Lecture 2: First Algorithms

First quantum algorithms date back to 1980's. But let's warm up with an older, classical algorithm. Much older. About 2500 years older, in fact!

Euclid's Algorithm

Input: $a, b \in \mathbb{Z}$
Output: $GCD(a,b)$ (greatest common divisor)

First ever (known, non-trivial) algorithm.
[Euclid, "Elements, Book 10", ~300 BC]

Algorithm:

1. while $b \neq 0$
2. $\quad t \leftarrow b$
3. $\quad b \leftarrow a \bmod b$
4. $\quad a \leftarrow t$
5. return a

Euclid actually used repeated subtraction, i.e.

1. while $a \neq b$
2. $\quad$ if $a > b$
3. $\quad\quad a \leftarrow a - b$
4. $\quad$ else
5. $\quad\quad b \leftarrow b - a$
6. return a

Exercise:
Prove Euclid's algorithm is correct, and performs $O(\log b)$ divisions.
Use this to argue GCD is in P.

Exercise (Exponentiation by squaring)
Prove that a^n can be computed using $O(\log n)$ multiplications.

Hint: Try calculating 13^9 by hand (i.e. pen & paper only, no calculator!)
Do any shortcuts occur to you?

1. Deutsch-Jozsa Algorithm

First quantum algorithm was due to Deutsch [Deutsch 1985].

First example of a computational problem where quantum computers have a computational advantage over classical.

Pick up the story with [Deutsch-Jozsa '92] which improved this to exponential advantage.

Problem (Deutsch-Jozsa)

Input: Black-box oracle
$f: \{0,1\}^n \to \{0,1\}$

Promise: f either constant or balanced

constant $\Rightarrow$ either $\forall x\ f(x)=0$
or $\forall x\ f(x)=1$

balanced $\Rightarrow$ $|\{x : f(x)=0\}|$
$= |\{x : f(x)=1\}|$

Output: decide which!

(Original Deutsch problem is $n=1$ case.)

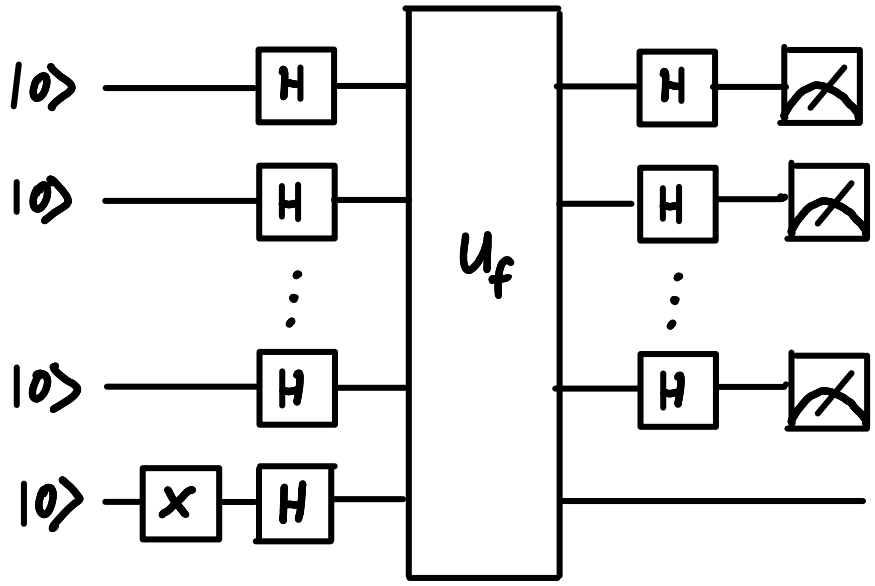
Input is given as a "black box" that evaluates f on its input.

Quantum black box U_f has to be unitary ($\Rightarrow$ reversible):

$$U_f \underbrace{|x\rangle}_{n \text{ qubits}} \underbrace{|y\rangle}_{1 \text{ qubit}} \longrightarrow |x\rangle\, |y \oplus f(x)\rangle$$

$$U_f\, |x\rangle\, |00\ldots0\rangle \longrightarrow |x\rangle\, |f(x)\rangle$$

Algorithm:



Output $\begin{cases} \text{"constant"} & \text{if measure } 00\ldots0 \\ \text{"balanced"} & \text{otherwise} \end{cases}$

Analysis:

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

$$H^{\otimes n}|00\cdots0\rangle = |++\cdots+\rangle$$

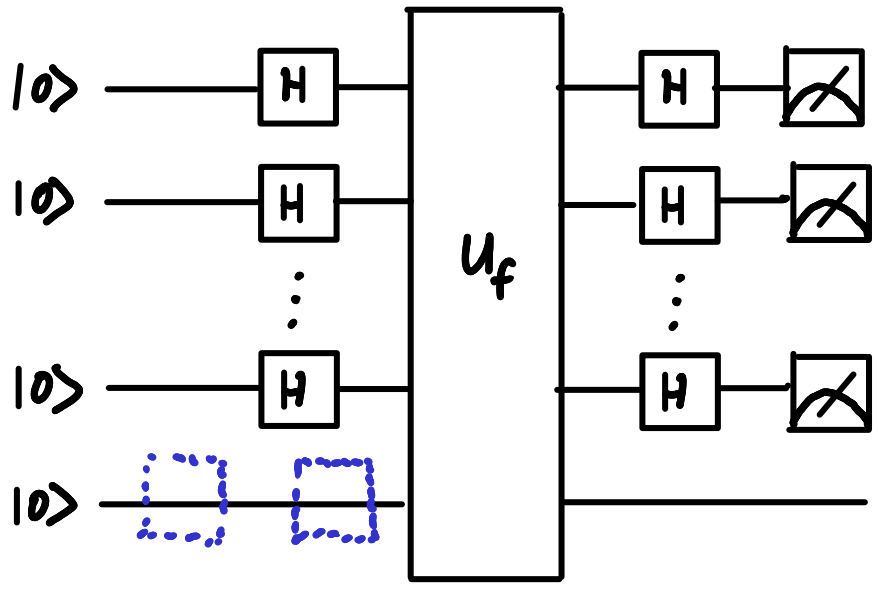
$$= \underbrace{\frac{1}{\sqrt{2}}(|0\rangle+|1\rangle)\ \frac{1}{\sqrt{2}}(|0\rangle+|1\rangle)\cdots\frac{1}{\sqrt{2}}(|0\rangle+|1\rangle)}_{n \text{ times}}$$

$$= \frac{1}{\sqrt{2^n}}(|00\cdots0\rangle + |10\cdots0\rangle + |01\cdots0\rangle + \ldots)$$

$$= \frac{1}{\sqrt{2^n}}\sum_x |x\rangle$$

→ feed superposition over all input bit strings x into "input" register of U_f.

Consider for a moment the circuit without the HX on the last qubit:



Input to U_f in this case is

$$|\psi\rangle = \frac{1}{\sqrt{2^n}} \sum_x |x\rangle |0\rangle$$

To compute $U_f |\psi\rangle$, we can just compute $U_f |x\rangle|0\rangle$ and then use linearity: (i.e. the fact that

$$U(\alpha|\varphi\rangle + \beta|\phi\rangle) = \alpha U|\varphi\rangle + \beta U|\phi\rangle)$$

$$U_f |x\rangle|0\rangle = |x\rangle|0 \oplus f(x)\rangle = |x\rangle|f(x)\rangle$$

so

$$U_f |\psi\rangle = \frac{1}{\sqrt{2^n}} \sum_x U_f|x\rangle|0\rangle = \frac{1}{\sqrt{2^n}} \sum_x |x\rangle|f(x)\rangle$$

"Quantum parallelism": output is superposition over function applied to all possible inputs simultaneously!

Warning: this alone is <u>not</u> enough to gain a quantum advantage!!

If we measure this state in the computational basis, it collapses to a single $|x\rangle|f(x)\rangle$ chosen uniformly at random.

Exercise: do the calculation and convince yourself of this.

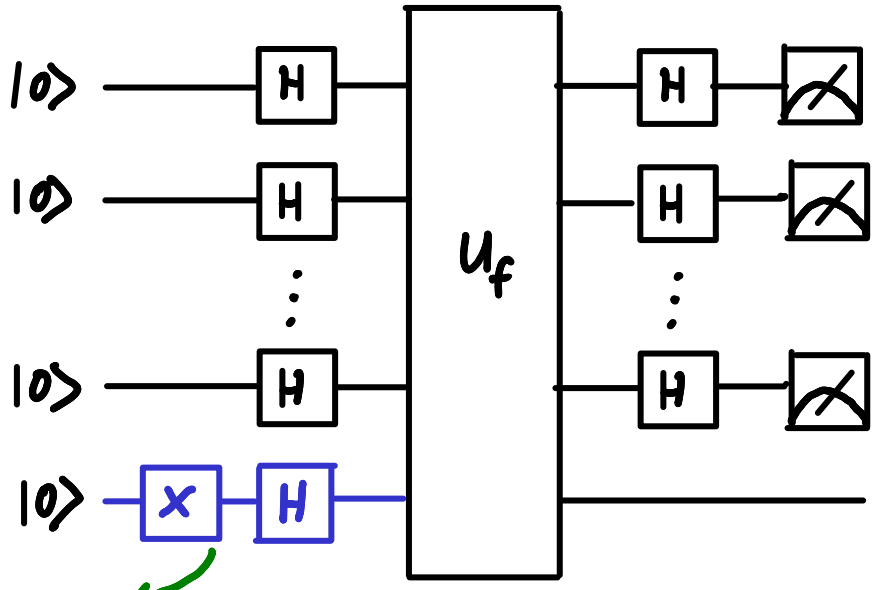
Can achieve exactly the same thing purely classically, by picking x uniformly at random & computing $f(x)$.

Can't use quantum parallelism to extract more than one output at a time ⟶ Is quantum parallelism useless?

No! But to exploit information encoded in the superposition, have to use interference in a clever way to extract "global" information about set of outputs.

This is typically the difficult part of designing quantum algorithms.

That's what the HX is for in the real Deutsch-Josza circuit:



remember: matrices compose "backwards"!

$$HX|0\rangle = H|1\rangle = |-\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$$

→ feed $|-\rangle$ into "output" qubit of U_f

⇒ Input to U_f is the state:

$$|\psi\rangle = \left(\frac{1}{\sqrt{2^n}} \sum_x |x\rangle\right) |-\rangle$$

Use linearity again:

$$\begin{aligned} U_f |x\rangle |-\rangle \\ &= U_f |x\rangle \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) \\ &= |x\rangle \frac{1}{\sqrt{2}}(|0 \oplus f(x)\rangle - |1 \oplus f(x)\rangle) \end{aligned}$$

cf. Def. of U_f

$$= \begin{cases} |x\rangle \frac{1}{\sqrt{2}} (|0\rangle - |1\rangle) & \text{if } f(x) = 0 \\ |x\rangle \frac{1}{\sqrt{2}} (|1\rangle - |0\rangle) & \text{if } f(x) = 1 \end{cases}$$

$$= \begin{cases} |x\rangle |-\rangle \\ -|x\rangle |-\rangle \end{cases}$$

$$= (-1)^{f(x)} |x\rangle |-\rangle$$

This uniquely quantum effect of picking up a phase depending on the input by initialising the "output" qubit to a suitable (non-computational basis) state is sometimes called _phase kick-back.

→ Output of circuit (before measurement):

$$U_f |\psi\rangle$$

$$= U_f \frac{1}{\sqrt{2^n}} \sum_x |x\rangle |-\rangle$$

$$= \frac{1}{\sqrt{2^n}} \sum_x U_f |x\rangle |-\rangle$$

$$= \frac{1}{\sqrt{2^n}} \sum_x (-1)^{f(x)} |x\rangle |-\rangle$$

What does this state look like if f is constant or balanced?

Constant: $\forall x \; f(x) = 0$ or $\forall x \; f(x) = 1$

$$U_f |\psi\rangle = \frac{1}{\sqrt{2^n}} \sum_x (\pm 1) \, |x\rangle |-\rangle$$

$$= \pm \frac{1}{\sqrt{2^n}} \sum_x |x\rangle |-\rangle$$

This is exactly the same state we fed into U_f (up to an irrelevant global phase $\pm$).

Since H gate is self-inverse ($H^{-1} = H$ so $H^2 = \mathbb{1}$), applying final column of Hadamards must just undo the original Hadamards & give us back $|00\cdots0\rangle$.

Explicitly:

$$(H^{\otimes n} \otimes \mathbb{1}) \, U_f \underbrace{(H^{\otimes n} \otimes HX) \, |00\cdots0\rangle}_{|\psi\rangle}$$

$$= (H^{\otimes n} \otimes \mathbb{1}) \, U_f |\psi\rangle = (H^{\otimes n} \otimes \mathbb{1}) |\psi\rangle$$

$$= (H^{\otimes n} \otimes \mathbb{1}) \; (H^{\otimes n} \otimes HX) \, |00\cdots0\rangle$$

$$= \mathbb{1} \otimes HX \; |00\cdots0\rangle$$

$$= |00\cdots\rangle |-\rangle$$

$\Rightarrow$ For constant f, always get $00\cdots0$ when we measure $|x\rangle$ register.

Balanced: $|\{x : f(x)=0\}| = |\{x : f(x)=1\}|$

$$U_f |\psi\rangle = \frac{1}{\sqrt{2^n}} \left(\sum_{x:f(x)=0} |x\rangle - \sum_{x':f(x')=1} |x'\rangle \right) |-\rangle$$

Since we have equal numbers of +'s & −'s (balanced), overlap $\langle\psi| U_f |\psi\rangle = 0$.

Hadamard gate is unitary & unitary transformations preserve inner products, so overlap is still 0 after the Hadamards

Explicitly:

$$\underbrace{\langle 00\cdots0|\langle -|}_{\langle\psi|} (H^{\otimes n} \otimes \mathbb{1})\, U_f\, \underbrace{(H^{\otimes n} \otimes HX)|00\cdots0\rangle}_{|\psi\rangle}$$

$$= \langle\psi| U_f |\psi\rangle$$

$$= \frac{1}{2^n} \left(\sum_x \langle x|\langle -| \right) \left(\sum_{a:f(a)=0} |a\rangle - \sum_{b:f(b)=1} |b\rangle \right) |-\rangle$$

$$= \frac{1}{2^n} \left(\sum_{a:f(a)=0} \underbrace{\langle a|a\rangle}_{=1} - \sum_{b:f(b)=1} \underbrace{\langle b|b\rangle}_{=1} \right) \underbrace{\langle -|-\rangle}_{=1}$$

$$= \frac{1}{2^n} \left(|\{x:f(x)=0\}| - |\{x:f(x)=1\}| \right) = 0$$

$\Rightarrow$ For balanced f, probability of measuring $00\cdots0$ is zero.

We have solved the Deutsch-Josza problem with a single query to U_f
+ $(2n+2)$ additional gates
+ n single-qubit measurements
$= 3n+2$ additional operations.

How many queries are required classically?

Worst case classically:

f balanced, but could query up to half $(= 2^n/2)$ of the bit-strings x before we see both outcomes $f(x) = 0$ & $f(x') = 1$

$\longrightarrow$ Classically, worst-case complexity is $2^n - 1$ queries

$\longrightarrow$ Quantum gives <u>exponential</u> advantage!

Discussion:

Is this a fair comparison?

Quantum oracle U_f is strictly more powerful than classical f: allows querying in superposition, in addition to querying on classical bit-strings (i.e. computational basis states).

However, if you build f out of unitary gates, you must be able to query in superposition to be consistent with quantum mechanics (linearity)

Consensus after ~30 years of reseach is this is a fair comparison.

Backed up by examples where we can "de-oracalise" the problem & construct the oracle efficiently out of gates

(Cf. Shor's algorithm, coming up later...)

Limitations:

- Oracle problem, not computational.

  Problem contains "hidden" ingredient (black-box function f / U_f, whose internal workings aren't specified). Not a standard computational problem specifying Boolean function to be computed on input bit-strings.

- Query complexity advantage, not computational complexity.

  Oracle complexity separations do not necessarily survive "de-oracling".

  Implementing black-box f / U_f could negate all the advantage (either by making q. alg. inefficient, or by "opening up the box" allowing more efficient classical algs.)

- Solving Deutsch-Jozsa problem isn't useful for much.

- Advantage only for exact computation; disappears if we tolerate any non-zero probability of error.

  Exact computation is not physically realistic. Always have some non-zero probability of error (cosmic ray flipping output; q. gates depend on continuous parameters so can only be approximated, cf. Solovoy-Kitaev; etc.)

Exercise: Construct a classical algorithm that succeeds with probability $1-\varepsilon$ using at most $O(1/\log \varepsilon)$ queries.

Historically, next significant development was [Bernstein-Vazirani '93].

Still an oracle problem, only gives polynomial separation between quantum & classical, but this separation survives in bounded-error setting. (Exercise)

However, we will skip ahead to ...

2. Simon's Algorithm

Problem (Simon's)

Input: Black box oracle
$f: \{0,1\}^n \longrightarrow \{0,1\}^n$

Promise: $f(x) = f(y) \Leftrightarrow x = y$ or $y \oplus s$

Output: "hidden shift" s

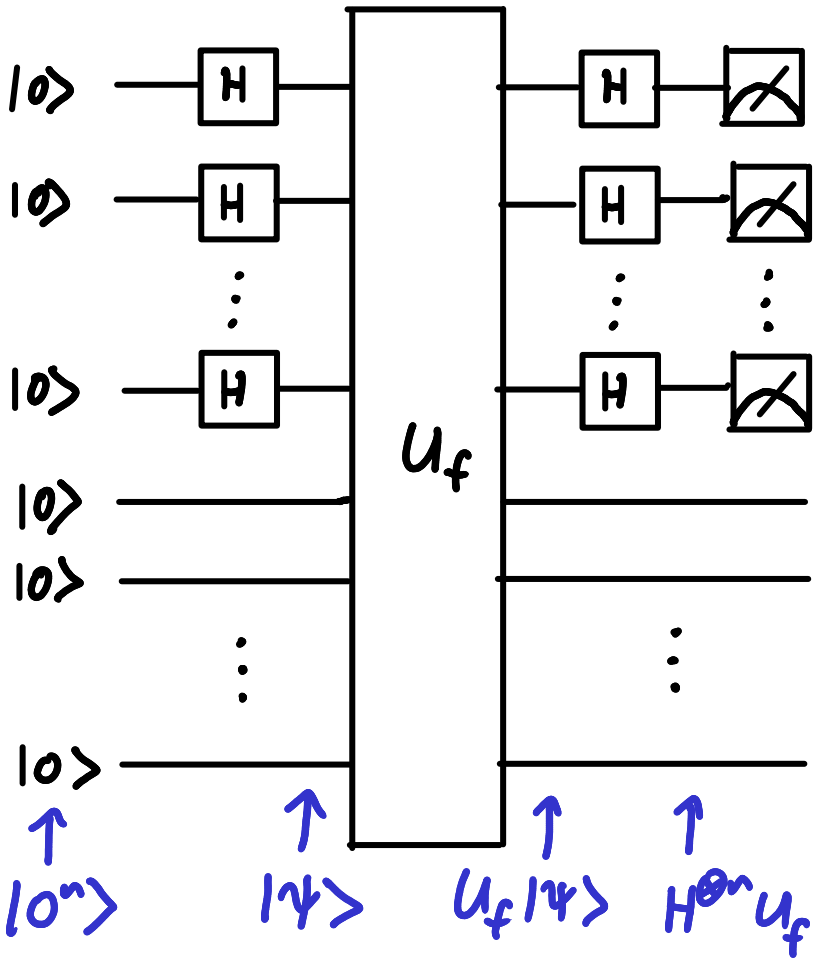
Note if $s = 0^n$ (i.e. all 0's), then f is one-to-one. Otherwise it's two-to-one.

E.g.

x	$f(x)$
000	101
001	010
010	000
011	110
100	000
101	110
110	101
111	010

$s = 110$

Algorithm



+ classical post-processing of measurement outcomes

Black box :

$$U_f |x\rangle |y\rangle = |x\rangle |f(x) \oplus y\rangle$$

Analysis

Input to U_f is superposition over all strings x : (cf. Deutsch-Jozsa)

$$|\psi\rangle = \frac{1}{\sqrt{2^n}} \sum_x |x\rangle |0^n\rangle$$

$$U_f |\psi\rangle = \frac{1}{\sqrt{2^n}} \sum_x |x\rangle |f(x)\rangle$$

(Note no phase kick-back this time.)

Now,

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

$$H|1\rangle = \frac{1}{\sqrt{2}} (|0\rangle - |1\rangle)$$

$$H|x_i\rangle = \frac{1}{\sqrt{2}} (|0\rangle + (-1)^{x_i}|1\rangle) \qquad x_i \in \{0,1\}$$

Pick up -1 in front of $|1\rangle$ when $|x_i\rangle = |1\rangle$

$$H^{\otimes n}|x\rangle = \frac{1}{\sqrt{2^n}} \sum_y (-1)^{??} |y\rangle$$

To get right sign factor, need to know if # of $|1\rangle$'s in $|y\rangle$ on qubits where $|x\rangle$ has a $|1\rangle$ is even or odd.
I.e. parity of # i s.t.
$x_i = y_i = 1 \iff x_i y_i = 1$

$$\longrightarrow ?? = \sum_i x_i y_i \bmod 2$$

$$= x_1 y_1 \oplus x_2 y_2 \oplus \cdots =: x \cdot y .$$

So

$$H^{\otimes n}|x\rangle = \frac{1}{\sqrt{2^n}} \sum_y (-1)^{x \cdot y} |y\rangle$$

Thus output of circuit before measurement is:

$$|\psi_{out}\rangle = (H^{\otimes n} \otimes \mathbb{1}^{\otimes n}) U_f |\psi\rangle$$

$$= \frac{1}{\sqrt{2^n}} \sum_x (H^{\otimes n} |x\rangle) |f(x)\rangle$$

$$= \frac{1}{2^n} \sum_x \sum_y (-1)^{x \cdot y} |y\rangle |f(x)\rangle$$

$$= \frac{1}{2^n} \sum_y |y\rangle \sum_x (-1)^{x \cdot y} |f(x)\rangle$$

What is probability of measuring the string z on $|\psi_{out}\rangle$?

Corresponding measurement operator is $|z\rangle\langle z| \otimes \mathbb{1}$, so

$$\Pr(z)$$

$$= \langle \psi_{out} | (|z\rangle\langle z| \otimes \mathbb{1}) |\psi_{out}\rangle$$

$$= \left(\frac{1}{2^n} \sum_x (-1)^{x \cdot z} \langle z| \langle f(x)| \right)$$

$$\cdot \left(\frac{1}{2^n} \sum_{x'} (-1)^{x' \cdot z} |z\rangle |f(x')\rangle \right)$$

$$= \frac{1}{4^n} \sum_{x,x'} (-1)^{(x+x') \cdot z} \langle f(x) | f(x') \rangle$$

$$= \frac{1}{4^n} \sum_{\substack{x,x': \\ f(x)=f(x')}} (-1)^{(x+x')\cdot z}$$

$$= \begin{cases} \frac{1}{4^n} \sum_x (-1)^{(x+x)\cdot z} & s = 0 \\ \frac{1}{4^n} \sum_x \left((-1)^{(x+x)\cdot z} + (-1)^{(x+x\oplus s)\cdot z} \right) & \text{otherwise} \end{cases}$$

cf. promise on f

$$= \begin{cases} \frac{1}{4^n} \sum_x \cancel{(-1)^{2x\cdot z}} \; (=1) \\ \frac{1}{4^n} \sum_x \left(1 + (-1)^{x\cdot z + x\cdot z \oplus s\cdot z} \right) \end{cases}$$

using $x \oplus s \cdot z = x \cdot z \oplus s \cdot z$

$$= \begin{cases} \frac{2^n}{4^n} = \frac{1}{2^n} & s = 0 \\ \frac{1}{4^n} \sum_x (2) = \frac{1}{2^{n-1}} & s \cdot z = 0 \\ \frac{1}{4^n} \sum_x (1-1) = 0 & s \cdot z = 1 \end{cases}$$

Classical post-processing

Both $s=0$ & $s\neq 0$ cases return string z chosen uniformly at random from set of strings satisfying $s \cdot z = 0$.

Repeat $n-1$ times
→ get $n-1$ strings z_i satisfying

$$\left.\begin{aligned} s\cdot z_1 &= 0 \\ s\cdot z_2 &= 0 \\ &\vdots \\ s\cdot z_{n-1} &= 0 \end{aligned}\right\}$$

$n-1$ linear equations in n unknowns $s_1 \dots s_n$ (bits of s)

If all z_i are linearly independent, we can solve efficiently for $s \neq 0$.
(E.g. Gaussian elimination also works for bits & addition $\oplus$ mod 2.)

$$\Pr(\{z_i\} \text{ linearly indep.}) \geq \prod_{k=0}^{n-2}\left(1-\frac{2^k}{2^n}\right) \geq 1-\sum_{k=0}^{n-2}\frac{1}{2^{n-k}}$$

using $(1-a)(1-b) \geq 1-a-b$

$$= 1-\sum_{k=2}^{n}\frac{1}{2^k} > \frac{1}{2}.$$

→ Succeed in identifying $s \neq 0$ with prob. $> \frac{1}{2}$.
(Note: any prob. > 0 would suffice.)

What if we fail?
Repeat whole procedure m times:

$\Pr(\text{succeed in identifying } s \neq 0)$

$= 1 - \Pr(\text{fail all } m \text{ times})$

$\geq 1 - \frac{1}{2^m}$

$\rightarrow$ Repeating $m = \log 1/\varepsilon$ times succeeds in identifying $s \neq 0$ with prob. $\geq 1-\varepsilon$.

If we've failed to identify $s \neq 0$ after $\log 1/\varepsilon$ repetitions, output $s = 0$.

For any $\varepsilon > 0$, quantum algorithm solves Simon's problem with probability $\geq 1-\varepsilon$ using $\log 1/\varepsilon$ queries to oracle ($\log 1/\varepsilon \cdot \text{poly}(n)$ total operations).

Classical lower bound

Intuition: need to find two inputs x, y s.t. $f(x) = f(y)$.

There are 2^n different inputs and no other structure to f

$\longrightarrow$ Need $\Omega(2^{n/2})$ queries to have good chance of finding $f(x) = f(y)$.

Proving this rigorously is not as easy as it looks! Often the case that classical lower bounds are harder to prove than constructing the quantum algs.

Discussion:

Simon's algorithm gives exponential separation between quantum & classical computation <u>even</u> when we allow some probability of error

However, other limitations similar to Deutsch-Jozsa (oracle problem; query complexity; not useful).