CSE439 Fall 2026 Week 9: Beginning Shor's Algorithm

First, let us review the quantum innards of Simon's Algorithm. The first Hadamard transform on the "x" half of the $2n$-dimensional Hilbert space, and then the reversible form F_f, produce the "functional superposition" of f, which---using our vector indexing notation---is defined by

$$\mathbf{u}(xy) \;= \frac{1}{\sqrt{N}} \text{ if } f(x) = y, \text{ else } 0.$$

The second Hadamard transform of the "x" half then gives $\mathbf{v}(xy) = \frac{1}{\sqrt{N}} \sum_{t \in \{0,1\}^n} (-1)^{x \bullet t} \mathbf{u}(ty)$, where we note that the "Boolean dot product" $\bullet$ actually is the inner product in our vector space over the binary field, i.e., with addition modulo 2. Either $x \bullet t = 0$, in which case the nth-order Hadamard matrix gives $+1$, or $x \bullet t = 1$, giving -1. Putting these together, we have:

$$\mathbf{v}(xy) \;= \frac{1}{N} \sum_{t \in \{0,1\}^n} \begin{cases} (-1)^{x \bullet t} & \text{if } f(t) = y \\ 0 & \text{otherwise.} \end{cases}$$

If f is 1-to-1, then for every y there is exactly one t such that $f(t) = y$, and that is the only nonzero term in the sum. This term has the same magnitude for all x. Note that if we get xy as the result of the measurement, it doesn't mean $f(x) = y$, only that we got amplitude from the t such that $f(t) = y$. Every xy combo thus has magnitude $\frac{1}{N} = \frac{1}{2^n}$. There are $N^2 = 2^{2n}$ such combos, so the total probability is N^2 times $\left(\frac{1}{N}\right)^2$, which is 1. That checks out, and we get a uniformly random x.

In the other case, the hidden string s is different from 0^n and $f(x_1) = f(x_2) \iff x_1 \oplus x_2 = s$ for all $x_1, x_2 \in \{0,1\}^n$, which makes f is 2-to-1 in this very particular manner. Now for any y that we get from the measured output xy, the definition of $\mathbf{u}$ guarantees that y is in the range of f. Hence there are exactly two strings t_1, t_2 such that $f(t_1) = f(t_2) = y$, and those obey $t_1 \oplus t_2 = s$. Thus the only two nonzero terms of the sum, for the given y and (any) x, yield

$$\mathbf{v}(xy) = \frac{1}{N}\left((-1)^{x \bullet t_1} \;+\; (-1)^{x \bullet t_2}\right).$$

Now by the fact that $t_2 = s \oplus t_1$ and $\oplus$ being the same as vector addition in our mod-2 space, we get $(-1)^{x \bullet t_2} = (-1)^{x \bullet (s \oplus t_1)} = (-1)^{x \bullet s \,+\, x \bullet t_1} \;=\; (-1)^{x \bullet t_1}(-1)^{x \bullet s}$, and so

$$\mathbf{v}(xy) = \frac{1}{N}\left((-1)^{x \bullet t_1}\left(1 \;+\; (-1)^{x \bullet s}\right)\right).$$

The nub is that whenever $x \bullet s = 1$, the sum in parentheses *cancels*, giving 0 as the amplitude of $\mathbf{v}(xy)$. Thus the only possible strings x we can get from the measurement are those for which $x \bullet s = 0$. Those are the same as those in the orthogonal subspace $\langle s \rangle^{\perp}$ of s. Because all the nonzero amplitudes are equal, we get a uniformly distributed member x of that subspace---and this makes the probability analysis of the outer classical part correct. This completes the proof...$\boxtimes$

...provided we trust that the probabilities really do sum to 1. Well, by "quantum conservation" and the laws of Nature, they have to. But it's good to do this final "Simon's sanity check": We have magnitude $\frac{2}{N}$ for each non-canceled amplitude (ignoring the sign). By f being 2-to-1 the range is half the target space, so there are 2^{n-1} possible values y. And any $x \in \langle s \rangle^{\perp}$ is possible (again note: this does not extail $f(x) = y$ from the final measurement), so there are 2^{n-1} of those. So there are $\left(\frac{N}{2}\right)^2 = \frac{N^2}{4}$ combos, each of probability $\frac{4}{N^2}$, so yes the probabilities sum to 1.

From Simon to Shor

The premise that f has a "hidden string" s is what makes the quantum part tick. Note that the proof analysis does not require us to *know* the string s, only to know tnat there is one. The outer classical part eventually enables to compute what s is---with high probability of success, that is. It is unknown whether there is a polynomial-expected-time quantum algorithm, able to query F_f repeatedly as Simon's does, that can distinguish a *general* 2-to-1 function f apart from 1-to-1 cases. This is widely disbelieved, but there is a technically-related problem that is "on the bubble": whether two given undirected graphs G_1 and G_2 are **isomorphic** (meaning: they are structurally the same graph but maybe with differently-numbered nodes). I believe this so-called **Graph Isomorphism** (**GI**) Problem belongs to BQP simply by my belief that **GI** is in classical P.

Peter Shor, in 1993, was likely the first to realize that the feature broadly represented by the hidden string s could be tied to a problem that has resisted solving for over 2,000 years:

Factoring: Given a natural number M, find a prime number p that divides M.

This is more than the problem of telling whether M itself is prime. *That* was placed into deterministic time roughly $O\left(n^6\right)$ almost 25 years ago, where $n \sim \log_2 M$ is the number of bits when M is written in binary. [Improving the "6" to match lower exponents long-known for classical randomized algorithms has proved to be a thorny problem unto itself.] The Sieve of Eratosthenes works in time $O(\sqrt{M}) = O\left(2^{n/2}\right)$ time. There are (classically randomized) algorithms that provably work in roughly $O\left(2^{n^{1/3}}\right)$ time, and some that possibly work in $O\left(2^{n^{1/4}}\right)$ or maybe even $O\left(2^{n^{1/5}}\right)$ time, but those still count as exponential time. There are substantial theoretical and experiential reasons for believing that

time $2^{n^{o(1)}}$, let alone polynomial time $2^{O(\log n)} = n^{O(1)}$, is impossible for classical (randomized) algorithms---and that this applies even when $M = pq$ is a product of just two primes. [Mind-you (1): the reasons are weaker than those for believing P ≠ NP, because the decision version of the problem---

FACT: Given a natural number M and a number k, is there a prime number $p < k$ that divides M?

---belongs to NP but is not NP-complete unless something mighty close to P = NP actually happens. The two versions are polynomial-time equivalent by classical binary search over k. Mind-you (2): both Lipton and I believe that **FACT** is actually in P, but never-mind why we think that.] It was hence a big shock when Shor proved:

Theorem: **Factoring** belongs to BQP, indeed has quantum circuits of size $\widetilde{O}(n^2)$ that give high probability of finding a prime factor.

Setup of Shor's Algorithm

In general, a **period** of a function f is a value r such that for all x,

$$f(x + r) = f(x).$$

The string s of the "promise property" in Simon's algorithm actually obeys this definition, even though it is a vector not a scalar. When Peter Shor read Simon's paper, he conceptualized that the final Hadamard transform *amplified* the periodic structure in the form of peaks and troughs of waves. The "trough" is how having $a \bullet s = 1$ made the two terms in the amplitude cancel, whereas having $a \bullet s = 0$ made them add with the same sign and hence concentrate the resulting probabilities on those cases.

Now, ahem, converting *periodic* structure into *peaks* is really the job of the *Fourier transform*, not the Hadamard transform. And the Fourier transform does this with numeric data, not just binary-string data. Shor conceptualized that replacing the final Hadamard transform with the **quantum Fourier transform** (**QFT**) might allow a similar concentration that makes a numeric period r emerge. And there is one such function and period of pre-eminent interest in cryptography... Incidentally, the QFT on n qubits is just the same as the ordinary Discrete Fourier Transform (DFT) on vectors of length $N = 2^n$. The circumstance that the QFT can be applied with $\widetilde{O}(n^2)$ quantum effort---so the theory of quantum circuits tells us---is what makes the difference.

Periodic Functions

The important example of a periodic function is **modular exponentiation**:

$$f_a(x) = a^x \bmod M.$$

Here a is a number in $\{0, 1, \ldots, M-1\}$ that is **relatively prime** to M. This means that a does not share a prime divisor with M. When $M = pq$ is the product of two different primes p and q, this simply means that a is not divisible by p or by q. If a and M did share a divisor p, then a^x would always be a multiple of p, and $a^x \bmod M$ is also a multiple of p because p divides M too. So you would not get all of the possible values modulo M. When a is relatively prime to M, what you always get is a number relatively prime to M. This is worth spelling out more than the text does:

Definition: $G_M = \{1\} \cup \{a : 1 < a < M \textit{ and } a \textit{ is relatively prime to } M\}$.

Theorem: G_M forms a **group** under multiplication.

A **group** is a set G with a distinguished element 1 together with an operation $\circledcirc$ that satisfies the following axioms:

- For all $g \in G$, $g \circledcirc 1 = 1 \circledcirc g = g$.
- For all $g \in G$ there is a unique $h \in G$ such that $gh = 1$ and $hg = 1$. We write $h = g^{-1}$.

For example, the $n \times n$ unitary matrices U form a group with $U^{-1} = U^*$. Well, the numbers in modular arithmetic form groups that are simpler to understand.

When $M = pq$ is a product of two primes, the size of G_M is exactly $(p-1)(q-1)$. (The general name for the size of G_M is the **totient** function of M, devised by and often named for the mathematician Leonhard Euler.) The consequence of G_M being a group that we need is:

Corollary: For all $a \in G_M$ there is a positive integer r such that $a^r \equiv 1 \bmod M$.

The least such r is exactly the period of $f_a(x)$ that we want to find. It always divides $|G_M|$, so when $M = pq$ we get that r divides $(p-1)(q-1)$. You might think this should narrow down possibilities, but:

- We don't actually get the value $m = (p-1)(q-1)$ factored for us---we don't even know m because we don't know how to factor $M =: pq$ to begin with.
- Compared to the number n of bits or digits of M, which is the complexity parameter we care about, the range of numbers less than m we might have to check is exponential in n.
- By the way, the number x in a^x can be exponential in n, so it looks like it takes too long to compute $f_a(x)$ to begin with. However, by **iterated squaring modulo** M we can compute the following values in $\widetilde{O}(n^2)$ time: $a_1 = a^2 \bmod M$, $a_2 = a_2^2 \bmod M = a^4 \bmod M$, $a_3 = a_2^2 \bmod M = a^8 \bmod M$, $a_4 = a_3^2 \bmod M = a^{16} \bmod M$, and so on up to $a_{n-1} = a_{n-2}^2 \bmod M = a^{n-1} \bmod M$. Then we need only multiply together those a_i such that

x as a binary number includes 2^i. This needs only $2n$ multiplications and mod-M reductions of n-bit numbers, so it is doable in $\widetilde{O}(n^2)$ time using an $\widetilde{O}(n)$-time integer multiplication algorithm. (Or we can say $O(n^3)$ time using the simple multiplication algorithm. The **RSA cryptosystem** uses mod exp'n too---and this time is largely why your credit card needed a chip.)

Nevertheless, if we *do* find the period r---for a "good" value a which we stand a fine chance of picking at random from G_M---then it was known long before Peter Shor found his algorithm in 1993 that we can go on to find p and q by classical efficient means.

Theorem: There is a **classical** randomized algorithm that, when provided a ***function oracle*** $g(M, a) =$ some integer multiple of the period of $f_a \bmod M$, finds a factor of M in expected polynomial time. That is, **Factoring** is in BPP^g.

That proof is the entire content of Chapter 12. Lipton and I bundled this up into a separate chapter so that instructors would have the freedom to skip it, as we'll do for the time being. (2024: It was in a replacement lecture done online via Zoom.) So we can focus on the task of finding r (or at least a multiple of r) via *quantum means*.

Shor's Theorem: We can build $\widetilde{O}(n^2)$-sized quantum circuits that given $M = pq$ and randomly-chosen $a \in G_M$ allow sampling values w via quantum measurement that with $O(n)$ samples give high probability of calculating (some integer multiple of) r.

One Other Useful Fact: The values $f(x), f(x+1), \ldots, f(x+r-1)$, when r is the **least** period, are all distinct when $f = f_a$ is eponentiation modulo M. [*Why* Shor's algorithm needs this property, even when measurement doesn't give you the least r, is still mysterious.]

Steps of Shor's Algorithm

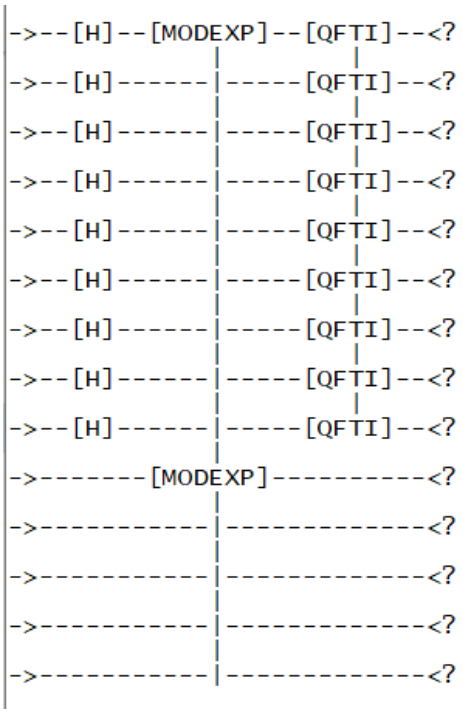
1. Given M, use classical randomness to guess a number a between 2 and $M-1$.
2. Use Euclid's algorithm to find $\gcd(a, M)$. If it gives a number $c > 1$, then "ka-ching!"---we got a divisor of M. Since both c and M/c are below $M/2$, we can recursively factor both of them.
3. If it gives $\gcd(a, M) = 1$, then we know $a \in G_M$. In the important $M = pq$ case, this had probability $\frac{(p-1)(q-1)}{pq}$ and so was pretty likely anyway. By the way, Euclid's algorithm also gives you a number b such that $ab = 1 \bmod M$. But it doesn't give you this b as a power of a (to wit, as $b = a^{r-1} \bmod M$), which is what you'd need to get r.
4. To give some slack, we choose a number $Q = 2^\ell \approx M^2$ and expand the domain of $f_a(x)$ to include x in the interval up to $Q-1$, not just up to $M-1$. The range is still 1 to $M-1$. So our domain is x in the range 0 to $2^\ell - 1$, which uses $\ell \approx 2n$ bits. This gives us quadratically many "ripples" of the period, which in turn helps the trigonometric analysis in the body of the proof.

5. The quantum circuit begins with ℓ-many Hadamard gates, followed by a quantum implementation of the $n^{O(1)}$ classical gates needed to compute modular exponentiation. This produces the functionally superposed quantum state

$$\Phi_f = \frac{1}{\sqrt{Q}} \sum_{x \in \{0,1\}^\ell} |x f_a(x)\rangle.$$

6. Apply the QFT (or its inverse) to the first ℓ qubits.
7. Then *measure* the whole result. Curiously, we ignore what happens in the "$f_a(x)$" portion of the circuit. The fact that those final n qubits were entangled with the first ℓ qubits is enough. So we let our output w in the "x-space" be the first ℓ bits of the measured result over the std. basis.

My own quantum circuit simulator draws an ASCII picture of the Shor circuit, here for $M = 21 = 3*7$ (where I guessed $a = 5$), which gave $\ell = 9$ since $2^9 = 512$ is the next power of 2 after $M^2 = 441$:



But there isn't any more to the quantum circuitry than that. It's all simply: compute a giant functional superposition and apply QFT (or its inverse) to it.

The analysis establishes that with pretty good probability already in one shot, the x part of the measured output xy reveals the period r by a followup classical means. (Note: we do not generally gate $y = f_a(x)$ here.) And with initial good probability over the choice of a, the resulting value r unlocks the key to factoring M. We will focus on understanding why the measured x has much to do with the period r to begin with. The basic point---which has been known for centuries---is that the Fourier transform converts *periodic data* to *peaked data*. Here is how the simple quantum circuit above applies this fact.

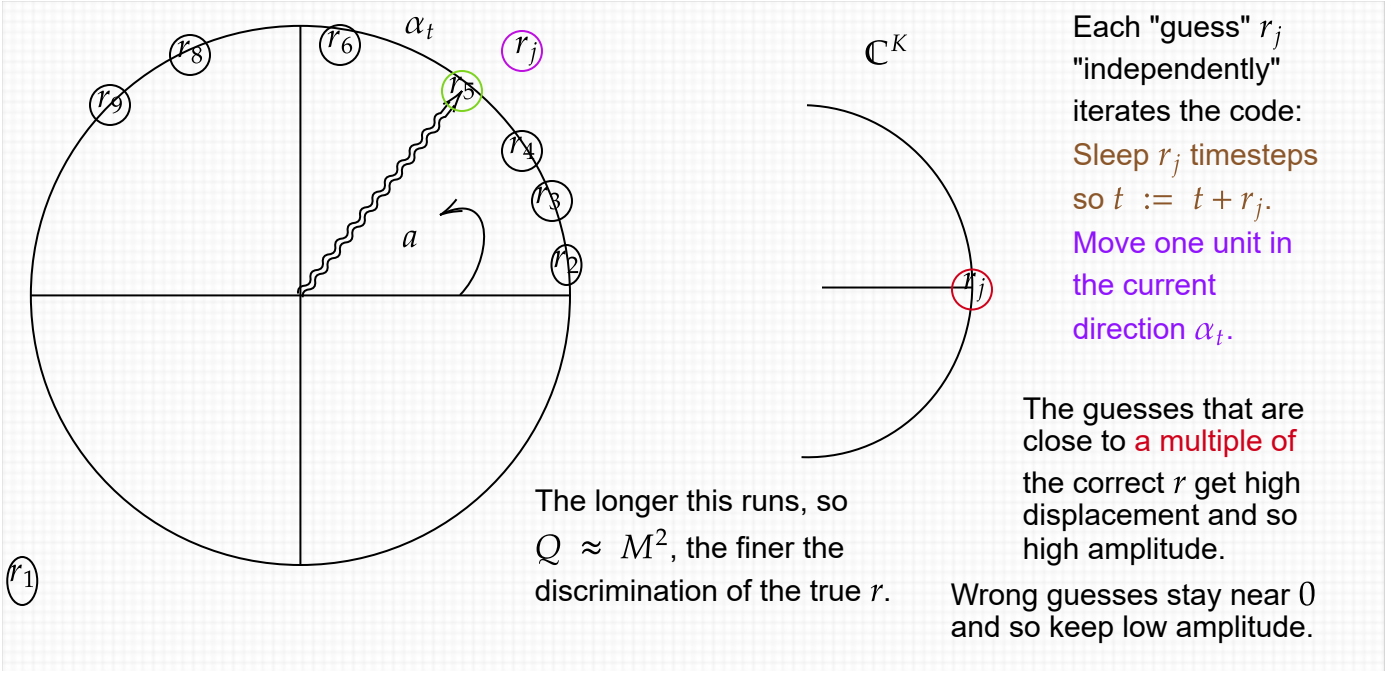
The Intuition

The following intuitive explanation draws on https://www.scottaaronson.com/blog/?p=208 by Scott Aaronson, but I've made is closer to the actual propof details.

Recall: $\mathbf{QFT}_\ell[j,k] = \omega^{jk}$, where j and k are treated as numbers not strings and ω is a principal 2^ℓ root of unity, wlog. $\omega = e^{2\pi i/Q}$. The angles of multiples and powers of ω are what we consider as values in $[0, Q-1]$. In particular, we may identify the value a with the angle $\alpha = e^{2\pi i a/Q}$. Multiplying by a has the effect of adding α to the overall phase angle.

Let r stand for the true (i.e., least) period of f. Then r is at most the size of the group G_M, minus one,so in particular, $r < M$. Let a be any element of the group G_M of size $(p-1)(q-1)$. Then we will picture a as inducing a "crazy clock" that jumps a units *counter*-clockwise at each time step. (If we used $\mathbf{QFT}^*$ then we'd picture the same thing going clockwise.)

Each row j of the QFT is an integer "guess" at the value of the period. For vividness, we will write j as r_j. Component v_j of the output vector v is the sum over columns k of $\omega^{jk} = \left(\omega^{r_j}\right)^k$ times the k-th entry of the functionally superposed powers-of-a vector. The first term is essentially $a\omega^{r_j \cdot 0} = a \cdot 1 = a$ itself. As we increment k to $k+1$, we increase the whole exponent by r_j. Going to a^{k+1} multiplies by a. Multiplying by another factor of ω^{r_j}, compared to the previous value in the sum, has the effect of adding the angle α repeatedly, r_j times. If r_j is (close to) a multiple of the period r, then we will come back to a (or close to it), meaning the phase α is (close to) the same. This is the condition under which successive terms in the sum *amplify* rather than tend to cancel each other.



It won't be perfect because the whole circle has Q = a power of 2 units whereas the true period r may have an odd divisor, or could be odd itself. But with fairly high probability, measurement---followed by figuring needed to get the guessed r_i from the measurement---yields a multiple of r. The true r is the least of the multiples. It is individually the most likely value returned and is also returned with reasonable probability. A non-least r might work anwyay. We can tell whether r works by seeing if the classical part gives us p or q, else we just try the quantum process again.

Heading into the analysis, however, we need to say exactly what the measured string w actually represents. In general, the angle α represented by a (when we actually use the complex plane to model the "crazy clock") will not be a whole-number fraction of the circle. But let us first suppose it is. Then the smallest period r (i.e., the true period) will go exactly once around the circle and back to angle α as represented by a. So suppose r_j is a correct guess of r. Then with high probability, the output w of the measurement has the sale angle α. Since angles add when we multiply complex numbers, this means $r\alpha$ takes us once around the circle. This in turn means that α is the *reciprocal* of r with regard to the circle. So w would be close to this reciprocal.

In the general case, we have to go some number t times around the circle before we get exactly back to a. That is, we have $r\alpha = t$ with respect to the circle. So $\alpha = \frac{t}{r}$ times whatever number Q represents the extent of once-around-the-circle in the units we are using. This finally means that w should be close to $\frac{tQ}{r}$ in these units. The w needs to be close enough to pull one final switcharoo: We don't know what t is either, but from $w \approx \frac{tQ}{r}$ we get $r \approx t\frac{Q}{w}$. Since r has to be an integer, we just need to find a t that multiply the fraction $\frac{Q}{w}$ into being real close to an integer. It turns out this will work when the additive error in the measured w relative to the "true amplifying direction" $\frac{tQ}{r}$ is at most ± 0.5 in the circle's units. Choosing Q high enough makes those units fine enough for this to work. The "analysis of the quantum part" tells how often the measured w is close enough to be "good." (As was the case with Simon's algorithm, the text re-uses the letter "x" to denote the particular string from the "x-space" that was obtained in the measurement.)