ qubits are used for an n bit number. There has been published an implementation that does not use the counting qubits and implements Shor's algorithm using only $2n + 3$ qubits.

CHAPTER 7 – REFERENCES

1) https://qiskit.org/textbook/ch-algorithms/quantum-fourier-transform.html
2) https://arxiv.org/abs/quant-ph/0205095v3
3) https://qiskit.org/textbook/ch-algorithms/shor.html