CPEN 400Q Lecture 16
Order finding and Shor's algorithm

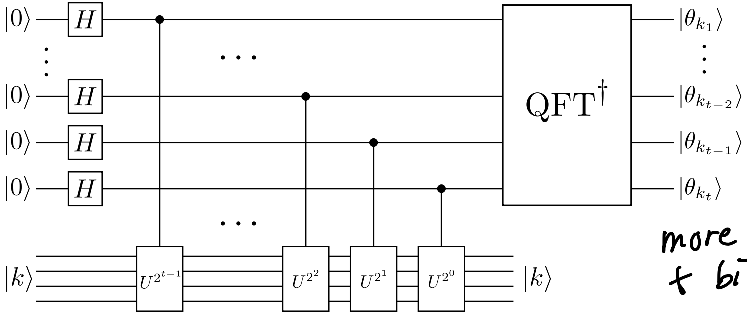
Wednesday 6 March 2024

Announcements

- Technical assignment 3 available later this week
- Midterm checkpoint meetings on Thurs/Fri

We dug into the details of **quantum phase estimation**, which estimates the eigenvalues of unitary matrices.

$$U|k\rangle = e^{2\pi i \theta_k}|k\rangle$$



more than t bits?

what if we don't know $|k\rangle$

Image credit: Xanadu Quantum Codebook node P.2

Reminder: where are we going?

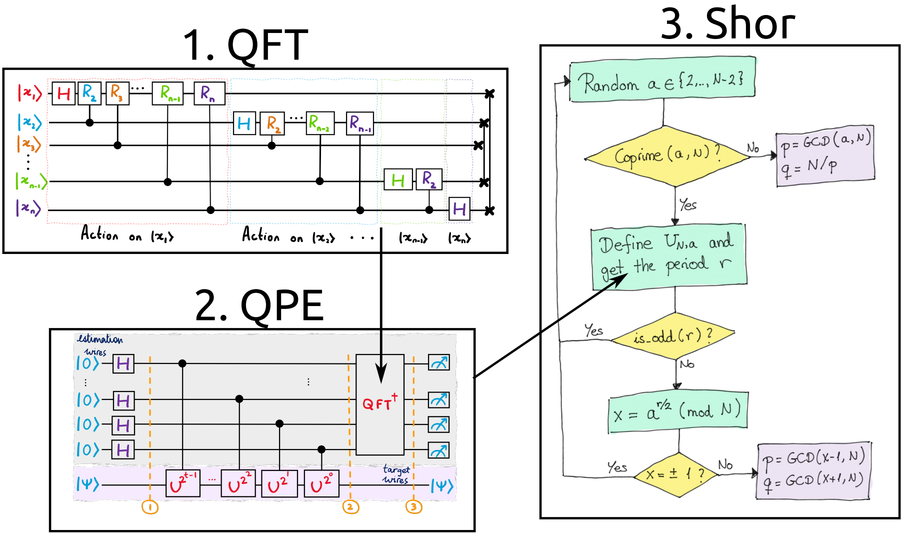


Image credit: Xanadu Quantum Codebook nodes F.3, P.3, S.4

Learning outcomes

- Use QPE to implement the order finding algorithm
- Implement Shor's algorithm in PennyLane

We defined a function

$$f(x) = a^x \bmod N$$

The *order* of *a* is the smallest *m* such that

$$f(m) = a^m \bmod N \underset{\uparrow \atop \text{equiv.}}{\equiv} 1 \bmod N$$

Order finding on a quantum computer

More formally, define

$$f_{N,a}(m) = a^m$$

$N = 5$
$a = 3$

$U_{5,3}|4\rangle = |3 \cdot 4 \bmod 5\rangle$
$= |2\rangle$

Define a unitary operation that performs

$$U_{N,a}|k\rangle = |ak \bmod N\rangle$$

$U|100\rangle = |010\rangle$

comp. basis state.

If m is the order of a, and we apply $U_{N,a}$ m times,

$$(U_{N,a})^m |k\rangle = |a^m k \bmod N\rangle = |k\rangle$$

So m is also the order of $U_{N,a}$! We can find it efficiently using a quantum computer.

Order finding on a quantum computer

Let U be an operator and $|\phi\rangle$ any state. How do we find the minimum r such that

$$U^r|\phi\rangle = |\phi\rangle$$

QPE does the trick if we set things up in a clever way:

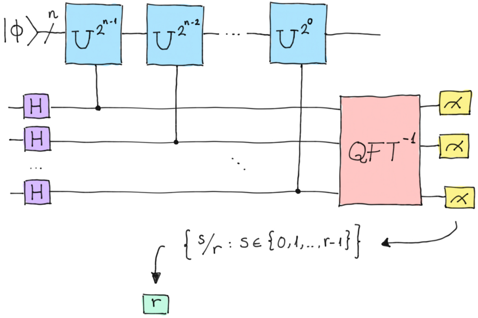


Image credit: Xanadu Quantum Codebook node S.3

Consider the state

$$|\psi_0\rangle = \frac{1}{\sqrt{r}}\left(|\phi\rangle + U|\phi\rangle + U^2|\phi\rangle + \dots U^{r-1}|\phi\rangle\right)$$

r order

If we apply U to this:

$$U|\psi_0\rangle = \frac{1}{\sqrt{r}}\left(U|\phi\rangle + U^2|\phi\rangle + U^3|\phi\rangle + \dots + \widetilde{U^r}|\phi\rangle\right)$$

$$= \frac{1}{\sqrt{r}}\left(U|\phi\rangle + U^2|\phi\rangle + \dots + |\phi\rangle\right)$$

$$= |\psi_0\rangle \quad \Rightarrow \text{eigenstate!}$$

Now consider the state

$$|\psi_1\rangle = \frac{1}{\sqrt{r}}\left(|\phi\rangle + e^{-\frac{2\pi i}{r}} U|\phi\rangle + e^{-2\cdot\frac{2\pi i}{r}} U^2|\phi\rangle + \dots + e^{-(r-1)\frac{2\pi i}{r}} U^{r-1}|\phi\rangle\right)$$

If we apply U to this:

$$U|\psi_1\rangle = \frac{1}{\sqrt{r}}\left(U|\phi\rangle + e^{-\frac{2\pi i}{r}} U^2|\phi\rangle + \dots + e^{-(r-1)\frac{2\pi i}{r}} \underbrace{U^r|\phi\rangle}_{|\phi\rangle}\right)$$

$$= \frac{e^{\frac{2\pi i}{r}}}{\sqrt{r}}\left(e^{-\frac{2\pi i}{r}} U|\phi\rangle + e^{-2\cdot\frac{2\pi i}{r}} U^2|\phi\rangle + \dots + |\phi\rangle\right)$$

$$= e^{\frac{2\pi i}{r}}|\psi_1\rangle$$

also an eigenstate!

This generalizes to $|\Psi_s\rangle$

$$|\Psi_s\rangle = \frac{1}{\sqrt{r}}\left(|\phi\rangle + e^{-s\frac{2\pi i}{r}} U|\phi\rangle + e^{-2s\frac{2\pi i}{r}} U^2|\phi\rangle + \dots + e^{-(r-1)s\frac{2\pi i}{r}} U^{r-1}|\phi\rangle\right)$$

It has eigenvalue

$$U|\Psi_s\rangle = e^{\frac{2\pi i s}{r}} |\Psi_s\rangle$$

$$\frac{s}{r}$$

$$U|k\rangle = e^{2\pi i \theta_k}|k\rangle$$

Idea: if we can create *any* one of these $|\Psi_s\rangle$, we could run QPE and get an estimate for s/r, and then recover r.

Order finding on a quantum computer

Problem: to construct any $|\Psi_s\rangle$, we would need to know r in advance!

Solution: construct the uniform superposition of all of them.

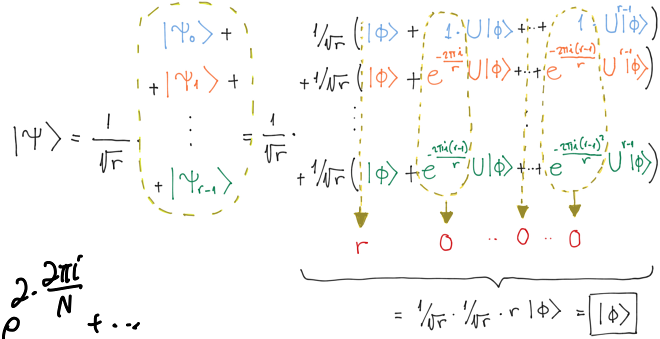
$$|\Psi\rangle = \frac{1}{\sqrt{r}} \sum_{s=0}^{r-1} |\Psi_s\rangle$$

But what does this equal?

$$U^r |\phi\rangle = |\phi\rangle$$

Order finding on a quantum computer

The superposition of all $|\Psi_s\rangle$ is just our original state $|\phi\rangle$!



$$e^{\frac{2\pi i}{N}} + e^{2 \cdot \frac{2\pi i}{N}} + \ldots$$

$\underline{\text{sum roots of unity}} = 0$

If we run QPE, the output will be s/r for one of these states.

Image credit: Xanadu Quantum Codebook node S.3

Shor's algorithm

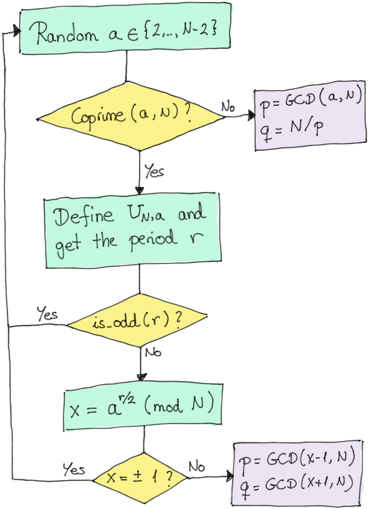


Image credit: Xanadu Quantum Codebook node S.4

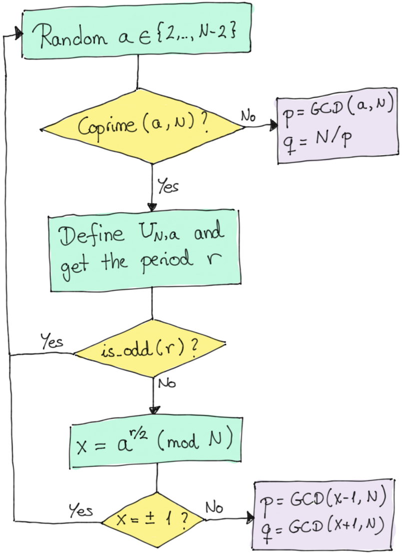
Shor's algorithm can factor a number N like

$$N = pq$$

where p, q are prime.

A quantum computer runs order finding to obtain p and q.

Everything else is number theory.



Non-trivial square roots

Idea: find a *non-trivial square root* of N, i.e., some $x \neq \pm 1$ s.t.

$$x^2 \equiv 1 \bmod N$$

If we find such an x,

$$x^2 \equiv 1 \bmod N$$

$$x^2 - 1 \equiv 0 \bmod N$$

$$(x-1)(x+1) \equiv 0 \bmod N$$

Then

$$(x-1)(x+1) = kN$$

for some integer k.

If

$$(x-1)(x+1) = kN = kpq$$

then $x - 1$ is a multiple of one of p or q, and $x + 1$ is a multiple of the other.

$$x-1 = sp$$

$$x+1 = tq$$

We can compute p and q by finding their *gcd* with N:

$x-1 = sp, \quad N = qp \qquad \Rightarrow \quad p = \gcd(x-1, N)$

$$q = \gcd(x+1, N)$$

Non-trivial square roots and factoring

$$x^2 \equiv 1 \mod N$$

It's actually okay to find any *even* power of x for which this holds:

$$x^r = x^{2r'} = (x^{r'})^2 \equiv 1 \mod N$$

We can use order finding to find such an r. If it is even, we can obtain x and factor N.

Shor's algorithm

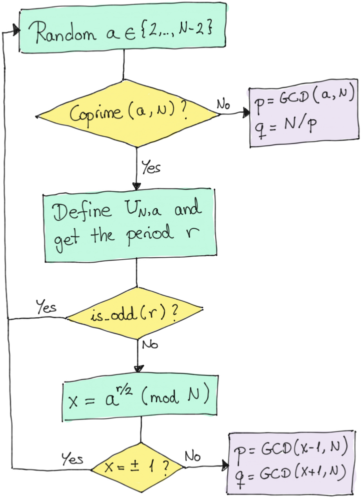
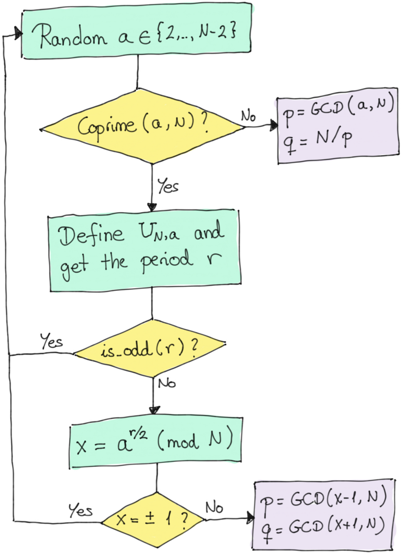


Image credit: Xanadu Quantum Codebook node S.4

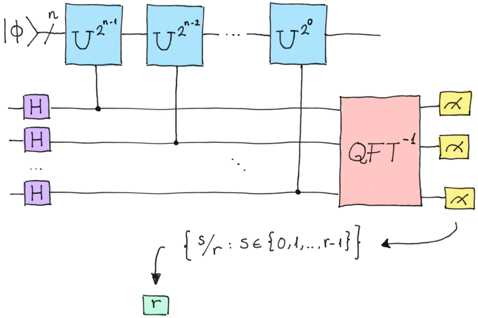
GCD: polynomial w/Euclid's algorithm

Modular exponentiation: can use exponentiation by squaring, other methods to reduce operations and memory required



Is this really efficient?

Quantum part: let $L = \lceil \log_2 N \rceil$.



QFT: polynomial in number of qubits $O(L^2)$

Controlled-U gates: implemented using something called *modular exponentiation* in $O(L^3)$ gates.

Discussion

Form groups of 3-4, and consider the following questions:

1. Shor's algorithm was developed in 1994. Estimate the fraction of today's world population that can actually implement it.
2. Shor's algo can be used to break cryptosystems like RSA. Estimate the proportion of the world that would be affected if someone actually deployed it at scale.
3. Is it ethical to develop such an algorithm? Is it ethical to *teach* such an algorithm?
4. Look up some resource estimates; how long would it actually take to break 2048-bit RSA? How many qubits are needed?
5. Think critically about (a) who knows how to implement the algorithm, and (b) who will potentially have access to quantum hardware in the future. What issues can you foresee?
6. What are ways we can keep our cryptographic infrastructure secure in the future?

Next time

Content:

- Hands-on with quantum key distribution

Action items:

1. Midterm checkpoint meetings

Recommended reading:

- Codebook modules F, P, and S
- Nielsen & Chuang 5.3, Appendix A.5