An Undergraduate Introduction to Post-Quantum Cryptography

Juan Angel Luera
Quantum Physics III
(Dated: January 24, 2026)

In 1994 Mathematician Peter Shor announced his new factoring algorithm, now known as Shor's Algorithm. This algorithm was a proposition of how to factor numbers using quantum computers. Almost 30 years later the consequences of this discovery are more real than ever and compromise the cryptographic infrastructure that modern computing is based on. On this paper I aim to thoroughly introduce these concepts at an undergraduate level of physics understanding.

I. INTRODUCTION TO POST-QUANTUM CRYPTOGRAPHY

Most of current day encryption algorithms are cipher algorithms. In other words these algorithms use a key to encrypt data and use the same or another key to decrypt it. Algorithms that use the same key to encrypt and decrypt data are called symmetric-key algorithms. These type of algorithms have been in usage for millennia for various purposes. In general this type of cryptography is considered to be safe, but there's one major draw back. How do you send a key safely? For the message to be decrypted it's necessary that both the sender and the receiver of the message share a common key. If there are no safe channels of communication or if the key somehow gets leaked the whole system falls apart[4].

In 1976 the first protocol for public-key cryptography or asymmetric encryption was described by Whitfield Diffie and Martin Hellman. Public-key encryption works by having two keys both of which can used to decrypt or encrypt (Let's call them public and private) with the caveat that whatever is encrypted by the public key is only decrypted by the private key and vice-versa. This asymmetric construction means there is no need to transmit a key safely. Furthermore, this construction can be used in authentication protocols. A popular application of this idea is Rivest-Shamir-Adleman (RSA) encryption. At a simple level RSA works by having a large public key N which is the product of 2 prime numbers[3]

$N = p \cdot q$ where p and q are two distinct large prime numbers

and the private key is one of the two digits p or q. The reason why this work relies on factoring. Factoring numbers is quite a difficult endeavor even for our most powerful supercomputers and is classified as nondeterministic polynomial time (NP), so effectively the private key is hidden as a factor of N. In fact, the fastest algorithm that we have, the number field sieve runs exponentially slower than Shor's algorithm.[6]

Number Field Sieve run time: $exp(\Theta(n^{1/3}\log^{2/3} n))$
Shor's Algorithm run time: $O(n^2 \log n \log\log n)$

As time passed we have developed new encryption methods, but a lot of these are vulnerable to Shor's algorithm similarly to RSA.

This did not go unnoticed, over the last couple of years there have been efforts to create encryption algorithms that are quantum resistant. In fact, in 2016 the national institute of standards and technology (NIST) made request for Nominations of public key post-quantum cryptographic algorithms.[2]

II. HOW SHOR'S ALGORITHM WORKS

There are several ways to understand Shor's Algorithm. One of the simplest is to reduce it to the problem of order finding.
Recall that to solve public key encryption we need to factor a large number N which is a product of two large prime numbers $N = p * q$
Now take an arbitrary number $x \leq N - 1$ that is coprime to N (it's not p or q).Then define r to be a number such that $x^r \mod N = 1$. For an even r and $x^{r/2} \mod N \neq 1$ there is a strong probability that $\gcd(x^{r/2} \pm 1 \mod N, N)$ are p and q. From here it becomes obvious that to solve the factoring algorithm we need to find r, in other words we need to solve for the order of x.
It's not immediately obvious how such a problem can be solved, so it's often useful to decompose the problem into smaller bits. One of the bits that needs to be understood before delving into the details of Shor's Algorithm is the quantum fourier transform (QFT).

Quantum Fourier Transform

$$|j\rangle \rightarrow \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} e^{2\pi i jk/N} |k\rangle \tag{1}$$

Inverse Quantum Fourier Transform

$$|k\rangle \rightarrow \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} e^{2\pi i jk/N} |j\rangle \tag{2}$$

The QFT is very similar to the discrete fourier transform (DFT) of classical computing. It takes a discrete number of quibits/bits from one space to another

(time,space, etc to frequency domain). However, what's not similar it's the construction of such transform. While on classical computing it's trivial to build systems that perform sums over over large spaces, the implementation of such logic is not as clear for Quantum systems. A big challenge is how to implement a gate that acts on the whole space. Well, it turns out that with a clever representation of the QFT you can implement it with two distinct gates the Hadamard and the controlled R_k

$$\textit{Hadamard Gate } \tfrac{1}{\sqrt{2}}\begin{pmatrix}1 & 1\\ 1 & -1\end{pmatrix}$$

$$R_k \textit{ Gate } \begin{pmatrix}1 & 0\\ 0 & e^{2\pi i/2^k}\end{pmatrix}$$

If we take N to be 2^n we can represent the state $|j\rangle$ in a binary representation $|j_1 j_2 ... j_n\rangle$. It's also important to consider the binary fraction representation $0.j_l j_{l+1} ... j_m = j_l/2 + j_{l+1}/2^2 + ... + j_m/2^{m-l+1}$. On this representation the QFT can be given in the following product representation after some algebra.

$$|j_1 j_2 ... j_n\rangle \rightarrow \frac{(|0\rangle + e^{2\pi i 0.j_n}|1\rangle)(|0\rangle + e^{2\pi i 0.j_{n-1}j_n}|1\rangle)...(|0\rangle + e^{2\pi i 0.j_1 j_2 ... j_n}|1\rangle)}{2^{n/2}}$$

This representation of the quantum fourier transform is useful because it permits us to construct efficient quantum circuits.

Notice that if you take the $|j_1 j_2 ... j_n\rangle$ state and apply a Hadamard (H) to the first bit you get

$$\tfrac{1}{2^{1/2}}(|0\rangle + e^{2*\pi i 0.j_1}) |j_2 j_3 ... j_n\rangle$$

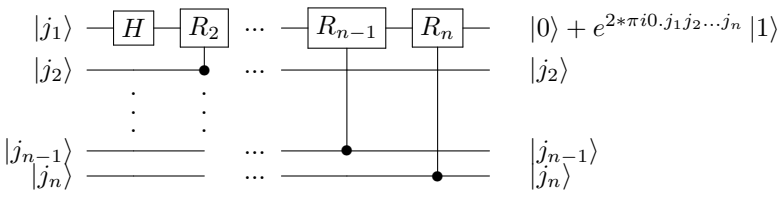
since $H|j_1\rangle$ will equal $(|0\rangle + |1\rangle)$ if j_1 is equal to 0 and $(|0\rangle - |1\rangle)$ if j_1 is equal to 1. If you apply a controlled R_2 gate to this state you obtain.

$$\tfrac{1}{2^{1/2}}(|0\rangle + e^{2*\pi i 0.j_1 j_2}) |j_2 j_3 ... j_n\rangle$$

Continue applying the controlled R-gates R_3, R_4 ... R_n and you are left with the state.

$$\tfrac{1}{2^{1/2}}(|0\rangle + e^{2*\pi i 0.j_1 j_2 ... j_n}|1\rangle) |j_2 j_3 ... j_n\rangle$$

Quantum circuit for one bit QFT



Lastly, do this for all the bits and you will recover the Quantum Fourier Transform.

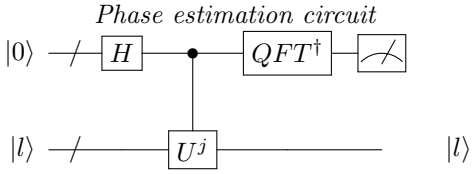
Phase Estimation Algorithm

The QFT is the key for the next step in the order finding algorithm, phase finding. Suppose you have a state $|l\rangle$ that is an eigenstate of some operator U with eigenvalue $e^{2\pi i\theta}$. The goal of the phase finding algorithm is to find this θ

$$U|l\rangle = e^{2\pi i\theta}|l\rangle$$

Assuming you can construct a controlled U^{2^j} then it is possible to find the phase θ. The first step is to initiate a state with two registers $|0\rangle^{\otimes n}|l\rangle$ for an arbitrary length $|l\rangle$. Next, apply a Hadamard transform to the $|0\rangle^{\otimes n}$, your state is now $|+\rangle^{\otimes n}|l\rangle$. Then, you apply controlled U^{2^j} on the $|l\rangle$ state from the corresponding j^{th} qubit. Due to phase kickback each of the qubits will acquire a phase on the $|1\rangle$ of the superposition state. Lastly, you apply the inverse QFT to the first register. This will take your state to $\left|\tilde{\theta}\right\rangle|l\rangle$ where $\left|\tilde{\theta}\right\rangle$ is approximately the $|\theta\rangle$ state. Measuring the first register qubits will give you approximately the phase.

Phase estimation circuit



Order Finding Algorithm

The order finding algorithm is not too distinct from the phase finding algorithm. The main distinction is that order finding acts on the unitary operator

$$U|a\rangle \equiv |xa \mod N\rangle$$

Consider the states defined by

$$|u_s\rangle \equiv \tfrac{1}{\sqrt{r}}\sum_{k=0}^{r} exp[\tfrac{-2\pi isk}{r}]\,|x^k \mod N\rangle$$

Acting on these states with the Unitary we described previously results in

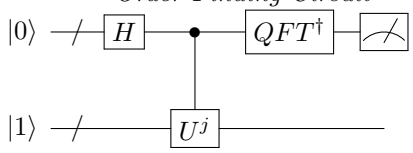
$$\begin{aligned} U|u_s\rangle &= \tfrac{1}{\sqrt{r}}\sum_{k=0}^{r} exp[\tfrac{-2\pi isk}{r}]\,|x^{k+1} \mod N\rangle \\ &= exp[\tfrac{2\pi is}{r}]\,|u_s\rangle \end{aligned}$$

From this observation we see that using the phase estimation procedure on this unitary and eigenstate will return a phase $\frac{s}{r}$ .For us to use the phase estimation procedure to solve this problem there two requirements that need to be met. First we need a way to perform U^{2^j} unitaries efficiently. There is a known way to do this called *modular exponentiation*. The second requirement is preparing the $|u_s\rangle$ state. We do not currently know any clever ways to do this, however, we can take advantage of a clever observation that.

$$\tfrac{1}{\sqrt{r}}\sum_{s=0}^{r-1}|u_s\rangle = |1\rangle$$

and have our second register qubits be $|1\rangle$ Admittedly we don't know r, so we can't initiate an r qubit state. However, as long as we have a large enough initial state we can get a good approximation.

Order Finding Circuit



After performing the phase estimation routine we obtain an approximate $\frac{s}{r}$ that we can use to obtain r through clever math. It should be pointed out there is a chance that this algorithm fails in the case that s and r aren't co-prime. However, this pretty unlikely and on average won't affect the run time too much.

Having solved for r we should in theory be able to use to factor large numbers and break all public-key factoring based encryption algorithms. This is true however not currently feasible. We are still away from building quantum systems large enough that they could factor the numbers we use for encryption. Moreover, this algorithm only cracks factoring based systems.

III. QUANTUM KEY DISTRIBUTION

Now that we understand how Shor's algorithm works and its limitations we can discuss ways to circumvent it. If we recall, the symmetric key algorithms that we discussed earlier aren't prone to the same weaknesses as public-key algorithms. Rather their weakness is how to transmit private keys safely, assuming an unsafe channel of communication. One such proposition is quantum key distribution (QKD) protocols. QKD protocols are used to transmit secret keys through unsafe channels using quantum effects. The first protocol invented that accomplished this was proposed by Charles H. Bennet and Giles Brassard in 1984.

BB84 protocol

Suppose Alice wants to send a secret message to Bob. Alice encrypts the message, but now needs to send a key to Bob safely, so he can de-crypt this message. However a third person Eve is secretly eavesdropping on their conversation. Eve can tamper with their communication and pretend to be either Bob or Alice. Assuming there exists a quantum channel and both Alice and Bob have quantum computers of high fidelity, it's possible for Alice to send a key secretly. The first step is for Alice to prepare a sequence of N qubits which are in either the $\{0, 1\}$ or $\{+, -\}$ basis randomly. (example below)

Alice prepares states

$|0\rangle \ |1\rangle \ |+\rangle \ |+\rangle \ |0\rangle \ |0\rangle \ |-\rangle \ |-\rangle \ |1\rangle \ |0\rangle \ |+\rangle \ |0\rangle$

Then Alice sends these states to Bob through the quantum channel. Bob then measures these states randomly in in the $\{0, 1\}$ or $\{+, -\}$ basis.

Bob measures in

$0/1 + / - + / - \ \ 0/1 + / - 0/1 \ \ + / - 0/1 \ \ 0/1 + / - \ \ + / - 0/1$

Bob gets

$|0\rangle \ |+\rangle \ |+\rangle \ |0\rangle \ |-\rangle \ |0\rangle \ |-\rangle \ |1\rangle \ |1\rangle \ |+\rangle \ |+\rangle \ |0\rangle$

Next, Alice announces the basis of each of her stated, and Bob let's her know on which states he measured with the right basis. Lastly, they discard the states that were measured with the wrong basis.

States where they agree

$|0\rangle \quad |+\rangle \quad |0\rangle \ |-\rangle \quad |1\rangle \quad |+\rangle \ |0\rangle$

Then they take these left over states (which for a long enough string of bits should be approximately N/2) and measure and compare enough bits so that they have k qubits left. By comparing the results they can realize whether their communication has been tampered with. If all the bits match it means no one was eavesdropping on their communication and they use the leftover bits to as the private key.

Fixing BB84

The BB84 protocol is not perfect, for example any error would completely invalidate your results. Besides it not being robust, it also requires technology that doesn't currently exist. For the protocol to work one would need quantum memories and quantum channels that transmit data over long distances without dephasing. However, it provides a fundamental idea of protocols that are quantum resistant (as far as we know). New propositions of QKD protocols utilize similar concepts of using quantum effects to transmit secret keys. These new protocols also incorporate ideas of fault tolerant quantum computing and quantum correction codes to increase the robustness. While still far away from being used practically QKD protocols are an interesting and promising area of research that could be the solution to our cryptographic dilemma.

IV. FUTURE OF CRYPTOGRAPHY

Shor's algorithm is an incredible algorithm that demonstrates the power of quantum computing. Although, it's discovery does pose a threat to our current cryptographic infrastructure we already have possible solutions. Lattice based algorithms are a promising form of public-key based classical algorithms that offer a quantum resistant solution as far as we are aware. Even if down the line we discover quantum algorithms that break our new classical solutions we can also use quantum based or at least quantum aided systems of encryption (eg. QKD). It's too soon to say what will be the long term solution to our encryption conundrum, but with all the different research and solutions that are being proposed it's almost certain we will find one.

Acknowledgments

[1] Bernstein and Lange, *Post-Quantum Cryptography* (Nature, 2017), page 188-194

[2] Computer Security Division, *Post-quantum cryptography*,(CSRC 2016)

[3] Diffie and Hellman, *New Directions in Cryptography*, (1976)

[4] IBM, *Cipher algorithm and keys* (https://www.ibm.com/docs/en/ztpf/1.1.0.15?topic=concepts-cipher-algorithms-keys)

[5] Ioannou and Mosca, *A new spin on quantum cryptography: Avoiding trapdoors and embracing public keys* ,(2011), pages 255-274

[6] Nielsen and Chuang, *Quantum Computation and Quantum Information 10th Anniversary Edition* (Cambridge University Press, 2010), page 216-247

[7] Peter Shor, 8.370 ,(Lecture 13 and 8)