Next time:

Shor's alg for discrete logarithm and sketch of how to solve factoring.

Def (Discrete Logarithm problem)

Let g, h be elements of abelian group $(G, \cdot)$. Define for integer $x \in \mathbb{Z}$, $g^x = g \cdot g^{x-1}$ (inductively)

Identify smallest positive integer such that $g^x = h$ (if one exists).

Simplification: $N = |\langle g \rangle|$, the order of g is known.

$\langle g \rangle = \{g, g^2, \ldots, g^{N-1}, 1\}$.

Why solve this problem? It's a generalization of many pertinent cryptographic questions.

For an example, turn no further than Diffie-Helmen cryptographic key exchange.

Alice & Bob communicate over a public channel that Eve is listening to. They want to agree on a random secret s that Eve doesn't know.

① Alice & Bob publically agree on an element $g \in G$ of known order N. I.e., $|\langle g \rangle| = N$. For example, $G = \mathbb{Z}_p^*$ for prime p and g is an element of order $N = (p-1)/2$.

② Alice privately chooses $a \in \mathbb{Z}_N$ and computes $A = g^a$ and publishes A.

③ Bob privately chooses $b \in \mathbb{Z}_N$ and computes $B = g^b$ and publishes .

④ Alice computes $s = B^a$ and Bob computes $s' = A^b$.

If executed correctly, $s = s'$.

The claim is that Eve cannot guess s from A, B, g alone.

This is an instance of discrete log.

Because if Eve computes $a = DL(g, A)$, then Eve can compute $B^a = s$.

A quantum alg for discrete log via AHSP

To apply AHSP, we need to create a function f that hides a coset.

Define $f: \mathbb{Z}_N \times \mathbb{Z}_N \to G$ defined by

$$f(a, b) = g^a h^b.$$

What subgroup does this hide?

$$1 = f(a, b) = g^a (g^x)^b = g^{a+xb}.$$

Then $a + xb \equiv 0 \mod N \Rightarrow a = -xb$

so $H = \{(-xb, b) : b \in \mathbb{Z}_N\}$.

Observe that $H = \langle (-x, 1) \rangle$.

To verify observe that $f(a,b) = f(a',b')$

implies $g^{a+xb} = g^{a'+xb'} \Rightarrow (a-a', b-b') \in H$

Recall that our goal is to calculate x.

Running AHSP outputs an element of $H^\perp$.

$(c,d) \in H^\perp \iff \chi((a,b),(c,d)) = 1 \quad \forall \ (a,b) \in H.$

$\omega^{ac+bd} = 1$ where $\omega = \exp(2\pi i/N)$.

$\Rightarrow \omega^{-cx+d} = 1 \Rightarrow cx = d$

Then $H^\perp = \{(c, cx) : c \in \mathbb{Z}_N\} = \langle (1, x) \rangle$.

So, running the AHSP alg. will produce a uniformly random $c \in \mathbb{Z}_N$ along with cx.

If $\gcd(c, N) = 1$, then c^{-1} is well defined and

$$x = c^{-1} d.$$

Ex. $N = 5$, $c = 3$, then $c^{-1} = 2$.

Fact When $\gcd(c, N) = 1$, the Extended Euclidean Alg. computes c^{-1} in time $O(\log N)$ arithmetic.

So, when $\gcd(c, N) = 1$, we recover x.

If $\gcd(c, N) \neq 1$, try again until it does.

Fact $\Pr_{c \in \mathbb{Z}_N} \left[\gcd(c, N) = 1 \right] \geq \Omega\left(\frac{1}{\log N}\right)$.

Pf. Assume N has k prime factors $p_1 < p_2 < \dots < p_k$.

So $N = p_1^{a_1} p_2^{a_2} \cdots p_k^{a_k}$ where $a_j \geq 1$.

$$\log N = \sum_i a_i \log p_i \geq k.$$

Claim $\#\{c : \gcd(c,N) = 1\} = N \prod_{i=1}^{k} (1 - \frac{1}{p_i})$.

Ex. If N has 2 prime factors, then $\gcd(c,N) = 1$ if c isn't divisible by p_1 or p_2.

N/p_1 divisible by p_1. N/p_2 divisible by p_2.

$N/(p_1 p_2)$ divisible by p_1 and p_2.

so $N\left(1 - \frac{1}{p_1} - \frac{1}{p_2} + \frac{1}{p_1 p_2}\right) = N\left(1 - \frac{1}{p_1}\right)\left(1 - \frac{1}{p_2}\right)$.

So, $\Pr_c\left[\gcd(c,N) = 1\right] = \prod_{i=1}^{k} \left(1 - \frac{1}{p_i}\right)$

$$\geq \prod_{i=1}^{k} \left(1 - \frac{1}{i+1}\right)$$

$$= \prod_{i=1}^{k} \left(\frac{i}{i+1}\right)$$

$$= \frac{1}{k+1}$$

$$\geq \Omega\left(\frac{1}{\log N}\right).$$

Therefore, we need $O(\log N)$ iterations to find x.

Computing $f(a,b) = g^a h^b$ requires $O(\log N)$ multiplications and computing F_N also $O(\log N)$ elementary gates.

Total computational cost $O(\text{polylog } N)$ – considered breaking cryptography.

We made one crucial assumption for this alg to work.
We needed to know $|\langle g \rangle| = N$.

What if that wasn't known?

Def (Order finding). given $g \in G$ (abelian), calculate min r s.t. $g^r = 1$.

Note: Factoring $N = pq$, product of primes, reduces to solving order finding as well.

Sketch of order finding alg:

For $\ell \in \mathbb{Z}^+$, let $f(\ell) = g^\ell$ which can be efficiently computed in $\log \ell$ repeated squarings of x classically.

ℓ :	0	1	2	3	...	r-1	r	r+1	r+2	...
$f(\ell)$:	1	g	g^2	g^3		g^{r-1}	1	g	g^2	...

observed periodicity

We want to extract this period — similar to a hidden shift.

Trouble is how do we setup the hidden shift problem?

We could pick a large Q and compute f on $\mathbb{Z}_Q$.

Identify Q sufficiently large such that $Q > 2|G|^2$ (guaranteed).

Consider for a moment if we knew Q was a multiple of r.

Then $f(l) = f(l')$ iff $l' - l$ is a multiple of r.

So, we can solve r using AHSP algorithm.

Issues:

① No easy way of computing Q which was a multiple of r.

② We only know an efficient circuit for QFT over $Q = 2^k$.

Resolution: Approximate hidden shift is good enough.

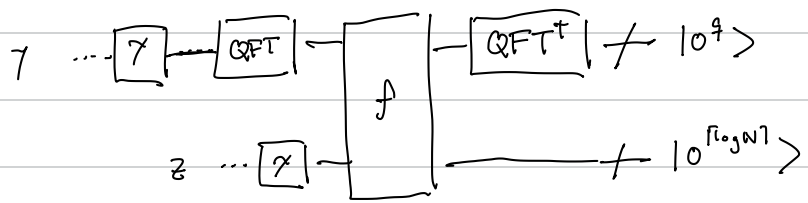
Pick Q as first power of 2 greater than $|G|^2$. ($Q = 2^q$)

Now map $l \mapsto f(l)$ for $l \in \mathbb{Z}_Q$ doesn't line-up perfectly but is "mostly correct".

Algorithm:

① Generate $\frac{1}{\sqrt{Q}} \sum_{l=0}^{Q} |l\rangle |f(l)\rangle$ and measure second register for output $\in \mathbb{Z}_N$.

② Apply $QFT(\mathbb{Z}_Q)$ and measure to get y.



What values of y are we likely to measure?

Let $\ell, \ell + r, \ell + 2r, \ldots, \ell + (J-1)r$ be the sols. to $f^{-1}(z)$.
where $J = \lfloor Q/r \rfloor$.

After collapse to z and QFT_1 state is

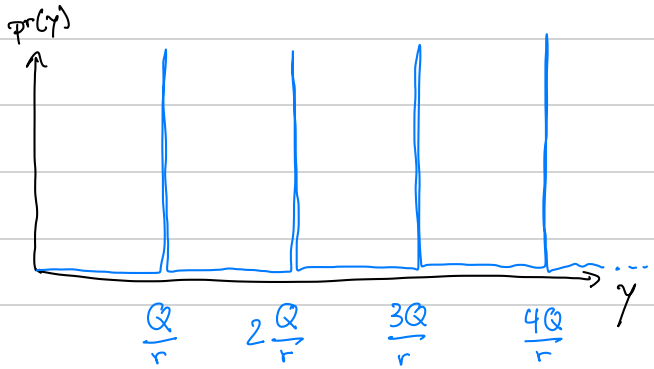
$$\frac{1}{\sqrt{J}} \sum_{j=0}^{J-1} QFT\, |\ell + rj\rangle .$$

$$= \sum_{y \in \mathbb{Z}_Q} \frac{1}{\sqrt{QJ}} \sum_{j=0}^{J-1} \omega^{y(\ell + rj)} |y\rangle \qquad \text{where } \omega = e^{2\pi i/Q}.$$

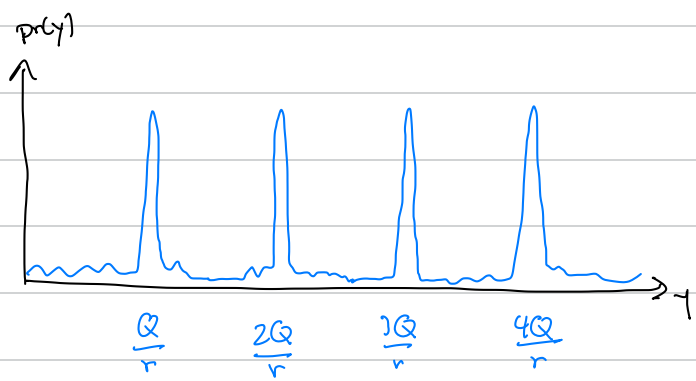
$$= \sum_{y \in \mathbb{Z}_Q} \underbrace{\frac{1}{\sqrt{QJ}} \omega^{y\ell} \sum_{j=0}^{J-1} \omega^{yrj}}_{\text{amplitude on } y.} |y\rangle$$

Recall, when $r \mid Q$, only $y = k \cdot \frac{Q}{r}$ are measured and we calculated GCD of samples y_i to learn Q/r.

Cartoon:



But instead, when $r \nmid Q$, the picture looks more like,



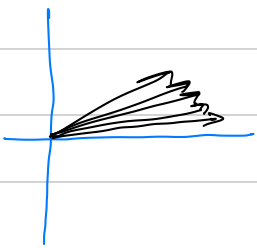
Still peaks at $\frac{Q}{r}$ but is noisy.

This is b.c. algorithm needs to output an integer

but because $r \notin \mathbb{Q}$, constructive and destructive interferences will only mostly align

When $yr \approx 0 \mod \mathbb{Q}$

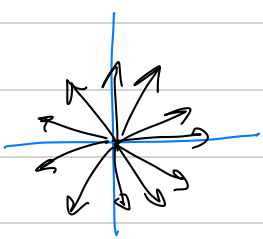
amplitude sum looks like



constructive interference

When yr far from $0 \mod \mathbb{Q}$

amplitude sum looks like



destructive interference

going to skip the rest of the details and move on to next subject.