CS 150 Quantum Complexity Theory

Lecture : Quantum Algorithms

Feb 21, 2024

Lecturer: Saeed Mehraban *Scribe: Preliminary notes*

1 Overview

So far we described the quantum formalism an saw a few important protocol in quantum computing such as Hadamard test and teleportation. We saw the role of entanglement in these protocol. We are now ready to get started with quantum algorithms. Quantum algorithms will be a large part of our focus during this semester. We first start with the quantum black box model, which is an idealized way of describing input to quantum computations. We will describe algorithms due to Deutsch-Josza, Bernstein-Vazirani and Simon. The problems these algorithms solve involve learning properties of Boolean functions. While these problems seem very abstract, they are the backbone of some of the algorithms we will describe later. Next, we will describe the celebrated Shor's algorithm for factoring large numbers. One of the main elements of this algorithm is the so-called quantum Fourier transform which we will describe in detail. Next we describe quantum phase estimation, Hamiltonian simulation and energy estimation. After that we will go over the Grover's search algorithm. We will then describe two special topics: quantum algorithms for linear systems which have applicaitons in machine learning and the hidden subgroup problem which generalizes the Shor's algorithm in several ways.

2 The quantum black-box model

How do we describe the input to a quantum computation? Suppose we have a Boolean function $f : \{0,1\}^n \to \{0,1\}$ and would like to provide access to instances of this function to a quantum computer. For this purpose, we use the so-called black-box model. A black-box representation for a function is a box that takes a string of bits $x \in \{0,1\}^n$ as input and outputs a single bit equal to $f(x)$. The black-box model is also sometimes called the oracle model. We immediately face a problem. How do we query an oracle in a reversible way? the input to the back box is n bits and the output is a single bit. It turns out we can implement the oracle in two ways: the phase oracle and the index oracle. The phase query works according to $O_f : |x\rangle \to (-1)^{f(x)}|x\rangle$. How is this implementation reversible? If you query this oracle twice $O_f(O_f(|x\rangle)) = (-1)^{f(x)}O_f(|x\rangle) = (-1)^{f(x)+f(x)}|x\rangle = |x\rangle$. As a result the oracle is the inverse of itself and is hence reversible. The oracle is furthermore linear: $O_f(|v\rangle + |w\rangle) = O_f(|v\rangle) + O_f(|w\rangle)$. The second model we consider for quantum oracles is the index oracle. The oralce takes two set of registers as input. In the first set we encode the input and in the second set we input an arbitrary string that has the same size as the size of the output of the function. For the case of f, since f outputs one bit, the second

register takes one bit. If the output of f was three bits the second set would take three bits. The way the oracle acts is similar to a controlled-NOT oparation. O_f takes $|x, w\rangle$ as input and produces $|x, w \oplus f(x)\rangle$. Recall that for $a, b \in \{0,1\}^m$, $c = a \oplus b$ is the bit-wise XOR of the two bits, i.e., $c_i = a_i \oplus b_i$ for $1 \leq i \leq m$. Why is this oracle reversible? If we query O_f twice we obtain $O_f(O_f(|x\rangle|w\rangle)) = O_f(|x\rangle|w \oplus f(x)\rangle) = |x\rangle|w \oplus f(x) \oplus f(x)\rangle$. Similar to the phase oracle, the index oracle is also linear.

Exercise: Prove that the above two notions are equivalent by allowing ancillas.

2.1 The Deutsch-Josza Algorithm

A function is called constant if it outputs the same bit 0 or 1 on every input. It is called balanced if the number of inputs that produce the 0 output is the same as the number of inputs that produce 1. For instance the NOT function is a balanced function. In the Deustch-Josza problem, we have black-box access to a function $f : \{0,1\} \to \{0,1\}$, and wish to see wether it is constant or balanced, ie is $\alpha_f := f(0) \oplus f(1) = 0$ or 1. Classically we need to make two queries. Why? There are four possible function : $\{0,1\} \to \{0,1\}$: $f(x) = 0, f(x) = 1, f(x) = x$ and $f(x) = NOT(x)$. Suppose we query the function on the 1 input and suppose we obtain the output 0. We know that the function cannot be the constant $f(x) = 1$ or the balanced $f(x) = x$, but we can't distinguish between $f(x) = 0$ and $f(x) = NOT(x)$. DJ showed that quantumly you can do this using 1 query: We first apply a Hadamard gate, then phase query the function, then apply the Hadamard gate again. We can show that if the function is balanced, then we will sample 1 from the output with probabibility 1, and otherwise 0. Here is the analysis.

$$|0\rangle \xrightarrow{H} \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \tag{1}$$

$$\xrightarrow{O_f} \frac{1}{\sqrt{2}}((-1)^{f(0)}|0\rangle + (-1)^{f(1)}|1\rangle) \tag{2}$$

$$\propto |(-1)^{\alpha_f}\rangle \tag{3}$$

$$\xrightarrow{H} |0\rangle \text{ if constant or } |1\rangle \text{ if balanced.} \tag{4}$$

• **Generalization to multi-qubit** We can consider the generalization of the Deutsch-Josza (DJ) problem to functions taking many input bits. Similar to the $\{0,1\} \to \{0,1\}$ functions, we can define constant function to be functions that output the same value, 0 or 1, on every input. Similarly, we define the balanced function to be one for which, out of the $N = 2^n$ possible inputs, $N/2$ yield 0 and $N/2$ yield 1 (so they are called balanced). Consider the following problem.

Problem 1 (Generalized Deutsch-Josza)**.** *Given a Boolean function $f : \{0,1\}^n \to \{0,1\}$ and the promise that f is either constant of balanced decide which one is the case.*

Out of the 2^{2^n} boolean functions taking n-bit string to one bit, there are $\binom{2^n}{2^{n-1}} \sim 2^{2^n - n/2}$ balanced functions. Why? Using counting argument (similar to what we did before), we can

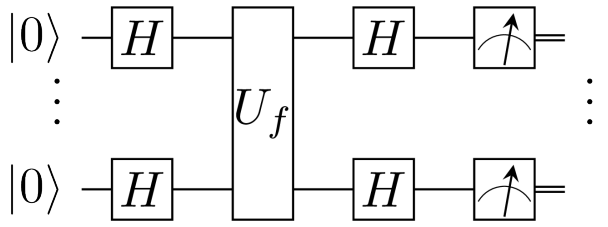


Figure 1: The circuit for multi-qubit Deutsch-Josza algorithm

deduce that extereme majority of balanced functions require exponential-size circtuits. There are however only two constant functions $f(x) = 0$ and $f(x) = 1$. The generalized DJ problem is called a promise problem because we are "promised" that the black-box function is either constant or balanced. There are $2^{2^n}(1 - \frac{1}{2^{n/2}})$ functions that are neither constant of balanced and we are promised that those instances are given to us as input.

Claim 2.1. *There is a quantum algorithm that decides balanced vs. constant using 1 single query.*

Proof. We use the circuit in Figure 2.1.

$$|0^{\otimes n}\rangle \xrightarrow{H^{\otimes n}} \frac{1}{\sqrt{2^n}} \sum_{x\in\{0,1\}^n} |x\rangle \tag{5}$$

$$\xrightarrow{O_f} \frac{1}{\sqrt{2^n}} \sum_{x\in\{0,1\}^n} (-1)^{f(x)}|x\rangle \tag{6}$$

$$\xrightarrow{H^{\otimes n}} \frac{1}{2^n} \sum_{x\in\{0,1\}^n} (-1)^{f(x)} \sum_{y\in\{0,1\}^n} (-1)^{x\cdot y}|y\rangle \tag{7}$$

Now examine the amplitude for $|0^{\otimes n}\rangle$ which appears with probability $\frac{1}{4^n}|\sum_{x\in\{0,1\}^n}(-1)^{f(x)}|^2$. If f is constant this probability is 1, and if balanced it is 0. □

Remark 2.2. Since there are doubly-exponentially different balanced functions one needs exponential queries to solve the generalized DJ problem classically. Another way to see this is by noticing that if we query f on any subset of inputs less than 2^{n-1} and we get the same value say 0, we cannot be still sure whether the function is constant or balanced. We hence get an exponential-to-one query improvement.

Exercise: What is the randomized query complexity of the Deutszch-Josza problem?

2.2 The Bernstein-Vazirani Algorithm

For $x, y \in \{0,1\}^n$, we use the notation $x \cdot y = x_1 \cdot y_1 \oplus \ldots \oplus x_n \cdot y_n$. The Bernstein-Vazirani Problem is the following problem

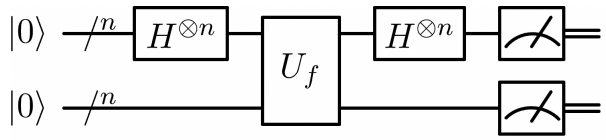


Figure 2: The circuit for Simon's algorithm (from Wikipedia)

Problem 2 (Bernstein-Vazirani)**.** *Given a Boolean function $f : \{0,1\}^n \to \{0,1\}$ with $f(x) = s \cdot x$ for a secret $s \in \{0,1\}^n$, find s.*

We need $\Theta(n)$ classical queries. To see the $O(n)$ upper-bound note $s_i = f(0^{i-1}10^{n-i+1})$. For the lower-bound we use an information theoretic argument: if we make less than n queries there is always more than one candidates for s that are consistent with all the queries.

Claim 2.3. *There is a quantum query algorithm that achieves the goal with only* 1 *query.*

Proof. We use the circuit in Figure 2.1 again. Note the output of the circuit is

$$|\psi_{\text{out}}\rangle = \frac{1}{N} \sum_{x,y \in \{0,1\}^n} (-1)^{f(x)-x\cdot y} |y\rangle. \tag{8}$$

using $f(x) = s \cdot x$ we obtain

$$|\psi_{\text{out}}\rangle = \frac{1}{N} \sum_{x,y \in \{0,1\}^n} (-1)^{(s-y)\cdot x} |y\rangle = |s\rangle. \tag{9}$$

Here we used the observation that $\sum_{x \in \{0,1\}^n} e^{a \cdot x} = N$ if $a = 0$ and 0 otherwise. □

2.3 Simon's Algorithm

We wish a problem that we witness exponential speedup for a quantum algorithm. Simon's problem exactly achieves this:

Problem 3. *Given $f : \{0,1\}^n \to \{0,1\}^n$ and the promise that $f(x) = f(y)$ iff $x = y \oplus s$ for a secret key s. Find s.*

Example 2.4. $s = 110$

$$\begin{aligned} f(000) &= 1, \\ f(001) &= 2, \\ f(010) &= 3, \\ f(011) &= 4, \\ f(100) &= 3, \\ f(101) &= 4, \\ f(110) &= 1, \\ f(111) &= 2 \end{aligned}$$

Claim 2.5. *There is a quantum algorithm that achieves this using polynomially many queries.*

Proof. We use the query model to create $|0^n\rangle \otimes |0^n\rangle \to \frac{1}{\sqrt{N}} \sum_x |x\rangle|f(x)\rangle$ we then measure the f register to obtain $\frac{|x\rangle+|y\rangle}{\sqrt{2}}$ where $x \cdot s = y \cdot s$. We then apply Hadamard $H^{\otimes n}$ to the state to obtain state $\propto (1 + (-1)^{(x-y)\cdot z})|z\rangle \propto (1 + (-1)^{s\cdot z})|z\rangle$ we measure to obtain z knowing that $z \cdot s = 0$. Taking many independent samples we end up with a system of equations:

$$\begin{aligned} z_1 \cdot s &= 0 \\ z_2 \cdot s &= 0 \\ &\vdots \\ z_n \cdot s &= 0 \end{aligned}$$

□

Exercise: *Show classically we need* $\Theta(2^{n/2})$.

3 Shor's problem

- *Simon's problem does not really solve a real-world problem. It provides oracle separation but not a real separation for an artificially designed problem.*
- *After Simon's algorithm, Shor observed that with some (perhaps non-trivial) amount of work, one can transform the algorithm in to an algorithm for factoring on a quantum computer.*
- *He uses a tool now known is quantum Fourier transform. As a matter of fact all other algorithms we have been talking use a form of Fourier analysis. They in particular use Fourier transform over* $\mathbb{Z}_2^n$*. Shor used Fourier transform over integers (the cyclic group in particular).*
- *Simon's problem involved a function* f *and a secret key* s *such that* $f(x + s) = f(x)$*, for all* x*. In a way, Simon's problem finds the period of this function.*
- *Shor's algorithm concerns* $\mathbb{Z}_N$*:* $f : [N] \to [N]$*, with the promise* $f(x) = f(x + r) = f(x + 2r) = \ldots$ *arguments* $\mod N$*: Period Finding over integers.*
- *In what follows, we first introduce Fourier analysis and quantum Fourier transform.*
- *We then move to defining the problem of Factoring and describe Shor's quantum algorithm for it.*

3.1 Quantum Fourier Transform

Let $\omega = e^{2\pi i/N}$, be the N'th root of 1. Let $f : [N] \to \mathbb{C}$ be complex numbers. We define the Fourier transformation of f as

$$\hat{f}(k) = \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} f(j)\omega^{jk}$$

It is insightful to note ω satisfies the following identity

$$1 + \omega + \omega^2 + \ldots + \omega^{N-1} = 0.$$

To see this, let $S = 1 + \omega + \ldots + \omega^{N-1}$. Therefore $1 + \omega S = S + \omega^N$. Hence

$$S = \frac{1 - \omega^N}{1 - \omega} \tag{10}$$

since $\omega^N = 1$, therefore $S = 0$. Next consider the following sum for some integer l:

$$S_l := 1 + \omega^l + \omega^{2l} + \ldots + \omega^{(N-1)l}.$$

First we observe that if l is an integer multiple of N then $\omega^l = 1$, therefore $S_l = N$. Otherwise, using Equation 10 (replacing ω with ω^l):

$$S_l = \frac{\omega^{lN} - 1}{\omega - 1} = 0.$$

therefore

$$\frac{1}{N} S_l = \begin{cases} 1 & \text{if } l \text{ is an integer multiple of } N \\ 0 & \text{otherwise} \end{cases} \tag{11}$$

One of the implications of the above Harmonic identity is that

$$f(j) = \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} \hat{f}(k)\omega^{-jk}$$

To see this, we perform the following calculation

$$\begin{aligned}
\frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} \hat{f}(k)\omega^{-jk} &= \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} (\frac{1}{\sqrt{N}} \sum_{l=0}^{N-1} f(l)\omega^{lk})\omega^{-jk} && (12) \\
&= \frac{1}{N} \sum_{k=0}^{N-1} \sum_{l=0}^{N-1} f(l)\omega^{(l-j)k} && (13) \\
&= \sum_{l=0}^{N-1} f(l) \frac{1}{N} (\sum_{k=0}^{N-1} \omega^{(l-j)k}) && (14) \\
&= \sum_{l=0}^{N-1} f(l)(\frac{1}{N} S_{l-j}) && (15) \\
&= f(j) && (16)
\end{aligned}$$

To see the last line (Equation 16) we note that the only possibility for $l-j$ to be an integer multiple of N is that $l=j$. We now use Equation 11 to conclude that the only term that survives in the sum $\sum_{l=0}^{N-1} f(l)(\frac{1}{N}S_{l-j})$ is $f(j)$.

The Fourier transform maps the constant function to pulses and vice-versa. In particular, let δ be such that

$$\delta(j) = \begin{cases} 1 & j=0 \\ 0 & j \neq 0 \end{cases}$$

and $c:[N]\to\mathbb{C}$ be such that $c(j)=1/\sqrt{N}$ for all $j\in[N]$, then $\hat{c}=\delta$ and $\hat{\delta}=c$. In other words, if a function is very flat, then its Fourier tranform will be highly spiked (and vice versa).

The quantum Fourier transform: *Based on this background, we can now define the quantum Fourier transform. The Hilbert space is $\mathbb{C}^N$. QFT acts on this Hilbert space. In particular, it maps the basis according to*

$$QFT: |j\rangle \to \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} \omega^{jk} |k\rangle \tag{17}$$

$$QFT = \frac{1}{\sqrt{N}} \begin{pmatrix} 1 & 1 & 1 & \dots & 1 \\ 1 & \omega & \omega^2 & \dots & \omega^{N-1} \\ \vdots & \vdots & \vdots & \dots & \vdots \\ 1 & \omega^{N-1} & \omega^{2(N-1)} & \dots & \omega^{(N-1)^2} \end{pmatrix} \tag{18}$$

Exercise: *Show that this map is unitary.*

Let's solve some simple examples. Let's compute

$$QFT|0\rangle = \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} \omega^{0\times k}|k\rangle = \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} |k\rangle \tag{19}$$

We obtain a uniform superposition. Now imagine $|\psi\rangle = \frac{1}{\sqrt{N}}\sum_{j=0}^{N-1}|j\rangle$. Then

$$QFT|\psi\rangle = \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} QFT|j\rangle \tag{20}$$

$$= \frac{1}{N} \sum_{j=0}^{N-1} \sum_{k=0}^{N-1} \omega^{jk}|k\rangle \tag{21}$$

$$= \frac{1}{N} \sum_{k=0}^{N-1} (\sum_{j=0}^{N-1} \omega^{jk})|k\rangle \tag{22}$$

$$= |0\rangle. \tag{23}$$

Next, imagine we start with a superposition over even numbers. For simplicity let $N = 2L$

$$|\psi\rangle = \frac{1}{\sqrt{L}} \sum_{l=0}^{L-1} |2l\rangle \tag{24}$$

Now we apply the Quantum Fourier Transform

$$QFT|\psi\rangle = \frac{1}{\sqrt{L}} \sum_{l=0}^{L-1} QFT|2l\rangle \tag{25}$$

$$= \frac{1}{\sqrt{LN}} \sum_{l=0}^{L-1} \sum_{k=0}^{N-1} \omega^{2kl} |k\rangle \tag{26}$$

$$= \frac{1}{\sqrt{LN}} \sum_{k=0}^{N-1} (\sum_{l=0}^{L-1} \omega^{2kl}) |k\rangle. \tag{27}$$

In the sum above only two terms survive $k = 0$ and $k = L = N/2$; the reason is that at these two points $\omega^k = 1$ and the corresponding amplitude becomes $\frac{1}{\sqrt{NL}} \sum_{l=0}^{L-1} 1 = \frac{1}{\sqrt{2}}$; for other terms we get zero because ω^{2k} is a nontrivial ($\neq 1$) L'th root of identity. therefore the output becomes $\frac{1}{\sqrt{2}}(|0\rangle + |N/2\rangle)$. In general, using similar ideas, we can show for $N = s \cdot L$, if

$$|\psi\rangle = \frac{1}{\sqrt{L}} \sum_{l=0}^{L-1} |s \cdot l\rangle$$

then

$$QFT|\psi\rangle = \frac{1}{\sqrt{s}} \sum_{j=0}^{s-1} |j \cdot N/s\rangle.$$

Implementation of the quantum Fourier transform: *Recall the binary notation for numbers; a number $j \in \mathbb{N}$ can be represented by $j = j_0 2^0 + j_1 2^1 + \ldots + j_{n-1} 2^{n-1}$, where $j_l \in \{0, 1\}$ and $j_{n-1} \ldots j_1 j_0$ is the binary representation. Similarly, $0.j_l \ldots j_m = \frac{j_l}{2} + \ldots + \frac{j_m}{2^{m-l+1}}$. We can show that the quantum Fourier transform maps:*

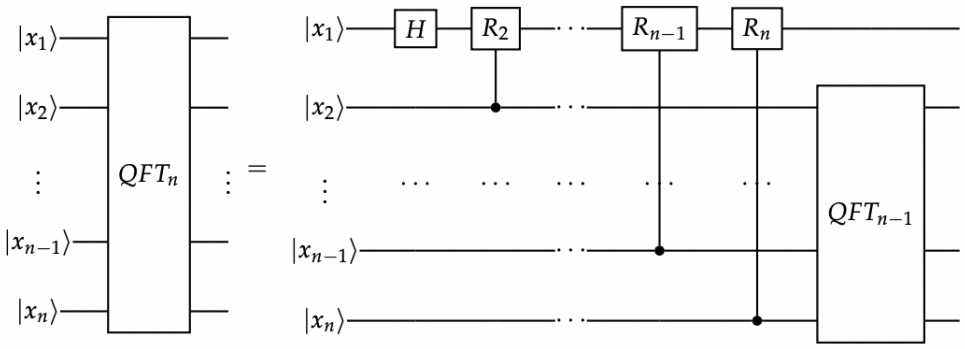
$$|j_1, \ldots, j_n\rangle \mapsto \frac{1}{\sqrt{2^n}}(|0\rangle + e^{2\pi i 0.j_n}|1\rangle)(|0\rangle + e^{2\pi i 0.j_{n-1} j_n}|1\rangle) \ldots (|0\rangle + e^{2\pi i 0.j_1 \ldots j_{n-1} j_n}|1\rangle) \tag{28}$$

For proof, see Nielsen-Chuang chapter 5. We can implement this using a number of controlled rotations like

$$R_k = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & e^{2\pi i \frac{2^k}{2^n}} \end{pmatrix} \tag{29}$$

Figure represents a recursive implementation of the quantum Fourier transform. QFT can be implemented using $O(n^2)$ elementary gates.

Figure 3: Recursive implementation of QFT



Exercise: *Show the above recursive implementation of the quantum Fourier transform works.*

3.2 The factoring problem

Problem 4 (Factoring)**.** *Given a composite number N output its prime factorization, i.e.,p, q s.t. $N = pq$, in time polynomial in $|N|$ number of digits in N in, say, binary representation.*

- *Basis for the RSA cryptosystem.*
- *The cyclic group: $\mathbb{Z}_N := \{x : gcd(x, N) = 1\}$.*
- *The size of $\mathbb{Z}_N$*
  - *If N is prime then $|\mathbb{Z}_N| = N - 1$.*
  - *If N is the product of two distinct primes p, q then $|\mathbb{Z}_N| = \phi(N) := (p - 1)(q - 1)$.*
- *The order of x in $\mathbb{Z}_N$ is the smallest integer r s.t. $x^r \mod N = 1$.*
- *Let $f_x(l) = x^l \mod N$ the period of f_x is the same as the order of x in $\mathbb{Z}_N$.*

Problem 5 (Period finding)**.** *Find the order of x in $\mathbb{Z}_N$ in polynomial time $|N|$.*

Theorem 3.1 (Miller '70)**.** *With constant probability a uniform random element x of $\mathbb{Z}_N$ has order $2r$ such that both $gcd(N, x^r + 1)$ and $gcd(N, x^r - 1)$ are non-trivial factors of N.*

Corollary 3.2. There is a reduction from Factoring to Period finding.

- *Note the similarity with the Simon's problem: We have $f : [N] \to [N]$, $f_x(l) = x^l \mod N$, and the promise that $f_x(a) = f_x(b)$ iff $a - b$ is a period of f.*

3.3 Shor's algorithm

Here we outline the steps of the Shor's algorithm. We only keep the high-level idea. We note that these steps are very similar to the steps in Simon's algorithm. We first query the function $f(l) = x^l \mod N$ *on all relevant inputs, then measure the funciton register to obtain a superposition over all pre-images of a specific value* $f(r)$*. We then use quantum Fourier transform (and extra postprocessing) to learn the period. The quantum Fourier transform is in place of the Hadamard transform in Simon's algorithm.*

- *In the first step of the algorithm we find* Q *(a power of 2) that is essentially close to* N^2*.* $\log Q$ *is the number of qubits we use to store the input register. The reason for this choice is to have enough number of qubits to produce a superposition over all relevant input instances.*

- *Next we prepare*

$$\frac{1}{\sqrt{Q}} \sum_{r=1}^{Q-1} |r\rangle |f_x(r)\rangle = \frac{1}{\sqrt{Q}} \sum_{r=1}^{Q-1} |r\rangle |x^r \mod N\rangle \tag{30}$$

  Note $x^r \mod N$ *can be prepared using repeated squaring.*

- *Next we measure the second register to obtain*

$$\frac{1}{\sqrt{l}} \sum_{i=1}^{l-1} |r_0 + is\rangle |f_x(r_0)\rangle \tag{31}$$

  where $l = \frac{Q - r_0 - 1}{s}$*.*

- *Apply QFT to the first register*

$$\frac{1}{\sqrt{lQ}} \sum_{i=0}^{l-1} \sum_{r=0}^{Q-1} \omega^{r(r_0+is)} |r\rangle |f_x(r_0)\rangle \tag{32}$$

  QFT can be applied using a circuit of size $\log^2 N$ *composed of Hadamards and controlled phases.*

- *Now measure and obtain* $|r_1\rangle |f_x(r_0)\rangle$ *with probability*

$$\frac{1}{Ql} |\sum_{i=0}^{l-1} \omega^{r_1(r_0+is)}|^2 = \frac{1}{Ql} |\sum_{i=0}^{l-1} \omega^{ir_1 s}|^2 \tag{33}$$

  If $r_1 s$ *is close to a multiple of* Q *then the above probability is close to* 1 *otherwise* 0*. Assuming we obtain one of these instances, once we measure the register we obtain* $r_1 s = mQ$ *for some integer* m*. We divide* r_1 *by* Q *and obtain* m/s *we can obtain* s *using a procedure known as the continued fraction procedure as we will outline below.*

Continued Fraction: *The continued fraction is a method which given a real number, r, computes an approximation of this number as a fraction of integers of the form*

$$r \approx a_0 + \cfrac{1}{a_1 + \cfrac{1}{a_2 + \cfrac{1}{a_3 + \cfrac{1}{\ldots + \cfrac{1}{a_n}}}}} =: \frac{P_n(r)}{Q_n(r)}$$

I like this example in Vazirani's lecture notes. We try this approximation for the number π. We can do it for the estimation $\pi \approx 3.14$.

$$\begin{aligned}
\pi &\approx 3.14 && (34)\\
&= 3 + \frac{1}{\frac{100}{14}} && (35)\\
&= 3 + \frac{1}{7 + \frac{2}{14}} && (36)\\
&\approx 22/7 && (37)
\end{aligned}$$

We can obtain other candidates by considering more decimals in π.

$$\begin{aligned}
\pi &\approx 3.1415 && (38)\\
&= 3 + \frac{1}{\frac{10000}{1415}} && (39)\\
&= 3 + \frac{1}{7 + \frac{95}{1415}} && (40)\\
&= 3 + \frac{1}{7 + \frac{1}{14 + \frac{1}{\frac{85}{95}}}} && (41)\\
&= 3 + \frac{1}{7 + \frac{1}{14}} && (42)\\
&\approx 311/99 && (43)
\end{aligned}$$

Lemma 3.3. *If r is rational equal to P/Q then $Q = Q_n(r)$ for some $n = O(\log Q)$.*

Back to the Shor's algorithm. Recall that we have stored an estimation of m/s in the output of our quantum algorithm. It turns out that if we obtain m/s with good enough accuracy then using the above lemma, s can be obtained using $s' = Q_n(r)$ for some $n = O(\log N)$. The point is that we can verify that s' is indeed the period; otherwise we try again.

4 Quantum phase estimation

Another application of the quantum Fourier transform is phase estimation. The problem is as follows:

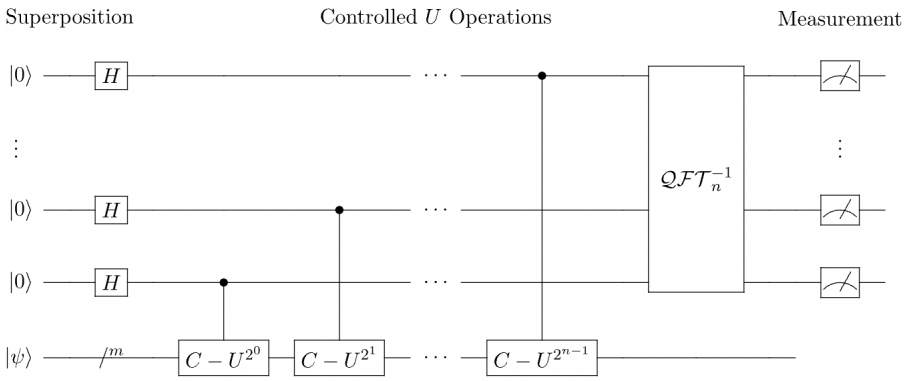


Figure 4: The circuit for phase estimation (from Wikipedia)

Problem 6 (Phase estimation)**.** *Given black-box access to controlled-U for a unitary U and one of the eigenvectors of U, output an estimation to the corresponding eigenvalue.*

Let $|u\rangle$ be the corresponding eigenvector. The algorithm is as follows: We start with $|0^t\rangle \otimes |u\rangle$ (t is large enough to store the final value for the eigenvalue with appropriate precision). We number the registers containing zeros by 1 to t. We apply Hadamards to the zeros (i.e., qubits 1 to t) to prepare a uniform superposition. We then apply controlled-U^{2^j}, controlled on the j'th qubit, to $|u\rangle$. After this operation, we will obtain

$$\frac{1}{\sqrt{2^n}}(|0\rangle + e^{2\pi i 0.\phi_t}|1\rangle)(|0\rangle + e^{2\pi i 0.\phi_{t-1}\phi_t}|1\rangle)\dots(|0\rangle + e^{2\pi i 0.\phi_1 \dots \phi_{t-1}\phi_t}|1\rangle)|u\rangle \tag{44}$$

We then apply the inverse Fourier transform to the first t qubits and read an estimation of ϕ.

Solving period finding via phase estimation: *We consider U to be the following unitary U : $|y\rangle \mapsto |xy \mod N\rangle$. Like before, let r be the period of x in N, i.e., $x^r = 1 \mod N$. We can show that*

$$|u_s\rangle = \frac{1}{\sqrt{r}} \sum_{k=0}^{r-1} \exp(\frac{-2\pi i s k}{r})|x^k \mod N\rangle$$

is an eigenvector of U with eigenvalue

$$e^{\frac{2\pi i s}{r}}$$

the main challenge is to implement $|u_s\rangle$. Well, if we don't know r, we can do that, but that is the same as solving the original period-finding question. The main observation is that

$$|1\rangle = \frac{1}{\sqrt{r}} \sum_{s=0}^{r-1} |u_s\rangle \tag{45}$$

Therefore, the output state is within close distance to $\frac{1}{\sqrt{r}} \sum_{s=0}^{r-1} |u_s\rangle \otimes |s/r\rangle$. So for each s we obtain an estimate of s/r with probability $\sim 1/r$ within $O(\log N)$ bits of accuracy. We can use the

continued fraction algorithm to find the closest rational number to s/r and obtain s' and r' such that $s/r = s'/r'$. Assuming s and r are relatively prime, we obtain $r = r'$; we can test if r' is the real r and if it was not we can try again.

5 Energy estimation

We saw that using quantum phase estimation we can estimate eigenvalues of a unitary matrix U, given access to powers of controlled-U operations and the specific eigenvectors. Can we use this algorithm to estimate the eigenvalues of physical observables? Recall that a physical observable O is a Hermitian operator whose eigenvalues model the attainable physical measurements out of a physical observable. In particular if O has eigenvalues and eigenvectors $\lambda_i, |i\rangle$, respectively, then if we measure O in the eigenstate $|\psi_j\rangle$ we obtain $\langle j|O|j\rangle = \lambda_j$. One of the most important observables in physics is the Hamiltonian or energy observable. A Hamiltonian is an observable whose eigenvalues are a system's energy levels. It is important to note that the Hamiltonian also describes the time evolution of a closed physical system. It turns out that if a system is described according to a Hamiltonian H then after t steps it has evolved according to the unitary matrix

$$e^{-iHt}$$

Recall that if A is a Hermitian matrix, then e^{iA} is unitary. To get a better sense of why that happens, we note the well-known Schrödinger equation describing the time evolution of a system is according to the differential equation

$$i\partial_t|\psi(t)\rangle = H|\psi(t)\rangle$$

If the system at time zero starts with $|\psi(0)\rangle$ then the solution to the Schrödigner equation at time t is according to $|\psi(t)\rangle = e^{-iHt}|\psi(0)\rangle$.

Suppose we are given one of the eigenstates of a system with Hamiltonian H. How do we measure the energy of the system at that level? If we can prepare the unitary e^{-iH} and controlled operations based on it, then we can input the eigenstate to the phase estimation circuit to estimate and measure the energy. The reason this works is that if E_j is an eigenvalue of H with eigenvector $|j\rangle$ then e^{-iE_j} is an eigenvalue of e^{-iHt}. The reason is due to the spectral decomposition. Recall the spectral decomposition of $H = \sum_j E_j|j\rangle\langle j|$. We discussed that for any function f of H we get $f(H) = \sum_j f(E_j)|j\rangle\langle j|$.

It remains to explain how we can implement $U = e^{-iHt}$ using basic quantum gates. This is the topic of the quantum simulation algorithm, which we will discuss next. Let me remark that assuming we can implement U using basic two-qubit gates, we can also implement controlled-U^j for $j \geq 1$. To get from controlled-U to controlled-U^j we just repeat the former j times. Suppose $U = g_T \dots g_1$ where each g_i is a two-qubit gate. We notice that $c-U = \prod_i c-g_i$. To see this, we observe that for any operation A, $c-A = |0\rangle\langle 0| \otimes I + |1\rangle\langle 1| \otimes A$. Each gates $c-g_i$ is a quantum gate acting on constant (3) qubits and can be implemented using constant many elementary gates; the reason is that any quantum circuit acting on l qubits can be implemented using $2^{O(l)}$ elementary gates.

I would like to remark that if, instead of one of the eigenstates, we feed the phase estimation algorithm with an arbitrary quantum state $|\psi\rangle$, we obtain the following. We find the decomposition of $\psi = \sum_j \psi_j |j\rangle$ according to the orthonormal eigenbasis of H; recall that because H is Hermitian, its eigenstates can be made orthonormal. The quantum phase estimation algorithm maps $|0\rangle|\psi\rangle \mapsto\approx \sum_j \lambda_j |0\rangle|\tilde{E}_j\rangle$, where $\tilde{E}_j$ is an estimation of the energy E_j. If we sample from this distribution we will obtain $\tilde{E}_j$ with probability $|\lambda_j|^2$. By taking many samples and obtaining an average over them we get an estimation of the value $\sum_j |\psi_j|^2 \tilde{E}_j \approx \langle\psi|H|\psi\rangle$. In order to make the error less that ϵ we need to make around $\Omega(\frac{\|H\|_\infty^2}{\epsilon^2})$ measurements.

6 Quantum simulation algorithm

In the previous section, we discussed the energy observable known as Hamiltonians, which capture the energy levels of the system and also describe the time evolution of a closed system. Let H be the Hamiltonian of a system. We explained that after time t the evolution of a system is described according to the unitary matrix $U = e^{-iHt}$. We say a Hamiltonian is k-local if it can be written as $H = \sum_i H_i$, where each H_i acts upon k qubits. Quantum simulation, which is Feynman's original idea of simulating quantum physics using quantum computers, can be captured according to:

Problem 7 (Quantum simulation)**.** *Given access to terms H_j of a k-local Hamiltonian $H = \sum_{j=1}^m H_j$, with $\max_j \|H_j\|_\infty = h$ prepare an efficient unitary circuit that estimates the unitary e^{-iH}. Suppose further more for any terms H_j there are at most K terms that don't commute with it.*

The main idea is based on a formula due to Trotter, also known as the Lie product formula. In particular, for two matrices A and B, the Trotter formula is

$$e^{A+B} = \lim_{N\to\infty} (e^{A/N} e^{B/N})^N \tag{46}$$

In what follows, we prove this formula, analyze its rate of convergence, and explain how we can use it to simulate quantum Hamiltonians. If A and B are two matrices that commute, then $e^{A+B} = e^A e^B$. So if the terms of the Hamiltonian H commute with each other, then $e^{-iH} = \prod_j e^{-iH_j}$. Since each e^{-iH_j} acts at most on k qubits, then we can prepare it using a quantum circuit of size at most $2^{O(k)}$. So we can implement this unitary using $m \cdot 2^{O(k)}$ steps.

What happens when the terms of the Hamiltonian do not commute with each other? For matrices A and B, if they don't commute, then $e^{A+B} \neq e^A e^B$. In particular

$$e^{A+B} - e^A e^B = \sum_{k=0}^{\infty} \frac{(A+B)^k}{k!} - \sum_{k=0}^{\infty} \frac{A^k}{k!} \sum_{k=0}^{\infty} \frac{B^k}{k!} \tag{47}$$

$$= \frac{(A+B)^2}{2} - \frac{A^2}{2} - \frac{B^2}{2} - AB + h.o. \tag{48}$$

$$= -\frac{1}{2}[A,B] + h.o. \tag{49}$$

In general, we can show

$$e^{\sum_i A_i} - \prod_i e^{A_i} = -\frac{1}{2}\sum_{i<j}[A_i, A_j] + h.o. \tag{50}$$

The main idea is to simulate approximately $e^{-iH/N}$ and repeat N times. This way the commutation between two terms in the Hamiltonian decays as $\|[\frac{H_i}{N}, \frac{H_j}{N}]\|_\infty \sim \frac{h^2}{N^2}$. For an appropriately chosen N we show that this simulation gives a good estimation. In particular,

$$\|e^{-iH/N} - \prod_j e^{-iH_j/N}\|_\infty = O(\frac{Kmh^2}{N^2}) \tag{51}$$

Next, we have to repeat the above procedure for N times. There is a well-known observation in quantum information that the error in quantum circuits grows linearly.

Lemma 6.1. *Let $A = \prod_{j=1}^N A_j$ and $B = \prod_{j=1}^N B_j$ be unitary matrices such that $\|A_i - B_i\|_\infty \leq \epsilon$ for all $1 \leq i \leq N$ then $\|A - B\|_\infty \leq N\epsilon$.*

Exercise: *Prove this lemma.*

Applying this lemma to the Equation 52 we obtain

$$\|e^{-iH} - (\prod_j e^{-iH_j/N})^N\|_\infty = O(\frac{Kmh^2}{N}) \tag{52}$$

Error analysis: *If we choose $N = \Omega(\frac{Kmh^2}{\epsilon})$ we can estimate e^{-iH} within error ϵ. We can implement $(\prod_j e^{-iH_j/N})^N$ in time $T = mN2^{O(k)}$. As a result, we can simulate e^{-iH} within error ϵ in time $T = \frac{Km^2h^22^k}{\epsilon}$.*

Remark 6.2. In order to simulate e^{-iHt} we need to replace H_j with H_jt and hence in the error analysis we need to replace h with ht. Assuming $k, h = O(1)$ we conclude that the Hamiltonian system after t time steps can be simulated in time $T = O(\frac{Km^2t^2}{\epsilon})$. Our intuition suggests that we should be able to simulate the system in a time linear in t. Using more careful analysis of recent work has been able to achieve the optimal bound $O(ts + \log\frac{1}{\epsilon})$, where s is the sparsity, ie, the maximum number of nonzero terms in each row of the Hamiltonian in the computational basis. See e.g. Berry-Childs-Kothari 2015 and Low-Chuang 2016.

7 Grover's search

The search problem can be phrased according to

Problem 8 (Search)**.** *Given oracle access to $f : [N] \to \{0, 1\}$ such that $\exists x, f(x) = 1$, find such x.*

Remark 7.1. Any classical algorithm needs at least $N - 1$ queries in the worst case and $N/2$ on average.

Figure 5: The circuit for Grover's algorithm

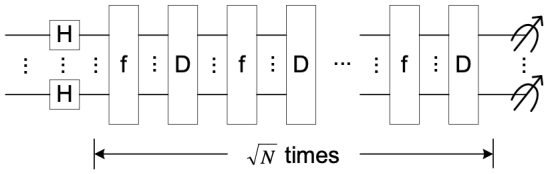
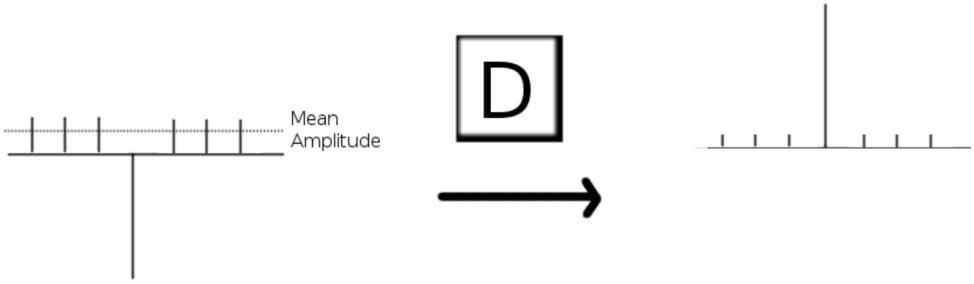


Figure 6: The effect of the reflection operator Grover's algorithm



We assume we have an oracle $O_f : \sum_x \alpha_x|x\rangle \mapsto \sum_x \alpha_x(-1)^{f(x)}|x\rangle$. The following algorithm is due to Grove and solves the search problem in $O(\sqrt{N})$ times. It is interesting to note that this bound is tight but applies to unstructured searches.

The Grover Algorithm:

1. *Prepare $|\psi\rangle := \frac{1}{\sqrt{N}} \sum_x |x\rangle$.*
2. *Query f and get: $\frac{1}{\sqrt{N}} \sum_x (-1)^{f(x)}|x\rangle$*

- *We define a diffusion (also known as reflection) operation*

$$D = 2|s\rangle\langle s| - I = H^{\otimes n}(2|0\rangle\langle 0| - 1)H^{\otimes n} \tag{53}$$

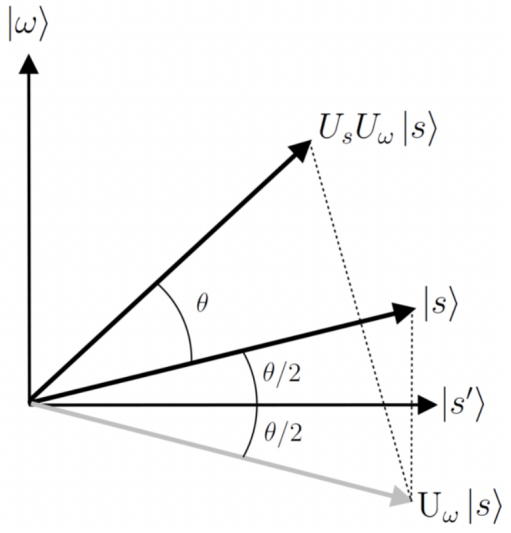
This operator serves as a reflection. For $|\alpha\rangle = \sum_x \alpha_x|x\rangle$ Define $S := \frac{\sum_i \alpha_i}{N}$. Then D maps this state to $\sum_x \alpha'_x|x\rangle$ where $\alpha'_x = 2S - \alpha_x$.

3. *apply the reflection operator*
4. *Repeat for $O(\sqrt{N})$ time*

Theorem 7.2 (Analysis of Grover)**.** *Grover's algorithm succeeds after time $O(\sqrt{N})$.*

Proof. Let x_0 be the solution to our search. We look at the space spanned by $|\omega\rangle := |x_0\rangle$ and $|s'\rangle := \frac{1}{\sqrt{N-1}} \sum_{x \neq x_0} |x\rangle$. Let $|s\rangle = \frac{1}{\sqrt{N}} \sum_x |x\rangle$. See Figure 7. The effect of the oracle query is

Figure 7: Analysis of Grover (from Wikipedia)



given by $U_\omega := (I - 2|\omega\rangle\langle\omega|)$; geometrically the effect of this operator is tantamount to reflecting the sign for the component of the state along the direction $|\omega\rangle$. Similarly the effect of the diffusion (or reflection) operator is given by $U_s = (2|s\rangle\langle s| - I)$, geometrically equivalent to reflecting along $|s\rangle$. Initially, after applying the Hadamard operation, the state of the quantum computer is in $|A_0\rangle := |s\rangle = \frac{1}{\sqrt{N}}|\omega\rangle + \sqrt{\frac{N-1}{N}}|s'\rangle$. In each round we first query then function then the diffusion. Let $G = U_s U_\omega$. The state of our quantum computer after t iterations of G is $|A_t\rangle := G^t|A_0\rangle$.

Initially, the overlap between $|A_0\rangle$ and $|\omega\rangle$ is $\langle A_0|\omega\rangle = \frac{1}{\sqrt{N}}$. Geometrically, this is the same as saying that the angle between this state with the $|s'\rangle$ is $\sin\Delta = \sqrt{\frac{1}{N}}$. In each iteration the angle will increase by 2Δ. Hence after $\sim \sqrt{N}$ iterations we obtain an angle around $\pi/2$ and hence we are at state $|\omega\rangle$. It is important to note that if we keep repeating this algorithm we will deviate from the correct answer.

□

8 Solving systems of linear equations (Optional)

One of the most important computational problems throughout sciences and engineering is solving linear systems. The problem is, given a square matrix and a target vector $\vec{b}$, find vector $\vec{x}$ such that $A\vec{x} = \vec{b}$. We consider the following versions of this problem for which we can gain exponential quantum speedup.

Problem 9 (Linear systems)**.** *Given a $N \times N$ Hermitian matrix A, and a unit vector $|b\rangle$, and another Hermitian matrix M, find vector $\langle x|M|x\rangle$, where $|x\rangle$ is such that $A|x\rangle = |b\rangle$.*

- *In 2008 Harrow Hassidim Lloyd proposed a quantum algorithm that runs in $\log N \cdot O(\kappa^2)$.*

- *Best classical algorithm runs in time $O(N\kappa)$ (or $O(N\sqrt{\kappa})$ for PSD matrices).*

The Algorithm: *Let $\lambda_i, |u_i\rangle$ be the eigenvalue and eigenvectors of A. Consider the decomposition of $|b\rangle = \sum_{i=1}^{N} \beta_i |u_i\rangle$ in the eigenbasis of A.*

1. *Use quantum simulation to prepare e^{iAt}.*
2. *Use quantum phase estimation mapping : $|b\rangle \otimes |0\rangle \mapsto \approx \sum_{j=1}^{N} \beta_j |u_j\rangle \otimes |\lambda_j\rangle$.*
3. *Add an additional ancilla and rotate it conditioned on the value of the $|\lambda\rangle$ register and obtain*

$$\approx \sum_{j=1}^{N} \beta_j |u_j\rangle \otimes |\lambda_j\rangle (\sqrt{1 - \frac{C^2}{\lambda_j^2}}|0\rangle + \frac{C}{\lambda_j}|1\rangle)$$

4. *Uncompute the $|\lambda\rangle$ register and obtain*

$$\approx \sum_{j=1}^{N} \beta_j |u_j\rangle (\sqrt{1 - \frac{C^2}{\lambda_j^2}}|0\rangle + \frac{C}{\lambda_j}|1\rangle)$$

5. *Measure the ancilla register and condition on obtaining a 1 to obtain the state*

$$\approx \sqrt{\frac{1}{\sum_{j=1}^{N} C^2 |\beta_j|^2 / |\lambda_j|^2}} \sum_{j=1}^{N} \beta_j \frac{C}{\lambda_j} |u_j\rangle$$

6. *Up to normalization we obtain some state $\approx \sum_{j=1}^{N} \beta_j / \lambda_j |u_j\rangle = A^{-1}|b\rangle = |x\rangle$. The normalization is the probability of obtaining 1.*
7. *Measure POVM $\{M, I - M\}$ to obtain $\langle x|M|x\rangle$*

Analysis:

- *Assuming A is $s-$sparse, we can perform quantum simulation for e^{iAt} in time $O(\log N)s^2 t$.*
- *If A is not Hermitian, then define*

$$\Lambda = \begin{pmatrix} 0 & A \\ A^\dagger & 0 \end{pmatrix}$$

  and solve $\Lambda|y\rangle = \begin{pmatrix} |b\rangle \\ 0 \end{pmatrix}$, where $|y\rangle = \begin{pmatrix} 0 \\ |x\rangle \end{pmatrix}$.

- *If $f(i,j) = \sum_{k=i}^{j} |\langle k|b\rangle|^2$ is efficiently computable, then we can prepare $|b\rangle$ efficiently; otherwise we can assume $|b\rangle$ is given to us as a subroutine in some other algorithm.*
- *in order to succeed we need to choose $C = O(1/\kappa)$ and succeed with probability $\Omega(1/\kappa^2)$.*

9 The hidden subgroup problem (Optional)

The most immediate generalization of Shor's problem which involves the (Abelian) cyclic group is to extend it to families of problems known as the "Hidden-Subgroup Problems" over arbitrary groups. Shor and Kitaev showed that we get quantum polynomial-time algorithms for Abelian groups. We don't know if the same is possible for arbitrary groups.

Problem 10 (The Hidden Subgroup Problem)**.** *Let $(G, \cdot)$ be a group and $H \leq G$ be a subgroup. Suppose $f : G \to [N]$ is such that for $x, y \in G$, $f(x) = f(y)$ iff $x = hy$ for some $h \in H$. (In other words of x, y belong to a left coset of H.) Find the generators of H.*

- *The Simon's problem is a special case: $G = (\mathbb{Z}_2^N, +)$ and $H = \{0, s\}$.*
- *Shor's problem is also a special case: $G = (\mathbb{Z}_N, +)$ and $H = \{\ldots, -2s, -s, 0, s, 2s, \ldots\}$ and $f_x(l) = x^l \mod N$.*

Theorem 9.1 (Shor-Kitaev)**.** *The Hidden subgroup problem over finite Abelian groups can be solved within* BQP.

- *We know if SAT is reducible to HSP then the polynomial Hierarchy collapses.*
- *HSP is within* $\mathrm{NP} \cap \mathsf{coAM}$

Theorem 9.2. *Graph Isomorphism $\leq$ HSP*

Proof. For a graph C let Automorphism group of C be the set of permutations $Aut(C) := \{\pi \in S_n : \pi(C) \cong C\}$. Given two graphs C_1, C_2 let $C = C_1 \cup C_2$. Now if $C_1 \ncong C_2$ then $Aut(C) = Aut(C_1) \times Aut(C_2)$. Here is the reduction $G = S_n$, $H = Aut(C)$, $f_C(\pi) = \pi(C)$. □

9.1 Query complexity of HSP (optional)

Theorem 9.3 (Ettinger-Höyer-Knill)**.** *HSP can be solved using polynomial many queries to $f : G \to \mathbb{N}$, where f satisfies $\forall x, y \in G, f(x) = f(y)$ iff $x = hy$ for some $h \in H \trianglelefteq G$.*

Proof sketch. Suppose we have

1. Prepare the superposition: $\frac{1}{\sqrt{|G|}} \sum_{x \in G} |x\rangle$
2. Query f to prepare: $\frac{1}{\sqrt{|G|}} \sum_{x \in G} |x\rangle|f(x)\rangle$
3. Measure the second register to get a superposition: $|C\rangle = \frac{1}{\sqrt{|H|}} \sum_{h \in H} |hy\rangle$
4. Repeat this for $K \approx \log^2 |G|$ times to get: $|C_1\rangle, \ldots, |C_K\rangle$.

We claim is that there is a measurement to give us H. The measurement is not necessarily efficient. We observe there are at most $|G|^{\log|G|}$ different subgroups of G. The reason is that there are at most $\log|G|$ generators for G, and there are, therefore at most $\binom{|G|}{\log|G|}$ many distinct subgroups.

1. Prepare $|\psi_H\rangle = |C_1\rangle \otimes \ldots \otimes |C_K\rangle$.

We observe that if $H \neq H' \trianglelefteq G$ then $|H \cap H'| \leq |H|/2$. To see this, we observe that since $H \neq H'$ there exists a nontrivial $x \in H/H'$. Therefore for any $y \in H \cap H', xy \in H/H'$. Therefore $|H \cap H'| \leq |H/H'|$ and since $|H| = |H \cap H'| + |H/H'|$, $|H \cap H'| \leq |H|/2$. Therefore $|\langle H, H'\rangle| \leq \frac{1}{2}$. This because if $|H\rangle = \frac{1}{\sqrt{|H|}} \sum_{h\in H} |hy\rangle$ and $|H'\rangle = \frac{1}{\sqrt{|H'|}} \sum_{h'\in H'} |h'y'\rangle$ then $\langle H, H'\rangle = \frac{|H\cap H'|}{\sqrt{|H||H'|}}$. That means if two subgroups are all the same, they are exactly the same. Therefore $|\langle \psi_H, \psi_{H'}\rangle| \leq \frac{1}{2^K}$. There are at most $|G|^{\log|G|}$ different such ψ_H's, therefore, it is enough to take $K \approx \log^2|G|$ to reduce the error below the required amount.

□

Exercise: *How can we perform the measurement above?*