Quantum Computing

Roger Fan

December 11, 2022

1 Introduction

In classical physics, physical systems are always assumed to be in one state at a time. However, modern quantum mechanics tells us that this is not actually how the world works. Instead, quantum systems can be in a *superposition* of many different *classical states* at the same time, and when we measure the system, the system then collapses into one of these classical states. In practice, we model a *pure quantum state* $|\phi\rangle$ as a linear combination of its N classical states, which are written as $|0\rangle, |1\rangle, \ldots, |N-1\rangle$:

$$|\phi\rangle = \alpha_0 |0\rangle + \alpha_1 |1\rangle + \cdots + \alpha_{N-1} |N-1\rangle$$

Each coefficient α_i is called the amplitude of $|i\rangle$ in $|\phi\rangle$, and is a *complex number*. These quantum states are really vectors in the N-dimensional Hilbert space spanned by the orthonormal basis vectors $|0\rangle, |1\rangle, \ldots, |N-1\rangle$, and we may write $|\phi\rangle$ as a column vector:

$$|\phi\rangle = \begin{bmatrix} \alpha_0 \\ \vdots \\ \alpha_N \end{bmatrix}$$

The conjugate transpose of $|\phi\rangle$, which is denoted $\langle\phi|$, is also of importance.

$$\langle\phi| = \begin{bmatrix} \overline{\alpha_0} & \overline{\alpha_1} & \cdots & \overline{\alpha_{N-1}} \end{bmatrix}$$

We call $|\phi\rangle$ a "ket", and $\langle\phi|$ a "bra". We may put these two together to make $\langle\phi|\psi\rangle$, called a "bracket," which is the dot product $\langle\phi| \cdot |\psi\rangle$. Note that $\langle\cdot|\cdot\rangle$ is really a Hermitian inner product, which is the analogue of inner products in complex vector spaces.

1.1 Qubits

A *qubit* is a quantum bit; that is, a quantum state with only two classical states. Qubits can hold much more information than their classical counterparts, and are expressed as

$$|\phi\rangle = \alpha |0\rangle + \beta |1\rangle$$

From the theory of quantum mechanics, Born's Rule tells us that when measured, $|\phi\rangle$ collapses into the state $|0\rangle$ with probability $|\alpha|^2$ and state $|1\rangle$ with probability $|\beta|^2$. Thus $|\alpha|^2$ and $|\beta|^2$ *must sum to* 1.

The ℓ_2 norm of a complex vector $\vec{v} = (a_0, \cdots, a_{n-1})$ is $\sum_{i=0}^{n-1} |a_i|^2$, so we say that any valid qubit has a ℓ_2 norm of 1.

1.2 Tensor Product

When we work with more than one qubit, we must begin using the language of tensor products. Intuitively, we should expect this to be the case. For demonstration, say A and B are two random, classical bits, where A has a 0.5 chance of being 0 or 1, and B has a 0.3 chance of being 0 and a 0.7 chance of being 1. We may write the distributions for A and B as column vectors:

$$A = \begin{bmatrix} 0.2 \\ 0.8 \end{bmatrix} \begin{matrix} \leftarrow 0 \\ \leftarrow 1 \end{matrix} \qquad B = \begin{bmatrix} 0.3 \\ 0.7 \end{bmatrix} \begin{matrix} \leftarrow 0 \\ \leftarrow 1 \end{matrix}$$

The state of *both* A and B should be a distribution with 4 different outcomes, which we can write as a 4 dimensional probability vector

$$A \otimes B = \begin{bmatrix} 0.06 \\ 0.14 \\ 0.24 \\ 0.56 \end{bmatrix} \begin{matrix} \leftarrow 00 \\ \leftarrow 01 \\ \leftarrow 10 \\ \leftarrow 11 \end{matrix}$$

$A \otimes B$ is the *tensor product* of the two vectors A and B. This situation is completely analogous for quantum systems.

More formally, let P and Q be two quantum systems. Let B_P and B_Q be the sets of classical states of P and Q, which each are orthonormal bases of the vector spaces V and W, respectively.

Then, for all $b_P \in B_P$ and $b_Q \in B_Q$, we define $b_P \otimes b_Q$ as new orthonormal vectors, which span the space $V \otimes W$.

Let $x = \sum_{b \in B_P} x_b b$ be a quantum state of P, and let $y = \sum_{b \in B_Q} y_b b$ be a quantum state of Q. Then, the tensor product of x and y is defined as

$$x \otimes y = \sum_{b \in B_P} \sum_{c \in B_Q} x_b y_c b \otimes c$$

Example 1.2.1. *Let A and B be two qubits, which have quantum states $|\phi\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$ and $|\psi\rangle = \frac{1}{\sqrt{2}}(|0\rangle + i\,|1\rangle)$. Then, we may compute (using FOIL) that*

$$|\phi\rangle \otimes |\psi\rangle = \frac{1}{2}(|00\rangle + i\,|01\rangle - |10\rangle - i\,|11\rangle)$$

We have written $|0\rangle \otimes |0\rangle$ as $|00\rangle$ for shorthand. (Later, this will sometimes be denoted $|0\rangle\,|0\rangle$.) Note that $|00\rangle, |01\rangle, |10\rangle, |11\rangle$ are classical states of the quantum system including both *A and B. $|00\rangle$ is the classical state where both A and B have value* 0.

We call systems of n qubits a *n-qubit register*. However, not all states of n-qubit registers can be represented as tensor products. *Entanglement* is the quantum phenomenom when two qubits affect each other. For example, consider the 2-qubit register with state

$$\frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$$

The two qubits are always the same! Such a pair of qubits is called an EPR-pair, after Einstein, Podolsky, and Rosen.

1.3 Unitary Transformations

In classical computing, we use the logic gates AND, OR, and NOT to construct circuits. In the quantum case, there are also gates that act on qubits. In general, these gates are linear operations on qubits that can be represented as matrices. In fact, all such linear operations must be *unitary*, which means they preserve the ℓ^2 norm of vectors. Intuitively we should find this easy to believe, as these transformations take qubits to other qubits, which all must have a ℓ^2 norm of a 1. (Unitary matrices are also exactly the matrices U such that $UU^\dagger = I$, where $U^\dagger$ is the conjugate transpose of U.)

The *Hadamard* gate, or H gate for short, is a unitary transformation that acts on a single qubit. It is described by the matrix

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

In other words, H maps $|0\rangle$ to $\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$, and maps $|1\rangle$ to $\frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$. The reader may check that the Hadamard transform is its own inverse.

Another 1-qubit gate is the *phase gate* R_ϕ, which maps $|0\rangle$ to $|0\rangle$ but $|1\rangle$ to $e^{i\phi}|1\rangle$. This can be represented by the matrix

$$R_\phi = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\phi} \end{bmatrix}$$

There are some gates on two qubits, such as the *CNOT gate*. The CNOT gate takes in two qubits, and flips the second one if the first qubit is 1. As a matrix, it can be represented

$$CNOT = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

The CNOT gate may be upgraded to the *Toffoli Gate*, also known as the CCNOT gate, which takes in 3 qubits and negates the third if and only if the first two qubits are 1. In other words, it maps

$$\text{CCNOT}\,|A\rangle \otimes |B\rangle \otimes |C\rangle = |A\rangle \otimes |B\rangle \otimes |C \,\text{XOR}\, (A \,\text{AND}\, B)\rangle$$

These are all the gates that we will use. However, there are many more that exist. It is worth remembering that all quantum gates are linear operators, and to compute the effect of a quantum gate on a qubit, we often will compute what it does to the basis vectors.

Example 1.3.1. *Let a register of two qubits be in the state* $\frac{1}{\sqrt{2}}(|00\rangle + |10\rangle)$. *If* CNOT *is applied to this register, we may compute the resulting state by first computing*

$$\text{CNOT}\,|00\rangle = |00\rangle\,, \qquad \text{CNOT}\,|10\rangle = |11\rangle$$

Since CNOT *is a linear operator, we easily compute*

$$\text{CNOT}\,\frac{1}{\sqrt{2}}(|00\rangle + |10\rangle) = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$$

Often, we will have a register of multiple qubits where we will apply different gates to each of them. To compute the effects of such an operation, we use the tensor product of linear maps. Let V and W be two vector spaces, and let f and g be linear maps (which represent gates) from V to V and W to W, respectively. Then, if $v \in V$ and $w \in W$, define $f \otimes g$ as the linear map in $V \otimes W$ that maps

$$f \otimes g : (v \otimes w) \mapsto f(v) \otimes g(w)$$

f is the gate that we apply to the vector space V, and g is the gate we apply to the vector space W.

Example 1.3.2. *Given a 2-qubit register in the state $\frac{1}{\sqrt{2}}(|00\rangle+|11\rangle)$, let us compute the effect of using the Hadamard transform on the first qubit. The transformation we are applying is $H \oplus I$, as we apply H to the first qubit, and the identity transformation to the second. Then,*

$$(H \oplus I)\,|00\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \otimes |0\rangle$$

$$(H \oplus I)\,|11\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) \otimes |1\rangle$$

$$(H \oplus I)\frac{1}{\sqrt{2}}(|00\rangle + |11\rangle) = \frac{1}{2}(|00\rangle + |01\rangle + |10\rangle - |11\rangle)$$

For shorthand we will often denote $X \otimes X \otimes \cdots \otimes X$ as $X^{\otimes n}$. We conclude this section with an important example.

Example 1.3.3. *Given an n-qubit register in the state $|a\rangle$, where a is a length n binary string, what happens when we apply Hadamard gates to each qubit? The transformation is $\mathrm{H}^{\otimes n}$, and*

$$\mathrm{H}^{\otimes n}\,|a\rangle = \frac{1}{\sqrt{2^n}} \sum_{i \in \{0,1\}} (-1)^{i \cdot a}\,|i\rangle$$

Here, $i \cdot a$ is the dot product on length n binary strings. For example, $011 \cdot 110 = 1$.

1.4 Projective Measurement

We have seen that when we measure a quantum state, it collapses into a classical state. This is called a *wave function collapse*. We shall be mainly focused here on the simplified case of measuring certain qubits of a n-qubit register.

Let $\{0,1\}^n$ be the string of all binary n-digit strings. Given a state $|\phi\rangle = \sum_{i \in \{0,1\}^n} a_i\,|i\rangle$, if we measure the first qubit, by Born's rule, it will be a 1 with probability

$$\sum_{i \in S} |a_i|^2$$

Here, S is the set of all n-digit binary strings that begin with 0. What happens to $|\phi\rangle$ once we measure the first bit?

If the first bit is measured to be true, we would expect $|\phi\rangle$ to collapse into

$$|\phi\rangle \mapsto \sum_{i \in S} a_i |i\rangle$$

However, this new state does not have a ℓ^2 norm of 1, so we must fix this by dividing by a suitable factor:

$$|\phi\rangle \mapsto \frac{\sum_{i \in S} a_i |i\rangle}{\sqrt{\sum_{i \in S} |a_i|^2}}$$

Example 1.4.1. *Given the EPR-pair* $\frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$*, let us measure the first bit. The probability that the first bit is* 0 *is* $\left(\frac{1}{\sqrt{2}}\right)^2 = \frac{1}{2}$*. If we measure the first bit to be a* 1*, then* $|\phi\rangle$ *collapses to*

$$\frac{\frac{1}{\sqrt{2}} |00\rangle}{\frac{1}{\sqrt{2}}} = |00\rangle$$

This makes sense, since in any EPR-pair, if the first qubit is 0*, then the second qubit must also be* 0*.*

2 Quantum Algorithms

2.1 Quantum Circuit Model

In classical computing, a circuit is an acyclic directed graph consisting of n input nodes, which have n input bits, m output nodes, and some number of the logic gates AND, OR, and NOT. We say that a circuit computes the function $f : \{0,1\}^n \to \{0,1\}^m$ if for every length n binary string x, the circuit outputs $f(x)$.

It is known that circuits, in some sense, are equal in power to deterministic Turing Machines. That is, for every deterministic Turing machine that decides a language L in P time, it is possible to construct a circuit with n inputs that has polynomially many gates and only accepts (outputs 1 on) the length n strings in L.

Meanwhile, *quantum circuits* begin with n qubits and apply quantum gates to them *sequentially*. Notably, any polynomially-sized circuit can be simulated by a quantum circuit with polynomially many Toffoli gates, which can simulate AND, OR, and NOT gates on qubits. That is, quantum circuits are at least as powerful as classical ones.

The power of quantum computing comes with quantum parallelism, which is the idea that we may compute a function $f(x)$ on a superposition of states rather than just a single one. Say we have a classical circuit that computes the function $f : \{0,1\}^n \to \{0,1\}$. Then, we may construct an equivalent quantum circuit U using Toffoli gates that maps $|z\rangle |0^m\rangle \to |z\rangle |f(z)\rangle$ for all $z \in \{0,1\}^n$. Then, applying U to some superposition of all such z, we have

$$U\left(\sum_{z \in \{0,1\}^n} \alpha_z |z\rangle |0^m\rangle\right) = \sum_{z \in \{0,1\}^n} \alpha_z |z\rangle |f(z)\rangle$$

It seems as if we've done exponentially many computations with a single application of U! However, the values of $f(z)$ are not readily accessible to us, as they are locked inside a quantum state. After measuring the quantum state, we lose most of the information. Quantum algorithms will need to use quantum parallelism with the ideas of entanglement and interference to perform efficient computations.

2.2 Bernstein-Varizani

The inputs to quantum algorithms are not usually qubits, but rather *oracles*, which are black-boxed quantum gates. This may seem rather strange at first, but results will easily follow from this setting later.

We begin with the Bernstein-Varizani problem, which was one of the earliest quantum algorithms discovered.

Problem 2.2.1 (Bernstein-Varizani)**.** *Let $a \in \{0,1\}^n$ be unknown. For each $i \in \{0,1\}^n$, let $x_i = a \cdot i \pmod 2$. We are given an oracle O that maps*

$$O|i\rangle|b\rangle \to |i\rangle|b \operatorname{XOR} x_i\rangle$$

Compute a.

With a classical circuit, a simple algorithm would need n queries to O_x for information-theoretic reasons. For example, we could ask the oracle to compute $O|100\cdots0\rangle$, which would give us a_0, the first digit of a. Then, we could ask to compute $O|010\cdots0\rangle$, and so on, to compute all n digits.

However, things become quite different with quantum circuits. Construct a new qubit $|-\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$, and compute $O|i\rangle|-\rangle$:

$$O|i\rangle|-\rangle = |i\rangle \frac{1}{\sqrt{2}}(|x_i\rangle - |1-x_i\rangle) = (-1)^{x_i}|i\rangle|-\rangle$$

Let us construct a new oracle $O_\pm$ that takes in $|i\rangle$ and spits out the qubits of $O|i\rangle|-\rangle$ excluding the final qubit $|-\rangle$. That it, $O_\pm|i\rangle$ is $(-1)^{x_i}|i\rangle$.

Construct n qubits $|0\rangle$, for an initial state of $|0^n\rangle$. Apply Hadamard gates to each of the n qubits, which gives

$$\mathrm{H}^{\otimes n}|0^n\rangle = \frac{1}{\sqrt{2^n}} \sum_{i\in\{0,1\}^n} |i\rangle$$

Now, apply the oracle $O_\pm$ to these qubits, which gives

$$O_\pm\left(\frac{1}{\sqrt{2^n}} \sum_{i\in\{0,1\}^n} |i\rangle\right) = \frac{1}{\sqrt{2^n}} \sum_{i\in\{0,1\}^n} (-1)^{x_i}|i\rangle$$

Recall $x_i = a \cdot i$. The above expression looks familiar; in fact, it is exactly $H^{\otimes n}|a\rangle$! $H^{\otimes n}$ is its own inverse, so if we apply Hadamard transforms to each of our qubits, our qubits must now be in the state $|a\rangle$. By measuring each qubit, we have now computed a, and we are done.

Note that we have computed a by using only one query to O, which is significantly faster than classical algorithms. This problem may not seem very impressive, but this is only the beginning.

Quantum Fourier Transform

The *quantum fourier transform* computes the fourier transform of a qubit, in some sense. Formally, let $\omega_N = e^{2\pi i/N}$ be an N-th root of unity, and define the matrix F_N as the matrix that has ω_N^{jk} as its entry in cell (j, k):

$$F_N = \frac{1}{\sqrt{N}} \begin{bmatrix} & \vdots & \\ \cdots & \omega_N^{jk} & \cdots \\ & \vdots & \end{bmatrix}$$

F_N is a unitary transformation, and if $N = 2^n$, then it can be viewed as a quantum gate on n qubits. (In fact, note $F_2 = H$.) For the basis states $|k\rangle$, where $k \in \{0, 1\}^n$, we note that

$$F_N |k\rangle = \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} \omega_N^{jk} |j\rangle$$

If the reader is familiar with the discrete fourier transform, they must note that the quantum fourier transform does something fundamentally different. Rather than compute the discrete fourier transform, the quantum fourier transform encodes the information in a system of qubits.

Let $|k\rangle$ be a basis vector, so that k is a length n string. Again, let $N = 2^n$. For any integer j in binary, written as $j_1 \dots j_n$, $\frac{j}{2^n} = \sum_{l=1}^{n} j_l 2^{-l}$. Using this, we may cleverly observe that

$$\begin{aligned} F_N |k\rangle &= \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} e^{2\pi i jk/2^n} |j\rangle \\ &= \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} e^{2\pi i k(\sum_{l=1}^{n} j_l 2^{-l})} |j_1 \cdots j_n\rangle \\ &= \frac{1}{\sqrt{2^n}} \sum_{j=0}^{N-1} \prod_{l=1}^{n} e^{2\pi i j_l k/2^l} |j_1 \cdots j_n\rangle \\ &= \bigotimes_{l=1}^{n} \frac{1}{\sqrt{2}} (|0\rangle + e^{2\pi i k/2^l} |1\rangle) \end{aligned}$$

If the binary digits of k are $k_1 k_2 \dots k_n$, note $e^{2\pi i k/2^l} = e^{2\pi i 0.k_{n-l+1} \dots k_n}$. In the $n = 3$ case,

$$F_8 |k_1 k_2 k_3\rangle = \frac{1}{\sqrt{2}} (|0\rangle + e^{2\pi i 0.k_3} |1\rangle) \otimes \frac{1}{\sqrt{2}} (|0\rangle + e^{2\pi i 0.k_2 k_3} |1\rangle) \otimes \frac{1}{\sqrt{2}} (|0\rangle + e^{2\pi i 0.k_1 k_2 k_3} |1\rangle)$$

We now describe an algorithm to compute F_N of $|k\rangle$ using quantum gates. Define a new gate, the R_s gate, which acts on a single qubit:

$$R_s = \begin{bmatrix} 1 & 0 \\ 0 & e^{2\pi i/2^s} \end{bmatrix}$$

The *controlled-R_s* gate is a gate that takes in 2 qubits and applies R_s to the second qubit if and only if the first qubit is 1.

We begin with n qubits in the states $|k_1\rangle, \ldots, |k_n\rangle$, which also is the state $|k\rangle$. Apply the Hadamard transform on $|k_1\rangle$ to get

$$\mathrm{H}\,|k_1\rangle = \frac{1}{\sqrt{2}}(|0\rangle + (-1)^{k_1}\,|1\rangle) = \frac{1}{\sqrt{2}}(|0\rangle + e^{2\pi i 0.k_1}\,|1\rangle)$$

Now apply the controlled-R_2 gate on the first qubit, with the second qubit $|k_2\rangle$ as the control. That is, this gate will apply R_2 on the first qubit if k_2 is 1. The first qubit becomes

$$\frac{1}{\sqrt{2}}(|0\rangle + e^{2\pi i 0.k_1}\,|1\rangle) \mapsto \frac{1}{\sqrt{2}}(|0\rangle + e^{2\pi i 0.k_1 k_2}\,|1\rangle)$$

We will continue on in this fashion: use controlled-R_s gates on the first qubit, with the s-th qubit as the control. After performing these gates, we end up with the qubit

$$\frac{1}{\sqrt{2}}(|0\rangle + e^{2\pi i 0.k_1 k_2 k_3 \cdots}\,|1\rangle)$$

(As a spoiler, we have constructed the last qubit in the fourier transform of $|k\rangle$.) We now move on to the second qubit, $|k_2\rangle$. Apply the Hadamard transform:

$$\mathrm{H}\,|k_2\rangle = \frac{1}{\sqrt{2}}(|0\rangle + (-1)^{k_2}\,|1\rangle)$$

For each $s > 2$, apply the controlled-R_s gate on the second qubit, with the s-th qubit $|k_s\rangle$ as the control. After these gates, the second qubit will be in state

$$\frac{1}{\sqrt{2}}(|0\rangle + (-1)^{k_2 k_3 \cdots}\,|1\rangle)$$

(This is the second to last qubit in the fourier transform of $|k\rangle$.) We now proceed to the third qubit, which we analogously transform into $\frac{1}{\sqrt{2}}(|0\rangle + (-1)^{k_3 k_4 \cdots}\,|1\rangle)$, and so on. After doing this for all qubits, we are done, except that the qubits are all in reverse order. Reverse their order, and we are done! We have used on the order of n^2 quantum gates to compute the fourier transform of $|k\rangle$ from $|k\rangle$.

2.3 Shor's Factoring Algorithm

Arguably the most important quantum algorithm is Shor's factoring algorithm, which factors an integer in time polynomial in the number of digits. There is no known classical algorithm that is nearly as fast, and if Shor's algorithm is fully realized, it could break modern cryptography, which is reliant on the fact that factoring large integers is hard for a computer to do efficiently.

We now describe Shor's algorithm. Given some integer n, and some integer x, we will find the period of x in the multiplicative group $\mathbb{Z}_n^\times$. This will allow us to factor n, which we will describe later.

There is a classical circuit that, given a, will compute $x^a \pmod{n}$ in $O(\log a)$ time using repeated squaring. Convert this circuit into a quantum circuit, and let us call it the quantum oracle O_x.

Let $q = 2^p$ be the power of 2 such that $n^2 \leq q < 2n^2$. Construct a p-register of qubits, all with state $|0\rangle$. Then, apply Hadamard transforms to each one, so that the state becomes

$$\frac{1}{\sqrt{q}} \sum_{a=0}^{q-1} |a\rangle$$

Construct now a second p-register of qubits, all initially set to $|0\rangle$. Apply the quantum oracle O_x to our two registers of qubits, so that given some integer a represented in the first register, it will transform the second register into $x^a \pmod{n}$. Our system becomes

$$\frac{1}{\sqrt{q}} \sum_{a=0}^{q-1} |a\rangle \, |x^a \mod n\rangle$$

Perform a fourier transform on the first register of qubits, so that our system becomes

$$\frac{1}{q} \sum_{a=0}^{q-1} \sum_{c=0}^{q-1} \exp(2\pi i ac/q) \, |c\rangle \, |x^a \mod n\rangle$$

Now, measure all the qubits. Let $\{rc\}_q$ be the unique integer in the interval $(-q/2, q/2]$ that is congruent to rc modulo q. Shor shows in his original paper that for sufficiently large n, if $-r/2 \leq \{rc\}_q \leq r/2$, the probability of measuring any state $|c\rangle \, |x^a \mod n\rangle$ is at least $\frac{1}{3r^2}$.

If $-r/2 \leq \{rc\}_q \leq r/2$, then there exists some d such that

$$-r/2 \leq rc + dq \leq r/2$$

This is equivalent to

$$\left| \frac{c}{q} - \frac{d}{r} \right| \leq \frac{1}{2q}$$

Two fractions with denominators at most n are always at least $\frac{1}{n^2}$ away from each other. $r < n$, so it follows that $\frac{d}{r}$ is the *unique* fraction with denominator at most n that satisfies $\left| \frac{c}{q} - \frac{d}{r} \right| \leq \frac{1}{2q}$. We know c and q, so we may compute $\frac{d}{r}$ by using the *continued fraction expansion* of $\frac{c}{q}$, which computes the best fractional approximations to $\frac{c}{q}$.

Once we compute $\frac{d}{r}$, the denominator r is the period we wish to find. This only works, of course, if d and r are relatively prime, but this happens reasonably often. This algorithm from start to finish has a success probability of at least $\frac{1}{\log \log r}$, so we expect to repeat it on the order of $\log \log r$ times. Each repetition takes polynomial time in $\log n$, which makes it very efficient.

Let us now use this to factor n. This will not work when n is even or when n is a prime power, but these cases are efficiently computable by classical computers anyways. Let us choose an arbitrary $x < n$, and compute its period r. Repeat this process until we find some

x with even period, which has been shown to happen at least $\frac{1}{2}$ of the time. Because $x^r = 1 \pmod{n}$, we may factor this as

$$(x^{r/2} + 1)(x^{r/2} - 1) = x^r - 1 = 0 \pmod{n}$$

It can also be shown that $\gcd(x^{r/2} + 1, n)$ is a nontrivial divisor of n more than half the time as well. Thus, we may use repeat this algorithm to find a divisor of n, which is expected to take only polynomial time in $\log n$. This is exponentially faster than any known classical algorithms.

References

[1] Birgitta Whaley, Kevin Young, Mohan Sarovar. *Measurement in Quantum Mechanics*

[2] Ronald de Wolf. *Quantum Computing: Lecture Notes*

[3] Daniel Grier. *Quantum Complexity Theory*