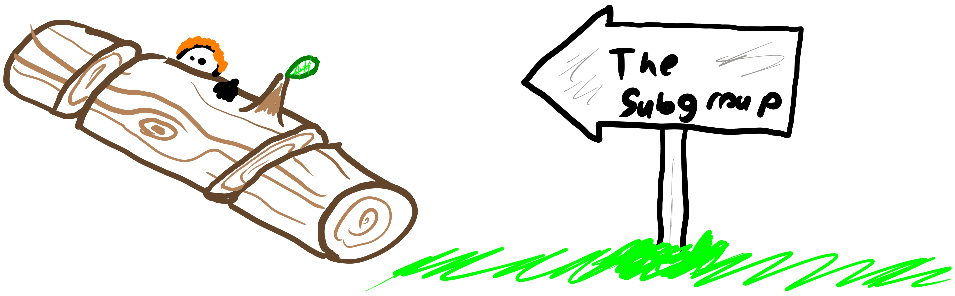
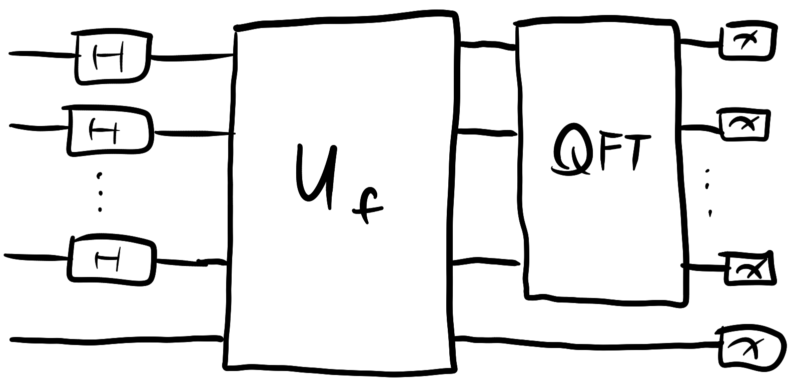
CMPT 476 Lecture 22

Discrete logarithms and the Hidden Subgroup problem



As Shor's algorithm was really just a query-efficient algorithm for finding the period of a function over $\mathbb{Z}_N$ its generality lends itself to other applications and generalizations. Today we'll look at some applications based on generalization of the QFT to other finite groups. If you aren't familiar with groups, I would encourage looking at the group theory appendix at this point.



(Diffie-Hellman & ECC)

Both the Diffie-Hellman key exchange and elliptic-curve cryptography are based on the assumed hardness of computing discrete logarithms

$$c = \log_b a \quad \text{where } a, b \in G,\ c \in \mathbb{Z}$$

One of Shor's insights was that his period finding algorithm also gives an efficient algorithm for the discrete logarithm problem in $\mathbb{Z}_p^\times$.

($\mathbb{Z}_p^\times$ — multiplicative integer group mod p)

Discrete logarithm problem in $\mathbb{Z}_p^\times$

formally, r is the order of a & b

Input: integers b, a, and r such that $a^r \equiv b^r \equiv 1 \bmod p$

Promise: $a = b^t$ for some $t \in \{0, 1, \dots, r-1\}$

Goal: Compute $t = \log_b a$

Ex.

Let $p = 7$, $b = 3$ and $a = 5$. I claim that $r = 6$:

powers	1	2	3	4	5	6
a	5	$25 \equiv 4 \bmod 7$	6	2	3	1
b	3	$9 \equiv 2 \bmod 7$	6	4	5	1

$b^5 \equiv 5 \bmod 7$

$\therefore t = 5$

In general, in $\mathbb{Z}_N^\times$ (or any cyclic group), a basic fact of group theory states that

$$\mathbb{Z}_N^\times = \langle b \mid b \in \mathbb{Z}_N^\times \rangle = \{b^0, b^1, \dots, b^{N-1} \mid b \in \mathbb{Z}_N^\times\}$$

So it's pretty easy to come up with instances of a & b as in the problem.

(Shor's approach to DLog)

Shor's approach was to use his period finding subroutine. However, applying it to $f(x) = a^x \bmod p$ would give the order/period r of a, which is independent of t!

His solution was to instead compute the period (s_1, s_2) of a two-parameter function

$$f(x, y) = a^x b^y \bmod p$$

The intuition is that the period of the above should be a function of t & r — since we know r maybe we can get t?

Note that $a^{x+s_1} b^{y+s_2} = b^{t(x+s_1)+y+s_2} = a^x b^y$ if and only if $ts_1 + s_2 \equiv 0 \bmod r$ (basic facts of group theory). So,

$$ts_1 + s_2 \equiv 0 \bmod r \Rightarrow ts_1 \equiv -s_2 \bmod r$$
$$t \equiv -s_2 s_1^{-1} \bmod r$$

At this point all we need to figure out is how to compute the two-parameter period. Recall that Simon's algorithm used $H \otimes H \otimes \cdots \otimes H$ to get the period of an n-parameter function

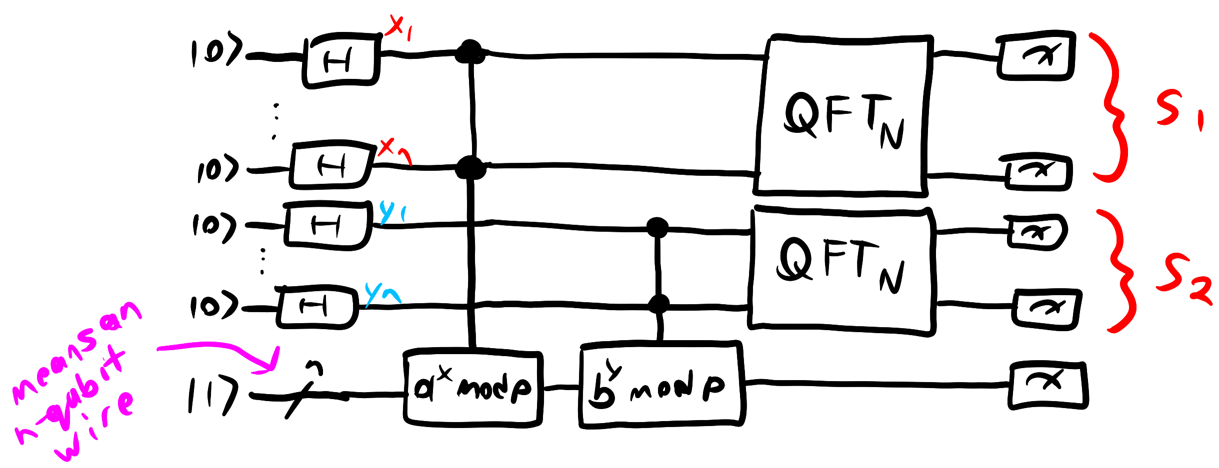
$$f : \mathbb{Z}_2^n = \mathbb{Z}_2 \times \mathbb{Z}_2 \times \cdots \times \mathbb{Z}_2 \longrightarrow \mathbb{Z}_2^m$$

where $H = QFT_2$? Well, let's try the same thing here on our function

$$f : \mathbb{Z}_N \times \mathbb{Z}_N \longrightarrow \mathbb{Z}_N$$

with two QFT_N's.

(Shor's DLog algorithm)



We won't go through the whole analysis (in lieu of a more general & clean one) but it suffices to note that the periodic state generated before the parallel QFTs is

$$\sum_{x,y\in\mathbb{Z}_N} |x\rangle|y\rangle|a^x b^y \bmod p\rangle$$

For a particular $a^x b^y \bmod p = c$ we can write the sum as

$$\sum_{x,y\in\mathbb{Z}_N \,|\, a^x b^y \bmod p = c} |x\rangle|y\rangle|c\rangle$$

$$= \sum_{s_1,s_2 \,|\, ts_1+s_2=0 \bmod r} |x_0+s_1\rangle|y_0+s_2\rangle|c\rangle$$

$$= \sum_{k\in\mathbb{Z}_r} |x_0+(-\tfrac{k}{t} \bmod r)\rangle|y_0+k\rangle|c\rangle$$

Taking the Fourier Transforms results in

$$\sum_{k\in\mathbb{Z}_r}\sum_{z_1,z_2} \omega_N^{(x_0+(-\frac{k}{t}\bmod r))z_1+(y_0+k)z_2}|z_1\rangle|z_2\rangle$$

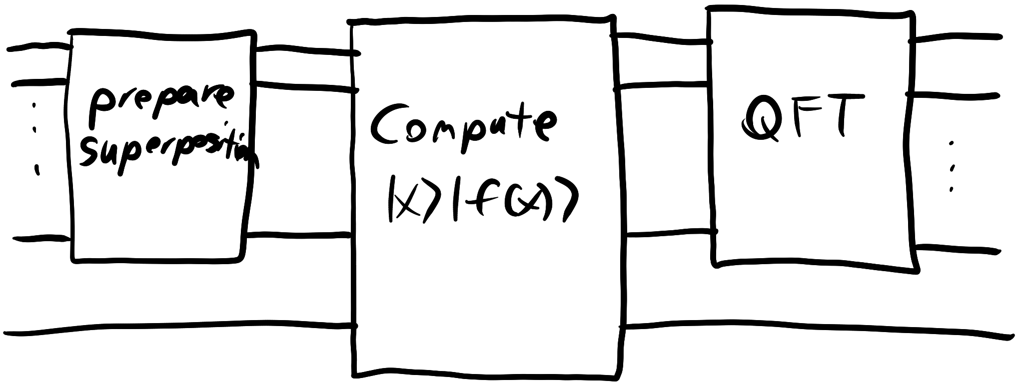
Which constructively inteferes roughly when

$$-\frac{k}{t}z_1+kz_2 \equiv 0 \bmod N \Rightarrow z_2 \equiv \frac{z_1}{t} \bmod N$$

$\therefore$ modulo modulus issues, $t = z_1 z_2^{-1} \bmod N$

(Hidden subgroup problems)

All of the algorithms we've studied so far have the same structure:



This is because they are all solving (versions of) the same problem: the (Abelian) Hidden Subgroup Problem.

First, recall that a group $G=(S,\cdot)$ is a set S equipped with an identity $e \in S$, a binary operator $\cdot : S \times S \to S$ and inverses a^{-1} $\forall a \in S$ such that

1. $(a \cdot b) \cdot c = a \cdot (b \cdot c)$
2. $a \cdot e = a = e \cdot a$
3. $a \cdot a^{-1} = e$

and a subgroup S of G is a subset of G that is a group with respect to G's operations

(see appendix for more details)

The Hidden Subgroup Problem (HSP)

input: a function $f: G \to X$ where G is a group and X a finite set (wlog $\mathbb{Z}_{\log|X|}$)

promise: there exists a subgroup S of G such that $f(x) = f(y)$ if and only if $xS = yS$

goal: find S

Ex.

Many problems of interest can be described as hidden subgroup problems. Here are a few:

Problem	Group	f	Subgroup
Deutsch	$(\mathbb{Z}_2, \oplus)$	any	$S=\{0\}$ if & only if f is constant
Simon	$(\mathbb{Z}_2^n, \oplus)$	any	$S=\{s\}$
Breaks RSA: Period finding	$(\mathbb{Z}_N, +)$	any	$S = r\mathbb{Z}_N$, r is order of a mod M
Breaks ECC: DLog	$(\mathbb{Z}_N \times \mathbb{Z}_N, +)$	$a^x b^y \bmod p$	$S = \mathbb{Z}_r \times (-k\mathbb{Z}_r)$, k is the Dlog $k = \log_b a$
Breaks lattice based crypto (Regev 03): Approx. Shortest Vector	Dih_n (Dihedral group)		
Graph isomorphism (would be a big deal...)	S_n (Symmetric group)		

These problems are all likely NP-intermediate (not NP-complete) but not in P either

Curious...

The last two problems above involve groups in which the binary operator is non-commutative — that is, they are non-Abelian groups. It turns out that Shor's algorithm generalizes to all finite Abelian HSP's thanks to the beautifully simple Fourier theory of finite Abelian groups, and the likewise beautiful classification of finite Abelian groups.

(Fourier theory of finite Abelian groups)

Let G be a finite Abelian group. A character of G is a homomorphism from $G \to T$ where T is the multiplicative group of roots of unity

$$T = (\{z \in \mathbb{C} \mid |z| = 1\}, \cdot)$$

The set $\hat{G}$ of (irreducible) characters of G is isomorphic to G.

Ex.

A character of $\mathbb{Z}_3$ is $\chi_1 : a \mapsto e^{2\pi i/3 \cdot a}$

Observe that for any $a \equiv b \bmod 3$, $\chi_1(a) = \chi_1(b)$.

Moreover, for any $a, b \in \mathbb{Z}_3$,

$$\chi_1(a+b) = e^{2\pi i/3 \cdot (a+b)} = e^{2\pi i/3\, a} e^{2\pi i/3\, b} = \chi_1(a)\chi_1(b)$$

In general, $\mathbb{Z}_N$ has characters

$$\chi_b : a \mapsto e^{2\pi i/N \cdot ab} \quad \forall b \in \mathbb{Z}_N$$

Which satisfy $\chi_{(b+c)} = \chi_b \cdot \chi_c$, giving an isomorphism of groups $\mathbb{Z}_N \simeq \hat{\mathbb{Z}}_N$ with $b \leftrightarrow \chi_b$.

The fourier transform of a function $f : G \to \mathbb{C}$ is a function $\hat{f} : \hat{G} \to \mathbb{C}$ such that

$$f(a) = \frac{1}{|G|} \sum_{\chi \in \hat{G}} \hat{f}(\chi)\chi(a)$$

In particular, $\hat{f}(\chi) = \sum_{a \in G} f(a)\chi(a)^*$

Ex. (normalized)

The fourier transforms of $\mathbb{Z}_2$ and $\mathbb{Z}_N$ correspond to the hadamard gate and QFT_N, respectively:

$$f(x) = \frac{1}{\sqrt{|\mathbb{Z}_2|}} \sum_{y \in \mathbb{Z}_2} \hat{f}(y) \overbrace{\chi_y(x)}^{(-1)^{xy}}$$

$$f(x) = \frac{1}{\sqrt{|\mathbb{Z}_N|}} \sum_{y \in \mathbb{Z}_N} \hat{f}(y) \overbrace{\chi_y(x)}^{e^{2\pi i/N \cdot xy}}$$

What about a group like $\mathbb{Z}_2^N = \mathbb{Z}_2 \times \mathbb{Z}_2 \times \cdots \times \mathbb{Z}_2$?
Well, the characters of $G \times H$ are the product of characters of G & H, so

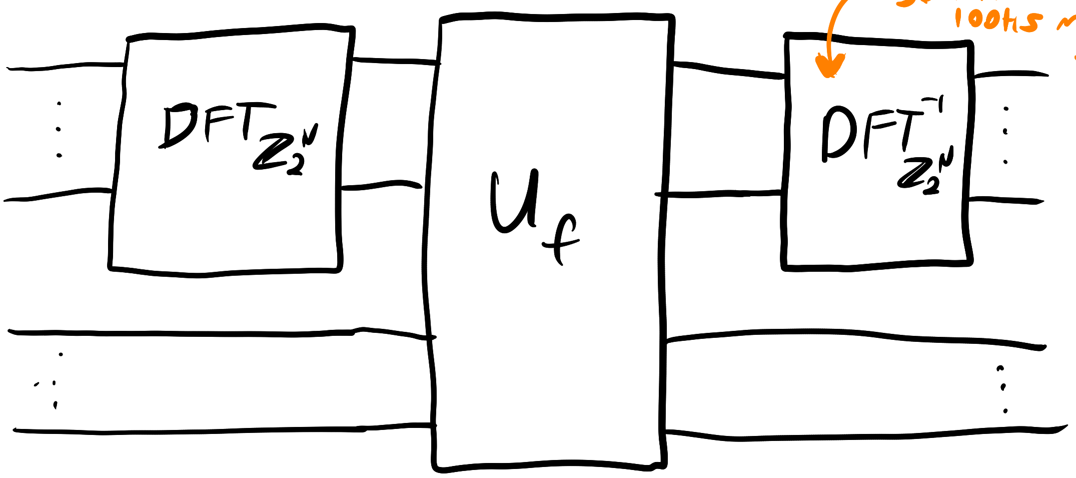
$$\chi_{x_1 \cdots x_N}(y_1 \cdots y_N) = \chi_{x_1}(y_1) \cdots \chi_{x_N}(y_N)$$
$$= (-1)^{x_1 y_1} \cdots (-1)^{x_N y_N}$$
$$= (-1)^{x \cdot y}$$

$\leftarrow$ dot product in $\mathbb{Z}_2$

So the fourier transform on $\mathbb{Z}_2^N$ is

$$f(x) = \frac{1}{\sqrt{|\mathbb{Z}_2^N|}} \sum_{y \in \mathbb{Z}_2^N} \hat{f}(y) (-1)^{x \cdot y}$$

But this is just $H \otimes H \otimes \cdots \otimes H$! So Simon's algorithm is really just



Doesn't need to be the inverse, but looks more symmetric this way :)

While we presented Shor's algorithm with $H^{\otimes n}$ as the initial Fourier Transform, note that

$$QFT_N |0\rangle = \frac{1}{\sqrt{N}} \sum_{y \in \mathbb{Z}_N} \omega_N^{0y} |y\rangle = \frac{1}{\sqrt{N}} \sum_{y \in \mathbb{Z}_N} |y\rangle$$

so $H^{\otimes n}|0\rangle = QFT_N|0\rangle$ and Shor's algorithm does have the form above as well.

Ex.

In the discrete log algorithm, we used two QFT_N's

$$QFT_N \otimes QFT_N$$

Like in Simon's example above, we can see that this is the Fourier transform on $G = \mathbb{Z}_N \times \mathbb{Z}_N$:

$$\begin{aligned} f(a,b) &= \frac{1}{|G|} \sum_{\chi \in \hat{G}} \hat{f}(\chi)\chi(a,b) \\ &= \frac{1}{|G|} \sum_{\chi_1,\chi_2 \in \hat{\mathbb{Z}}_N} \hat{f}(\chi_1,\chi_2)\chi_1(a)\chi_2(b) \\ &= \frac{1}{|G|} \sum_{k_1,k_2 \in \mathbb{Z}_N} \hat{f}(k_1,k_2)\omega_N^{k_1 a}\omega_N^{k_2 b} \\ &= \frac{1}{|G|} \sum_{k_1,k_2 \in \mathbb{Z}_N} \hat{f}(k_1,k_2)\omega_N^{k_1 a + k_1 b} \end{aligned}$$

Comparing with $QFT_N \otimes QFT_N$:

$$\begin{aligned} (QFT_N \otimes QFT_N)|a\rangle|b\rangle &= \frac{1}{\sqrt{N^2}} \sum_{k_1,k_2 \in \mathbb{Z}_N} \omega_N^{k_1 a}\omega_N^{k_1 b}|k_1\rangle|k_2\rangle \\ &= \frac{1}{\sqrt{N^2}} \sum_{k_1,k_2 \in \mathbb{Z}_N} \omega_N^{k_1 a + k_2 b}|k_1\rangle|k_2\rangle \end{aligned}$$

(QFT on products of $\mathbb{Z}_N$)

Let $G = \mathbb{Z}_{N_1} \times \mathbb{Z}_{N_2} \times \cdots \times \mathbb{Z}_{N_k}$. Then

$$QFT_G = QFT_{N_1} \otimes QFT_{N_2} \otimes \cdots \otimes QFT_{N_k}$$

(The Fourier Transform and the HSP)

Suppose we have $G = \mathbb{Z}_{N_1} \times \mathbb{Z}_{N_2} \times \cdots \times \mathbb{Z}_{N_k}$ and an HSP f, S on G. Can we do the same thing we did for Simon & Shor's algorithms?

Let's do the standard steps

1. Prepare $\frac{1}{\sqrt{|G|}} \sum_{g \in G} |g\rangle$ with QFT_G
2. Compute $\frac{1}{\sqrt{|G|}} \sum_{g \in G} |g\rangle|f(g)\rangle$ with oracle U_f
3. Measure to get $\frac{1}{\sqrt{|L|}} \sum_{g | f(g) = c} |g\rangle|c\rangle$

Now we know $f(g) = f(h)$ if & only if $g + S = h + S$, so we can pick a representative g_0 of the coset c and write the sum over S, giving the coset state

$$|g_0 + S\rangle = \frac{1}{\sqrt{|S|}} \sum_{s \in S} |g_0 + s\rangle$$

Applying the Fourier transform we get

$$\frac{1}{\sqrt{|S|}} \cdot \frac{1}{\sqrt{|G|}} \sum_{s \in S} \sum_{g \in G} \chi_g(g_0 + s)|g\rangle$$

$$= \frac{1}{\sqrt{|S| \cdot |G|}} \sum_{s \in S} \sum_{g \in G} \chi_g(g_0)\chi_g(s)|g\rangle \leftarrow \text{since } \chi_g \text{ is a homomorphism}$$

Now we could show interference using the fact that $\chi_g(s)$ is a root of unity, but a simpler algebraic proof exists using properties of group characters.

Define the kernel of S as

$$S^{\perp} = \{ t \in G \mid \chi_t(s) = 1 \; \forall s \in S \}$$

we claim that the final state is

$$\frac{1}{\sqrt{|S^{\perp}|}} \sum_{t \in S^{\perp}} \chi_t(g_0)|t\rangle$$

The proof relies on the fact that a character of a group is also a character of all of its subgroups, and the the orthogonality theorem

(Orthogonality theorem for characters)

Let $\chi_y, \chi_{y'}$ be charactors of a group. Then

$$\sum_{g\in G} \chi_y(g)\chi_{y'}(g)^* = \begin{cases} 0 & \text{if } y'\neq y \\ |G| & \text{otherwise} \end{cases}$$

Taking $\sum_{s\in S} \chi_g(s) = \sum_{s\in S} \chi_g(s)\chi_1(s)^*$ this sum is 0 (i.e. interferes) whenever $g \notin S^\perp$ (note that $\chi_1(s)=1$ for all s so $1\in S^\perp$).

(An algorithm for HSP on $\mathbb{Z}_{N_1}\times\cdots\times\mathbb{Z}_{N_k}$)

So, we've shown that at least when $N_i = 2^{n_i}$ for each i, we can sample from $S^\perp$ using Simon/Shor's algorithm. In Simon's algorithm we then used samples to find a linear basis of $S^\perp$. Since $\mathbb{Z}_{N_1}\times\cdots\times\mathbb{Z}_{N_k}$ works mostly like a vector space we can do the same

Algorithm sketch

1. Set $n \gtrsim \sum_i \log_2 N_i$
2. Use Simon/Shor/Fourier sampling to sample

$$t_1, \dots, t_{n+4} \in S^\perp$$

With at least $2/3$ probability, $t_1, \dots, t_{n+4}$ is a basis for $S^\perp$ — in group theory terms,

$$\langle t_1, \dots, t_{n+4}\rangle = S^\perp$$

3. Solve this linear system to get a basis of S

$$\begin{bmatrix} t_1 \\ \vdots \\ t_{n+4} \end{bmatrix} \begin{bmatrix} x_1/N_1 \\ \vdots \\ x_k/N_k \end{bmatrix} = \begin{bmatrix} 0 \bmod 1 \\ \vdots \\ 0 \bmod 1 \end{bmatrix}$$

(Abelian vs non-Abelian HSP)

Thanks to the **fundamental theorem of finite Abelian groups**, which states that for any such group G,

$$G \simeq \mathbb{Z}_{N_1} \times \mathbb{Z}_{N_2} \times \cdots \times \mathbb{Z}_{N_k}$$

the previous algorithm solves any HSP in <u>any</u> finite Abelian group in poly-time. Pretty cool! :)

The non-Abelian case is much harder, because the characters of a non-Abelian group are not simple roots of unity. Still, a poly-time algorithm for the non-Abelian case would have massive impacts via the shortest vector and graph isomorphism problems, and there's good reason to believe it should be possible.

The most famous result for the non-Abelian case is Kuperberg's sub-exponential-time algorithm for the HSP in dihedral groups. Specifically, Kuperberg's (first) algorithm runs in time

$$2^{O(\sqrt{\log|G|})}$$

Compared to the best-known classical algorithm which runs in exponential time

$$O(\sqrt{2}^{\log|G|})$$

A caveat of Kuperberg's algorithm however is (to the best of my knowledge) no one knows how to implement it in practice. Anecdotally, a few years back I was at a workshop aimed at examining the real impacts of QC on crypto. The question of implementing Kuperberg's algorithm came up — with researchers much more accomplished than I — we tried for a few days, before eventually giving up... I think there's still some code floating around...