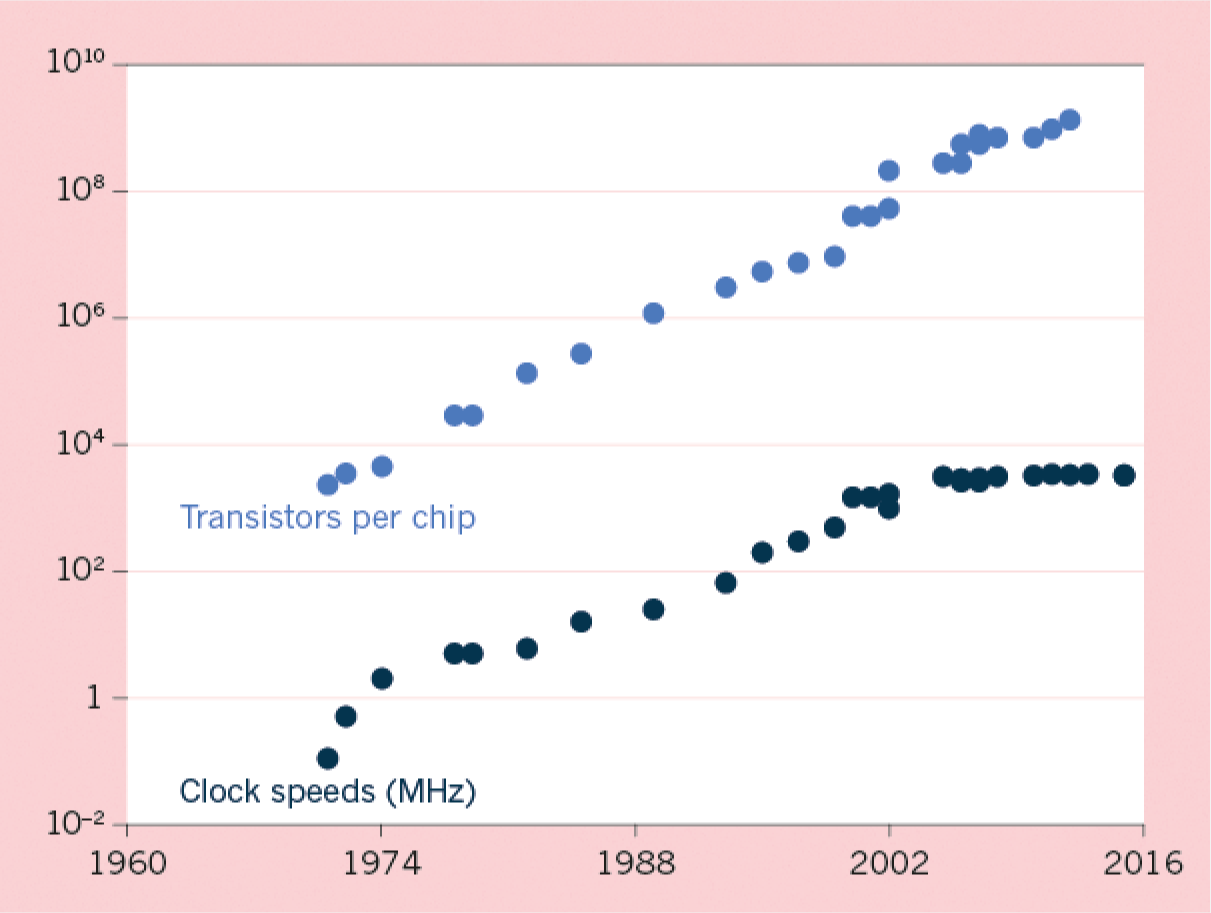
Quantum Algorithms

Shelby Kimmel

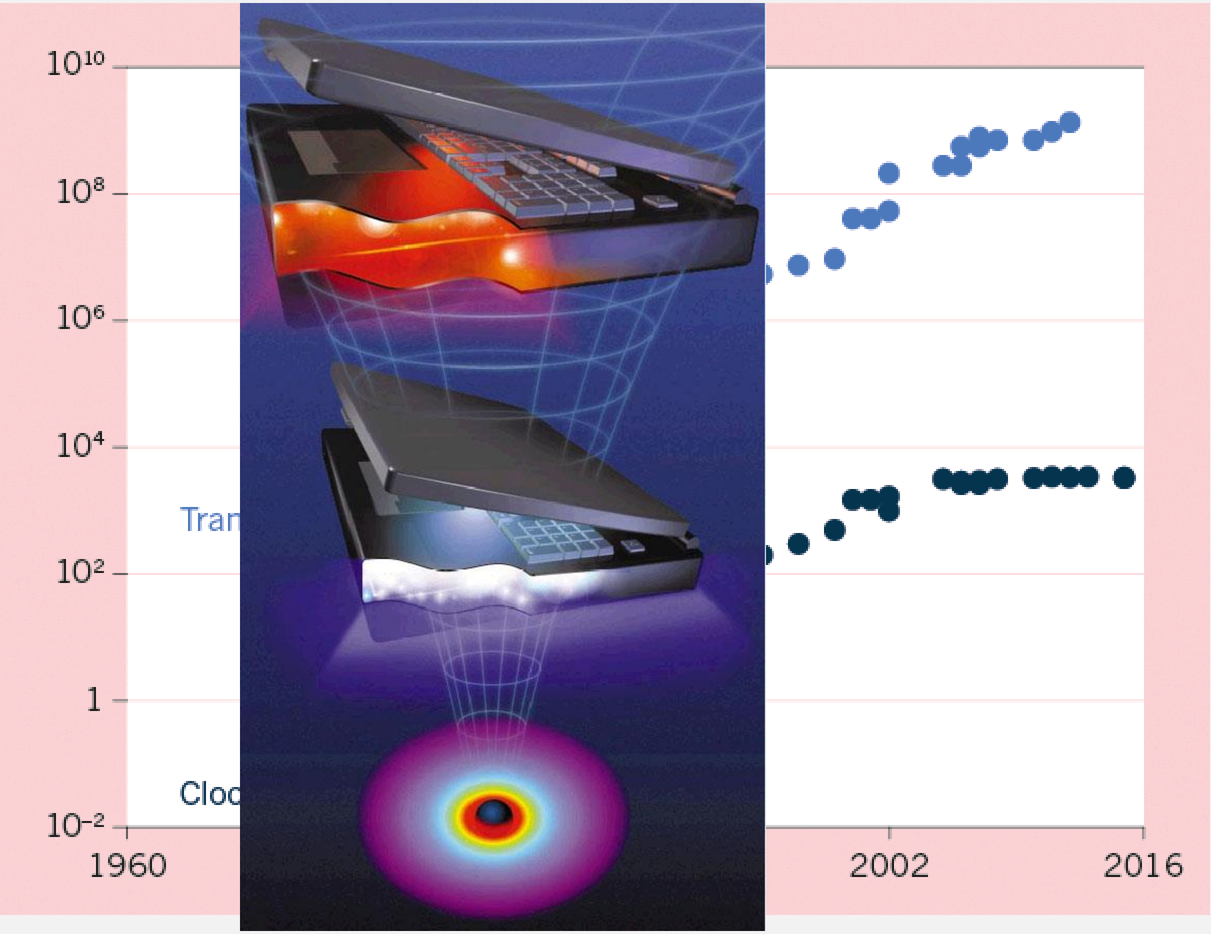
Joint Center for Quantum Information and Computer Science

How Good Can Computers Get?



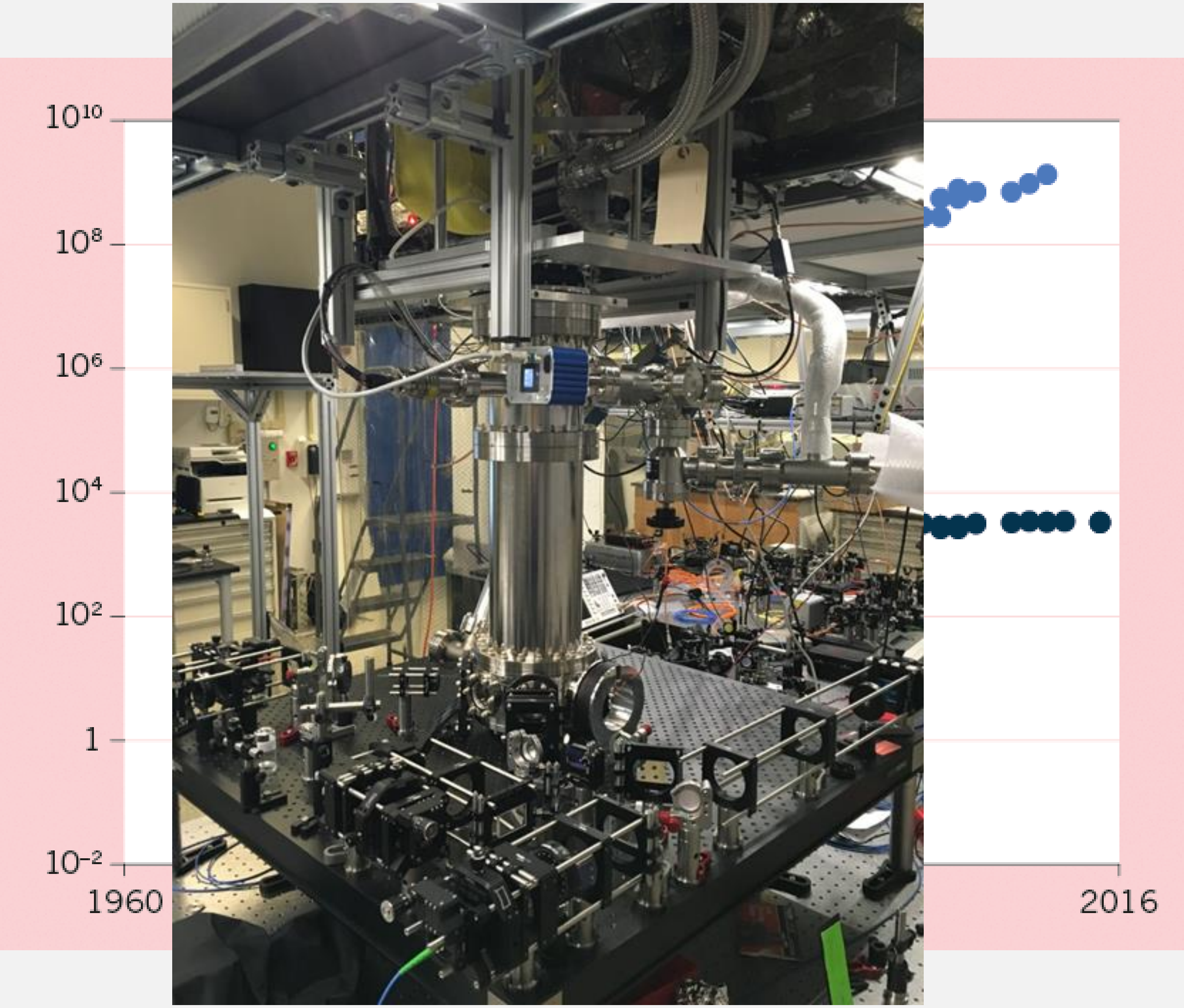
Waldrop, Nature, 2016

How Good Can Computers Get?



Lloyd, Nature, 2016

How Good Can Computers Get?



Monroe Lab (CryoSim)

Quantum Algorithm

Algorithm = instructions

Quantum Algorithm

Quantum Algorithm = instructions

Outline

1. What problems have fast quantum algorithms?
2. Metaphorical interlude: why do quantum computers have an advantage?
3. When is there a provable quantum advantage?

Fast and Exciting Quantum Algorithms

Factoring

Quantum Chemistry

Factoring

3107418240490043721350750035888567930037346022842727545720161948823206
4405180815045563468296717232867824379162728380334154710731085019195485
29007337724822783525742386454014691736602477652346609

$$= p \times q$$

Factoring

3107418240490043721350750035888567930037346022842727545720161948823206
4405180815045563468296717232867824379162728380334154710731085019195485
29007337724822783525742386454014691736602477652346609

=

1634733645809253848443133883865090859841783670033092312181110852
389333100104508151212118167511579

×

1900871281664822113126851573935413975471896789968515493666638539
088027103802104498957191261465571

Factoring

- Best classical algorithm: exponential in cube root of number of digits d:

$$\sim e^{\sqrt[3]{d}}$$

Rubinstein 2013

- Best quantum algorithm: cubic in number of digits:

$$\sim d^3$$

Shor 1997

Factoring

Why do we care?

- Security of modern electronic commerce relies on public-key cryptosystems (e.g. sharing credit care info over internet).
- Public-key cryptosystems are only safe if factoring (and similar problems) are difficult.

- If we build a quantum computer, we can break current cryptosystems.

Fast and Exciting Quantum Algorithms

Factoring

Quantum Chemistry

Quantum Chemistry

Current classical computers can only simulate molecules with less than ~ 70 electronic states.

- Number of bits scales exponentially in number of states

Quantum computers only require ~ 1 qubit per electronic state

- Can simulate on small quantum computers (in principle)

Poulin et al 2014, Wecker et al 2014

Quantum Chemistry

Exist quantum algorithms for

- Thermal Rate Constant = rate of chemical reaction
- Energy structure of molecules
- Simulating solid state systems (superconductors, spin glasses, metamaterials)

Quantum Chemistry

Exist quantum algorithms for

- Thermal Rate Constant = rate of chemical reaction
- Energy structure of molecules
- Simulating solid state systems (superconductors, spin glasses, metamaterials)

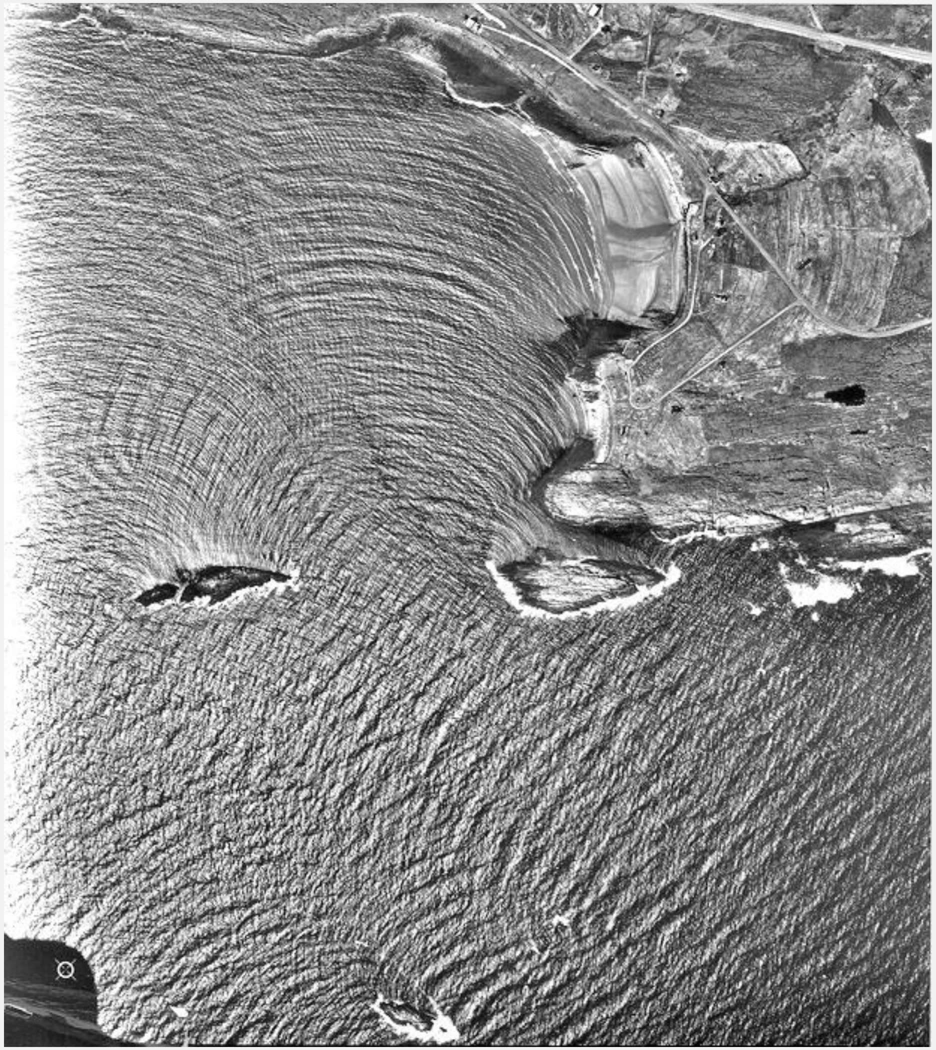
Applications

- New drug development
- New devices/technology (batteries, solar cells, better classical computers)
- Carbon capture

Outline

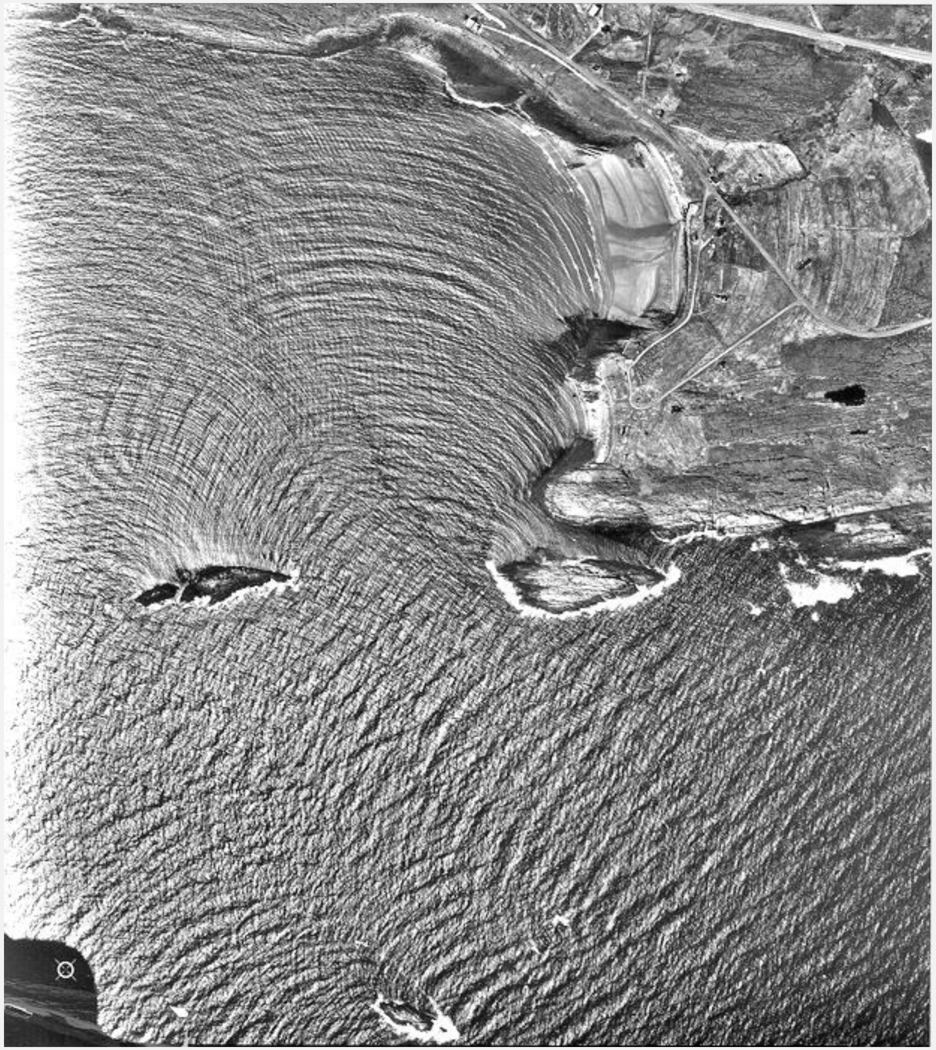
1. What problems have fast quantum algorithms?
2. Metaphorical interlude: why do quantum computers have an advantage?
3. When is there a provable quantum advantage?

Metaphor for quantum computer



Metaphor for quantum computer

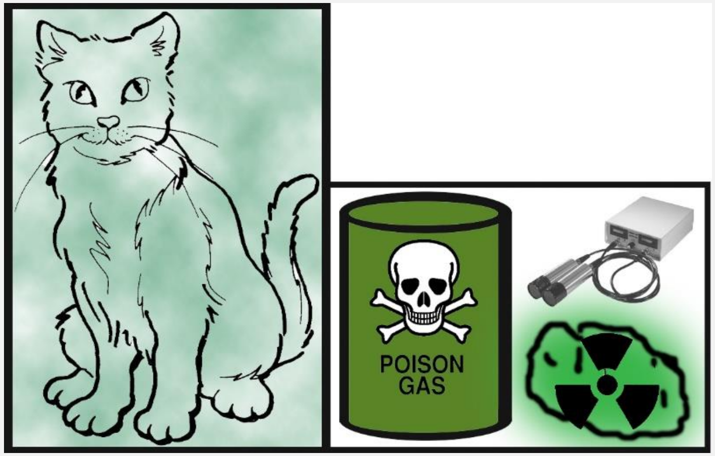
- Writing algorithm is like engineering wave size and location on a beach



Metaphor for quantum algorithms

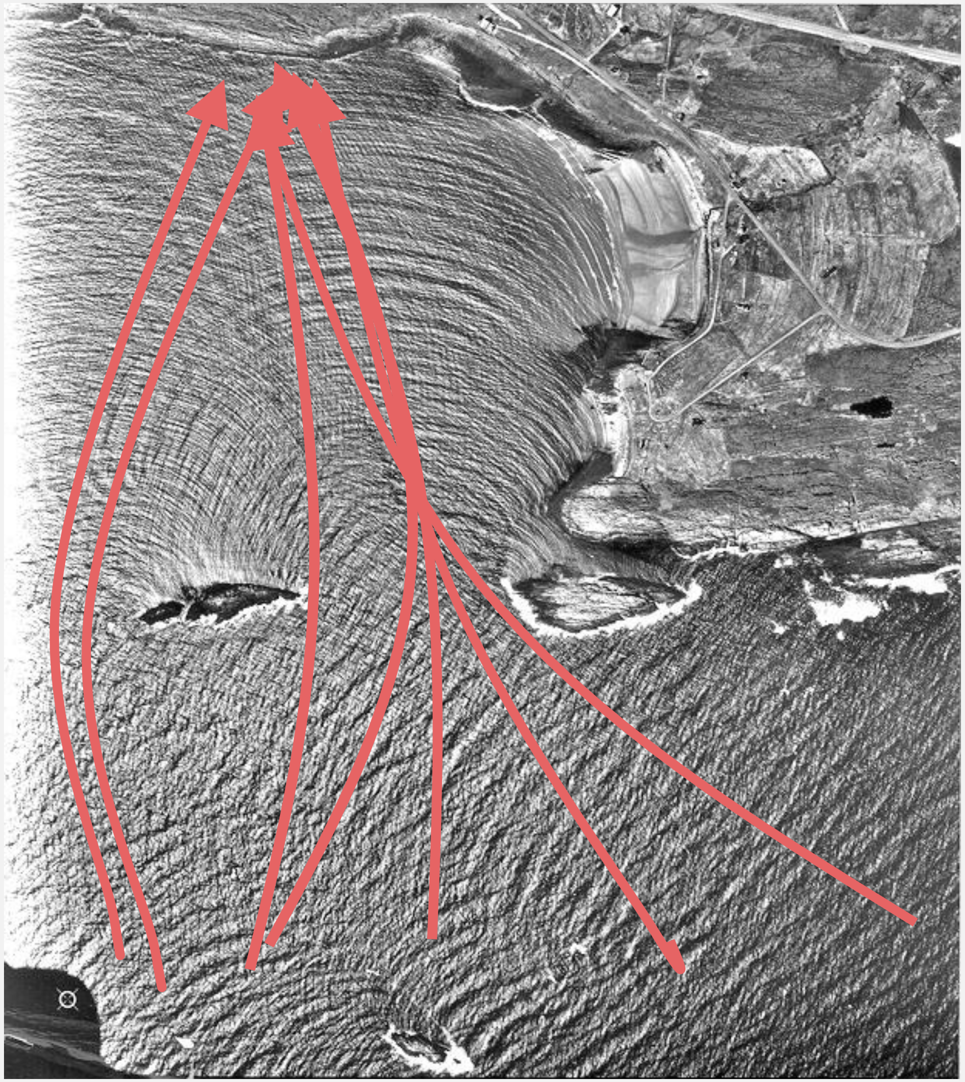
What makes quantum computers powerful?

- Superposition – "can be in all states at once"



Quantum Advantage

- Superposition

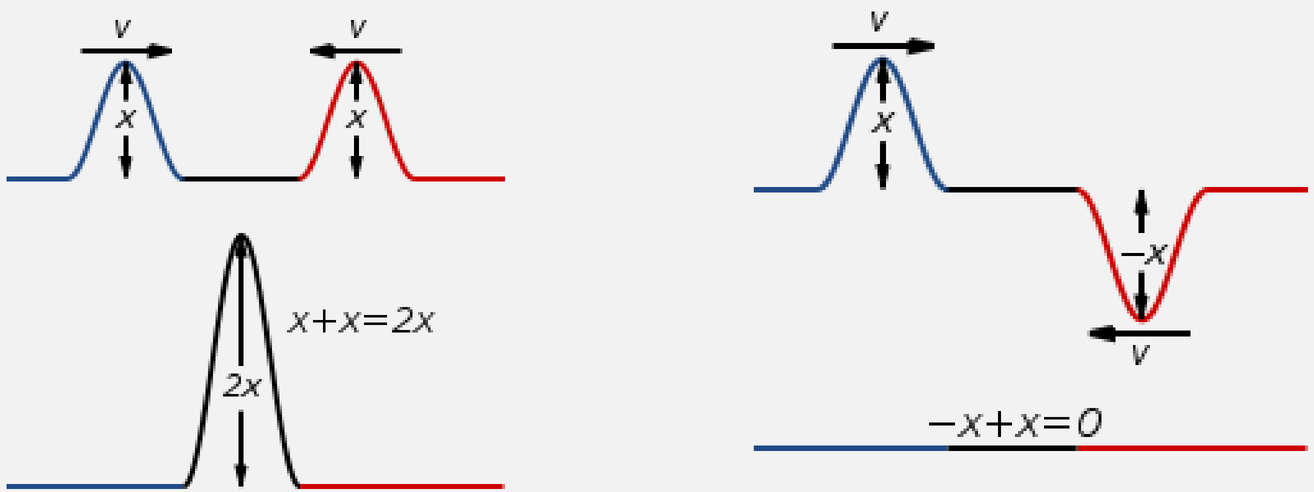


What makes quantum computers powerful?

- Superposition – "can be in all states at once"
- Interference

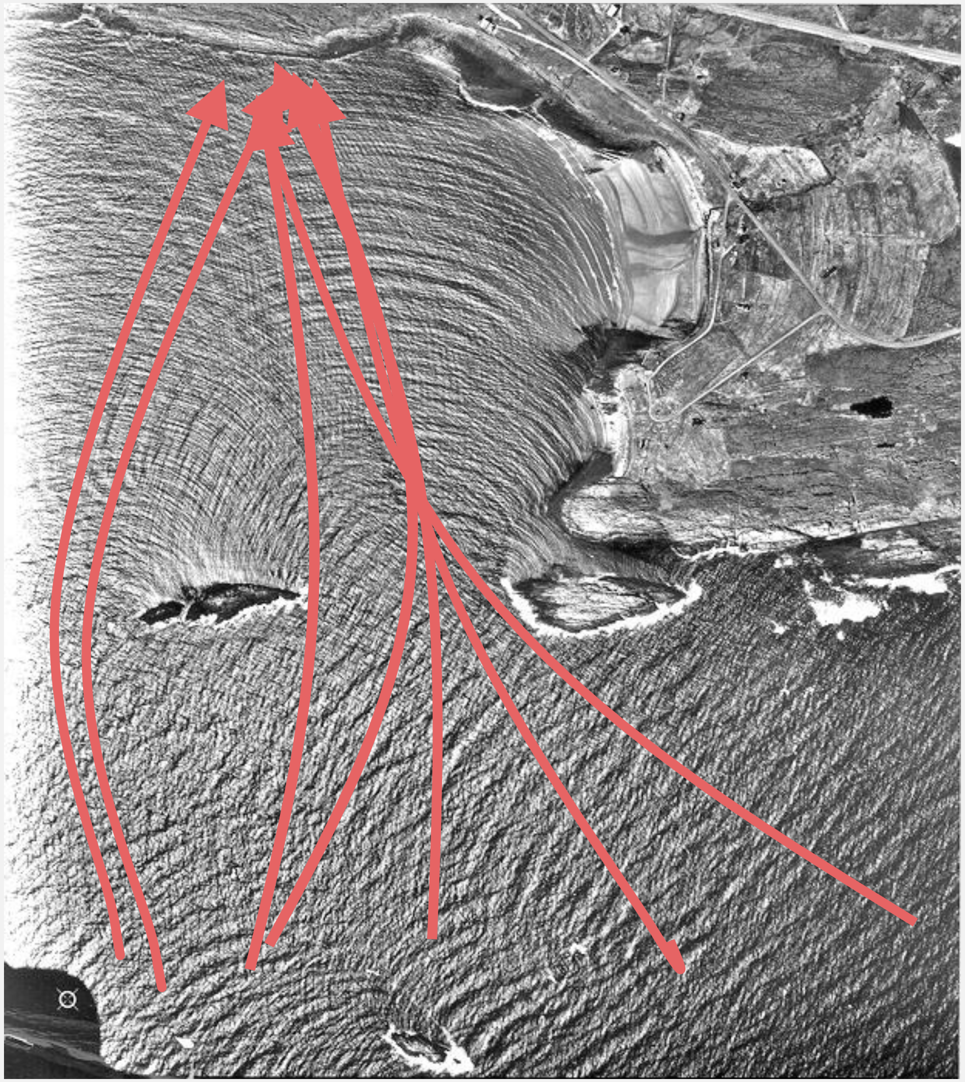
Quantum Advantage

- Interference



Quantum Advantage

- Superposition
  + interference



Outline

1. What problems have fast quantum algorithms?
2. Metaphorical interlude: why do quantum computers have an advantage?
3. When is there a provable quantum advantage?

Proving Quantum Advantage is Difficult!

- Best classical algorithm: exponential in cube root of number of digits d:

$$\sim e^{\sqrt[3]{d}}$$

There could be a better algorithm!

- Best quantum algorithm: cubic in number of digits:

$$\sim d^3$$

New Model – Functions

- Explicit description:

$$f(x) = 2x^2 - 3$$

New Model – Functions

- Explicit description:

$$f(x) = 2x^2 - 3$$

- Black Box description

$$x \rightarrow \boxed{f} \rightarrow f(x)$$

$$0 \rightarrow \boxed{f} \rightarrow -3$$

$$1 \rightarrow \boxed{f} \rightarrow -1$$

$$2 \rightarrow \boxed{f} \rightarrow 5$$

New Model – Functions

- Problem: Given a black box function f, does the function have property P?
- Cost: "Query Complexity" = Number of times you need to use the box
  (Don't count other operations)

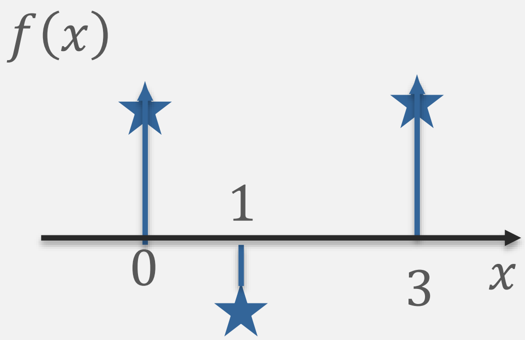
New Model – Functions

- Ex: Given black box access to f, and promised f is quadratic or linear, determine which.

$$f(x) = ax^2 + bx + c \qquad\qquad f(x) = ax + b$$

New Model – Functions

- Ex: Given black box access to f, and promised f is quadratic or linear, determine which.



$$0 \to \boxed{f} \to 1$$

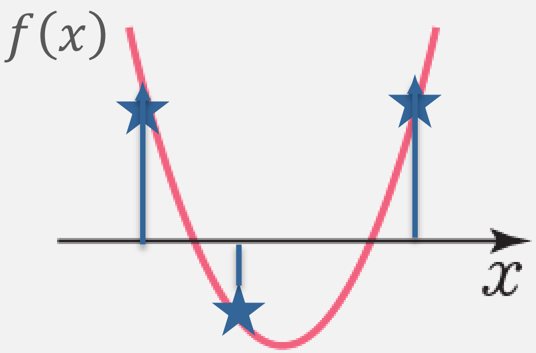
$$1 \to \boxed{f} \to -1$$

$$3 \to \boxed{f} \to 1$$

New Model – Functions

- Ex: Given black box access to f, and promised f is quadratic or linear, determine which.

  Query Complexity = 3



$$0 \rightarrow \boxed{f} \rightarrow 1$$

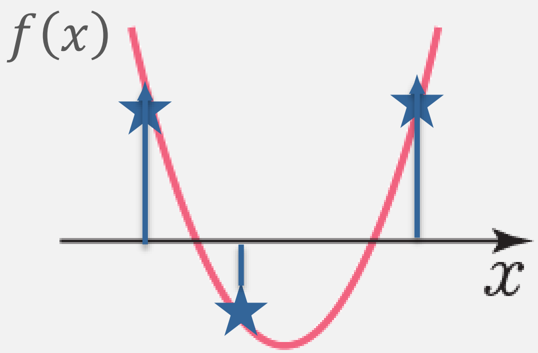
$$1 \rightarrow \boxed{f} \rightarrow -1$$

$$3 \rightarrow \boxed{f} \rightarrow 1$$

New Model – Functions

- Ex: Given black box access to f, and promised f is quadratic or linear, determine which.

  Query Complexity = 3



$0 \rightarrow \boxed{f} \rightarrow 1$

$1 \rightarrow \boxed{f} \rightarrow -1$

$3 \rightarrow \boxed{f} \rightarrow 1$

Only queries are counted!

Quantum Black Box

Input is quantum state that encodes input value

$$|x\rangle \rightarrow \boxed{f} \rightarrow |f(x)\rangle$$

Output is quantum state that encodes output value

Black box is a unitary operation that encodes f

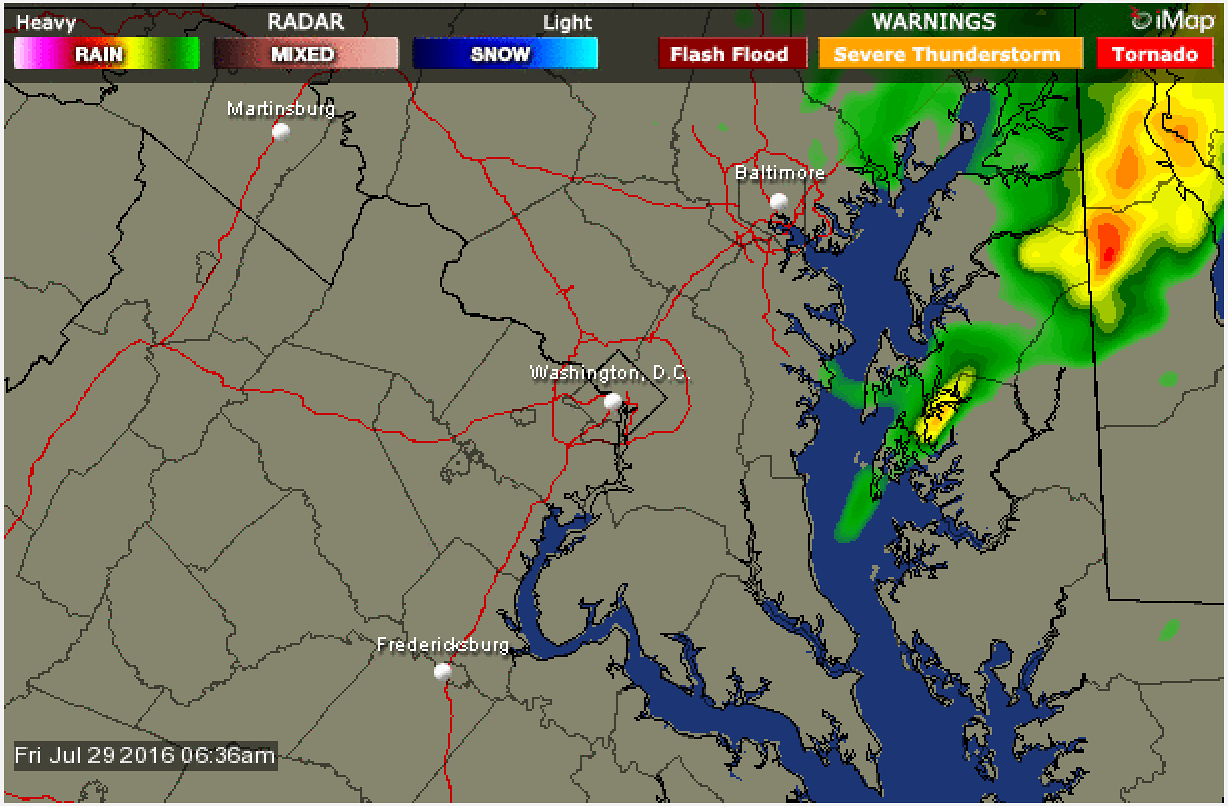
Quantum Black Box

- Problem: Given a quantum black box of f, does the function have property P?

- Cost: "Quantum Query Complexity" = Number of times you need to use the box

  (Free use of quantum computer, unlimited time, size)

Example: Weather Predictions



Washington Post

Query Complexity Examples

Boolean functions: $x = \{1,2,3,\ldots,n\}$, $f(x) = \{0,1\}$

$\boldsymbol{x}$	$\boldsymbol{f(x)}$
1	0
2	1
3	0
4	1
5	1
6	0

Query Complexity Examples

Boolean functions: $x = \{1,2,3, \ldots, n\}$, $f(x) = \{0,1\}$

Property of f
Even Parity Are there an even # of 1-valued outputs?

x	$f(x)$
1	0
2	1
3	0
4	1
5	1
6	0

Query Complexity Examples

Boolean functions: $x = \{1,2,3,\ldots,n\}$, $f(x) = \{0,1\}$

Problem	Quantum Query Complexity	Classical Query Complexity
Even Parity Are there an even # of 1-valued outputs?	$\frac{n}{2}$	n

Beals et al 1998

Query Complexity Examples

Boolean functions: $x = \{1,2,3, \dots, n\}$, $f(x) = \{0,1\}$

Property of f
All Zeros Are all outputs 0-valued? (Search)

x	$f(x)$
1	0
2	1
3	0
4	1
5	1
6	0

Query Complexity Examples

Boolean functions: $x = \{1,2,3, \dots , n\}$, $f(x) = \{0,1\}$

Problem	Quantum Query Complexity	Classical Query Complexity
All Zeros Are all outputs 0-valued? (Search)	$\sim\sqrt{n}$	$\sim n$

Grover 1997

Query Complexity Examples

More general functions with promises

Property
Period finding Promised f is periodic, find the period

x	$f(x)$
1	0
2	4
3	3
4	0
5	4
6	3

Query Complexity Examples

More general functions with promises

Problem	Quantum Query Complexity	Classical Query Complexity
Period finding Promised f is periodic, find the period	1	$\sim\sqrt[4]{n}$

Chakraborty et al 2010

Query Complexity Examples

More general functions with promises

Property
Hidden shift Promised $f(x) = g(x + s)$ for known function g. Find s.

x	$f(x)$
1	3
2	1
3	1
4	6
5	5
6	2

x	$g(x)$
1	1
2	6
3	5
4	2
5	3
6	1

Query Complexity Examples

More general functions with promises

Problem	Quantum Query Complexity	Classical Query Complexity
Hidden shift Promised $f(x) = g(x+s)$ for known function g. Find s.	$\sim \log n$	$\sim \sqrt{n}$

Gavinsky et al 2011

Quantum Advantage

Small Quantum Speed-up	Large Quantum Speed-up
No promise on function	Promise on function (e.g. periodic, shifted function)

x	$f(x)$
1	0
2	1
3	0
4	1
5	1

Quantum Advantage

Small Quantum Speed-up	Large Quantum Speed-up
No promise on function	Promise on function (e.g. periodic, shifted function)

x	$f(x)$
1	0
2	4
3	3
4	0
5	4

Quantum Advantage

Small Quantum Speed-up	Large Quantum Speed-up
No promise on function	Promise on function (e.g. periodic, shifted function)
Outcome depends on local property (changing one output changes the property)	Outcome depends on global property. (if change one output, still close to desired property)

x	$f(x)$
1	0
2	0
3	0
4	0
5	0

Quantum Advantage

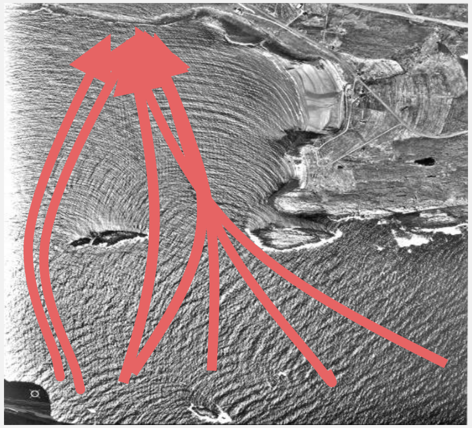
Small Quantum Speed-up	Large Quantum Speed-up
No promise on function	Promise on function (e.g. periodic, shifted function)
Outcome depends on local property (changing one output changes the property)	Outcome depends on global property. (if change one output, still close to desired property)

x	$f(x)$
1	0
2	0
3	0
4	0
5	0

x	$f(x)$
1	0
2	4
3	3
4	0
5	4

Quantum Advantage

Small Quantum Speed-up	Large Quantum Speed-up
No promise on function	Promise on function (e.g. periodic, shifted function)
Outcome depends on local property (changing one output changes the property)	Outcome depends on global property. (if change one output, still close to desired property)



More on quantum algorithms

- http://www.scottaaronson.com/blog/?p=208 Shtetl-Optimized "Shor I'll Do It"