Shor's Algorithm

Pranish Bhagat

August 2021

Abstract

The security of messages encoded via the widely used RSA public key encryption system rests on the enormous computational effort required to find the prime factors of a large number N using classical (conventional) computers. In 1994 Peter Shor showed that for sufficiently large N, a quantum computer could perform the factoring with much less computational effort. This project endeavors to explain, in a fashion comprehensible to the nonexpert, the RSA encryption protocol; the various quantum computer manipulations constituting the Shor algorithm; how the Shor algorithm performs the factoring; and the precise sense in which a quantum computer employing Shor's algorithm can be said to accomplish the factoring of very large numbers with less computational effort than a classical computer. It is made apparent that factoring N generally requires many successive runs of the algorithm. My analysis reveals that the probability of achieving a successful factorization on a single run is about twice as large as commonly quoted in the literature.

1 Introduction

In this project, we will be talking about encryption and implementing one of the most famous algorithms in the field: Shor's Algorithm. Shor's Algorithm is the premier example of a quantum algorithm that shows the power of quantum computation compared to classical computation today. This is because Shor's Algorithm provides a nearly exponential speed-up over the fastest known classical algorithm in finding prime factors of large numbers. But why is this so pertinent? Finding factors for large numbers is the basis of RSA encryption, which is a public key sharing scheme used for secure data transactions like credit card transactions. Under RSA encryption, messages can be encrypted with a code called a public key, which can be shared openly. Once a message has been encrypted with the public key, which is a very large number, it can only be decrypted by a private key, which is the prime factor of that large number. The larger the number, the more computational power it takes to find a prime factor and the more secure the encryption is. So the security of RSA relies on the fact that finding a prime factor of a large number, takes an extremely long time. To give you an idea of how long this would take, lets take the example of a 232 decimal digit semi prime number. A semi prime number is just the product of two prime numbers and its the hardest type of number to factorize. To factorize this number, you need roughly a 1000 years of processing time on a typical laptop. Quantum computers running Shor's algorithm could one day shorten this way significantly but quantum processors today are far from being able to do this. In this project, we will focus on how to implement Shor's algorithm in Qiskit.