Meeting 8.1 Some quantum algorithms

I. Simon's problem

II. Reducing Factoring to period-Finding

III. Phase estimation and period-Finding

I. Simon's problem

First, recall Deutsch's problem can be solved in $O(1)$ on quantum computer.

Input: a black box function

$$F : \{0,1\}^n \longrightarrow \{0,1\}$$

which is promised to be either:

i) constant, or

ii) balanced, meaning $\# F^{-1}(0) = \# F^{-1}(1)$.

Problem: Decide whether F is constant or balanced.

Classically, requires $2^{n-1}+1$ evaluations of F.

"Oracle separation" of BQP and P.

Since BPP is "realistic" classical computing, can we separate BPP and BQP?

Warning: $P \subseteq BPP \subseteq BQP \subseteq PSPACE$, and we don't know if $P \neq PSPACE$!

Is there an ORACLE separation of BPP and BQP?

Simon's problem

Given black box/oracle function

$$f : \{0,1\}^n \longrightarrow \{0,1\}^k$$

replace w/ X s.t. $|X| \geq 2^{n-1}$

$(k \geq n-1)$

which is promised to satisfy

$$f(x) = f(y) \text{ if and only if } x - y \in \{0, s\}$$

for some $s \in \{0,1\}^n$.

Problem: Find s.

Can't be solved in BPP^f. Even a probabilistic algorithm requires at least $2^{n/2}$ queries to oracle to find $x \neq y$ with $f(x) = f(y)$.

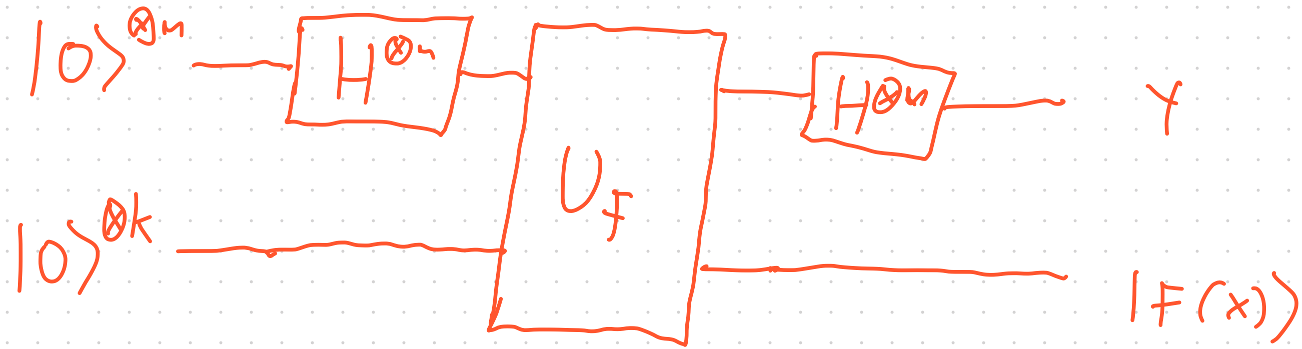
Simon's algorithm

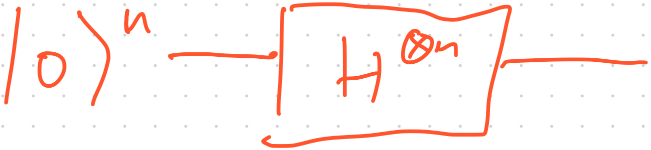
Suppose have usual "quantum oracle" for f

$$U_f : (\mathbb{C}^2)^{\otimes n} \otimes (\mathbb{C}^2)^{\otimes k} \longrightarrow (\mathbb{C}^2)^{\otimes n} \otimes (\mathbb{C}^2)^{\otimes k}$$

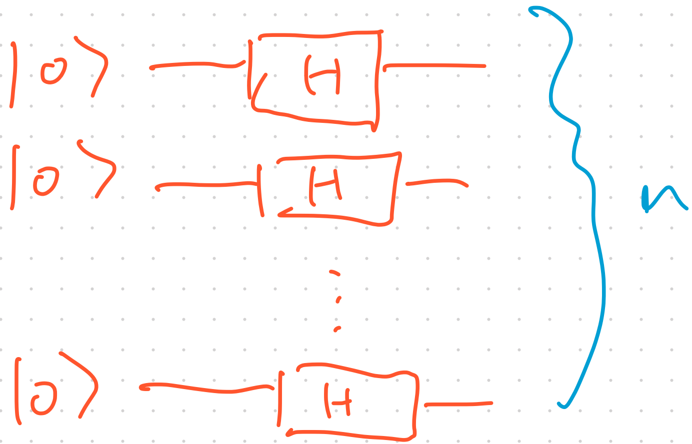
$$|x, y\rangle \longmapsto |x, y \oplus f(x)\rangle.$$

Use simple circuit





shorthand for



Output is

$$(H^{\otimes n} \otimes Id) \circ U_F \circ (H^{\otimes n} \otimes Id) |0^n\rangle \otimes |0^k\rangle$$

$$= (H^{\otimes n} \otimes Id) \circ U_F \left(\frac{1}{2^{n/2}} \sum_{x=0}^{2^n-1} |x\rangle |0^k\rangle \right)$$

$$= H^{\otimes n} \otimes Id \left(\frac{1}{2^{n/2}} \sum_x |x\rangle |F(x)\rangle \right)$$

$$= \frac{1}{2^n} \sum_{x,y} (-1)^{x \cdot y} |y\rangle |F(x)\rangle$$

dot product over $\mathbb{Z}/2\mathbb{Z}$.

$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$$

$$H^{\otimes n} \sum_x |x\rangle = \frac{1}{2^{n/2}} \sum_{x,y} (-1)^{x \cdot y} |y\rangle$$

$$= \frac{1}{2^n} \sum_{x,y} (-1)^{x \cdot y} |y\rangle |F(x)\rangle$$

If we measure the y output in computational basis, then probability of seeing a specific bit string $y \in \{0,1\}^n$ is

$$\left\| \frac{1}{2^n} \sum_{x} (-1)^{x \cdot y} |F(x)\rangle \right\|^2$$

Now sum of $I = \text{Image}(F)$:

$$\left\| \frac{1}{2^n} \sum_{x} (-1)^{x \cdot y} |F(x)\rangle \right\|^2 = \left\| \frac{1}{2^n} \sum_{z \in I} \left[(-1)^{x_z \cdot y} + (-1)^{(x_z + s) \cdot y} \right] |z\rangle \right\|^2$$

where $f^{-1}(z) = \{ x_z, x_z + s \}$

$$\left\| \frac{1}{2^n} \sum_{z \in I} \left[(-1)^{x_z \cdot y} + (-1)^{(x_z + s)\cdot y} \right] |z\rangle \right\|^2 = \begin{cases} 0 & \text{if } y \cdot s = 1 \bmod 2 \\ 1/2^{n-1} & \text{if } y \cdot s = 0 \bmod 2 \text{ and } s \neq 0^n \\ 1/2^n & \text{if } s = 0^n \end{cases}$$

"destructive interference"

"Constructive interference"

Take-away:

Get uniform distribution on

$$\{0, s\}^\perp = \{ x \in \{0,1\}^n \mid x \cdot s = 0 \bmod 2 \}$$

Performing experiment ℓ times, get $x_1, x_2, \dots, x_\ell$ such that $x_i \cdot s = 0 \bmod 2$ for all s. Generate $\{0, s\}^\perp$ with probability

$$\geq 1 - \frac{|\{0, s\}^\perp|}{2^\ell} = 1 - \frac{1}{2^{\ell - n + 1}}.$$

If $x_1, \dots, x_\ell$ generate, can recover s as (non-trivial) solution to

$$\begin{cases} x_1 \cdot s = 0 \mod 2 \\ x_2 \cdot s = 0 \mod 2 \\ \vdots \\ x_\ell \cdot s = 0 \mod 2 \end{cases}$$

What the heck just happened?

Not exactly clear, but it generalizes...

Hidden subgroup problem

Input: Finitely generated group G, set X and black box function

$$F : G \longrightarrow X$$

that is constant on cosets of $H \leq G$ (and distinct on distinct cosets).

Problem: Find generators of H.

Abelian hidden subgroup problems well understood. (Soluable in BQP?)

Many important special cases among them, including:

- Deutsch's problem
- Simon's problem

Contrived

- discrete log
- order-finding
- period-Finding

Useful!

Basic idea: can implement Fourier transforms on abelian groups on quantum computer

Rather than do this generally, let's cut to the chase:

Factoring.

II. Reducing factoring to period finding

Factoring Problem

Given integer N in binary, compute prime factorization

$$N = p_1^{k_1} \cdots p_\ell^{k_\ell}.$$

reduces to

Factor Finding

Given $N > 1$, find $1 < k < N$ that divides N, or, if not possible, return "Is Prime."

Note: Miller-Rabin (BPP) or Agrawal-Kayal-Saxena (P) primality test allow us to assume N composite.

Factor-Finding for composite integers reduces in BPP to

Order-Finding

Given N and $1 < x < N$ with $\gcd(x, N) = 1$, find smallest $r > 1$ such that

$$x^r \equiv 1 \mod N.$$

So, r is order of x in $(\mathbb{Z}/N\mathbb{Z})^\times$.

Factor - Finding $\rightsquigarrow$ Order - Finding

Two basic steps:

1. $x^2 \equiv 1 \mod N$ but $x \not\equiv \pm 1 \mod N$ yields Factor (either $\gcd(x-1, N)$ or $\gcd(x+1, N)$)

2. A randomly chosen $y \in (\mathbb{Z}/N\mathbb{Z})^\times$ has even order r and $y^{r/2} \not\equiv \pm 1 \mod N$ w/ large probability.

IF we have such a y, then

$$\gcd(y^{r/2} \pm 1, N) \text{ will}$$

be a Factor, by step 1.

Factor-Finding $\rightsquigarrow$ Order-Finding in BPP

Two precise theorems:

1. Suppose N has L bits, is composite, and x satisfies

$$\begin{cases} 1 < x < N \\ x^2 \equiv 1 \mod N \\ \cancel{x \not\equiv \pm 1 \mod N.} \end{cases}$$

Then either $\gcd(x-1, N)$ or $\gcd(x+1, N)$ is a nontrivial factor of N.

2. Suppose N odd, composite, and $N = p_1^{k_1} \cdots p_\ell^{k_\ell}$ is prime factorization. If $1 \le x \le N-1$ is a uniformly random integer w/ $\gcd(x, N) = 1$ and r is order of x in $(\mathbb{Z}/N\mathbb{Z})^\times$, then

$$\operatorname{Prob}\left(r \text{ even and } x^{r/2} \neq -1 \bmod N\right) \ge 1 - \frac{1}{2^\ell}.$$
$$\ge 1/2.$$

The reduction.

1. If N even, return 2. $(O(1))$
2. If $N = a^b$, $a \geq 1$, $b \geq 2$, return a. $(O(L^3))$
3. Choose random $1 < x < N-1$. If $\gcd(x, N) > 1$, return gcd. $(O(L^2))$
4. Find r, the order of x in $(\mathbb{Z}/N\mathbb{Z})^\times$. (Use quantum computer)
5. If r odd, pick another x. $(O(1))$
6. If r even, test if $\gcd(x^{r/2}+1, N)$ or $\gcd(x^{r/2}-1, N)$ is a factor. If neither is, then pick another x. $(O(L^2))$

Shows Factor-Finding in $\mathsf{fBPP}^{\text{order-finding}}$

Since $\mathsf{f}BPP \subseteq \mathsf{f}BQP$, if we can show

$$\text{Ordering-Finding} \in \mathsf{f}BQP,$$

then Factoring $\in \mathsf{f}BQP$ too.

III. Phase estimation and order-finding

Phase estimation is a general procedure for estimating eigenvalues of a unitary (or Hermitian) operator U when we have controlled-U^j operator accessible as oracles for every j.

Unitary U + evector $|u\rangle \rightsquigarrow \tilde{\theta}$ where $U|u\rangle = e^{2\pi i\theta}|u\rangle$

Controlled-U^j:

$$C\text{-}U^j: \quad |j\rangle \otimes |u\rangle \longmapsto |j\rangle \otimes U^j|u\rangle.$$

Quantum phase estimation protocol (sans protocol...)

Input: (i) Black box for $C-U^j$

(ii) eigenvector $|u\rangle$ with $U|u\rangle = e^{2\pi i \varphi_u}|u\rangle$

(iii) integer n

(iv) $\varepsilon > 0$ (e.g. $\varepsilon = 1/3$)

Output: n-bit approximation $\tilde{\varphi}_u$ to φ_u

Performance: $O(t^2)$ runtime, where $t = n + \lceil \log(2 + \frac{1}{\varepsilon}) \rceil$

$t = O(n)$

- One call to $C-U^j$
- Succeeds w/ probability at least $1-\varepsilon$.

I won't discuss circuits for phase estimation now but instead how to reduce order finding to it.

Want to find order of x in $(\mathbb{Z}/N\mathbb{Z})^\times$.

Use

$$U: |y\rangle \longmapsto \begin{cases} |xy \bmod N\rangle & \text{if } 0 \leq y \leq N-1 \\ |y\rangle & \text{if } N \leq y \leq 2^L \end{cases}$$

where L is # bits in description of N.

Eigenvectors of U: (not all of them...)

$$|u_s\rangle = \frac{1}{\sqrt{r}} \sum_{k=0}^{r-1} \exp\left[\frac{-2\pi i s k}{r}\right] |x^k \bmod N\rangle$$

For $0 \leq s \leq r-1$. Eigenvalues: $e^{2\pi i s/r}$

Issues: C-U's? Modular exponentiation...

$|u_s\rangle$? Prepare $\frac{1}{\sqrt{r}} \sum |u_s\rangle = |00\cdots01\rangle$

instead...

$s/r \rightsquigarrow r$? Continued Fraction trick...