compat=1.14

Shor's Algorithm

Elliott Pryor, Benjamin Bushnell, REID HELP

13 November, 2019

1 The Problem

The problem of factoring large numbers has existed for centuries. Euclid's algorithm provides a very efficient way to determine the greatest common divisor of two numbers (GCD). Say we look for $GCD(a, b)$ then a is a factor of b if the $GCD(a, b) > 1$. If we have one of the factors, it is very easy to find the other factor as we can just divide the number by its factor. However, factoring very large numbers using Euclid's algorithm is very time consuming as we have to try every single number less than the value we are factoring. We can make a few improvements to our guesses; however, classically this problem cannot be solved in polynomial time.

This is very important to modern security. We use RSA encryption as a way to encrypt our data.

For sections 2 and 3 assume we are trying to find:

$$ab = n$$

where a and b are factors of n and $a, b, c \in \mathbb{N}$.

2 Classical Computation

The classical part of this algorithm is really simple. We just have to guess a random number as our factor. Then we use Eucild's algorithm to verify that we didn't get extremely lucky and guess a factor. If we guess a factor, then we don't need to use the quantum part of the algorithm. Then we feed our guess (which we know does NOT share factors with the number in question) into the quantum algorithm.

3 The Quantum Algorithm

The quantum part of this algorithm is what turns the random number that we guessed into the actual factor. In short, it does this by finding the period of some function that we can relate to the factors. Given that that is a gross oversimplification of what happens, we will cover some of the math supporting Shor's Algorithm.

First, we must define a few relations.

Given $a, b \in \mathbf{Z}$