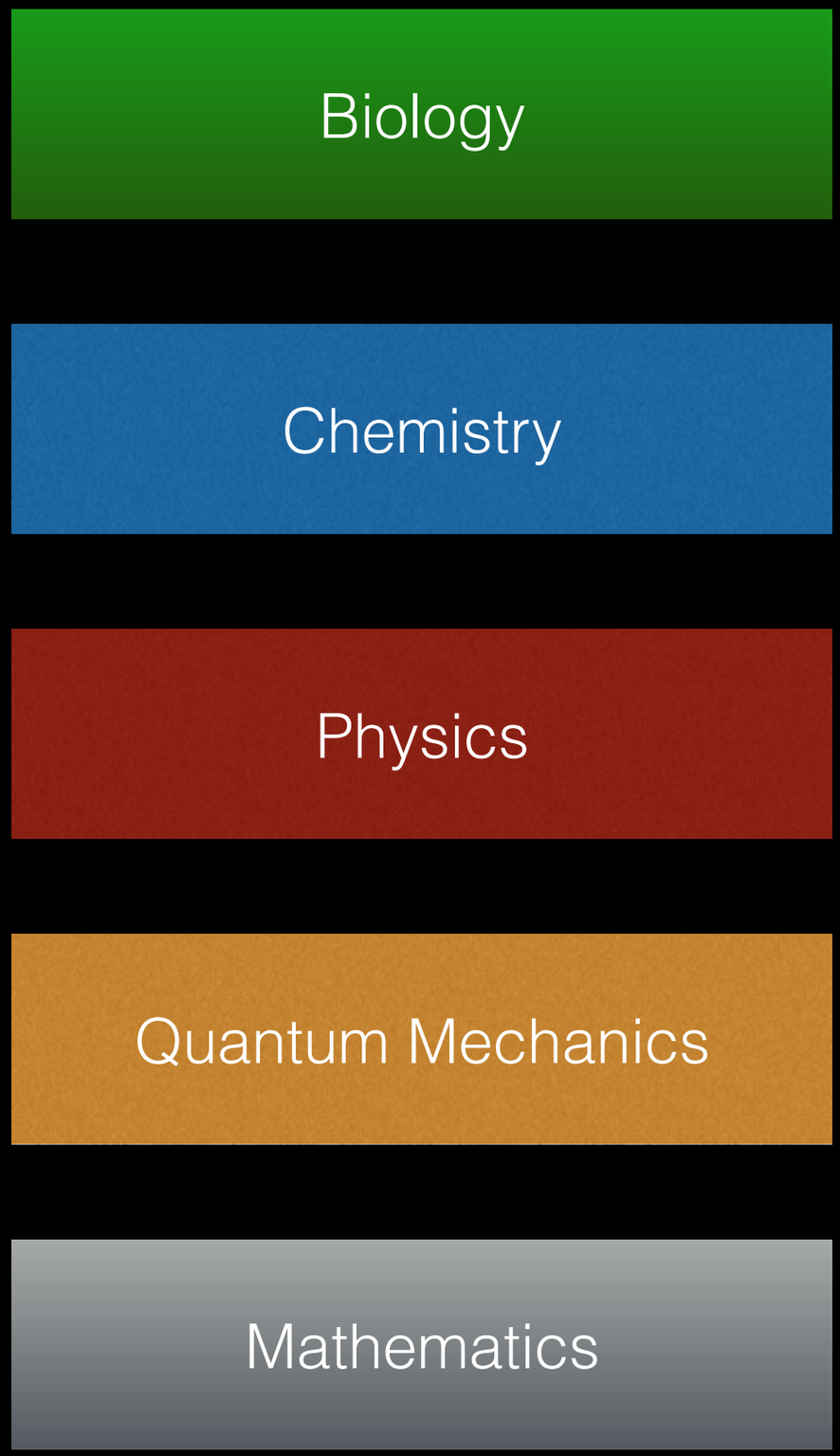
Quantum Computing

CSC 282 Fall 2013

Quantum mechanics is the operating system that other physical theories run on as applications.

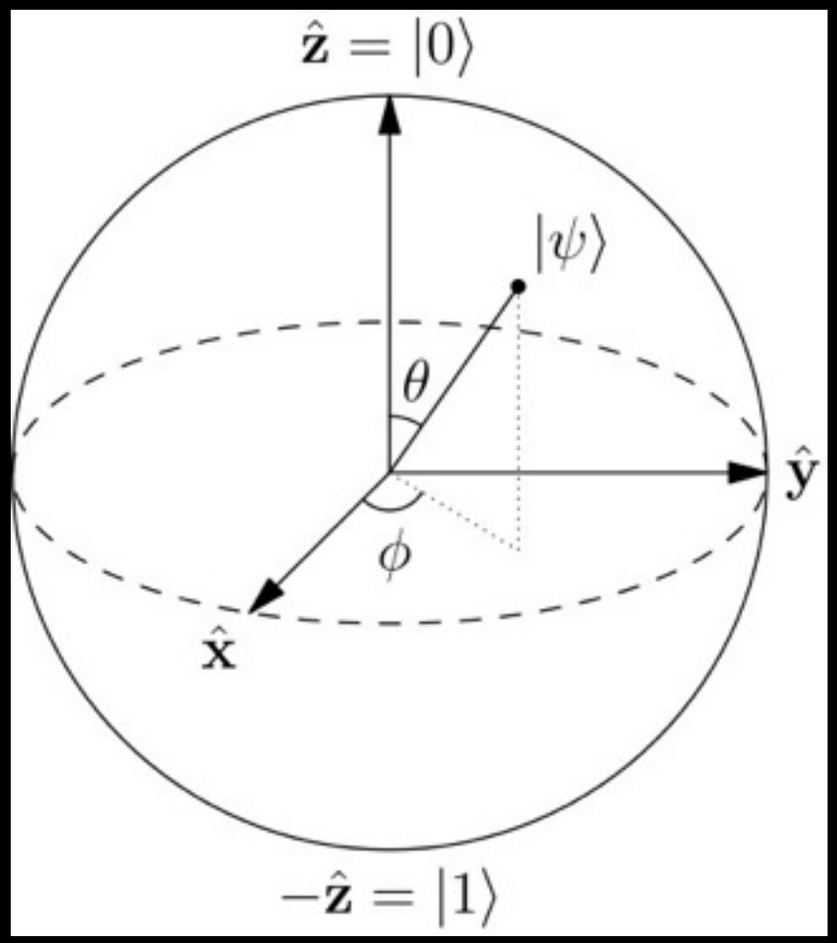
Scott Aaronson



Inevitability of Quantum Mechanics

- Quantum mechanics is what you would inevitably come up with if you started with probability theory, and then said, let's try to generalize it so that the numbers we used to call "probabilities" can be negative [complex] numbers

- Scott Aaronson



From Probability to Amplitudes and Back

- Classical bit $|0\rangle = \text{bit is } 0,\ |1\rangle = \text{bit is } 1$

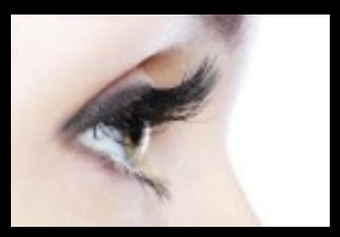
$$0 \le P(|0\rangle), P(|1\rangle) \le 1$$

$$P(|0\rangle) + P(|1\rangle) = 1$$

- Qubit: $|\psi\rangle = \alpha_0|0\rangle + \alpha_1|1\rangle$

$$0 \le |\alpha_0|, |\alpha_1| \le 1$$

$$|\alpha_0|^2 + |\alpha_1|^2 = 1$$



$$P(|0\rangle) = |\alpha_0|^2 \qquad P(|1\rangle) = |\alpha_1|^2$$

Multiple Qubits

- Quantum state of 2 qubits is a linear combination of 4 classical states

$$|\boldsymbol{\alpha}\rangle = \alpha_{00}|00\rangle + \alpha_{01}|01\rangle + \alpha_{10}|10\rangle + \alpha_{11}|11\rangle$$

- For n qubits, quantum state is a superposition of 2^n classical states

$$\sum_{x \in \{0,1\}^n} \alpha_x |x\rangle$$

Partial Measurements

- If only some of a entangled set of qubits are measured, the new superposition is the renormalized sum of the terms that are consistent with the measurement.

- Example: suppose first bit is observed to be 0. Then:

$$|\boldsymbol{\alpha}_{\mathbf{new}}\rangle = \frac{\alpha_{00}}{\sqrt{|\alpha_{00}|^2 + |\alpha_{01}|^2}}|00\rangle + \frac{\alpha_{01}}{\sqrt{|\alpha_{00}|^2 + |\alpha_{01}|^2}}|01\rangle$$

Compact Representation

- The quantum state of n qubits is characterized by 2^n amplitudes

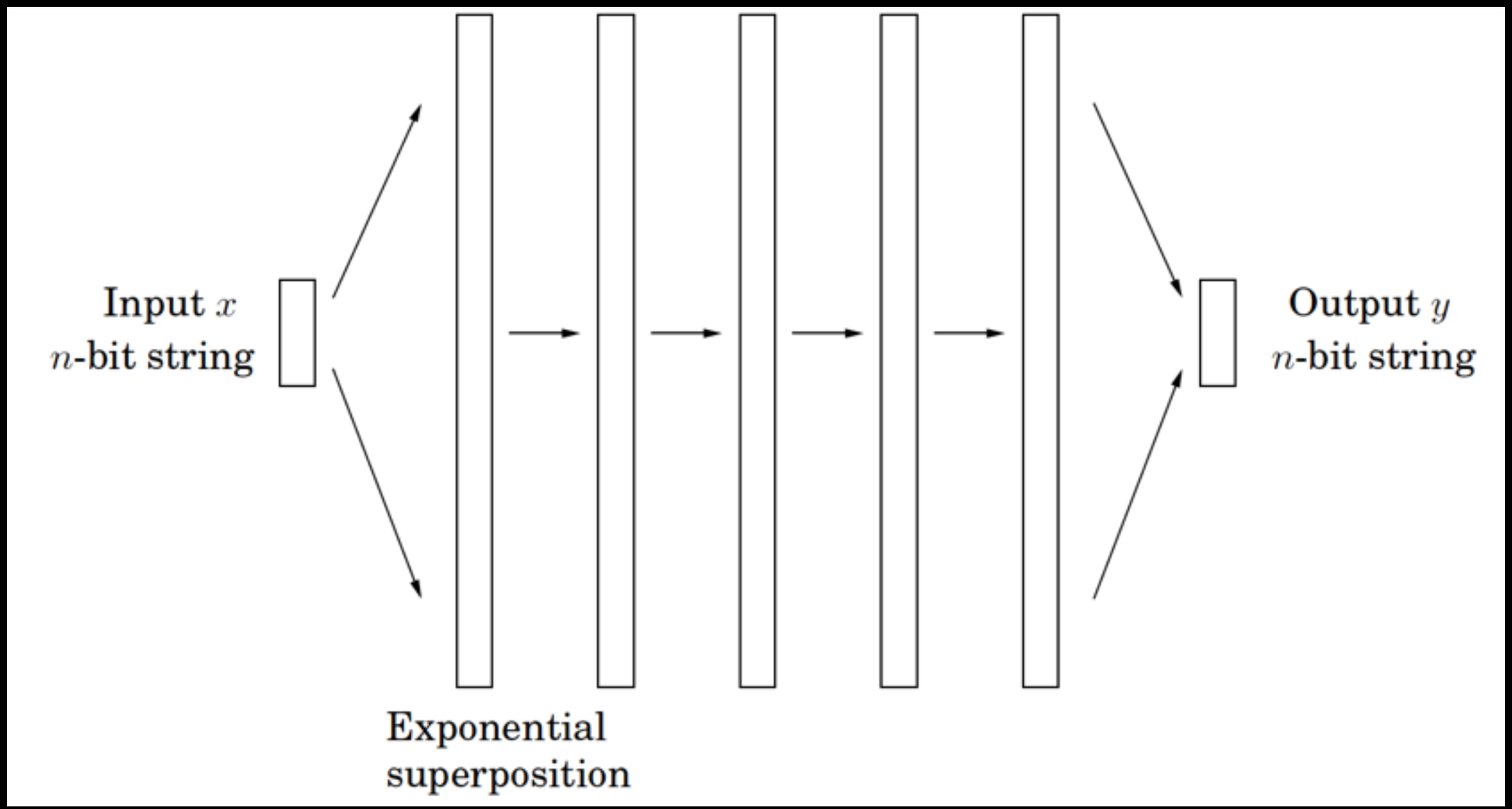
- Thus: a list of 2^n numbers (amplitudes) can be represented using only n quits

$$\sum_{x \in \{0,1\}^n} \alpha_x |x\rangle$$

- Equivalently: we can represent a list of k numbers using only log(k) qubits

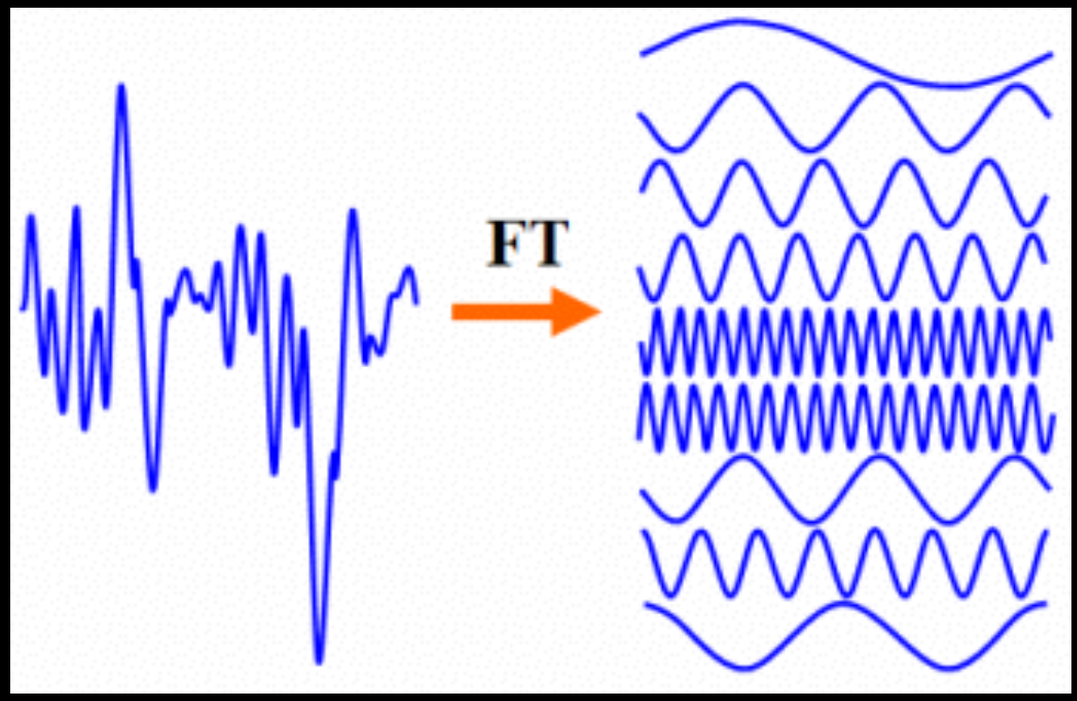
- It is easy to convert a list of k numbers *into* the superposition of log(k) quits - but the other direction is tricky!

General Form of a Quantum Algorithm



Discrete Fourier Transform

- Basis for signal processing: transform time domain to frequency domain and vice-versa
- Separates out the frequencies that contribute to a complicated signal
- Input: M-dimension vector α
  - samples over time
- Output: M-dimensional vector β
  - Amplitudes of frequencies



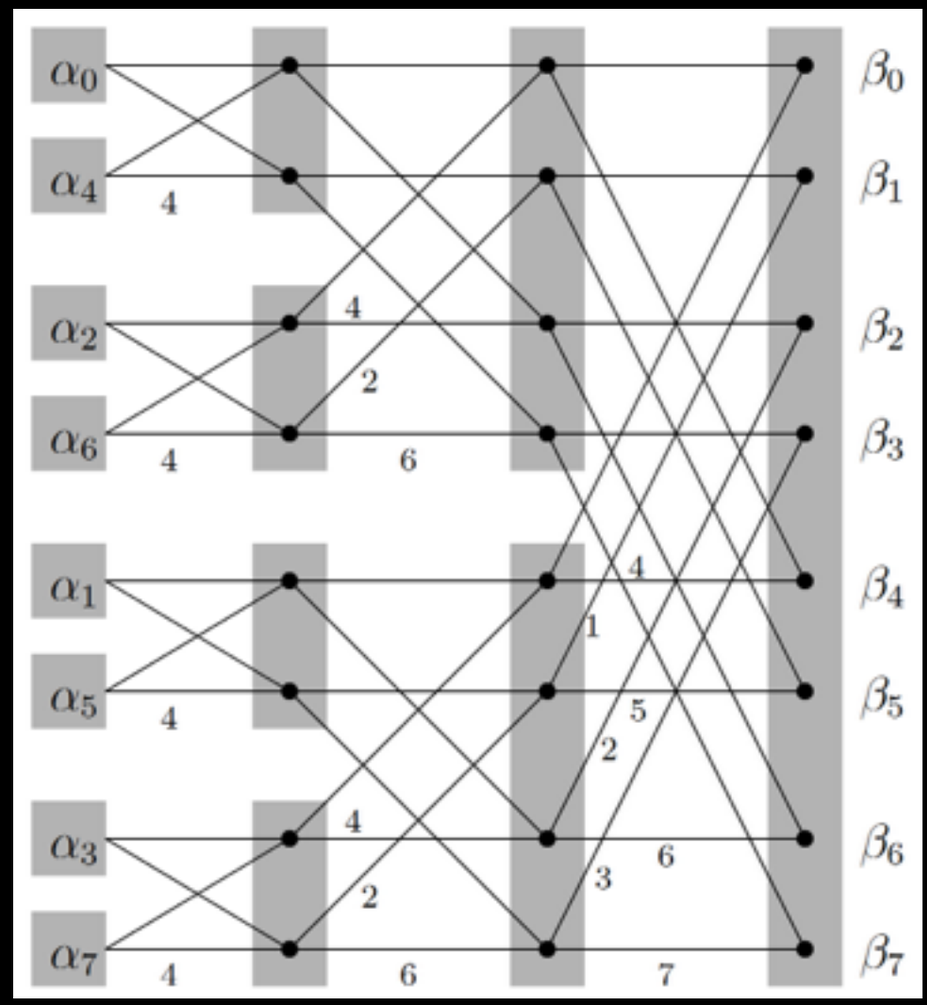
Computing Fourier Transform

$$\begin{bmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \\ \vdots \\ \beta_{M-1} \end{bmatrix} = \frac{1}{\sqrt{M}} \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \omega & \omega^2 & \cdots & \omega^{M-1} \\ 1 & \omega^2 & \omega^4 & \cdots & \omega^{2(M-1)} \\ & \vdots & & & \\ 1 & \omega^j & \omega^{2j} & \cdots & \omega^{(M-1)j} \\ & \vdots & & & \\ 1 & \omega^{(M-1)} & \omega^{2(M-1)} & \cdots & \omega^{(M-1)(M-1)} \end{bmatrix} \begin{bmatrix} \alpha_0 \\ \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_{M-1} \end{bmatrix}$$

- Using matrix multiplication: $O(M^2)$ $\omega = e^{2\pi i/M}$
- Fast Fourier Transform: O(M log M)
- Quantum Fourier Transform: $O(\log^2 M)$

The Exponential Superposition Trick

- The classic FFT represents the input α as a vector of length M, and involves log(M) stages
  - O(M log M)
- The quantum algorithm represents α as the superposition of log(M) bits, and involves log(M) stages
  - $O(\log^2 M)$



Getting the Answer Out

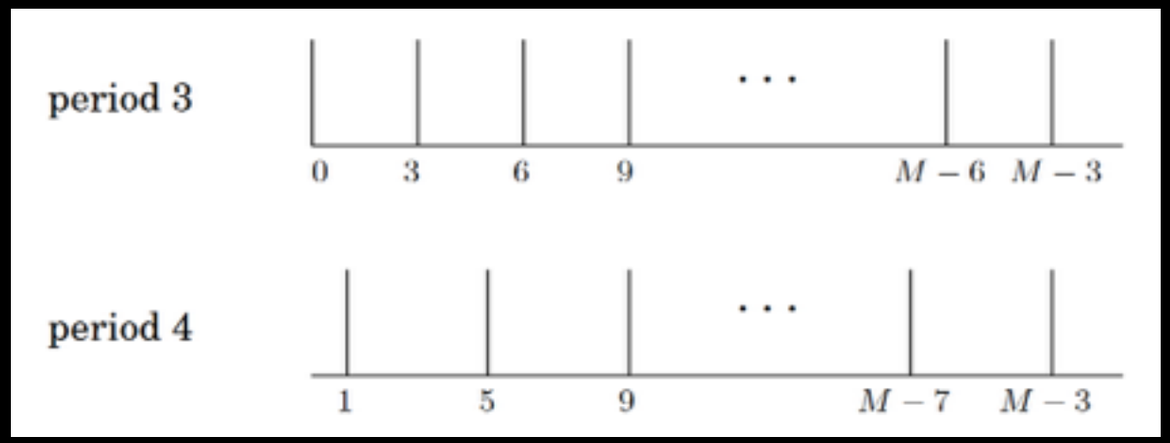
- Classic algorithm outputs the M-length vector β
- Quantum FT results in a superposition

$$|\beta\rangle = \sum_{j=0}^{M-1} \beta_j |j\rangle$$

- Reading it collapses the superposition, yielding just one of its classical states
- Outputs index *j* with probability $|\beta_j|^2$
  - *E.g.: most likely to return the most important frequency that make up the original signal*

Periodicity

- A special case where we can read out the complete answer from the quantum FT: when the input is periodic
- Exactly one non-zero value repeated every k positions in the input vector



- Many quantum algorithms (e.g. factoring) make use of this property

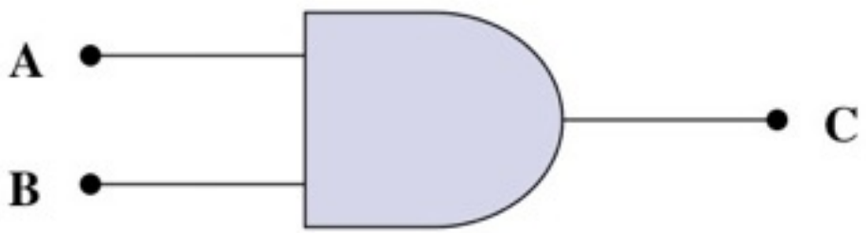
Quantum Circuits

Operations on Qubits - Reversible Logic

- Due to the nature of quantum physics, the destruction of information in a gate will cause heat to be evolved which can destroy the superposition of qubits.

Ex.

The AND Gate



Input		Output
A	B	C
0	0	0
0	1	0
1	0	0
1	1	1

In these 3 cases, information is being destroyed

- This type of gate cannot be used. We must use ***Quantum Gates***.

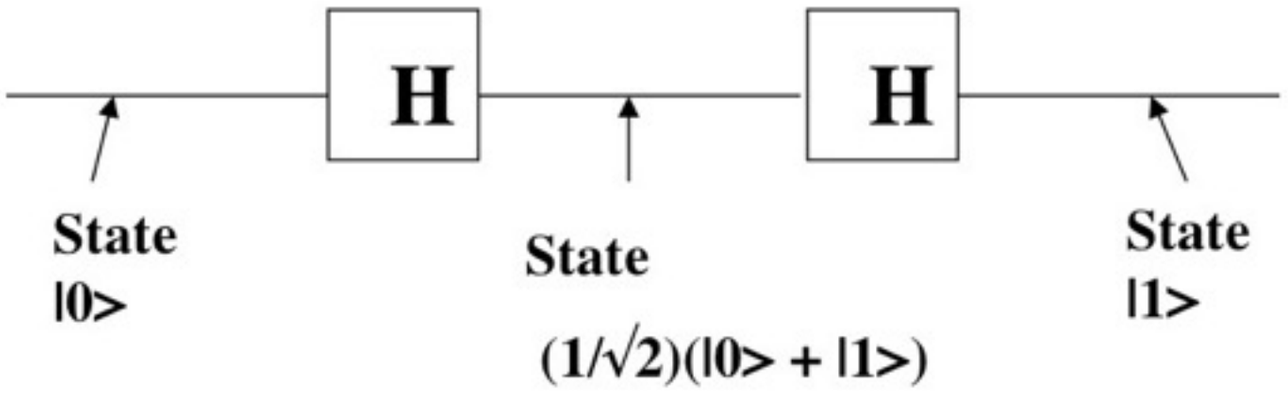
Quantum Gates

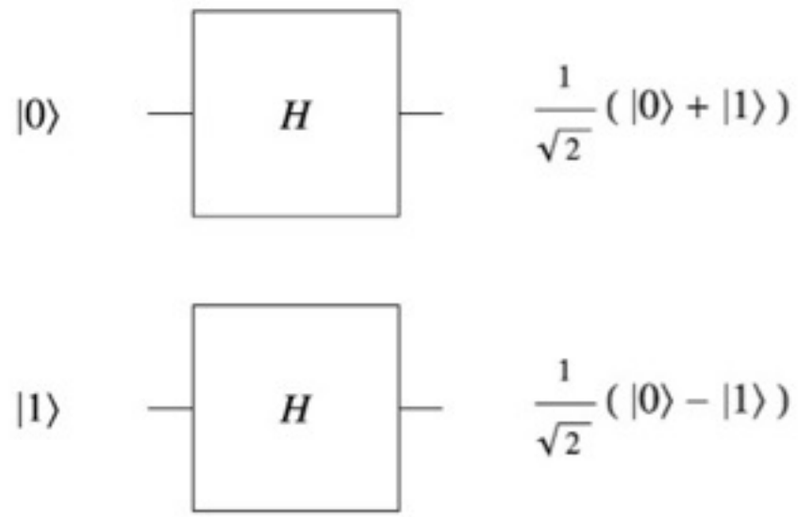
- Quantum Gates are similar to classical gates, but do not have a degenerate output. i.e. their original input state can be derived from their output state, uniquely. ***They must be reversible.***

- This means that a deterministic computation can be performed on a quantum computer only if it is reversible. Luckily, it has been shown that any deterministic computation can be made reversible.(Charles Bennet, 1973)

Quantum Gates - Hadamard

- Simplest gate involves one qubit and is called a ***Hadamard Gate*** (also known as a square-root of NOT gate.) Used to put qubits into superposition.

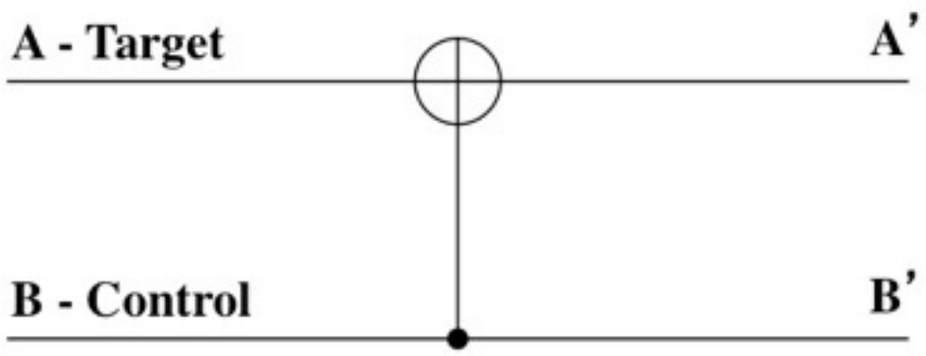




Note: Two Hadamard gates used in succession can be used as a NOT gate

Quantum Gates - Controlled NOT

- A gate which operates on two qubits is called a ***Controlled-NOT (CN) Gate.*** If the bit on the control line is 1, invert the bit on the target line.



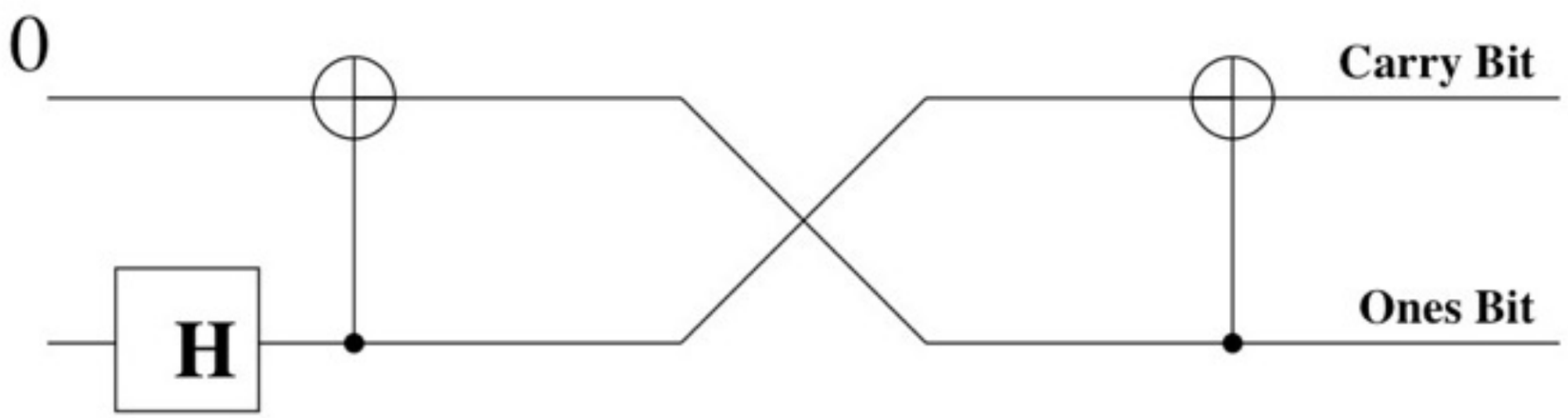
Input		Output	
A	B	A'	B'
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	1

Note: The CN gate has a similar behavior to the XOR gate with some extra information to make it reversible.

Example Operation - Multiplication By 2

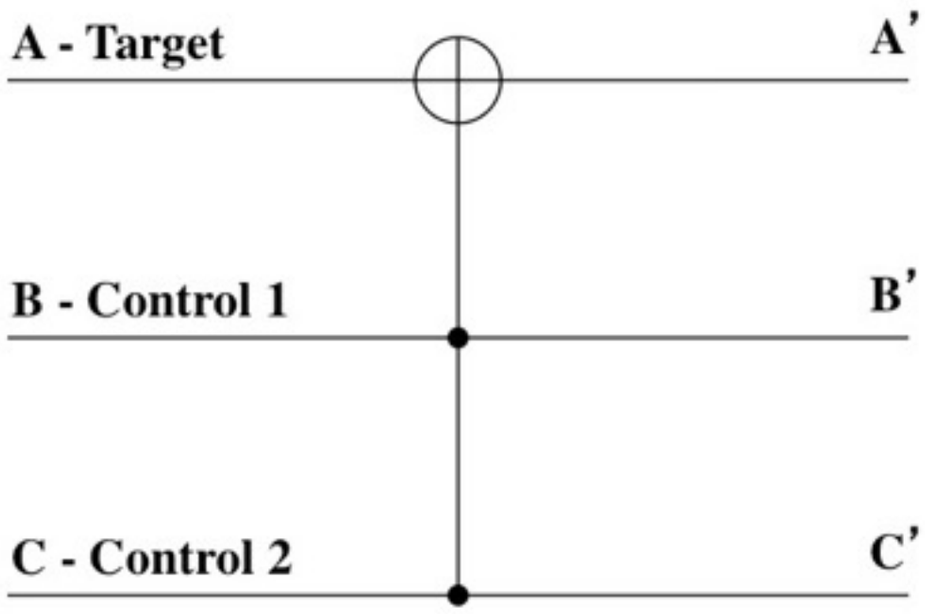
- We can build a reversible logic circuit to calculate multiplication by 2 using CN gates arranged in the following manner:

Input		Output	
Carry Bit	Ones Bit	Carry Bit	Ones Bit
0	0	0	0
0	1	1	0



Quantum Gates - Controlled Controlled NOT (CCN)

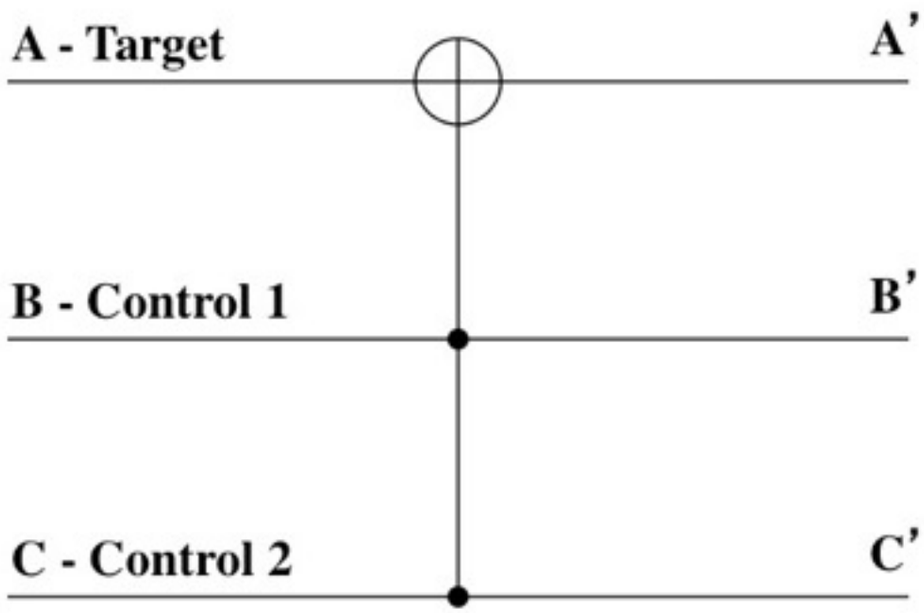
- A gate which operates on three qubits is called a ***Controlled Controlled NOT (CCN) Gate.*** Iff the bits on both of the control lines is 1,then the target bit is inverted.



Input			Output		
A	B	C	A'	B'	C'
0	0	0	0	0	0
0	0	1	0	0	1
0	1	0	0	1	0
0	1	1	1	1	1
1	0	0	1	0	0
1	0	1	1	0	1
1	1	0	1	1	0
1	1	1	0	1	1

A Universal Quantum Computer

- The CCN gate has been shown to be a ***universal*** reversible logic gate as it can be used as a NAND gate.



Input			Output		
A	B	C	A'	B'	C'
0	0	0	0	0	0
0	0	1	0	0	1
0	1	0	0	1	0
0	1	1	1	1	1
1	0	0	1	0	0
1	0	1	1	0	1
1	1	0	1	1	0
1	1	1	0	1	1

When our target input is 1, our target output is a result of a NAND of B and C.

Quantum Factoring

Shor's Algorithm

- Shor's algorithm shows (in principle,) that a quantum computer is capable of factoring very large numbers in polynomial time.

The algorithm is dependant on

- Modular Arithmetic
- Quantum Parallelism
- Quantum Fourier Transform

The Plan

- FACTORING is reduced to finding a *nontrivial square root* of 1 modulo N.
- Finding such a root is reduced to computing the *order* of a random integer modulo N.
- The order of an integer is precisely the *period* of a particular *periodic superposition*.
- Finally, periods of superpositions can be found by the *quantum FFT*.

Shor's Algorithm - Periodicity

- An important result from Number Theory:

$$\mathbf{F(a) = x^a \bmod N}$$ is a periodic function

- Choose N = 15 and x = 7 and we get the following:

$$\mathbf{7^0 \bmod 15 = 1}$$

$$\mathbf{7^1 \bmod 15 = 7}$$

$$\mathbf{7^2 \bmod 15 = 4}$$

$$\mathbf{7^3 \bmod 15 = 13}$$

$$\mathbf{7^4 \bmod 15 = 1}$$

$$\vdots$$

Shor's Algorithm - In Depth Analysis

To Factor an odd integer N (Let's choose 15) :

1. Choose an integer q such that $N^2 < q < 2N^2$ **let's pick 256**
2. Choose a random integer x such that GCD(x, N) = 1 **let's pick 7**
3. Create two quantum registers (these registers must also be entangled so that the collapse of the input register corresponds to the collapse of the output register)
   - Input register: must contain enough qubits to represent numbers as large as q-1. **up to 255, so we need 8 qubits**
   - Output register: must contain enough qubits to represent numbers as large as N-1. **up to 14, so we need 4 qubits**

Shor's Algorithm - Preparing Data

4. Load the input register with an equally weighted superposition of all integers from 0 to q-1. **0 to 255**
5. Load the output register with all zeros.

The total state of the system at this point will be:

$$\underbrace{\frac{1}{\sqrt{256}} \sum_{a=0}^{255} |a\rangle}, |000\rangle$$

Input Register

Output Register

Note: the comma here denotes that the registers are entangled

Shor's Algorithm - Modular Arithmetic

6. Apply the transformation x^a mod N to each number in the input register, storing the result of each computation

Note that we are using decimal numbers here only for simplicity.

The output function holds a repeating pattern, but it does not quite match our definition of a period function (all values are 0 except every P-th element).

So, we'll do a little more work before using the Quantum FFT to calculate the period.

		Output Register
	lod 15	1
	lod 15	7
	lod 15	4
		13
	lod 15	1
|5>	7^5 Mod 15	7
|6>	7^6 Mod 15	4
|7>	7^7 Mod 15	13

•
•

Shor's Algorithm - Superposition Collapse

7. Now take a measurement on the output register. This will collapse the superposition to represent ***just one*** of the results of the transformation, let's call this value c.

Our output register will collapse to represent one of the following:

|1>, |4>, |7>, or |13>

For sake of example, lets choose |1>

Shor's Algorithm - Entanglement

Now things really get interesting !

8. Since the two registers are entangled, measuring the output register will have the effect of partially collapsing the input register into an **equal superposition** of each state between 0 and q-1 that yielded c (the value of the collapsed output register.)

Since the output register collapsed to |1>, the input register will partially collapse to:

$$\frac{1}{\sqrt{64}}\ |0\rangle + \frac{1}{\sqrt{64}}\ |4\rangle + \frac{1}{\sqrt{64}}\ |8\rangle + \frac{1}{\sqrt{64}}\ |12\rangle, \ldots$$

The probabilities in this case are $\frac{1}{\sqrt{64}}$ since our register is now in an equal superposition of 64 values (0, 4, 8, . . . 252)

Shor's Algorithm - QFT

The QFT will essentially peak the probability amplitudes at integer multiples of $q/4$ in our case 256/4, or 64.

|0>, |64>, |128>, |192>, …

So we no longer have an equal superposition of states, the probability amplitudes of the above states are now higher than the other states in our register. We measure the register, and it will collapse with high probability to one of these multiples of 64, let's call this value p.

With our knowledge of q, and p, there are methods of calculating the period (one method is the continuous fraction expansion of the ratio between q and p.)

Shor's Algorithm - The Factors :)

10. Now that we have the period, the factors of N can be determined by taking the greatest common divisor of N with respect to $x \wedge (P/2) + 1$ and $x \wedge (P/2) - 1$. The idea here is that this computation will be done on a classical computer.

We compute:

$\text{Gcd}(7^{4/2} + 1, 15) = \mathbf{5}$

$\text{Gcd}(7^{4/2} - 1, 15) = \mathbf{3}$

We have successfully factored 15!

Next

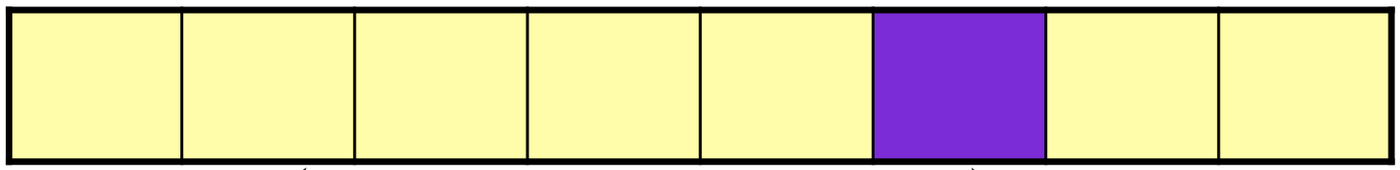
Quantum Search

Quantum Simulated Annealing

Limitations of Quantum Computing

Real Quantum Computers

Grover's Algorithm

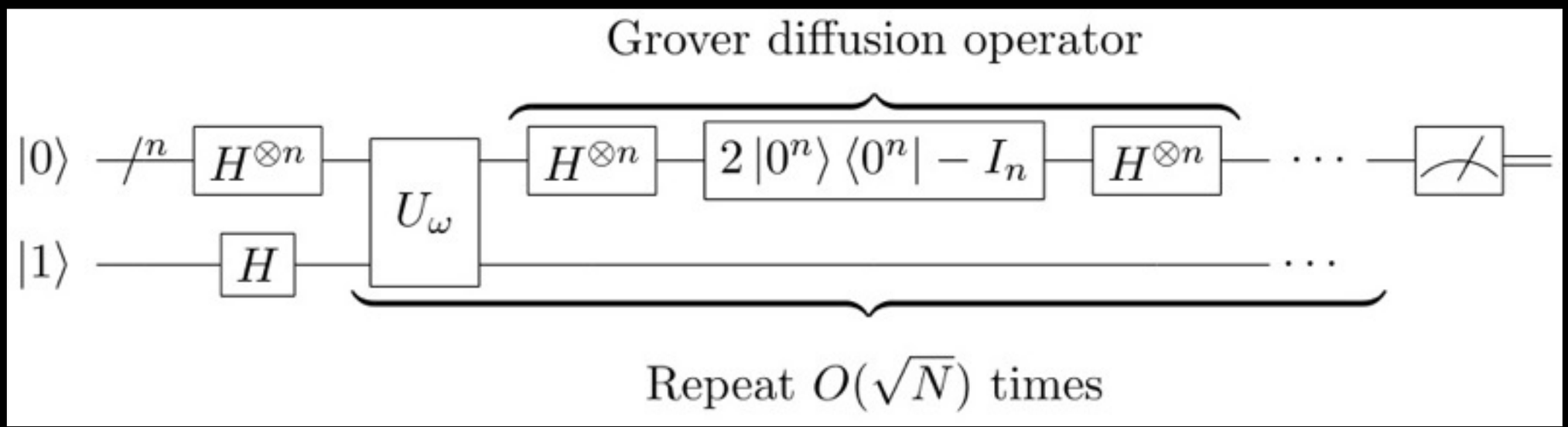


Unsorted database of n items

Goal: Find one "marked" item

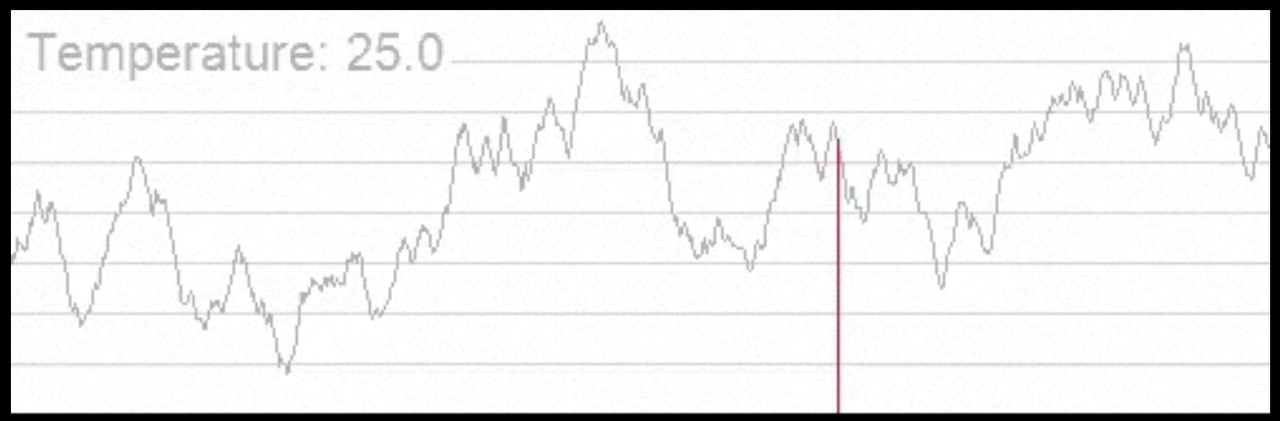
- Classically, order n queries to database needed
- **Grover 1996:** Quantum algorithm using order √n queries

- Given unsorted database of N entries,determine the index of the database entry that satisfies some search criterion
- Say $f(\omega) \in \{0,1\}$ defines criterion
  - Turn f into a quantum subroutine U_ω where
    - $U_\omega |\omega\rangle = -|\omega\rangle$
    - $U_\omega |x\rangle = |x\rangle$ for $x \neq \omega$



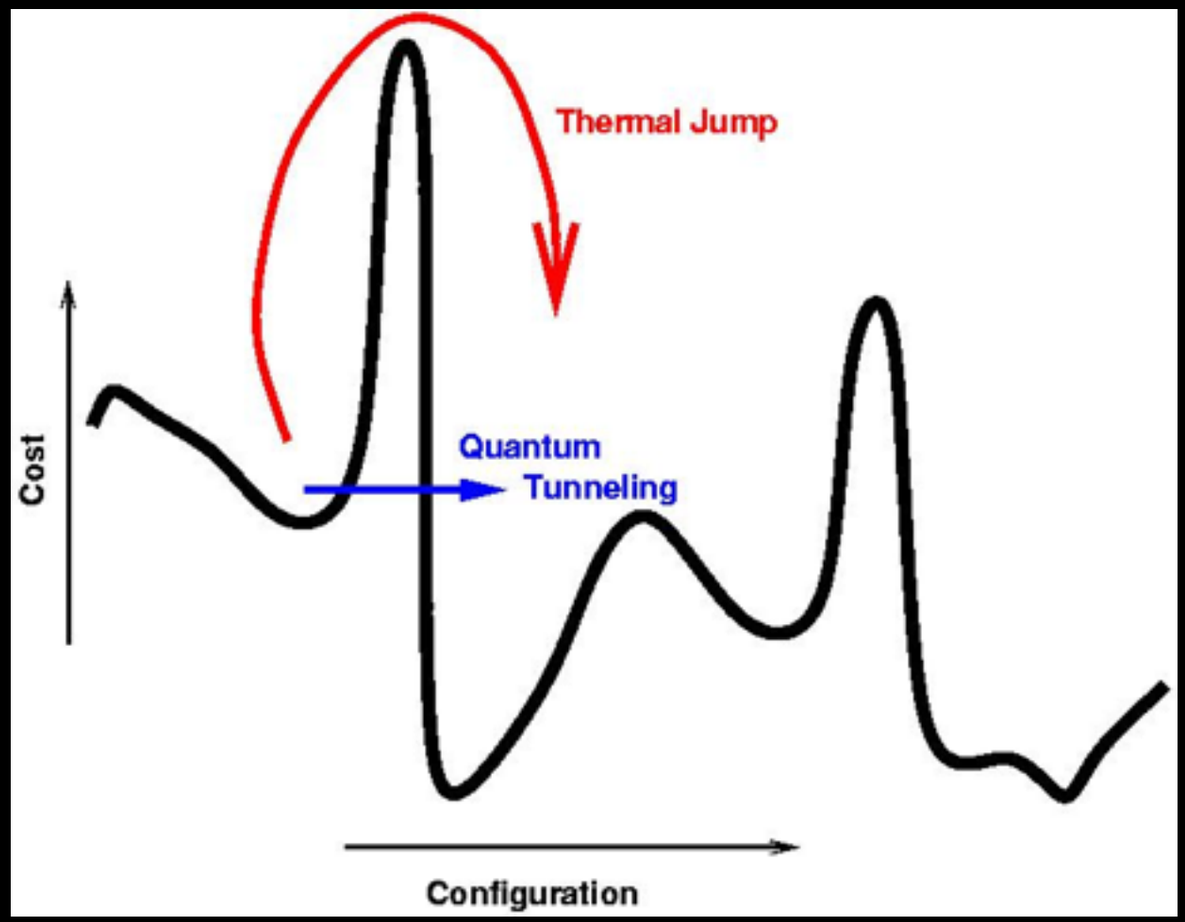
Simulated Annealing

- Goal: find a point that maximizes (or minimizes) an arbitrary function
- Start at a random point
- Make small changes in the point to increase the function
- Randomly make “wrong way” moves according to temperature - more likely when “hotter”
- If cooling is slow enough, guaranteed convergence to optimal point



Quantum Annealing

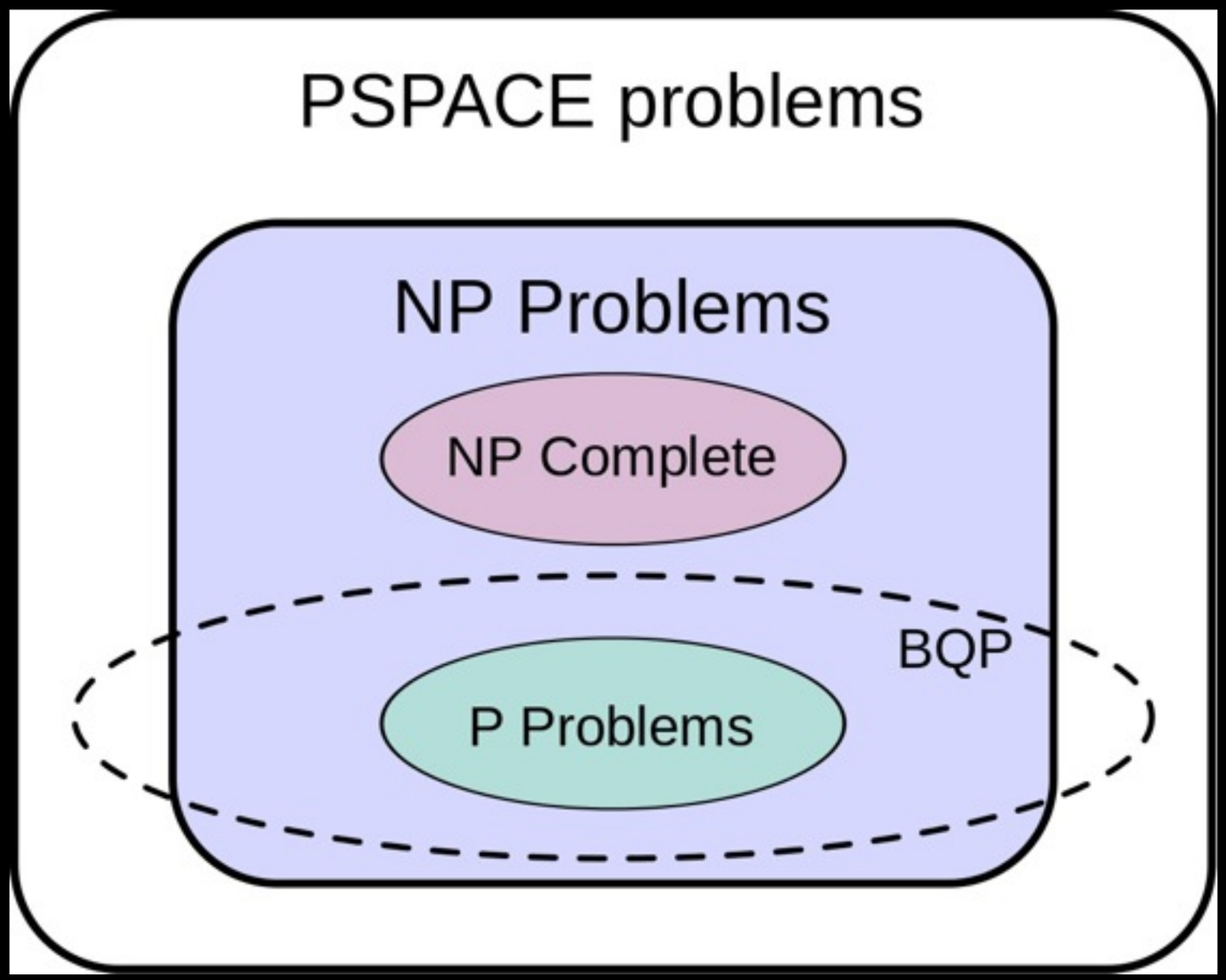
- Can “tunnel” to distance points in the state space
- Tunneling field strength determines size of neighborhood
- Finding the best point to tunnel to when neighborhood is size N requires
  - O(N) for classic computer
  - O(√N) for quantum computer



Quantum vs Classical Annealing

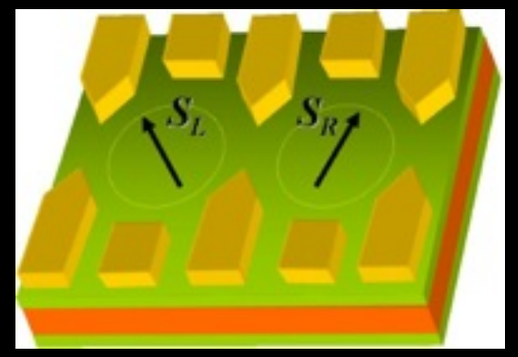
- Quantum annealing outperforms classical annealing when energy landscape has high, thin barriers surrounding shallow local minima
  - Classical annealing can’t climb out
- Quantum implementation beats classical simulation of quantum annealing when neighborhood size N is “too big”, but $\sqrt{N}$ is “not too big”
  - E.g. N = 1,000,000

Limitations of Quantum Computing



Implementing Quantum Computers

- Quantum dots
  - "Caged electrons"

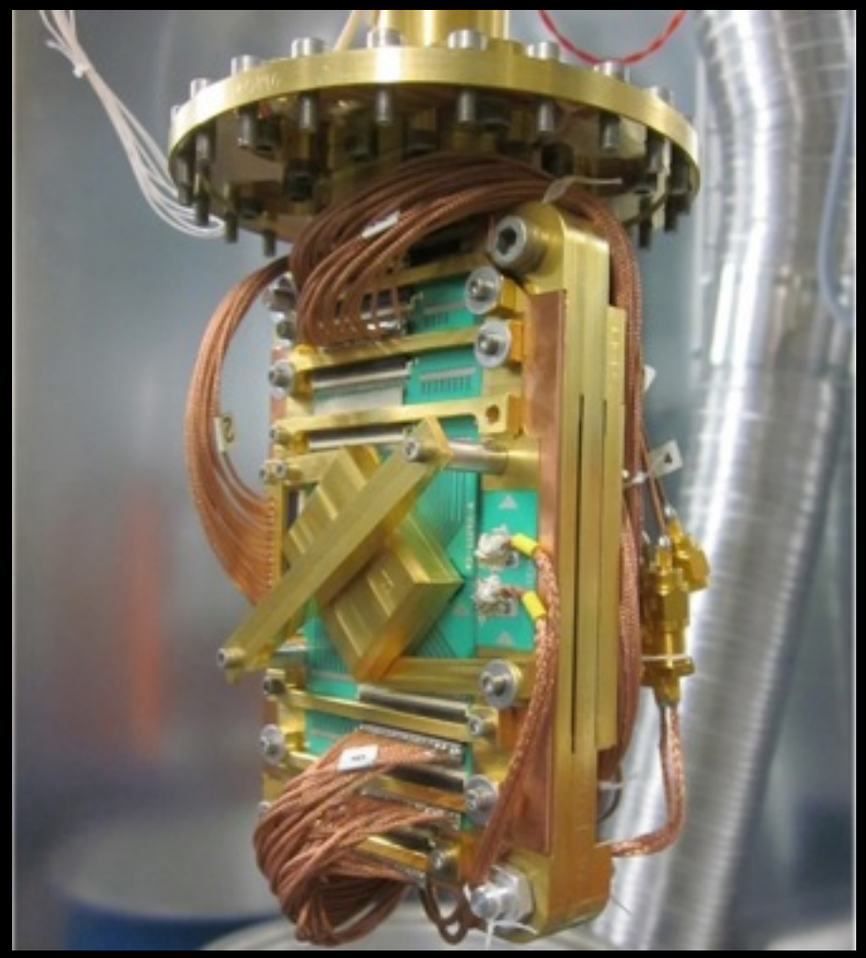


  - Have demonstrated certain 2-bit gates
- Nuclear magnetic resonance
  - Spin states of molecules as qubits

  - IBM 2001: 7-bit implementation of Shor's algorithms for factoring

D-Wave

- 512 qubits
- Performs quantum annealing
- Sold to Google (for NASA) and USC-Lockheed Martin
- Controversy: is it doing *simulated* quantum annealing or *quantum* quantum annealing?
- Has not yet outperformed ordinary computers on any real-world problem



Simulating Quantum Systems

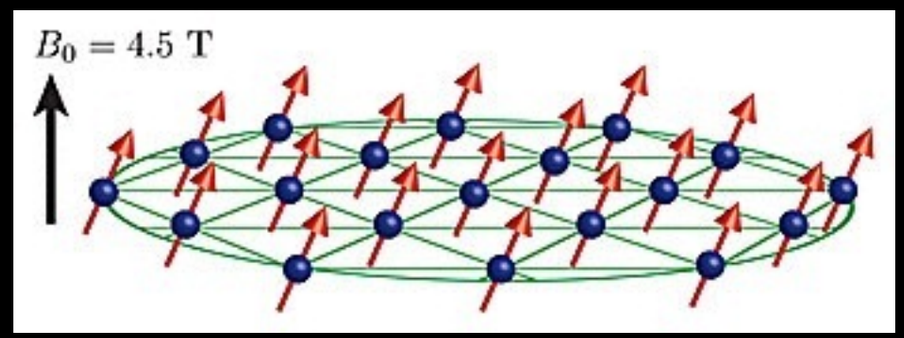
- Richard Feynman (1982) showed that a classical computer (Turing machine) requires exponential time to simulate a quantum system, and suggested idea of using a quantum computer

- Seth Lloyd (1996) showed that a quantum computer can simulate any quantum system efficiently

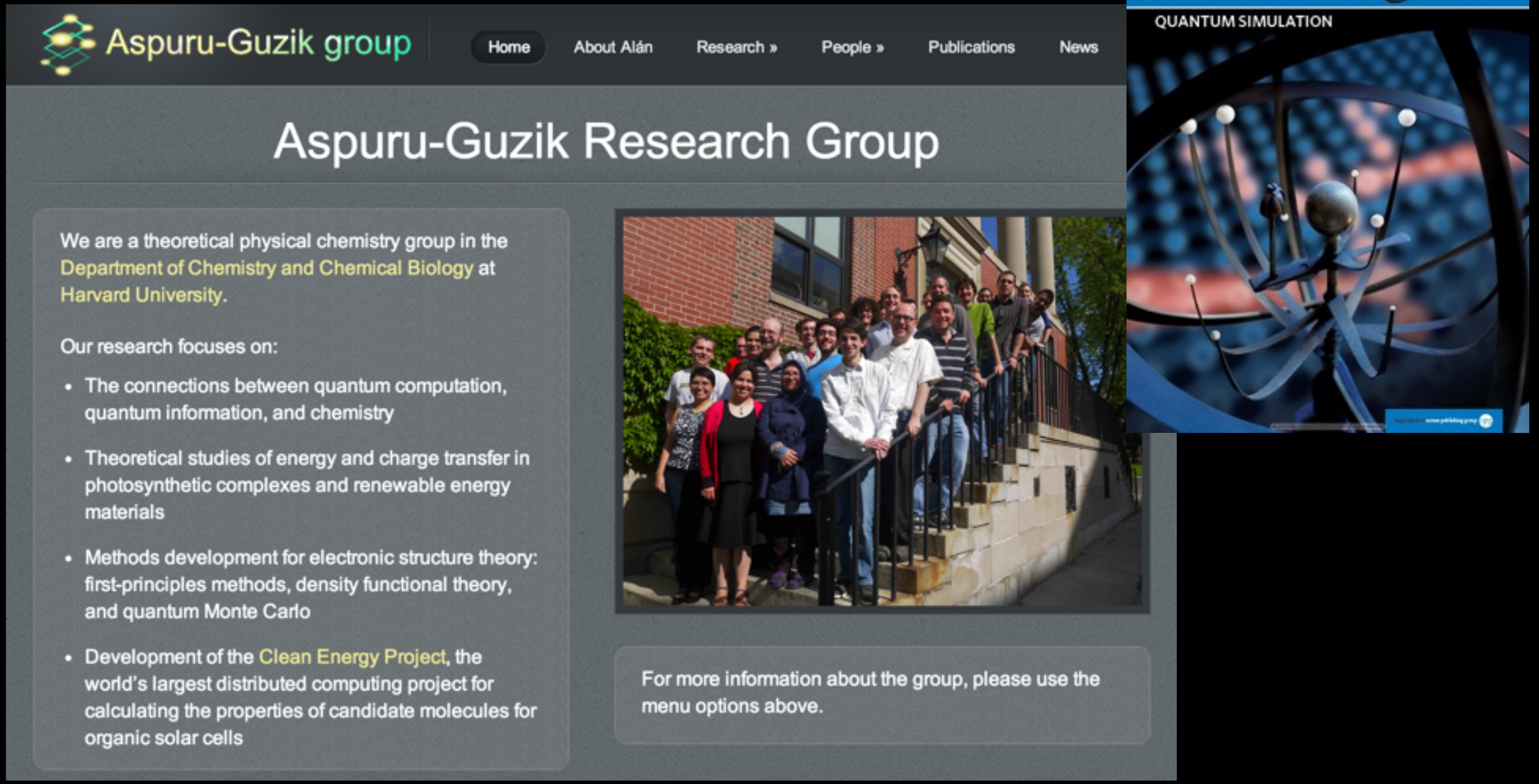
- Many quantum simulations of quantum systems have been built

  - NIST (2012) - built a simulator with 100s of qubits to simulate a quantum magnetic system

  - Simulator: crystal of beryllium ions



The Big Win: Quantum Simulation for Quantum Chemistry



Limitations of Quantum Computing