

**Shor’s Algorithm**

Stratakis Andreas

GitHub repository:

<https://github.com/astratakis/shors-algorithm>

TECHNICAL UNIVERSITY OF CRETE

ELECTRICAL AND COMPUTER ENGINEERING

JULY 2022

**CHAPTER I – 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 bit number in for some constant . 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 (although it is not yet prooven). The current best implementation runs in using

This problem is very important because number factorization is used in cryptography. Since it is very easy 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.

Luckily there are many other encryption protocols that cannot be (yet) be broken 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 II – A BETTER SOLUTION**

Shor’s algorithm takes as input a bad guess and the number , where g does not share any factors with and creates a much better guess that probably does share factors with .

For any number where :

The value is called the and when known it can be used to possibly derive factors of .

Given that :

Now the expression looks like a product of two integer numbers and it is divisible by some constant times . Note that in order for the values and to be integers, must be even.

Assuming that is even can be written as where . Note that one of the integers or could be a multiple of or in which case the problem would not be solved. It turns out that the probability of one of these factors sharing factors with is order of half. This means that as long as Shor’s algorithm runs in polynomial time, this process is worth repeating until a solution is produced.

**CHAPTER III – QUANTUM FOURIER TRANSFORM (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 and maps it to another vector according to the formula:

Similarly, the quantum Fourier transform acts on a quantum state and maps it to the quantum state where:

|  |  |  |
| --- | --- | --- |
|  |  |  |

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

, where

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 qubits, thus , where is the number of qubits.

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

A more intuitive way to understand the quantum Fourier transform is the following. Let’s assume a basis state in the Z axis of the Hilbert space of qubits. The transformation will be:

, where

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

**Example 1:**

Let’s assume that we have 1 qubit. This means that and

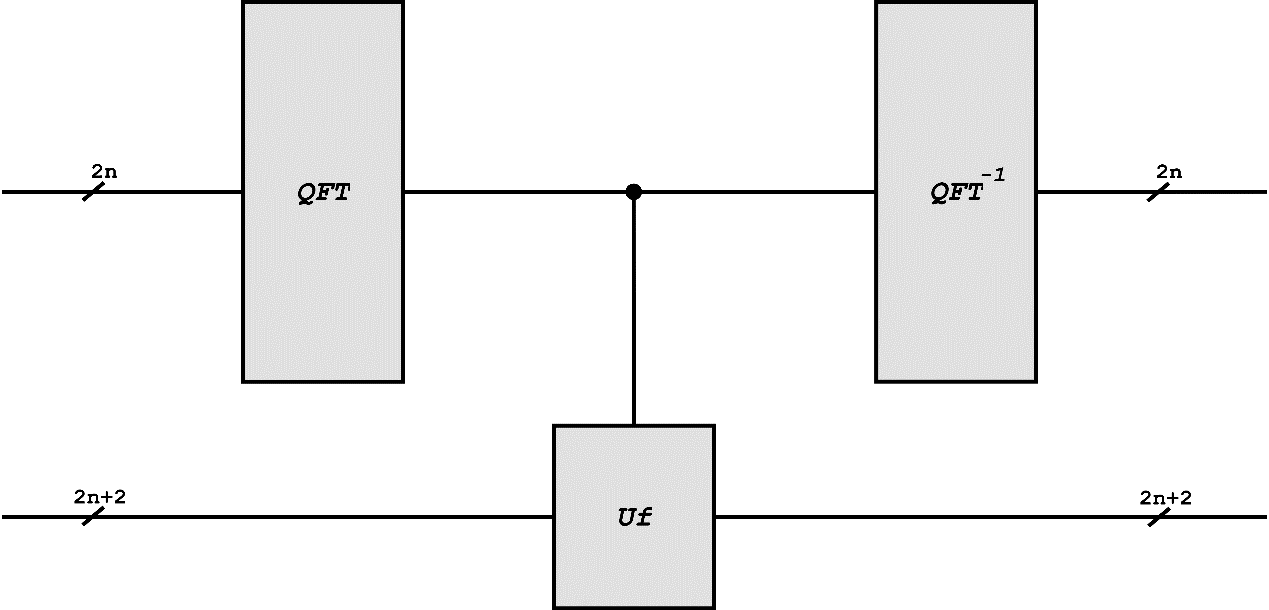
Notice that the QFT on 1 qubit is equal to the Hadamard transformation.

**Example 2:**

This is another example where qubits. This means that and . In this case the unitary QFT matrix is:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |

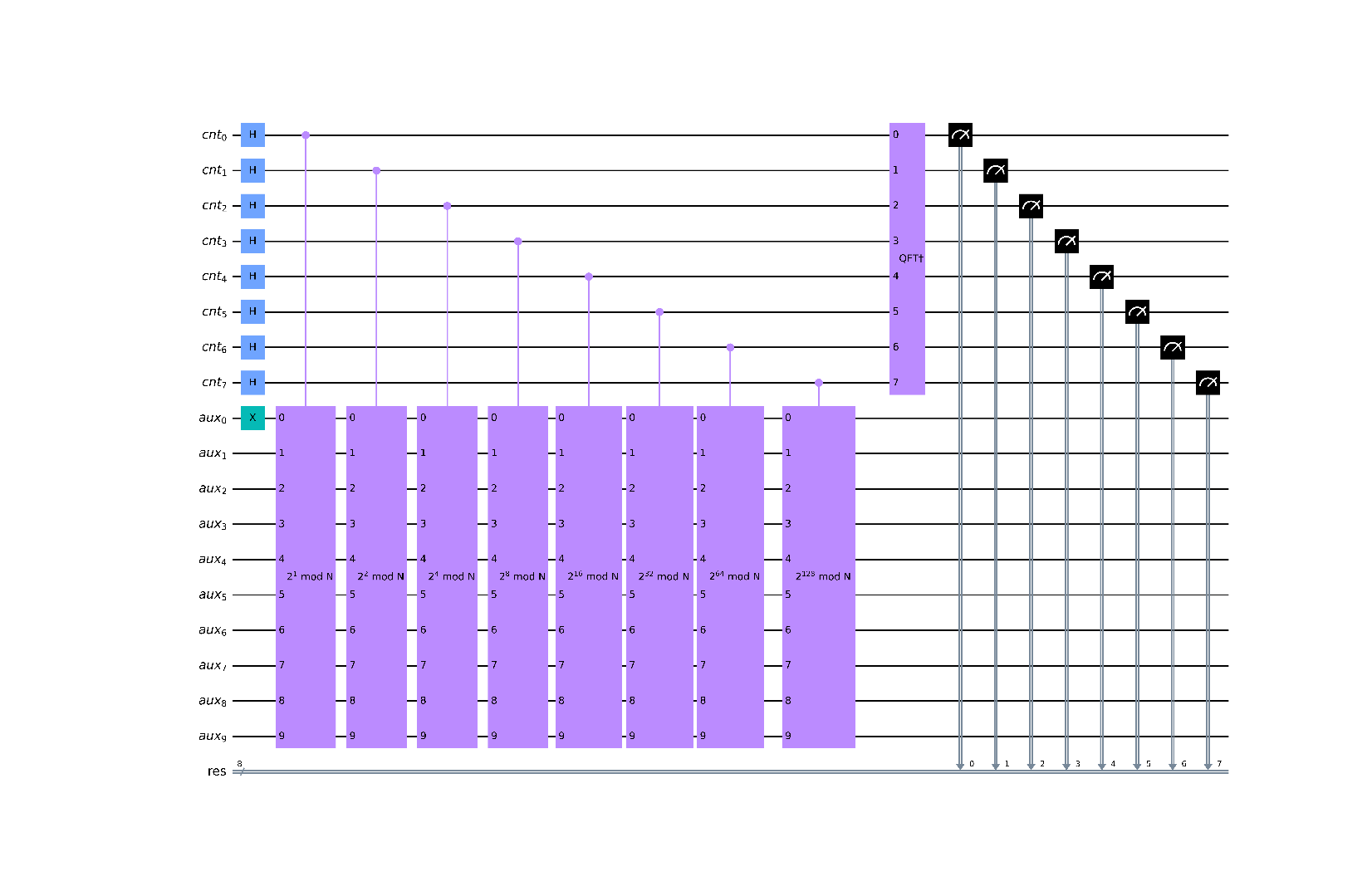
**CHAPTER III – SHORS ALGORITHM**



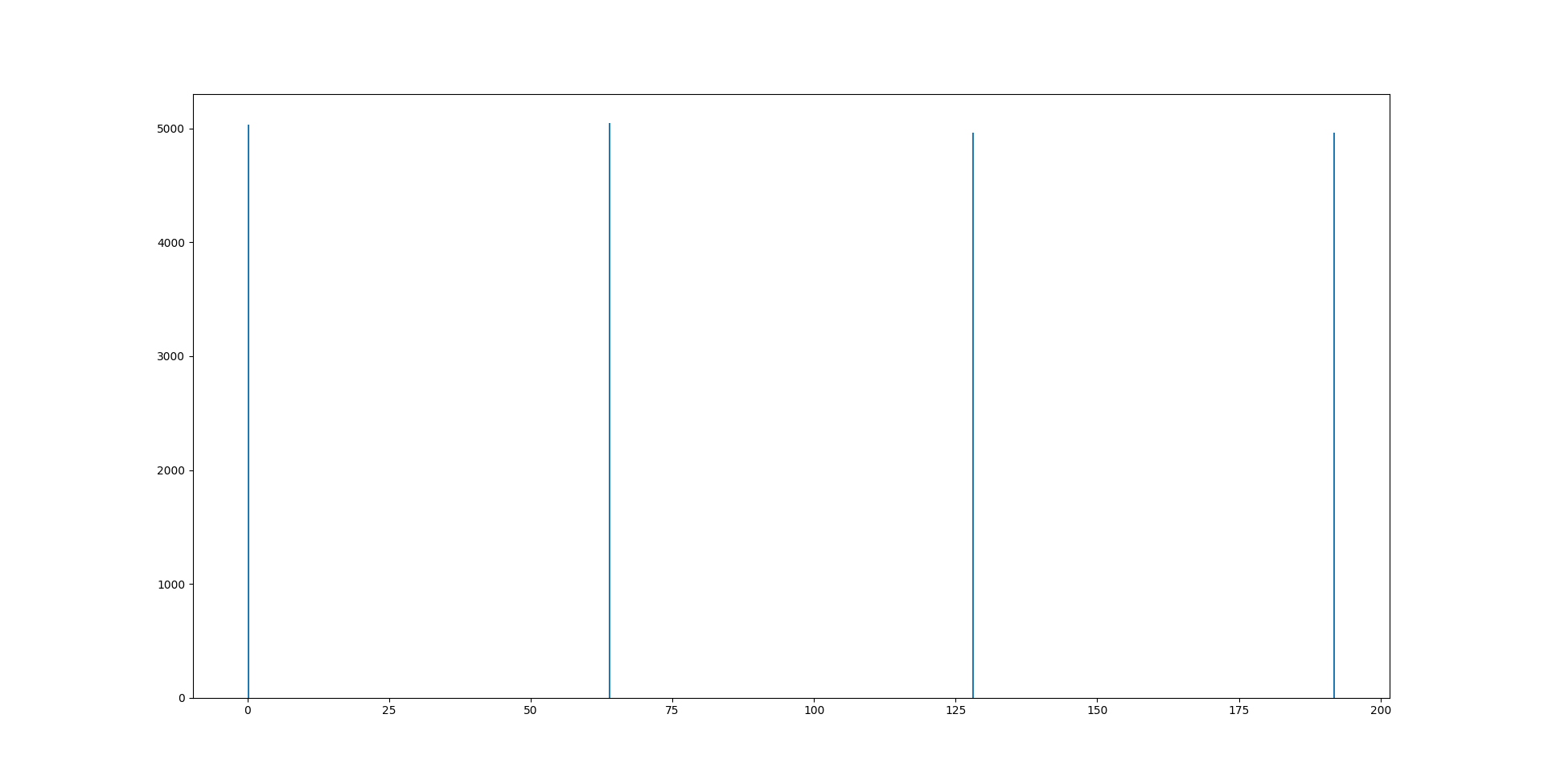
**CHAPTER IV – SIMULATION**

**Example 1:**

Shor’s circuit for and



*Quantum circuit*



*Measurement results*



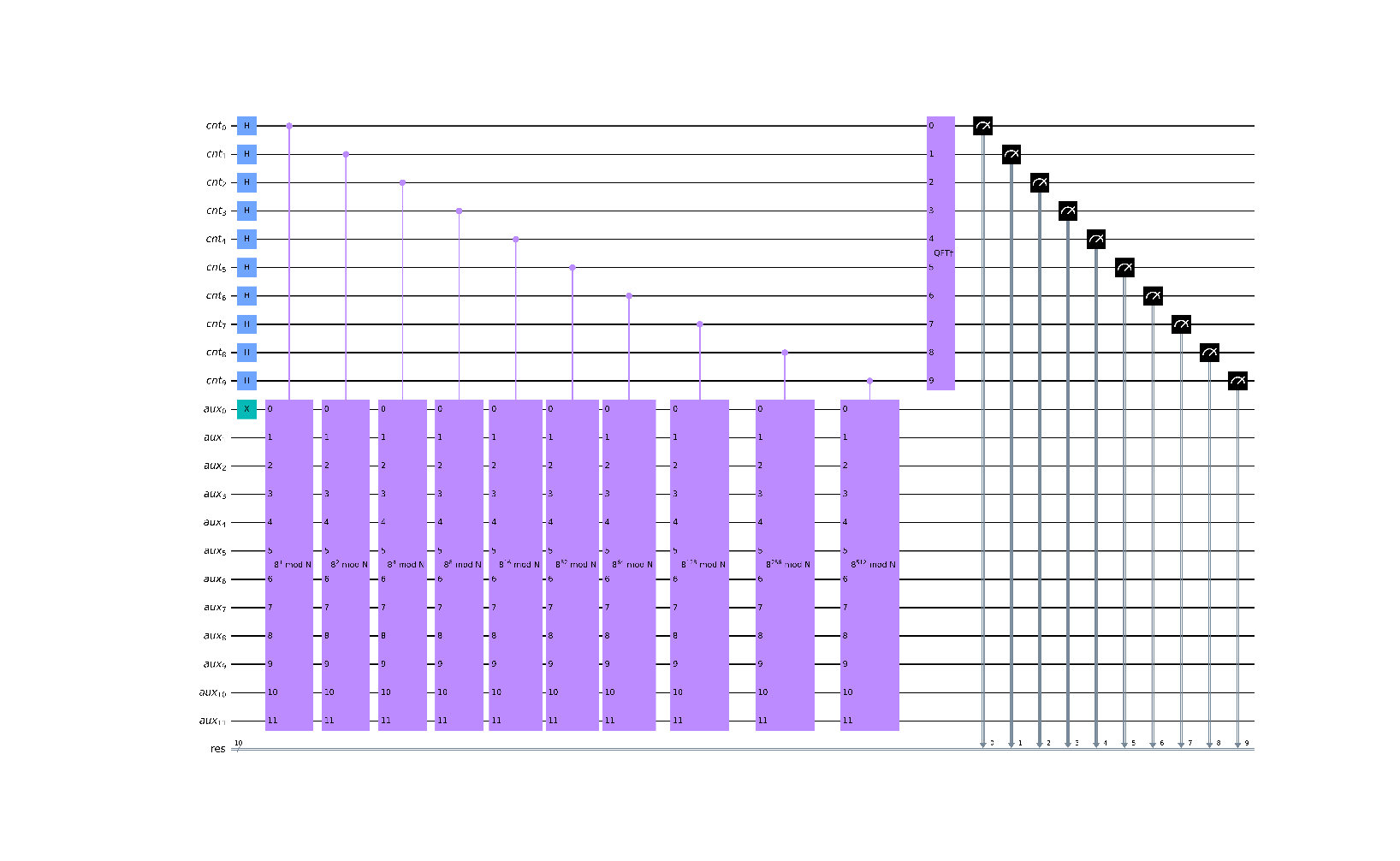
*Plot of best guesses*

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

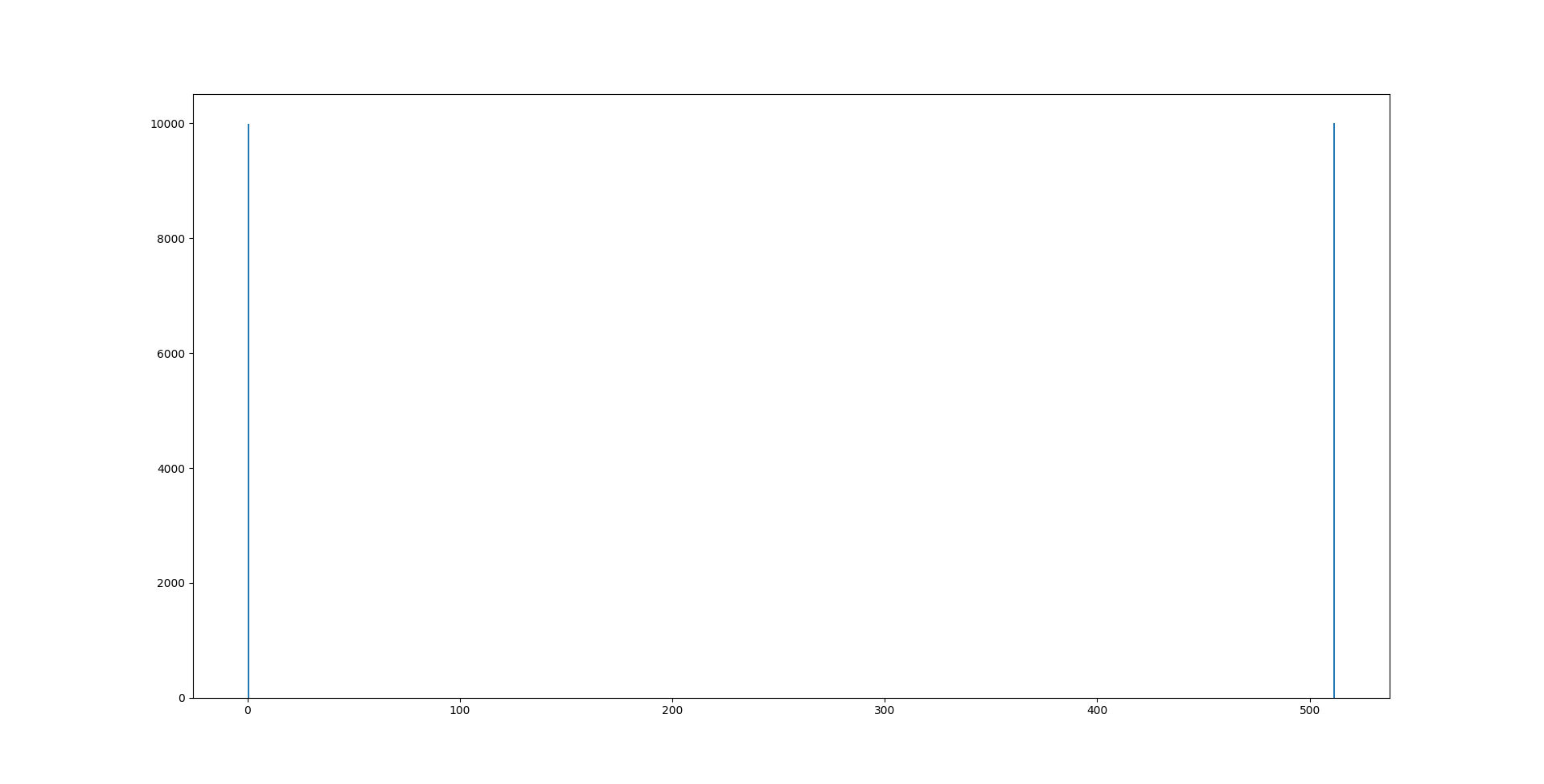
Therefore the factors of are and

**Example 2:**

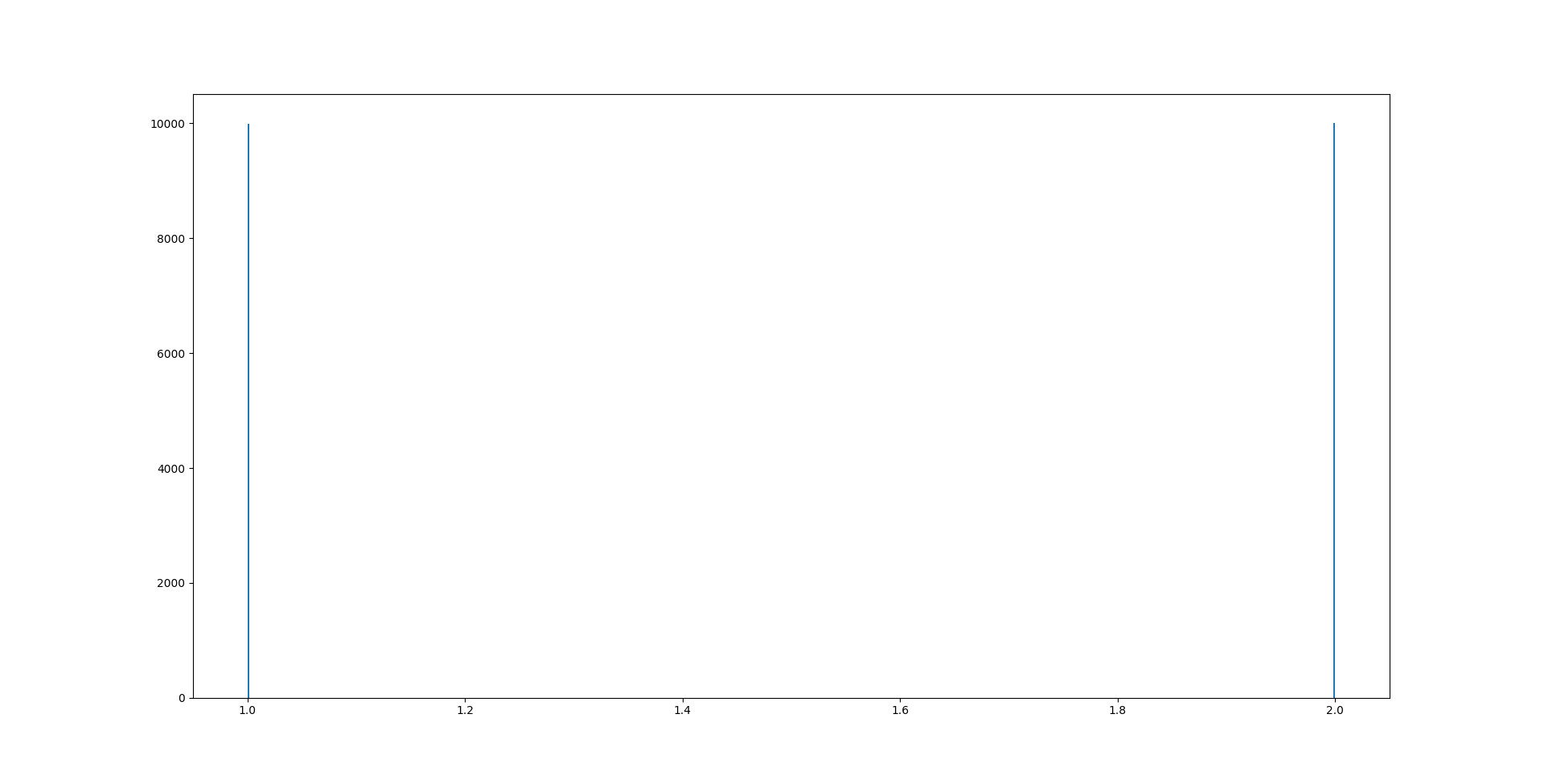
Shor’s circuit for and



*Quantum circuit*



*Measurement results*



*Plot of best guesses*

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

The first factor is

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:

Therefore the factors of are and

**Example 3:**