Quantum Computation in Cryptography

Rayhaan Patel

Shor's algorithm works by converting factoring $N = pq$ into a problem about finding the order, r of $g \pmod N$, and then solving this problem using the quantum Fourier transform.

Given the r and some random guess at one of the factors of N, g, we have that $g^r - 1 \equiv 0 \pmod N$, and if r is even, this can be factored to $(g^{\frac{r}{2}} + 1)(g^{\frac{r}{2}} - 1) \equiv 0 \pmod N$. Now, we can take $f = \gcd(N, g^{\frac{r}{2}} \pm 1)$, and if $f \neq 1$ and $f \neq N$, we will have factored N. Writing this as an algorithm, we have that

Definition 1. Shor's algorithm is defined as follows: Let $N = pq$ be an integer, where p and q are distinct primes, and N is odd.

(1) Choose a random number $1 < g < N$
(2) Compute $\gcd(g, N)$
(3) If $\gcd(g, N) \neq 1$, then $\gcd(g, N)$ is a factor of N, so we would have factored N.
(4) Find the order, r of g
(5) If r is odd, return to step 1.
(6) Otherwise, compute $f = \gcd(N, x^{\frac{r}{2}} + 1)$, and if $f \neq 1$ and $f \neq N$, then we have factored N, otherwise, go back to step 1.

Most of these steps can be preformed efficiently with standard computers, however, step 4, finding the order, r of g, is slow with classical computers, but can be vastly sped up using a quantum computer.

Now, we will take a look more closely at step 4 in the algorithm; computing the order, r of g.

Notice that $g^{r+i} \equiv g^r g^i \equiv g^i \pmod N$, which means that we can create a list of values of $g^i \pmod N$, and we look for how long it takes $g^i \pmod N$ to repeat, and this will be r. However, this is extremely inefficient using classical computers, as this would involve $O(e^n)$ computations on average; we would have to exponential g^i r times (where n is the length of N).

However, quantum computers give us a much faster way to do this computation using the Quantum Fourier Transform, which takes advantage of the unique properties of qubits (the quantum computing analog of bits) to compute r quickly.

First, we will introduce some notation used in quantum mechanics. Vectors are denoted by $|x\rangle$, which is called a "ket."

Given a Hilbert, H space with n dimensions, then we denote the elements of its orthonormal basis

$$|0\rangle, |1\rangle, \ldots, |n-1\rangle$$

and we write

$$|\psi\rangle, |\phi\rangle \ldots$$

and various other greek letters for generic elements of H. For any $|\psi\rangle \in H$, we write $\langle\psi|$ for the dual element in $H^\vee$ (a generalized transpose that also works for vector spaces over $\mathbb{C}$ $|\psi\rangle$.

Example 1. If $\dim H = 2$, and

$$|\psi\rangle = \begin{bmatrix} x \\ y \end{bmatrix}$$

, then

$$\langle\psi| = \begin{bmatrix} \overline{x} & \overline{y} \end{bmatrix},$$

where $\overline{x}$ is the complex conjugate of x. If $x, y \in \mathbb{R}$, then $\langle\psi|$ would be the transpose of $|\psi\rangle$.

This gives us a convenient way to write dot products; if $|\phi\rangle = \begin{bmatrix} z \\ w \end{bmatrix}$, then the dot product of $\langle\psi|$ and $|\phi\rangle$ is $\langle\psi\rangle\,\phi = \begin{bmatrix} \overline{x} & \overline{y} \end{bmatrix} \begin{bmatrix} z \\ w \end{bmatrix} = \overline{x}z + \overline{y}w$.

With classical computers, bits can be either 0's or 1's, so the space of possible states of the bit is a discrete set. However, in quantum computation, qubits (or quantum bits) are complex linear combinations of 0's and 1's.

Definition 2. Consider the normed complex vector space

$$H = \mathbb{C}^{\oplus 2}$$

(which is a vector space containing vectors with two complex coordinates, that has a way to measure size, akin to absolute value)

and let the orthonormal basis elements be $|0\rangle$, $|1\rangle$. A qubit is a nonzero element $|\psi\rangle \in H$ that can be written in the form

$$|\psi\rangle = a\,|0\rangle + b\,|1\rangle$$

such that at least one of a and b are nonzero (and $a, b \in \mathbb{C}$).

We also have that $|0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, |1\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$, for when we need to use vector-matrix multiplication. This allows us to have any complex linear combination of states, which are called *superpositions* of the states $|0\rangle, |1\rangle$, so instead of just having $|0\rangle$ and $|1\rangle$ as possible states, we can have states like $\frac{\sqrt{3}}{2}|0\rangle + \frac{1}{2}|1\rangle$, and $i\,|0\rangle + \sqrt{2}\,|1\rangle$. Typically, we normalize these states, so that for any vector, $a\,|0\rangle + b\,|1\rangle$, $|a|^2 + |b|^2 = 1$

When we choose to measure these qubits with respect to $|0\rangle, |1\rangle$ (this is how measuring qubits is typically done) something strange happens; we get $|0\rangle$ with a probability of $|a|^2$, and we get $|1\rangle$ with a probability of $|b|^2$.

This observation "collapses" the superposition, so if we measure the qubit to be $|1\rangle$, then its state is now $|\psi\rangle = |1\rangle$, not $|\psi\rangle = a\,|0\rangle + b\,|1\rangle$.

If we try to measure $|\psi\rangle$ again, we will still get $|1\rangle$, since $b = 1$ and $a = 0$, now that the qubit's superposition collapsed.

We can measure qubits with respect to other orthonormal bases, $|x\rangle, |y\rangle$. To do this, we rewrite $|\psi\rangle$ in the form $\alpha\,|x\rangle + \beta\,|y\rangle$, and then we have a $|\alpha|^2$ chance of getting $|x\rangle$ and a $|\beta|^2$ chance of getting $|y\rangle$.

The basis $|+\rangle = \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle, |-\rangle = \frac{1}{\sqrt{2}}|0\rangle - \frac{1}{\sqrt{2}}|1\rangle$ is a particularly notable basis.

We can also observe a qubit with respect to $\left|-\right\rangle, \left|+\right\rangle$, and still have a superposition with respect to $\left|0\right\rangle, \left|1\right\rangle$, since the qubit will collapse to either $\frac{1}{\sqrt{2}}\left|0\right\rangle + \frac{1}{\sqrt{2}}\left|1\right\rangle$, or $\frac{1}{\sqrt{2}}\left|0\right\rangle - \frac{1}{\sqrt{2}}\left|1\right\rangle$.

In order to preform computations with qubits, we need to use quantum logic gates to modify the bits. Since qubits are linear combinations of $\left|0\right\rangle, \left|1\right\rangle$, logic gates consist of linear maps.

Definition 3. For a map $U : H \to H$, we call U *unitary* if $U^\dagger$ is the inverse of U.

Quantum gates consist of unitary matrices, and because of their defining property, they are invertible.

Example 2. The Hadmard transformation is a unitary transformation that sends $\left|0\right\rangle$ to $\left|+\right\rangle$, and $\left|1\right\rangle$ to $\left|-\right\rangle$. If we set , then the Hadmard transformation can be written as $\begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{bmatrix}$

This is unitary; the Hadmard transformation's transpose is $\begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{bmatrix}$, and all of the arguments of the matrix are real, so they are their own complex conjugate, so $U^\dagger = U$, but we need $U^\dagger = U^{-1}$. To show this, we take $U^2 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$, which is the identity, as desired.

Now, we will look at tensor products, which can describe the states of multiple qubits at once. Given two vector spaces V, W, over the field k, we can combine these using a tensor product, resulting in $V \times W$ being the new space. The tensor product acts as a "wall" seperating the elements of V and W, but allowing us to consider combinations of them in a single state, which is extremely useful for when we need to consider multiple qubits, which are each in superpositions.

Definition 4. More formally, let V and W be vector spaces over the same field k. The *tensor product* $V \times W$ is the abelian group generated by elements of the form $v \times w$, with the relations

$$(v_1 + v_2) \times w = v_1 \times w + v_2 \times w$$

$$v \times (w_1 + w_2) = v \times w_1 + v \times w_2$$

$$(c \cdot v) \times w = v \times (c \cdot w),$$

where $c \in k$ is a scalar, and $v \in V, w \in W$.

Remember that elements of the form $v \times w$ *generate* $V \times W$; sums of vectors of the form $v \times w$ may not be expressable as a direct tensor product, but can still be elements of the space.

Now, to express the state of multiple qubits at once, say the qubits $\left|0\right\rangle, \left|1\right\rangle, \left|0\right\rangle, \left|0\right\rangle, \left|1\right\rangle$, we take $\left|0\right\rangle \times \left|1\right\rangle \times \left|0\right\rangle \times \left|0\right\rangle \times \left|1\right\rangle$, which represents the state of all qubits at once.

We will let $\left|x\right\rangle$ refer to $\left|x_n\right\rangle \times \left|x_{n-1}\right\rangle \times \cdots \times \left|x_1\right\rangle$, where $x = x_n x_{n-1} \dots x_1$ in binary.

For example, if $j = 5$, then $\left|j\right\rangle = \left|1\right\rangle \times \left|0\right\rangle \times \left|1\right\rangle$.

To set up the Fourier Transform, we let Q be a number, typically a power of 2. We will use $Q = 2^e > n$, where Q is the smallest power of 2 greater than n, since powers of two work nicely with qubits in the same way that they work well with normal bits. Let $f : \mathbb{Z} \to \mathbb{Z}/Q\mathbb{Z}$ be a periodic function with periodicity k, so that $f(x) = f(x+k)$ for all x, and k is the smallest value with this property. We must also assume that $f(i) \neq f(j)$, where $i, j < k, i \neq j$. For our case, $f(x) = g^x \pmod{n}$, and since $Q > n$, we can still expresss all of the values of $g^x \pmod{n}$ in $\mathbb{Z}/Q\mathbb{Z}$. We do not have to worry about $f(i) = f(j)$, where $i, j < k, i \neq j$, because then $g^i = g^j \pmod{n}$, so $g^{i-j} \equiv 1 \pmod{n}$, which is a contradiction, since $i - j < k$.

Let us take some integer j where $0 \leq i < Q$, and take the binary representation of j, using e digits. Then we create a quantum state out of j, using e qubits.

Example 3. If $Q = 32$ and $j = 5$, then j in binary is 00101, so we create a quantum state using the five qubits starting in $|0\rangle \times |0\rangle \times |1\rangle \times |0\rangle \times |1\rangle$, which we write as $|j\rangle$.

Let $\omega_N = e^{\frac{2\pi i}{N}}$

The Quantum Fourier transform is defined by

$$\mathcal{F}\,|j\rangle = \frac{1}{\sqrt{Q}} \sum_{k=0}^{Q-1} \omega_N^{jk}\,|k\rangle\,.$$

If we start with a collection of qubits in a superposition,

$$|\psi\rangle = \sum_{\ell=0}^{Q-1} a_\ell\,|\ell\rangle\,,$$

then

$$\mathcal{F}\,|x\rangle = \sum_{\ell=0}^{Q-1} \frac{a_\ell}{\sqrt{Q}} \left(\sum_{k=0}^{Q-1} \omega_Q^{jk}\,|k\rangle \right)$$

Now, we can complete Shor's algorithm. We start by initializing the qubit

$$|\psi\rangle = \frac{1}{\sqrt{Q}} \sum_{k=0}^{Q-1} |k\rangle\,,$$

which takes up e qubits, and we use e more qubits to represent $f(k)$, giving us $2e$ qubits. Since we have the first e qubits in a superposition of $|k\rangle$ and the last e in a superposition of $|f(k)\rangle$ We write this as

$$|\psi, f(\psi)\rangle = \frac{1}{\sqrt{Q}} \sum_{k=0}^{Q-1} |k, f(k)\rangle\,.$$

Now we take the quantum Fourier transform of the first e qubits, giving us

$$\begin{aligned} |\mathcal{F}\psi, f(\psi)\rangle &= \frac{1}{\sqrt{Q}} \sum_{k=0}^{Q-1} \frac{1}{\sqrt{Q}} \sum_{\ell=0}^{Q-1} \omega_Q^{k\ell}\,|\ell, f(k)\rangle \\ &= \frac{1}{Q} \sum_{m=0}^{Q-1} \sum_{\ell=0}^{Q-1} |\ell, m\rangle \sum_{k:f(k)=m} \omega_Q^{k\ell} \end{aligned}$$

Now we measure all $2e$ qubits, giving us $2e$ qubits of the form $|\ell, m\rangle$.

The probability of getting any given pair is

$$\left|\frac{1}{Q}\sum_{k:f(k)=m}\omega_Q^{k\ell}\right|^2=\frac{1}{Q^2}\left|\sum_{b=0}^{(Q-k_0-1)/r}\omega_Q^{\ell rb}\right|^2$$

where k_0 is the smallest value such that $f(k_0)=m$, and r is the period, which is what we want.

Since for any integer, T, we have that

$$\sum_{k=0}^{N-1}e^{2\pi i\frac{k}{N}}=0,$$

since this gives us a bunch of roots of unity, which cancel out (for any given nth root of unity, α, $-\alpha$ is also an nth root of unity, so $e^{2\pi i\frac{k}{N}}$ will eventually reach both points, making them cancel). The only way we can prevent terms from canceling is by making sure $\omega_Q^{\ell rb}$ is close to 1, or that $\frac{\ell r}{Q}$ is close to an integer, c.

So, after preforming a measurenment, $\frac{\ell r}{Q}$ is most likely close to an integer, and we know ℓ and Q, so we can rearrange this to get that $\frac{\ell}{Q}$ is close to $\frac{c}{r}$. We can then look at the continued fraction expansion of $\frac{\ell}{Q}$, and create a list of approximations of $\frac{c}{r}$, and check if the denominators of any of these fractions are r, by taking $g^r \pmod n$. If this does not work for all of the denominators, $d < \pmod n$, then we can try another g, and start the algorithm again, but this happens rarely.

For quantum computers, it is easy to compute the fourier transform of the superposition, as shown in [Sho99], and since the rest of the algorithm is not computationally difficult, Shor's algorithm can factor in polynomial time. Thus, quantum computers give us an efficient way to factor $N = pq$, and crack RSA encryption.

References

[Sho99] Peter W Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM review*, 41(2):303–332, 1999.