HIGH SCHOOL EDITION
Journal of Student Research

Volume 13 Issue 1 (2024)

Implementation of Shor's Algorithm and Its Demonstrated Quantum Efficiency

Chenxuan Wang

The Experimental High School Attached to Beijing Normal University, China

ABSTRACT

The unique principles of quantum physics introduce an element of uncertainty into our previously deterministic world. Quantum computing, derived from quantum physics, enables the representation of a qubit in both |0> and |1> states, thus encoding more information and unleashing a realm of new possibilities. This paper conducts an analysis of the classical RSA encryption protocol and its vulnerability to the quantum algorithm—Shor's algorithm. Furthermore, this paper rigorously establishes the mathematical theories required for the analysis. Finally, the algorithm is thoroughly explored, and the potential efficiencies of quantum computing are discussed. As the study of quantum computing deepens, its application is poised to surpass that of traditional computers in various fields, leaving a profound impact.

Introduction

Compared with classical computers, the process of quantum computing involves some special physical properties about particles which gives quantum computing a unique advantage. The two most important ones are superposition and entanglement.

The nature of superposition is uncertainty, which is what makes quantum physics so different from classical physics. In classical physics, everything is predictable. Newton's second law is the basis for the derivation of most formulas in classical mechanics. In the quantum world, however, this is not the case. According to the current research, when not being observed, each particle manifests as a probability wave and only randomly collapses into a definite state after observation. In the case of a computer, a classical computer can only express two states, 0 and 1. Quantum computers, on the other hand, can involve superpositions of all the different probability ratios between 0 and 1.

The superposition property of quantum is the basis of most quantum computing algorithms, making more complex calculations and problem solving possible. For example, the problem in factoring a large number is that with a classical computer, the calculation time increases exponentially with the number of digits, while with a quantum computer, the time only increases in polynomial with the number of digits[1].

At the same time, the superposition of qubits can also be applied in Key distribution[2]. By randomly selecting and sending the fixed state of the qubit or the superposition state, and randomly selecting a method of decryption, the two parties can complete the encrypted information transmission and avoid being overheard.

The other is entanglement, in which two qubits become entangled with each other. The implication is that when the two qubits collapse, their results correlate. For example, there are four maximal entangled Bell states:

$$\frac{1}{\sqrt{2}}(|00\rangle + |11\rangle, \frac{1}{\sqrt{2}}(|00\rangle - |11\rangle, \frac{1}{\sqrt{2}}(|01\rangle + |01\rangle, \frac{1}{\sqrt{2}}(|01\rangle - |10\rangle)$$

Therefore, when one of the qubits is measured, in this example if it is measured as |0>, the other will necessarily collapse to $|01\rangle$ when measured. Because these two qubits don't have a $|01\rangle$ state.

Through quantum entanglement, people have made great progress in cryptography and the transmission of information[3]. For example, when two parties share a pair of entangled qubits in advance, one party can transmit one qubit to the other in a classical channel through two classical bits, or use one qubit to transmit two classical bits, which is impossible for classical computers to accomplish.

As the theory of quantum computing is gradually upgraded, the configuration of quantum computers is becoming more and more important. Since quantum superpositions collapse rapidly when observed, quantum computers must avoid any contact with the outside world. There are two types of quantum computers that are most common today. One is to use the current direction of the LC circuit as two states, but this method requires the use of semiconductors to produce superconducting effects at low temperatures. This requires a strict cooling environment and can only be maintained for a short time. Another method is to trap an electron in specific locations through electromagnetic fields. A laser is used to change the energy of an electron by changing its orbital or spin direction as 0 or 1 states. There are also other ways to create quantum superpositions, but achieving these complex states requires the development of more powerful hardware conditions. Therefore, with the gradual improvement of the theory of quantum physics, more accurate and powerful facilities need and will ensue as well.

RSA Encryption

Introduction to RSA

The main function of RSA encryption algorithm is to realize the encryption transmission and decryption acceptance of specified information. It uses different public encryption keys and private keys for encryption and decryption. Its security mainly depends on prime factorization calculation of the large number. Since the classical computer often cannot complete the decryption in the effective time, safety is ensured.

Introduction to Relevant Math Theories

Congruence

First, we define that a and b are congruent modulo n, when a and b have the same remainder when divided by n. It is shown as below:

$$a \equiv b (\mathrm{mod}\ n)$$

Euclidean Algorithm

Second, we define greatest common divisor (gcd) as the greatest common factor number that divides two or more numbers exactly. We can use Euclidean algorithm to compute gcd：

Let $a = b \cdot q + r$, where a,b,q,r are all positive integers. To calculate the greatest common divisor $d = (a, b)$ of a and b, we must first prove that $(a, b) = (b, r)$,Let u be any number that divides both a and b:

$$a = s \cdot u, b = t \cdot u$$

Then u is also divisible by r, because$r = a - b \cdot q = s \cdot u - t \cdot u \cdot q = (s - t \cdot q)u$

Similarly, let v be any number that divides both b and r:

$$b = s' \cdot v, r = t' \cdot v$$

So v is also divisible by a, because $a = b \cdot q + r = s' \cdot v \cdot q + t' \cdot v = (s' \cdot q + t')v$,

Therefore, any common factor of a and b is any common factor of b and r, so the greatest common factor of a and b must also be equal to the greatest common factor of b and $r \cdot (a, b) = (b, r)$ is proved.

It can be seen that when calculating gcd, we can divide the first two numbers to get the remainder, and then use the former divisor (that is, the smaller of the two original numbers) as the dividend to divide the remainder obtained by the first operation to get a new remainder, so repeat the remainder divided by the divisor, until the new remainder is equal to 0, then the last divisor is gcd. Since the remainder must be smaller than the divisor, the new remainder obtained from each operation must be smaller than the old remainder, that is, the remainder must eventually equal 0.

Extended Euclidean Algorithm

Third, we already know that an inverse of a number is denoted by x−1 is a number when multiplied with x yield identity. Now we define to be modulo multiplicative inverse of a when mod b, expressing like:

$$ax \equiv 1(\mathrm{mod}\ b)$$

Then, we can use Extended Euclidean Algorithm to solve it：

According to this algorithm, when we know the gcd of a and b, we can always find x and y so that $ax + by = \gcd(a, b)$. For example, if we know gcd $(26,11) = 1$, we will have the equation $26x + 11y = 1$. Then, we do the process of Euclidean Algorithm in 2.2.2:

$$\begin{gathered} 26 = 11 \times 2 + 4 \\ 11 = 4 \times 2 + 3 \\ 4 = 3 \times 1 + 1 \\ 3 = 1 \times 3 + 0 \end{gathered}$$

Then, we rewind them：

$$1 = 4 - (3 \times 1) \Rightarrow 1 = 4 \times 3 - 11 \Rightarrow 1 = 26 \times 3 - 11 \times 7$$

Now, we solve x and y to be x=3, y=-7. Since our goal is to solve x in $ax \equiv 1(\mathrm{mod}\ b)a^2 + b^2 = c^2$, if $\gcd(a, b) = 1$,we can first find the equation $ax + by = \gcd(a, b) = 1$. Then, since $ax + by(\mathrm{mod}\ b) = ax(\mathrm{mod}\ b) + by(\mathrm{mod}\ b) = ax(\mathrm{mod}\ b) + 0$, and $1(\mathrm{mod}\ b) = 1$, so we can finally get $ax(\mathrm{mod}\ b) = 1$. Therefore, the x we solve in $ax + by = \gcd\ (a, b)$ is also the x we want in $ax \equiv 1(\mathrm{mod}\ b)$.

Euler Totient Function

We define φ(n) as the Euler Totient Function, which counts all number between 1 to (n-1) that are co-prime to n. If n is prime, then all numbers from 1 to (n-1) are prime to n, that is, φ(n)=n-1. If n =p ·q (both p and q are prime numbers), then since there are p·q numbers from 1 to n, subtract all the multiples of p (q) and the multiples of q (p), plus the additional subtracted multiples of p and q (p ·q), we can get $\varphi(n) = pq - p - q + 1 = (p - 1)(q - 1)$.

Procedures of RSA

For preparing, we need to choose two distinct large prime numbers p and q. Then, we compute $n = p \cdot q$, in which n is the key length.Next, we choose a number e, such that $1 < e < \varphi(n)$ and $\gcd(e, \varphi(n)) = 1$.Then, we define d such that $d \cdot e \equiv 1(\mathrm{mod}\ \varphi(n))$. According to 2.2.3, if e does not meet the above "$1 < e < \varphi(n)$ and $\gcd(e, \varphi(n)) = 1$", then the extended Euclidean algorithm cannot be used to find a unique value of d under the definition of d, so e must satisfy the above conditions.

Let m be the information that needs to be transmitted. First, we encrypted by$c \equiv m^e (\text{mod } n)$.n and m^e is already known, so that we can calculate c and send it to the receiver.

For the receiver, they will have private key d that satisfy "$d \cdot e \equiv 1(\text{mod } \varphi(n))$" to decrypted by doing $m = c^d(\text{mod } n)$. We can prove this as following:

If $h \equiv c^d(\text{mod } n)$, which is $h \equiv m^{d \cdot e}(\text{mod } n)$,since $d \cdot e \equiv 1(\text{mod } \varphi(n))$, according to alternate definition in congruence, then $(d \cdot e - 1)$ is divisible by $\varphi(n)$, which is $d \cdot e - 1 = k \cdot \varphi(n)$ and receiver get sh $\equiv m^{k \cdot \varphi(n)+1}(\text{mod } n)$。 According to Euler Theorem: "$a^{\varphi(n)} \equiv 1(\text{mod } n)$", we can get $m^{\varphi(n)} \equiv 1(\text{mod } n)$. Also because $a(\text{mod } n) * b(\text{mod } n) = a * b(\text{mod } n)$ which can be proved as follow:

$a = ng + y$, so $a(\text{mod } n) = y; b = nf + z$, so $b(\text{mod } n) = x$; $a * b = ngf + ngz + nfy + yz$, whose first three terms are all divisible byn, therefore, $a * b(\text{mod } n) = y * z = a(\text{mod } n) * b(\text{mod } n)$,$m^{k \cdot \varphi(n)}(\text{mod } n) = \left(m^{\varphi(n)}(\text{mod } n)\right)^k = 1(\text{mod } n)$

Therefore,

$h \equiv mk \cdot \ \varphi(n) + 1(\text{mod } n) = \ mk \cdot \ \varphi(n) \cdot m(\text{mod } n) = 1 \cdot m$ (mod n) , meaning h=m.

Safety Discussion

According to the process in 2.3, we learn that the security of RSA protocol is reflected in the fact that d is a private key, and no one can know it. Only when d is decrypted can one get the transmitted information m. However, if we want to calculate d based on the known conditions n and e, it is equivalent to solving Modulo Multiplicative Inverse of e when mod φ(n). We can certainly use Extended Euclidean Algorithm in 2.2.3 to do that as long as we know φ(n). However, according to 2.2.4,$\varphi(n) = (p - 1)(q - 1)$. Unless we know p and q, we have no way to know φ(n).

So now, the problem is factorizing n to $n = p \cdot q$. If we use classical calculation method, the time need to be spent will grow exponentially with n's digits, so 2048 bits or more can be considered safe。 But if we use Quantum calculation, things might be different. Shor's algorithm down below is aimed to deal with this problem and make RSA not safe anymore.

Shor's Algorithm

Introduction to Shor's Algorithm

The main function of Shor's algorithm is to accelerate the prime factorization of large composite numbers. Since Shor's algorithm is a quantum algorithm, its operation process is based on the principle of quantum mechanics. Therefore, by virtue of the superposition and entanglement of the quantum state itself, quantum computing has parallel computing capabilities beyond classical computing, so as to quickly complete the decomposition of integers, which is expected to crack RSA encryption technology.

Procedures of Shor's Algorithm

Overview of Shor's Algorithm

We still use n to represent the large number that needs to be factored. First, we find an x such that $1 < x < n$. Next, we find the minimum value of r who satisfy $r \geq 1$ and $x^r \equiv 1(\text{mod } n)$. We call r at this point the order of x mod n. $1 < x < n$, so that $x^r \equiv 1(\text{mod } n)$'s result can only be1, 2……n-1, n-1 in total. Therefore, as we increase the power of x, there must be x^k and x^j that satisfy $x^k \equiv x^j(\text{mod } n)$,then $x^h = kn + x^j$, $x^{h-j} = (kn \cdot$

$x^{-j}) + 1$. In this case, x^{-j} is referred to Modulo Multiplicative Inverse of x^j, rather than $1/x^j$. Therefore, x^{-j} is an integer and $x^{h-j} = 1(\text{mod } n)$..

However, if x and n are not prime, that is, x is an integer multiple of p or q, then this relation is no longer satisfied. For example, let x=3 and n=15, then x^k (k is any real number)is always multiple of 3. Since n is also multiple of 3, then x (mod n) must be a multiple of 3, meaning there is no case involve $x^r \equiv (\text{mod } n)$. However, the probability of this situation is very small. Even if it occurs, we can use the Euclidean Algorithm in 2.2.2 to calculate gcd(x,n), get p or q, and then obtain the factorization method to n. Since the amount of computation in gcd is polynomial in the number of n's digit, rather than exponential in general case, so it does not take a long time.

When we find order of x-r, we only need to make $y = x^{r/2}$ (mod n).Now, $(y+1)(y-1) = y^2 - 1 = x^r - 1$. Since x^r (mod n) $= 1$, so that $(y+1)(y-1)(\text{mod } n) = x - 1(\text{mod } n) = 0$,that is $(y+1)(y-1) = k \cdot n = k \cdot p \cdot q$(k belongs to all real numbers)。At this point, there is 75% possibility that $(y+1) = a \cdot p$, $(y-1) = b \cdot q$,$a \cdot b = k$, and a, bare both positive integers. Now, $\gcd(y+1, n) = p$, $\gcd(y-1, n) = q$. If $\gcd(y+1, n) = 1$, $\gcd(y-1, n) =$ nor vice versa, then randomly select another x-value and repeat the above operation.

The Calculation Method of Order r

Set-up: First, after we know n, we find q and let $q=2^i$. Next, we build a quantum computer that has two registers. Each register has ℓ qubits. Then, if each qubit has ⌈0 > or ⌈1 > two states, then a system composed by ℓ qubits will totally has 2^i states that can be superposed, which is q different states. If the two registers a and b are superposed together, they will eventually have q^2 states superposed on each other. Expressed as：

$$|\vartheta\rangle = \sum_{a=0}^{q-1}\sum_{b=0}^{q-1} C_{a,b}|a,b\rangle, \sum_{a=0}^{q-1}\sum_{b=0}^{q-1}|c(a,b)|^2 = 1$$

Step one: With the above conditions and circumstances in place and all qubits in both registers set to $|0\rangle$, the value of the ℓ qubits in each register is still 0 in decimal, so the common state of both registers can be expressed as $|\psi\rangle = |0,0\rangle$. Next, we apply the Hadamard gate $H = \frac{1}{\sqrt{2}}\begin{pmatrix}1 & 1\\ 1 & -1\end{pmatrix}$ to each of the l qubits in register a.

Because the ℓ qubits in register a are treated as multiplications in calculations, when they are all applied to the Hadamard gate, that is, when each one changes from $|0\rangle$ to $1/\sqrt{2}\,(|0\rangle + |1\rangle)$, the result of register a is:

$$\left(\frac{1}{\sqrt{2}}\right)^q (|00\ldots0\rangle + |00\ldots1\rangle + |00\ldots10\rangle + \cdots\ldots + |11\ldots1\rangle) = |\psi\rangle = \sum_{a=0}^{q-1}\left(\frac{1}{\sqrt{2}}\right)^q|a\rangle,$$

Because the sum of the squares of each of the coefficients has to be equal to 1, but the sum of squares now is equal to $q \cdot (\frac{1}{2})^q$, not equal to 1. Therefore, we need to multiply the whole by its reciprocal $\frac{2^q}{q}$, that is multiply each of the original unsquared coefficients by $\sqrt{\frac{2^q}{q}}$, such that:

$$|\psi\rangle = \sum_{a=0}^{q-1}\left(\frac{1}{\sqrt{2}}\right)^q \cdot \sqrt{\frac{2^q}{q}}|a\rangle|0\rangle = \sum_{a=0}^{q-1}\sqrt{\frac{1}{q}}|a\rangle|0\rangle\,|a\rangle|0\rangle = |\psi\rangle = \frac{1}{\sqrt{q}}\sum_{a=0}^{q-1}|a\rangle|0\rangle$$

Step two: We randomly select a number that satisfies 1 <x <n, and then the ℓ qubits in register b are transformed as follows:

$$|a,0 \rightarrow |a,x^a \bmod n\,\rangle$$

Where, a indicates the value after converting the binary number composed of the l qubits in register a to decimal. According to 3.2.2.2, the current a consists of a superposition of q different numbers. The transformation of register b is to make the l qubits in register b also form multiple superposition states, and the decimal value of each superposition state meets the relationship of $\mathrm{b} = \mathrm{x}^{\mathrm{a}} \bmod \mathrm{n}$ with the decimal value of each superposition state in register a. In this way, we get the new complete superposition state:

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

<u>Step three</u>: Below, we measure each qubit in register b. Since observation can collapse a quantum state with superposition to a definite value, it is possible to obtain a unique value k consisting of ℓ qubits, where k is the result of x to some power mod n. Since the values of register b have been determined, register a also no longer contains all values from 0 to q-1, but only all values of a that conform to $\mathrm{x}^{\mathrm{a}} \bmod \mathrm{n} = \mathrm{k}$. If the above equation is satisfied when register a=a0, and r is the order of $\mathrm{x}^{\mathrm{r}} \equiv 1(\bmod \mathrm{n})$, then we can deduce that $\mathrm{a} = \mathrm{a0} + \mathrm{n}\cdot\mathrm{r}$, because $\mathrm{x}^{\mathrm{a0+nr}} = \mathrm{x}^{\mathrm{a0}}\cdot\mathrm{x}^{\mathrm{nr}}$, $\mathrm{x}^{\mathrm{a0+nr}}(\bmod \mathrm{n}) = \mathrm{x}^{\mathrm{a0}}(\bmod \mathrm{n})\cdot\mathrm{x}^{\mathrm{nr}}(\bmod \mathrm{n}) = \mathrm{k}\cdot 1^{\mathrm{n}} =$. So, the new state becomes:

$$|\psi\rangle = \frac{1}{\sqrt{M}}\sum|\mathrm{a},\mathrm{k}\rangle,$$

In which M represents the total number of items, that is how many different value a can have. We can follow the course of deriving $\frac{1}{\sqrt{q}}$ in 3.2.1 to find the coefficient $\frac{1}{\sqrt{M}}$ in this case.The number a composed of qubits in register a in the new state should satisfy a∈A, $\mathrm{A} = \{\mathrm{a0}, \mathrm{a0} + \mathrm{r}, \mathrm{a0} + 2\mathrm{r}, \ldots, \mathrm{a0} + (\mathrm{M} - 1)\mathrm{r}\}$. Since we have proved $\mathrm{x}^{\mathrm{a0+nr}}(\bmod \mathrm{n}) = \mathrm{k}$ above, meanwhile M= ｜ A ｜ , and M ≈ q/r, we can redescribe the superposition：

$$|\psi\rangle = \frac{1}{\sqrt{M}}\sum_{d=0}^{M-1}|\mathrm{a0} + \mathrm{dr},\ \mathrm{k}\rangle。$$

<u>Step four:</u> Next, we apply discrete Fourier transform $\mathrm{U_q}$ onto register a. What need to be careful is that matrix in quantum calculation multiplies with Matrices in quantum computing are multiplied by a vector of coefficients before each state in a superposition, rather than by the value of its state. Therefore, it can be seen from 3.2.2.4 that register a is formed by superposition of M different quantum states, and the coefficients of each state are equal to $\frac{1}{\sqrt{M}}$.However, since Uq is a matrix of q×q, it needs to operate on a column vector of $\mathrm{q}\times 1$, not $\mathrm{M}\times 1$. Therefore, in the real column vector used for calculation, the coefficient before

$$\frac{1}{\sqrt{q}}\cdot\frac{1}{\sqrt{M}}\begin{pmatrix} 1 & 1 & 1 & \square & \square & 1 \\ 1 & \delta^{1} & \delta^{2} & \square & \square & \delta^{q-1} \\ 1 & \delta^{2} & \delta^{4} & \square & \square & \delta^{2(q-1)} \\ \square & \square & \square & \square & \square & \square \\ \square & \square & \square & \square & \square & \square \\ 1 & \delta^{q-1} & \delta^{2(q-1)} & & & \delta^{(q-1)(q-1)} \end{pmatrix}\begin{bmatrix} 0 \\ 0 \\ 1 \\ 1 \\ 1 \\ 0 \end{bmatrix} = \frac{1}{\sqrt{qM}}\begin{bmatrix} \sum_{d=0}^{m-1} 1 \\ \sum_{d=0}^{m-1}\delta^{a0+dr} \\ \square \\ \sum_{d=0}^{m-1}\delta^{(q-1)(a0+dr)} \end{bmatrix}$$

$$|\psi\rangle = \frac{1}{\sqrt{\mathrm{qM}}}\sum_{\mathrm{c}=0}^{\mathrm{q}-1}\sum_{\mathrm{d}=0}^{\mathrm{m}-1}\exp\left(2\pi\mathrm{i}\cdot\frac{\mathrm{c}(\mathrm{a0}+\mathrm{dr})}{\mathrm{q}}\right)/\mathrm{c},\mathrm{k}\rangle$$

$$|\psi\rangle = \frac{1}{\sqrt{qM}}\sum_{c=0}^{q-1}\exp\left(\frac{2\pi i c a_0}{q}\right)\sum_{d=0}^{m-1}\exp\left(\frac{2\pi i c d r}{q}\right)/c\cdot k\rangle$$

$$|\psi\rangle = \frac{1}{\sqrt{qM}}\sum_{c=0}^{q-1}\exp\left(\frac{2\pi i c a_0}{q}\right)\sum_{d=0}^{m-1}\delta^d,\text{where } \delta = e^{2\pi i c r/q}$$

the state position included in register a would be $\frac{1}{\sqrt{M}}$., and the coefficient before the state position would be 0 if not included. After the common factor $\frac{1}{\sqrt{M}}$. is presented, it can be seen from the derivation in step three that the column vectors of $q \times 1$ should be a0, a0 + r, a0 + 2r,...,a0 + (M − 1)r appears as 1, and the rest appears as 0. Matrix multiplication yields:

Step five: When we measure register a, according to the above equation, the probability of measuring a certain state is the square of this state function. The e exponent part of the first half of the above formula has a modulus of 1, so the probability of obtaining a certain value c is：

$$P_r(c) = \frac{1}{qM}\left|\sum_{d=0}^{M-1}\delta^d\right|^2,\text{where } \delta = e^{2\pi i\frac{cr}{q}}$$

We find that when $\frac{cr}{q}$c is approximately equal to an integer, the coefficient of i in e's exponent $2\pi cri/q$ approaches an integer multiple of 2π, and $\delta = \exp\left(\frac{2\pi icr}{q}\right) \approx 1$. In this case, the probability that c is measured is approximately equal to $M/qM = 1/q$. When cr/q is not close to an integer, ζ is a complex number with real and imaginary parts, which can be obtained by summing the arithmetic sequence by $(1-\zeta M)/(1-\zeta) \approx 1$, where Pr (c) $\approx 1/qM \ll 1/q$. Therefore, observed probability distribution of c is concentrated around values such that $cr/q \approx$ d (d is an integer), observed probability distribution of C is concentrated around values such that $cr/q \approx$ d (d is an integer), That is, the obtained result c can basically be considered to be $c/q \approx d/r$.

Step 6: Now we have the equation $c/q = d/r$. c and q are known. Next, we use a classical computer to find fraction d/r, which is very similar to the value of c/q, and plug the obtained r into $xr \equiv 1 \pmod{n}$ to check whether it is correct.

The way to find d/r is to do continue fraction. The reason for this step is that as we can see from 3.2.1, r<n, and q is in the range 2n^2<q<3n^2, so the original value of c/q tends to be divided by two very large integers, so it needs to be approximated to find r. The overall idea of Continued fraction is to continuously take the reciprocal of fractions less than 1, and then decompose the reciprocal of continued fraction greater than 1 into integers and fractions, finally obtaining the following figure$\frac{1}{a+\frac{1}{b+\frac{1}{c+\cdots}}}$. For example:

$$\frac{415}{93} = 4+\frac{43}{93} = 4+\frac{1}{\frac{93}{43}} = 4+\frac{1}{2+\frac{7}{43}} = 4+\frac{1}{2+\frac{1}{\frac{43}{7}}} = 4+\frac{1}{2+\frac{1}{6+\frac{1}{7}}}$$

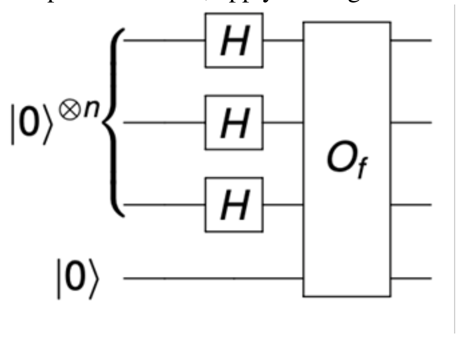
Discussion

Logics Behind Quantum Algorithm’s Advantage

By introducing Shor algorithm, the main purpose of this paper is to analyze the speed advantage of quantum algorithm compared with classical algorithm. In the general problem solving, if the solving time increases exponentially with the size of the problem, then the problem can be considered unsolvable. If it is only linear or polynomial growth, then it can be solved. Quantum algorithms use only linear or polynomial number of functions evaluations to replace exponential number of functions evaluations in classical calculation.

This advantage is due to the properties of qubits in quantum computing. Qubits follow the basic principles of quantum physics, of which the most relevant to computational efficiency is the superposition property of qubits. That is, a qubit can be at the superposition state of $|0>$ and $|1>$. This allows a qubit to contain both 0 and 1 's information at the same time. When the system has n qubits and each qubit is in a superposition state, the number of the entire system is composed of 2^{n} superpositions, containing 2^{n} information, not n. However, when processing this information, we are dealing with each qubit rather than each piece of information. So the number of functions needed to be evaluated is only n, but the actual information that's being processed is 2^{n}, and if we use classical computation we need to do 2^{n}. This is why quantum computing is able to reduce exponential problems to linear problems. The more standard process for this approach to quantum computing is as follows:

When we compute a function f, we keep the inputs unchanged and Exclusive OR (xor) the result to the output qubits. (x or, represent with $\oplus$; $a\oplus b$ means if $a = b$ than output 0, otherwise output 1). This type of circuit is called an oracle for f. If we input n variables, apply the H gate to them, and apply the oracle:



Picture 1 oracle for f

We will obtain the state:

$$\frac{1}{\sqrt{2^\wedge n}}\sum_{x=0}^{2^n-1}|\mathrm{x}>|f(x)>$$

In this way, we evaluate an exponential number of functions with just one call. However, it should also be noted that since the quantum superposition state collapses once measured then the superposition of multiple information no longer exists, so that it is important to design a quantum algorithm so that it maximize the probability of the state we want.

Discrete Fourier Transformation Application

The discrete Fourier transform is derived from Fourier series. The main content of Fourier series is that any function with period T can be represented by a series composed of superposition of a series of trigonometric functions $\mathrm{Ansin}(\mathrm{nwt} + \varphi\mathrm{n})$:

$$\mathrm{F(t)} = \mathrm{A}_0 + \sum_{n=0}^{\infty}\mathrm{Ansin}(\mathrm{n}\omega\mathrm{t} + \varphi\mathrm{n})(\text{A and n next to } \varphi \text{ are angular symbols})$$

Since each term is a function with the period $\mathrm{T} = 2\pi/\omega$ (the minimum period is $\mathrm{T} = 2\pi/\mathrm{n}\omega$, in $2\pi/\omega$, each can complete n minimum periods, so it is also a complete period for all). So, if the series converges in this period T, then the superposition function will also be a periodic function with the same period.

The sine and cosine functions are chosen because they are orthometric. "If in the interval $[a, b]$ a function series is given in which $\{\varphi k(x)\}$ （$k = 1,2,3 \ldots$） and it is .an integrable function on $[a, b]$. Now if on the interval $[a, b]$ it satisfies:

$$\int_a^b \varphi k(x)\varphi m(x)\, dx \begin{cases} = 0, when\ k \neq m \\ \neq 0, whern\ k = m \end{cases}$$

Then $\{\varphi k(x)\}$ is orthometric on the interval $[a, b]$."[2]

It can be proved that if $\varphi k(x)$ and $\varphi m(x)$ are equal to $\cos nx$, $\sin nx$ or $\cos nx$ and $\sin nx$ respectively, the above orthogonal condition can be achieved as long as the n of the two formulas are not equal. This can be analogized to the basic unit vectors i and j on the two axes of the plane cartesian coordinate system in vector algebra. The two vectors are perpendicular to each other, so the result of dot product is 0. In a rectangular coordinate system, all other vectors in the plane can be represented by these two orthogonal base unit vectors. In periodic function, especially in its physical application, it is also possible to orthogonal decompose a complex periodic function, so as to obtain a series of harmonic vibration superposition of different frequencies.

Since we get $|\psi\rangle = \frac{1}{\sqrt{M}} \sum_{d=0}^{M-1} |a0 + dr,\ k\rangle$ in the process of Shor algorithm, where register a is composed of $a0$, $a0 + r$, $a0 + 2r \ldots a0 + dr$ in superposition; since Fourier transform has the function of converting the time domain into the frequency domain; by applying Fourier transform to register a, the result can be used to analyze the probability of the occurrence of the specific value c measured by register a, so as to complete the Shor algorithm.

Another problem, since Uq is a higher-order matrix of $q \times q$, it is very complicated to calculate. Therefore, we can decompose it by matrix partitioning into $\ell(\ell + 1)/2$ matrices ($q = 2\ell$), each of size 2×2 or smaller. After such decomposition, it can also ensure that the quantum circuit built in the experiment is simpler and more feasible.

Conclusion

The unique principles of quantum physics make the world no more 100 percent certain. Quantum computing, which is developed from quantum physics, correspondingly realizes a qubit in both |0> and |1>, thus carrying more information, bringing more possibilities. Through the detailed analysis of RSA algorithm and Shor algorithm and the complete proof of related mathematical theories, we can conclude that quantum computing has great speed advantage over classical computing. This conclusion is also confirmed by exploring the underlying logic of the algorithm. As quantum computing is gradually studied more deeply, it will surpass traditional computers in more fields and bring great impact.

Acknowledgments

I would like to thank my advisor for the valuable insight provided to me on this topic.

References

[1] Peter W. Shor. Polynomial time algorithms for prime factorization and discrete logarithms on a quantum computer. SIAM J.Sci.Statist.Comput. 26 (1997).

[2] Charles H.Bennett, Gilles Brassard. Quantum Cryptography: Public Key Distribution And Coin Tossing. International.
[3] Dik Bouwmeester, Jian-Wei Pan, Claus Mattle, Manfred Eibl, Harald Weinfurter, Anton Zeilinger. Experimental quantum teleportation. Nature 390, 575-579(1997).