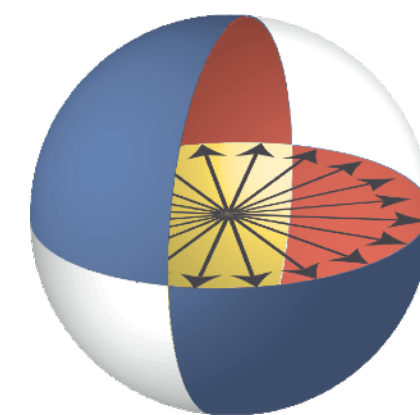


# Quantum algorithms

Andrew Childs

University of Maryland



JOINT CENTER FOR  
QUANTUM INFORMATION  
AND COMPUTER SCIENCE

# Overview

0. Introduction

1. Algebraic problems

2. Quantum walk

3. Hamiltonian simulation

4. Quantum linear algebra

5. Optimization

6. Machine learning

# 0. Introduction

# The origin of quantum speedup

Quantum computers allow for *interference between computational paths*



To perform a computation, we should arrange that

- paths to the solution interfere constructively
- paths to non-solutions interfere destructively

Quantum mechanics gives an *efficient representation of high-dimensional interference*

# Quantum computing $\neq$ exponential parallelism

Can we just explore all potential solutions in parallel and pick out the correct one?



**No!** The linearity of quantum mechanics prohibits this.

Specifically, unstructured search over  $N$  items requires  $\Omega(\sqrt{N})$  queries. [BBBV 97]

To get significant speedup, quantum computers need to exploit structure.

**Key question:** What kinds of problems have the right structure for quantum computers to exploit?

# Simon's problem

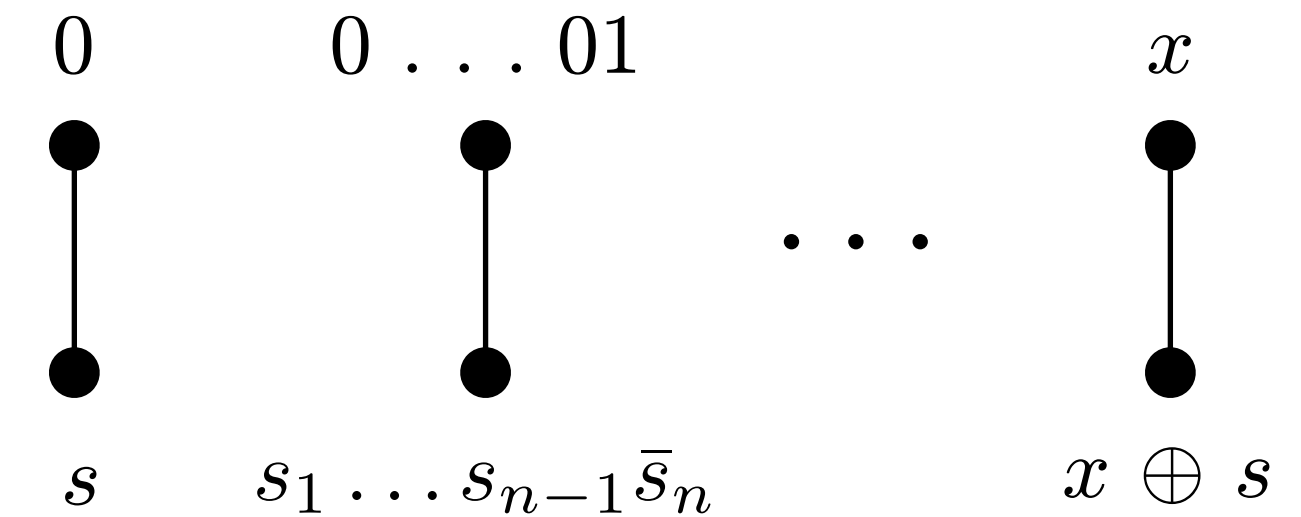
Given a black-box function  $f: \{0, 1\}^n \rightarrow R$

**Promise:** There is some  $s \in \{0, 1\}^n$  such that  $f(x) = f(y)$  if and only if  $x = y$  or  $x = y \oplus s$

**Problem:** Find  $s$

Classically, we can query random inputs until we find a collision. By the birthday problem, this takes about  $\sqrt{2^n}$  steps. This is essentially optimal.

But quantumly, there is a 1-query algorithm that learns a random  $x$  orthogonal to  $s$ . This can be repeated  $O(n)$  times to determine  $s$  with constant probability.





# The collision problem

Given a black-box function  $f: \{0, 1\}^n \rightarrow R$        $N = 2^n$

Promise:  $f$  is either 1-to-1 or 2-to-1



**Problem:** Determine which holds

Can be solved with  $O(N^{1/3})$  queries [Brassard, Høyer, Tapp 97]

- query  $K$  items
- search through remaining items for a duplicate
- cost  $O(K + \sqrt{N/K})$  is minimized with  $K = \Theta(N^{1/3})$

This is optimal! No exponential speedup. [Aaronson, Shi 01]

# The prospect of quantum speedup

The collision problem does not have enough structure to allow a fast quantum algorithm

Simon's problem is a special case with enough additional structure to give a fast quantum algorithm (but not a fast classical algorithm) → exponential speedup

**Major questions:** What problems have fast quantum algorithms?  
What structures enable exponential speedup?

Another important question: When can we get polynomial quantum speedup, and how much is possible?

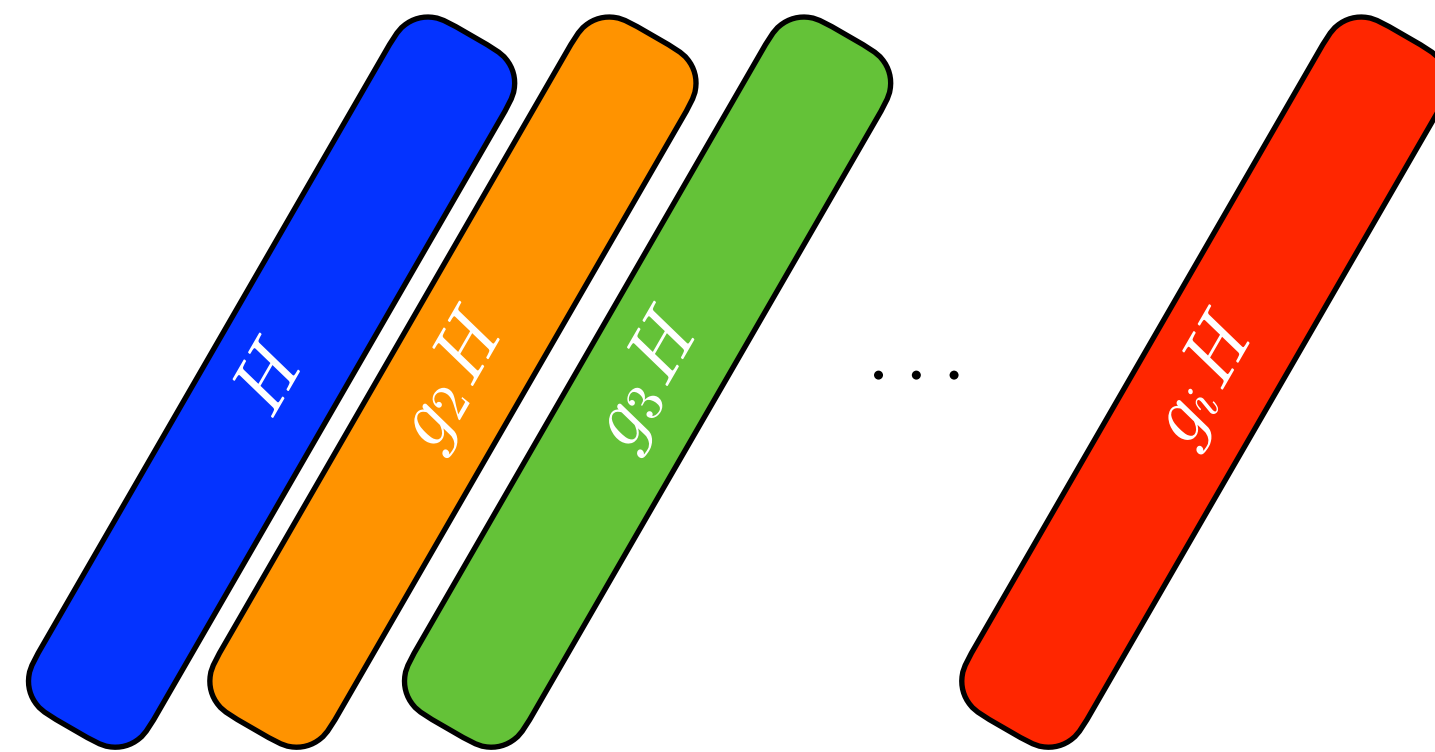


# I. Algebraic problems

# Hidden symmetry

Simon's problem exemplifies a more general class of problems with hidden symmetry

**Hidden subgroup problem:** Given a known group  $G$  and a black-box function  $f: G \rightarrow R$ . Promised that  $f$  is constant on cosets of some (unknown) subgroup  $H \leq G$  and distinct on different cosets. Goal: find (a generating set for)  $H$ .



“Standard method”:  $|G\rangle := \frac{1}{\sqrt{|G|}} \sum_{g \in G} |g\rangle \mapsto \frac{1}{\sqrt{|G|}} \sum_{g \in G} |g, f(g)\rangle$

Discarding second register gives a *coset state*  $|gH\rangle := \frac{1}{\sqrt{|H|}} \sum_{h \in H} |gh\rangle$  for a uniformly random (unknown)  $g$

# Abelian HSP applications

## Finite abelian groups:

- Discrete log [Shor 94]
- Decomposing abelian groups [Cheung, Mosca 01]
- Counting points on curves [Kedlaya 06]

## Infinite abelian groups:

- Factoring [Shor 94]
- Pell's equation ( $x^2 - dy^2 = 1$ ) [Hallgren 02]
- Unit group of a number field [Hallgren 05; Hallgren, Eisenträger, Kitaev, Song 14]
- Principal ideal problem, class groups [Hallgren 05; Biasse, Song 16]
- Ray class groups, Hilbert class fields [Hallgren, Eisenträger 10]

# Nonabelian HSP: Examples and applications

When  $G$  is a nonabelian group, polynomially many queries suffice.

Efficient algorithms known for specific HSPs: normal subgroups, metacyclic groups, Heisenberg/extraspecial groups, etc.

HSPs with exciting potential applications:

- Symmetric group: graph isomorphism, code equivalence
- Dihedral group: lattice problems [Regev 02]

A standard method algorithm for the symmetric group HSP would require highly entangled measurements [Hallgren, Moore, Rötteler, Russell, Sen, 06]

“Kuperberg sieve” solves the dihedral HSP in subexponential time

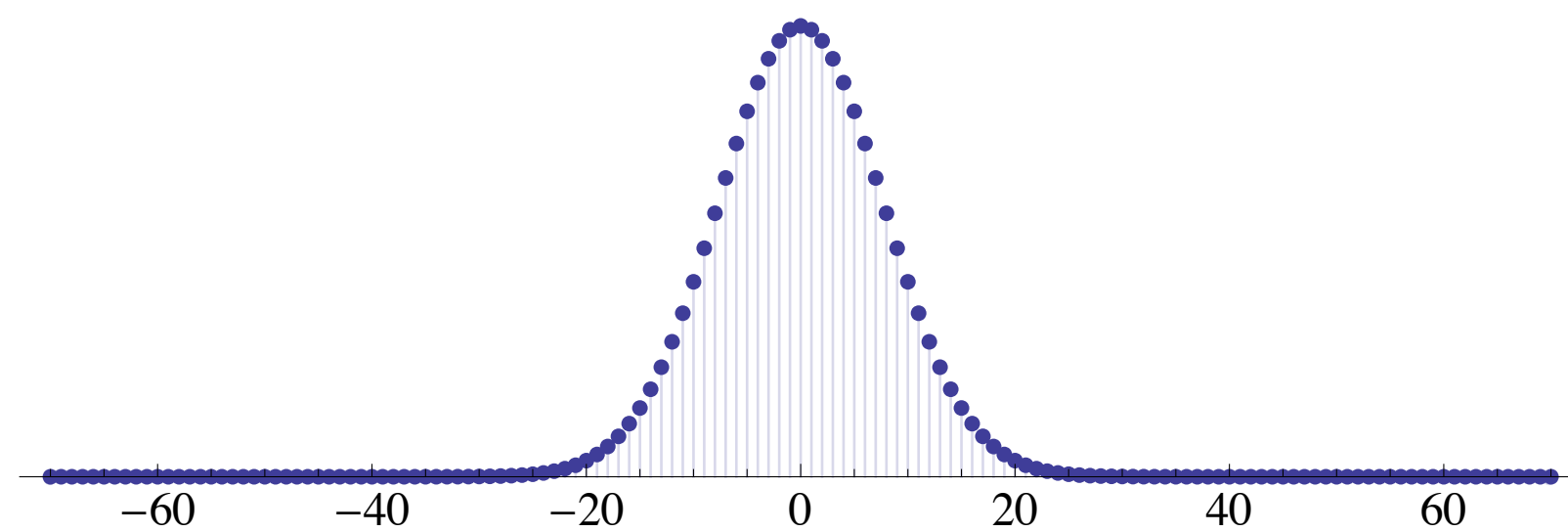
- No quantum speedup for lattice problems
- Subexponential quantum algorithm for elliptic curve isogenies [Childs, Jao, Soukharev 14]

## 2. Quantum walk

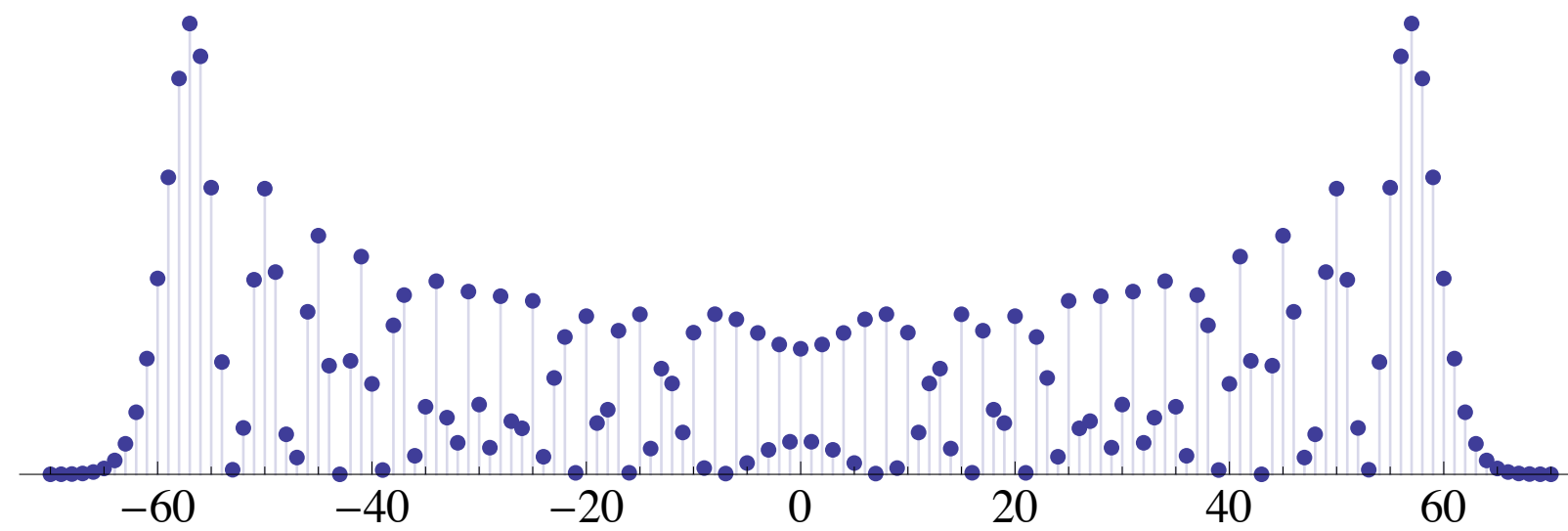
# From random to quantum walk

Quantum analog of a random walk on a graph.

**Idea:** Replace probabilities by quantum amplitudes.  
Interference can produce radically different behavior!



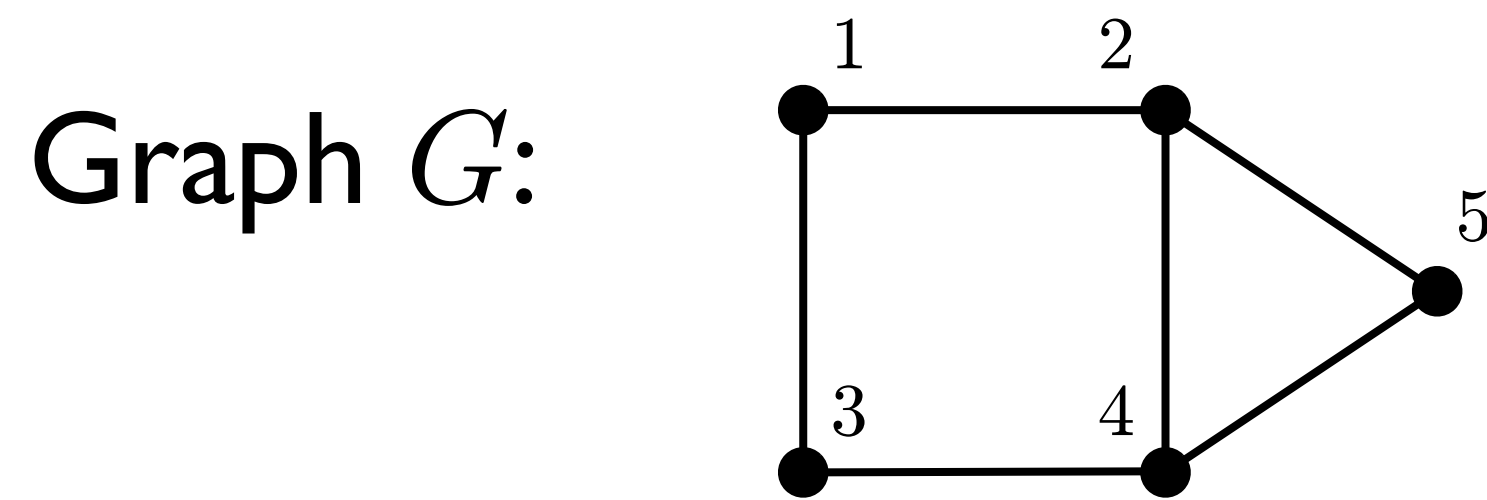
classical



quantum



# Continuous-time quantum walk



$$A = \begin{pmatrix} 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 \end{pmatrix}$$

*adjacency matrix*

$$L = \begin{pmatrix} 2 & -1 & -1 & 0 & 0 \\ -1 & 3 & 0 & -1 & -1 \\ -1 & 0 & 2 & -1 & 0 \\ 0 & -1 & -1 & 3 & -1 \\ 0 & -1 & 0 & -1 & 2 \end{pmatrix}$$

*Laplacian*

## Random walk on $G$

**State:** Probability  $p_v(t)$  of being at vertex  $v$  at time  $t$

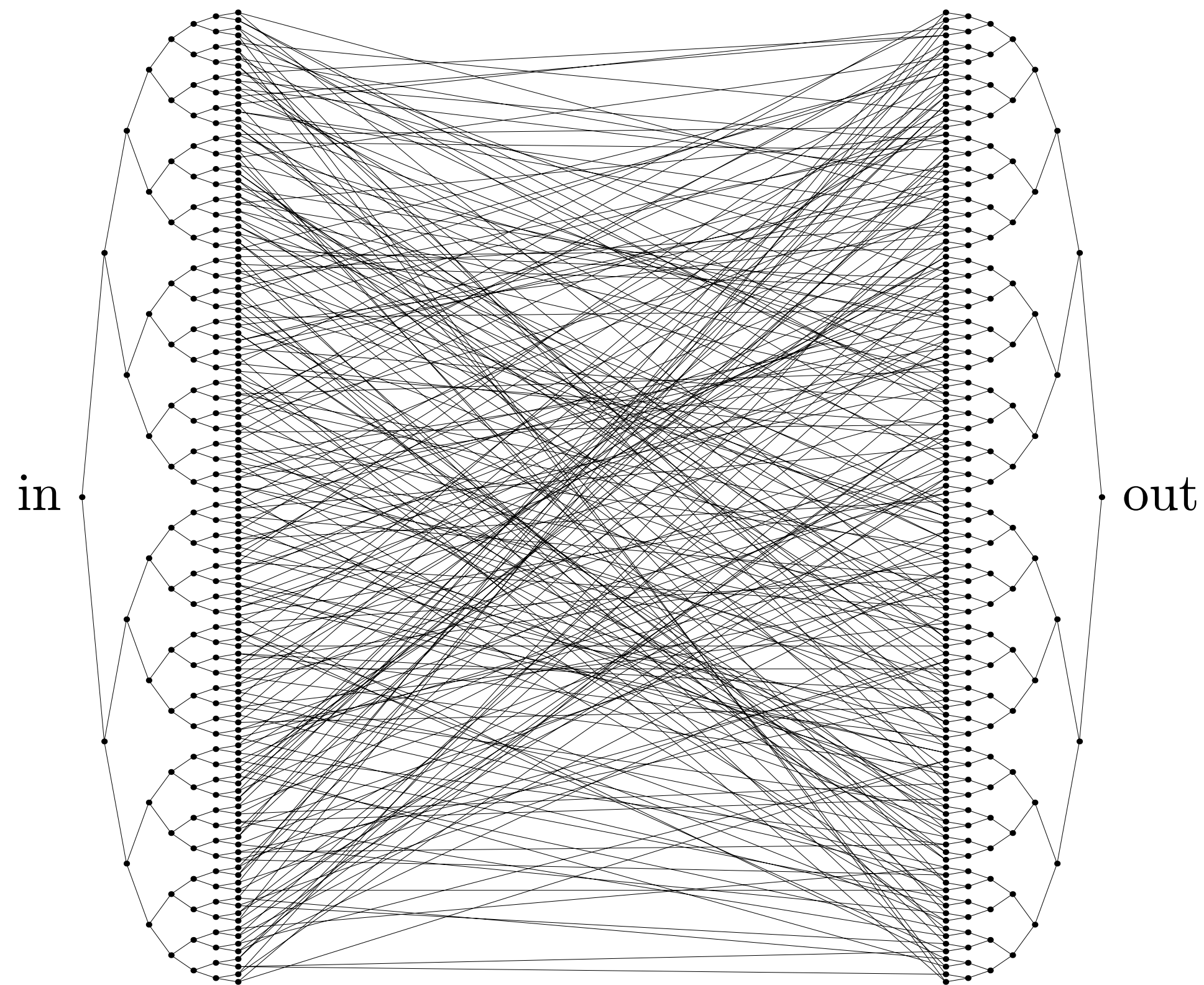
**Dynamics:**  $\frac{d}{dt} \vec{p} = L \vec{p}$

## Quantum walk on $G$

**State:** Amplitude  $a_v(t)$  to be at vertex  $v$  at time  $t$

**Dynamics:**  $i \frac{d}{dt} \vec{a} = L \vec{a}$   
 $i \frac{d}{dt} \vec{a} = A \vec{a}$

# Exponential speedup



**Problem:** Given the label of  $in$  and an adjacency-list black box for the graph, find the label of  $out$ .

Quantum walk from  $|in\rangle$  stays in the *column subspace* (uniform superpositions over vertices at fixed distance from  $in$ ).

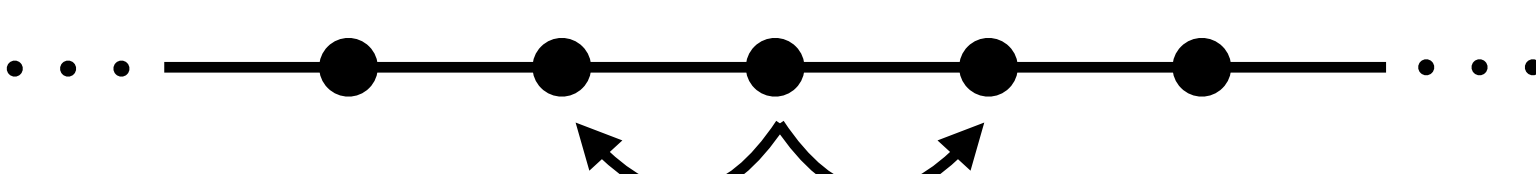
This walk rapidly reaches a state with significant overlap on  $|out\rangle$ .

Using polynomially many queries, a classical algorithm cannot distinguish the graph from an infinite binary tree rooted at  $in$ .

[Childs, Cleve, Deotto, Farhi, Gutmann, Spielman 03]

# Discrete-time quantum walk

A walk with discrete time steps is a little harder to define.

On a path:  $|x\rangle \mapsto \frac{1}{\sqrt{2}} (|x-1\rangle + |x+1\rangle)$ ?  **Not unitary!**

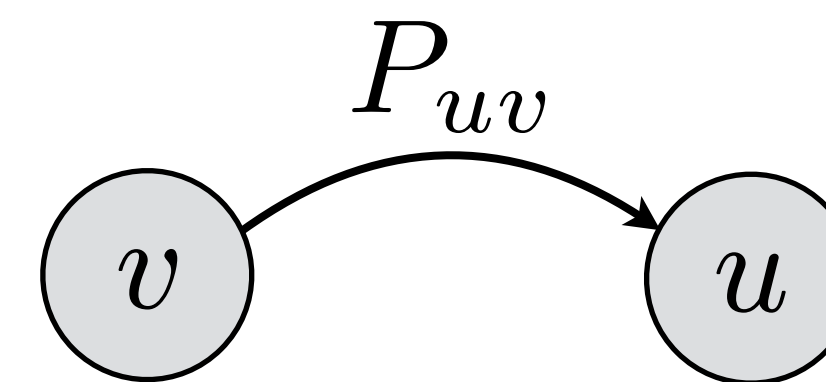
**Solution:** Introduce another register (“coin”) that remembers the previous position  
(reduces the potential for interference, but only slightly)

**Szegedy walk:** For a stochastic transition matrix  $P$ ,

- Reflect about  $\text{span}\{|\psi_v\rangle : v \in V\}$

where  $|\psi_v\rangle := \sum_{u \in V} \sqrt{P_{uv}} |v, u\rangle$

- Swap the edge direction:  $S := \sum_{u, v \in V} |u, v\rangle \langle v, u|$





# Quantum walk search

**Problem:** Given a graph  $G = (V, E)$  with a subset  $M \subseteq V$  of *marked vertices*. Using an oracle that tells whether a vertex is marked, determine whether  $M$  is empty.

**Classical strategy:** Take a random walk until we reach a marked vertex.

Time to hit a marked vertex is  $O(1/\delta\epsilon)$ , where

$$\delta = \text{spectral gap of walk} \quad \epsilon = |M|/|V|$$

(second-largest magnitude of an eigenvalue of transition matrix is  $1 - \delta$ )

**Quantum strategy:** Consider the Szegedization of the *absorbing walk* that remains at a marked vertex

Perform phase estimation on  $|\psi\rangle \propto \sum_{x \notin M} |\psi_x\rangle$

This state is invariant if  $|M| = 0$  and lives in eigenspaces with phase  $\Omega(\sqrt{\delta\epsilon})$  if  $|M| \neq 0$ , so  $O(1/\sqrt{\delta\epsilon})$  steps of the walk suffice to determine whether  $|M| = 0$ .

# Quantum walk search: examples

**Unstructured search:**  $G =$  complete graph on  $N$  vertices  $\delta = \Theta(1)$   $\epsilon = 1/N$

**Classical:**  $O(N)$  **Quantum:**  $O(\sqrt{N})$

**Element distinctness:** [Ambainis 04]

Given  $f: [N] \rightarrow R$ , are there distinct  $x, y \in [N]$  with  $f(x) = f(y)$ ?  $[N] := \{1, \dots, N\}$

**Classical:**  $\Omega(N)$

**Quantum:** Consider walk on Hamming graph  $H(N, K)$

vertices =  $[N]^K$ , edges between  $K$ -tuples that differ in one coordinate

store function values associated with the  $K$  inputs

$\delta = \Omega(1/K)$   $\epsilon = \Omega((K/N)^2)$

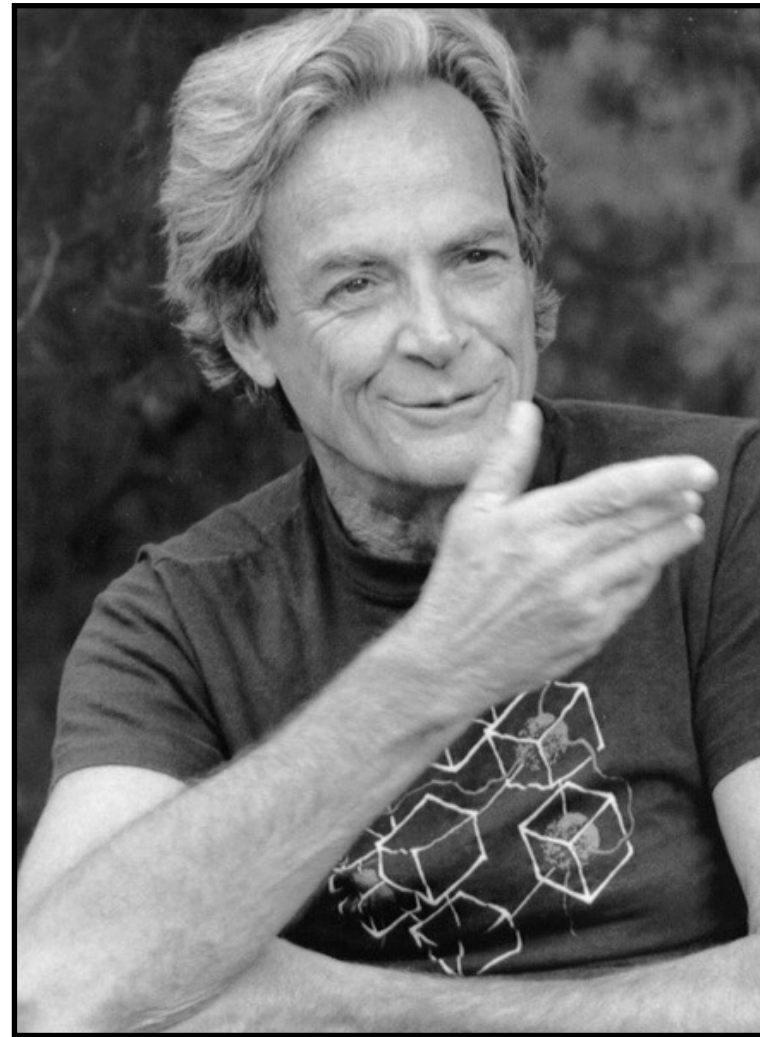
complexity  $K + N/\sqrt{K}$ , optimized with  $K = N^{2/3}$

This provides a powerful, general tool for search problems

# 3. Hamiltonian simulation



# Simulating Hamiltonian dynamics



“... nature isn’t classical, dammit, and if you want to make a simulation of nature, you’d better make it quantum mechanical, and by golly it’s a wonderful problem, because it doesn’t look so easy.”

Richard Feynman (1981)  
*Simulating physics with computers*

**Quantum simulation problem:** Given a description of the Hamiltonian  $H$ , an evolution time  $t$ , and an initial state  $|\psi(0)\rangle$ , produce the final state  $|\psi(t)\rangle$  (to within some error tolerance  $\epsilon$ )

A classical computer cannot even represent the state efficiently.

A quantum computer cannot produce a complete description of the state.

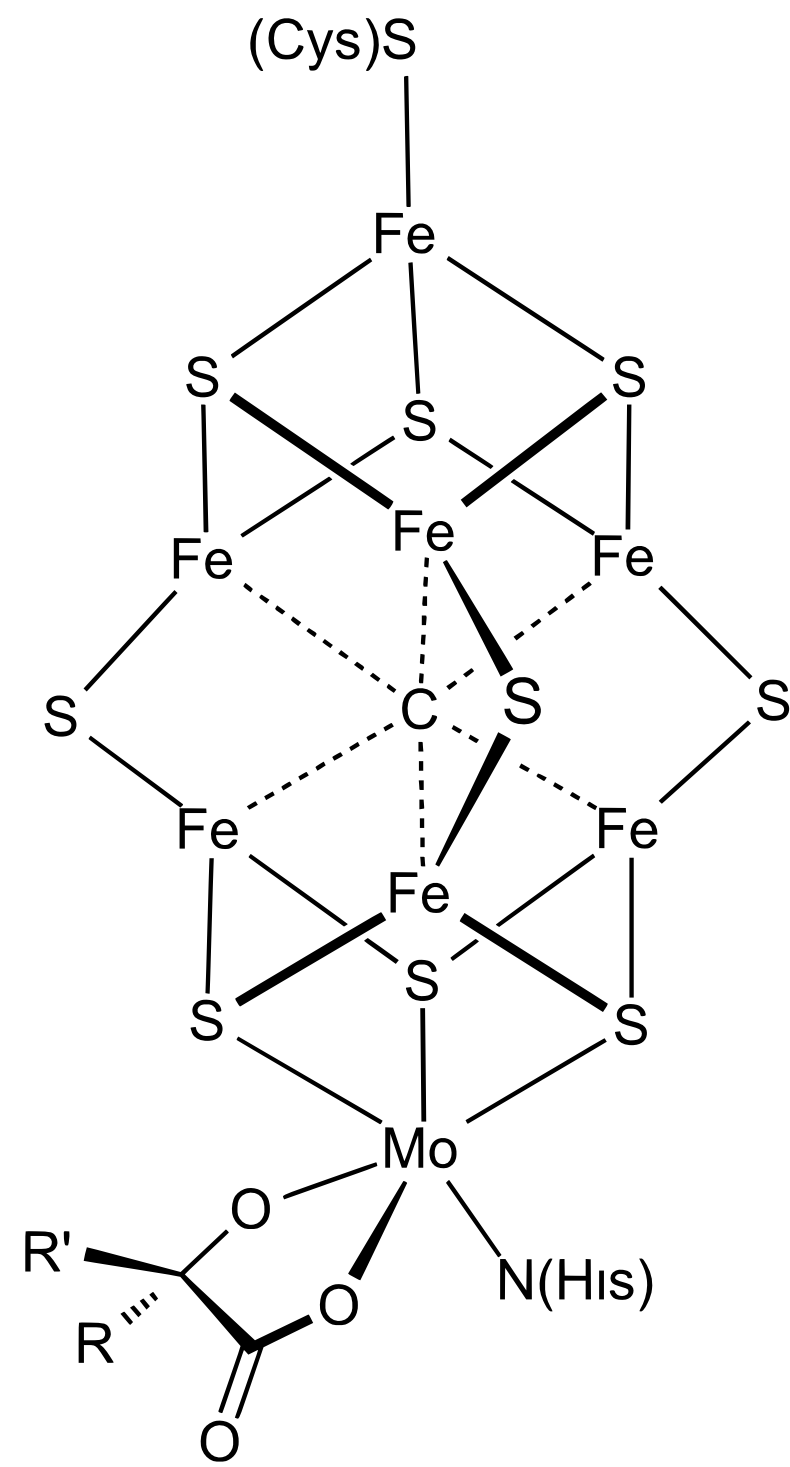
But given succinct descriptions of

- the initial state (suitable for a quantum computer to prepare it efficiently) and
- a final measurement (say, measurements of the individual qubits in some basis),

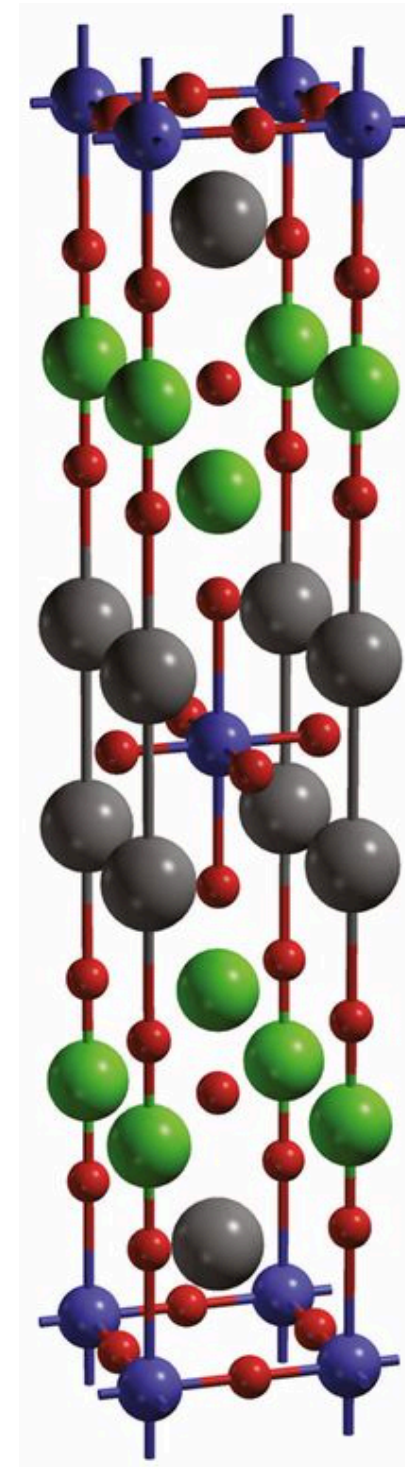
a quantum computer can efficiently answer questions that (apparently) a classical one cannot. Simulation is BQP-complete!



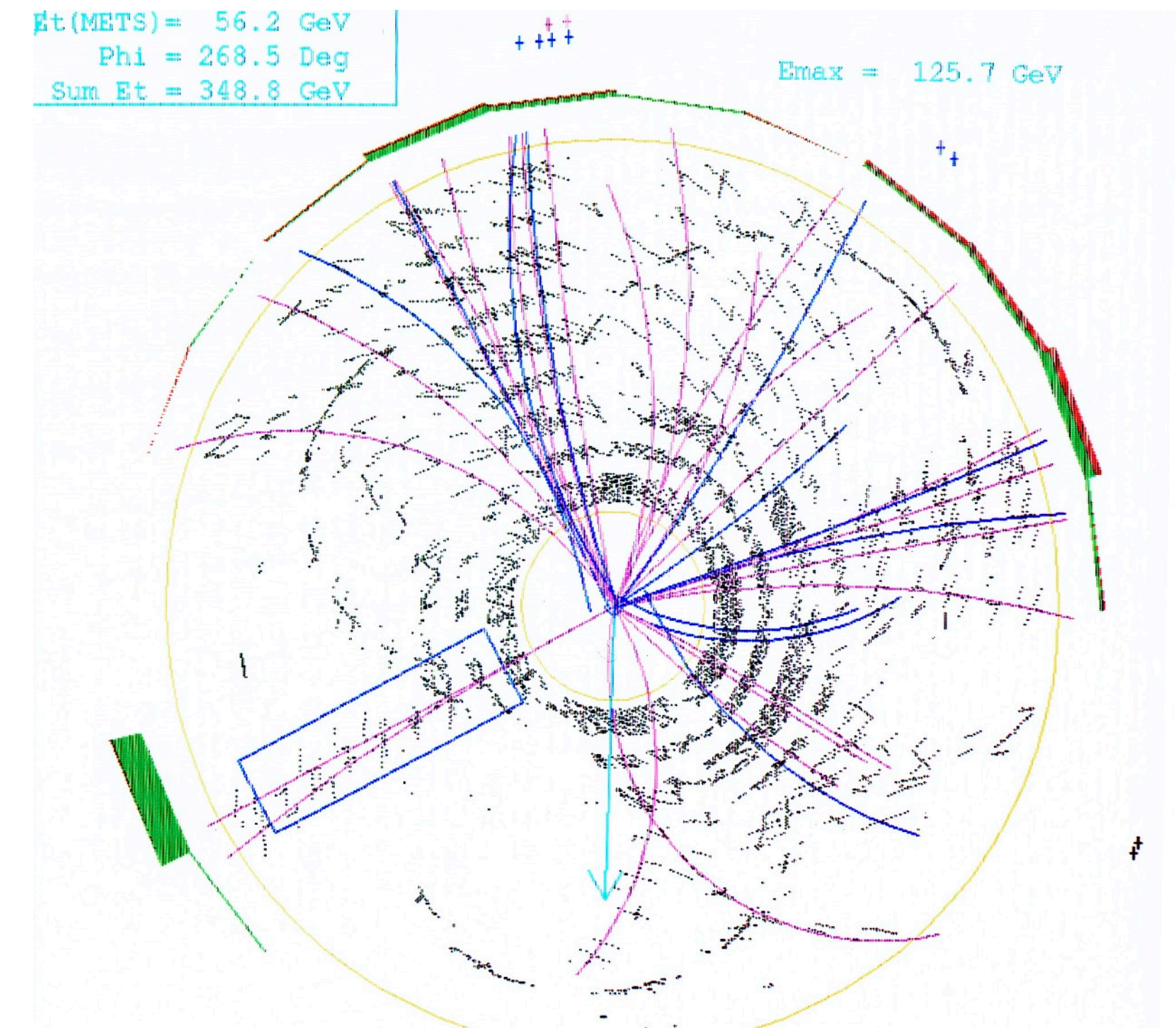
# Computational quantum physics



quantum chemistry  
(e.g., nitrogen fixation)

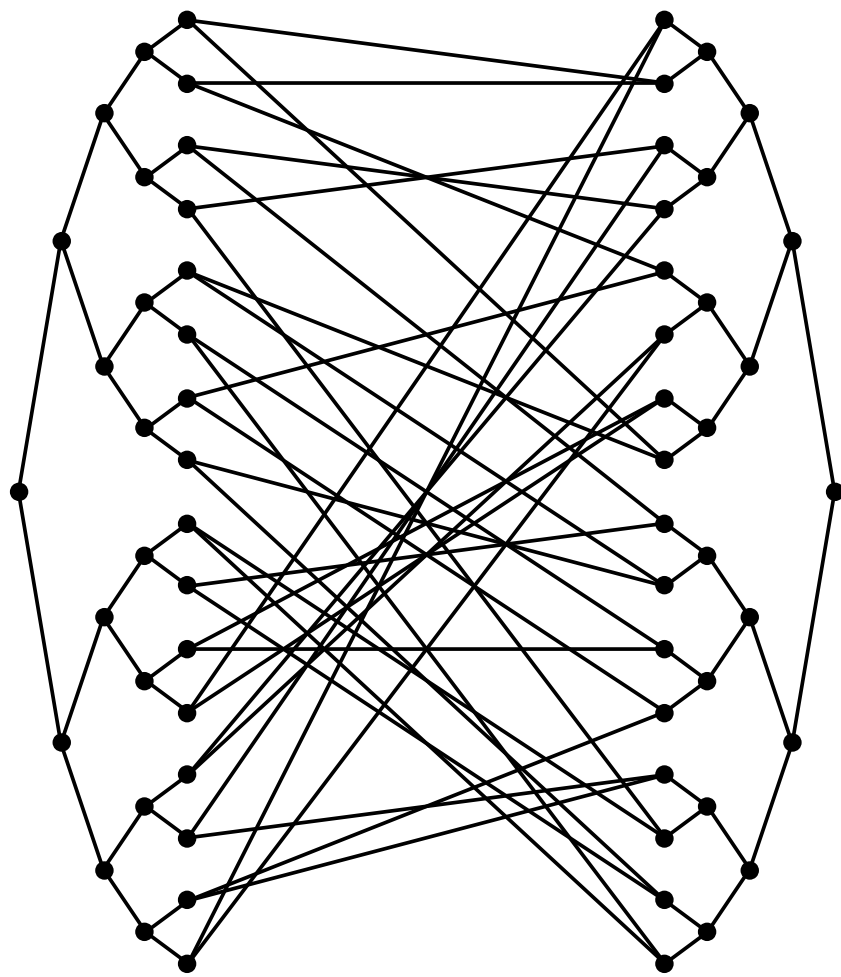


condensed matter physics/  
properties of materials

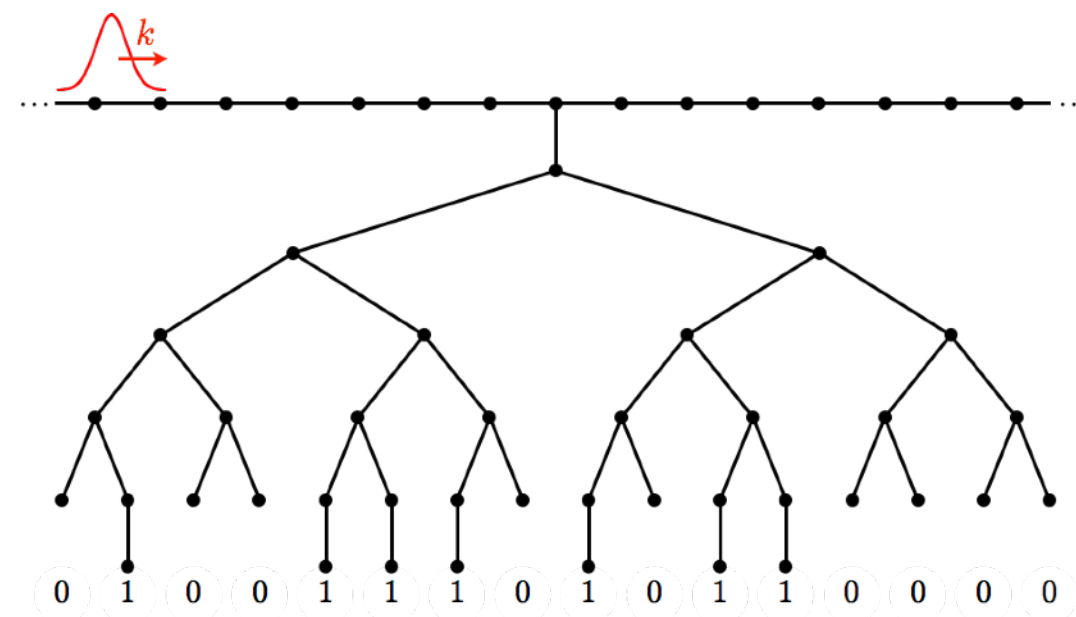


nuclear/particle  
physics

# Implementing quantum algorithms



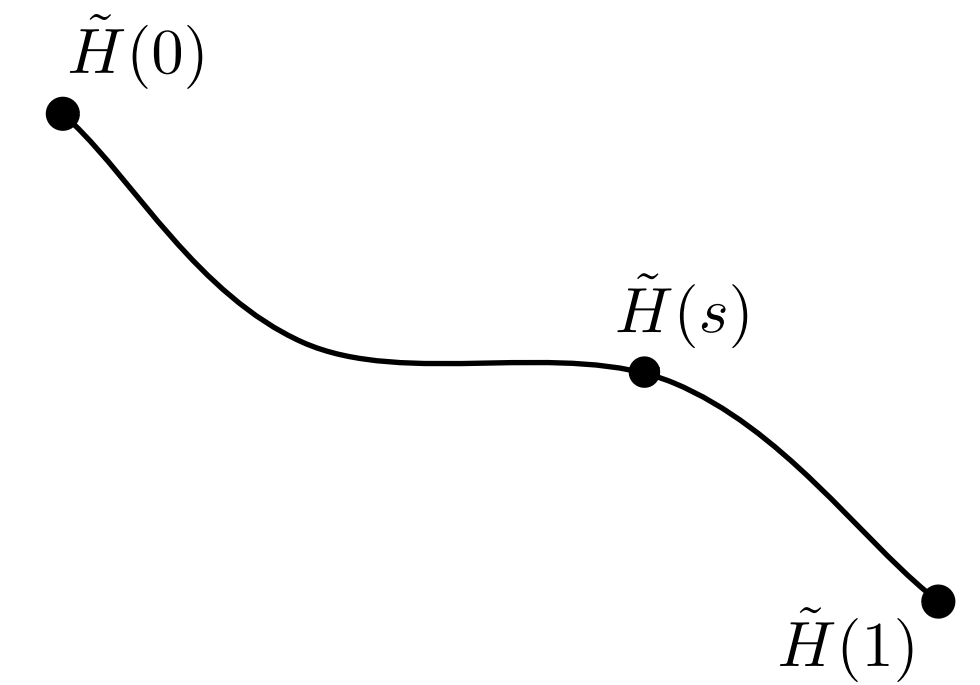
exponential  
speedup by  
quantum walk



evaluating  
Boolean  
formulas

$$A|x\rangle = |b\rangle$$

linear/  
differential  
equations,  
convex  
optimization



adiabatic  
optimization



# Product formulas

Suppose we want to simulate  $H = \sum_{\ell=1}^L H_{\ell}$

Combine individual simulations with the Lie product formula. E.g., with two terms:

$$\lim_{r \rightarrow \infty} \left( e^{-iAt/r} e^{-iBt/r} \right)^r = e^{-i(A+B)t}$$

$$\left( e^{-iAt/r} e^{-iBt/r} \right)^r = e^{-i(A+B)t} + O(t^2/r)$$

To ensure error at most  $\epsilon$ , take

$$r = O\left(\frac{\|H\|t^2}{\epsilon}\right) \quad \text{[Lloyd 96]}$$

Gives simulation of  $d$ -sparse Hamiltonians with complexity  $\text{poly}(d)$  [Aharonov, Ta-Shma 03]

To get a better approximation, use higher-order formulas.

E.g., second order:

$$\left( e^{-iAt/2r} e^{-iBt} e^{-iAt/2r} \right)^r = e^{-i(A+B)t} + O(t^3/r^2)$$

Systematic expansions to arbitrary order are known [Suzuki 92]

Using the  $2k$ th order expansion, the number of exponentials required for an approximation with error at most  $\epsilon$  is at most

$$5^{2k} L^2 \|H\| t \left( \frac{L \|H\| t}{\epsilon} \right)^{1/2k}$$

[Berry, Ahokas, Cleve, Sanders 07]

# Post-Trotter algorithms I

## Linear-time simulation

“No Fast-Fowarding Theorem”: simulation for time  $t$  has complexity  $\Omega(t)$

[Berry, Ahokas, Cleve, Sanders 07]

Applying phase estimation to a Szegedization of  $H$  gives an  $O(t)$  simulation

[Childs 10; Berry, Childs 12]

## High-precision simulation

Directly implement the truncated Taylor series of  $\exp(-iHt)$ , cost  $O\left(t \frac{\log(t/\epsilon)}{\log \log(t/\epsilon)}\right)$

LCU Lemma: implement  $U = \sum_j \beta_j V_j$  with complexity  $O(\sum_j |\beta_j|)$

This is the optimal dependence on  $\epsilon$

[Berry, Childs, Cleve, Kothari, Somma 14 & 15]

# Post-Trotter algorithms II

## Optimal tradeoff

Quantum signal processing (QSP) implements polynomials of a given “block-encoded” Hamiltonian (or more general matrix)

$$U = \begin{pmatrix} H & \cdot \\ \cdot & \cdot \end{pmatrix}$$

Gives  $d$ -sparse Hamiltonian simulation with cost  $O(dt + \log(1/\epsilon))$  [Low, Chuang 17]

QSP and “quantum singular value transformation” [Gilyén, Su, Low, Wiebe 19] provide versatile tools for other tasks

## Lattice Hamiltonians

Can do even better if the Hamiltonian has spatially local interactions

All above methods use  $\Omega(n^2)$  gates to simulate  $n$  spins with local interactions for constant time

Combining forward and backward evolution and applying Lieb-Robinson bounds, can improve this to  $\tilde{O}(n)$ , which is optimal [Haah, Hastings, Kothari, Low 18]

Also other algorithms using multiproduct formulas, interaction picture, randomization, other norms, ...



# Product formulas strike back

Numerical simulations suggest that product formulas can perform much better than straightforward bounds show

Can give tighter bounds using integral representations of the error, e.g.,

$$e^{-iBt}e^{-iAt} - e^{-i(A+B)t} = \int_0^t d\tau_1 \int_0^{\tau_1} d\tau_2 e^{-i(A+B)(t-\tau_1)} e^{i(\tau_2-\tau_1)B} [A, B] e^{-i\tau_2 B} e^{-i\tau_1 A}$$

Provides bounds that can take advantage of small commutators between terms

In particular, shows that product formulas nearly reproduce the complexity of [Haah, Hastings, Kothari, Low 18] for lattice Hamiltonians [Childs, Su, Tran, Wiebe, Zhu 19]

Can give even better bounds with information about the initial state

# 4. Quantum linear algebra

# Quantum linear systems algorithm

Given an  $N \times N$  system of linear equations  $Ax = b$ , find  $x = A^{-1}b$

Classical (or quantum!) algorithms need time  $\Omega(N)$  just to write down  $x$

What if we change the model?

- $A$  is sparse; given a black box that specifies the nonzero entries in any given row or column
- Can efficiently prepare a quantum state  $|b\rangle$
- Goal is to prepare a state  $|x\rangle \propto A^{-1}|b\rangle$

We can do this in time  $\text{poly}(\log N, 1/\epsilon, \kappa)$  where  $\kappa := \|A\| \cdot \|A^{-1}\|$

[Harrow, Hassidim, Lloyd 09]

Algorithm estimates the eigenvalues of  $A$  (in superposition) and replaces them by their inverse (using postselection)

Subsequent improvements do the same with complexity  $\kappa \text{poly}(\log(1/\epsilon))$  using variable-time amplitude amplification and LCU [Ambainis 12; Childs, Kothari, Somma 17]

# Differential equations

We can apply a similar framework to other linear-algebraic tasks. For example:

Given a system of linear differential equations  $\frac{d}{dt}x = Ax + b$  with the ability to prepare  $|b\rangle$  and  $|x(0)\rangle$ , and a sparse matrix oracle for  $A$ , prepare  $|x(T)\rangle$  for some desired final time  $T$

Approach: apply a finite difference approximation to give a linear system; solve it with the QLSA [Berry 14]

Generalizations give improved performance and also handle time-dependent coefficients, partial differential equations, some nonlinear differential equations, ...

# Applications?

Linear equations and differential equations are ubiquitous. Surely we can use this for something?

Proposals: electromagnetic scattering, machine learning, finance, ...

The input/output requirements impose serious constraints. So far there is no compelling end-to-end application with rigorous evidence for speedup. Can we find one?



# 5. Optimization

# Discrete optimization

Grover's algorithm  $\Rightarrow$  quadratic speedup for minimization [Dürr, Hoyer 96]

## Graph algorithms

- shortest paths [Dürr, Heiligman, Høyer, Mhalla 04]
- minimum spanning trees [Dürr, Heiligman, Høyer, Mhalla 04]
- maximum flows/matchings [Ambainis, Špalek 07]

Speeding up exponential-time algorithms for NP-hard problems (SAT, subset sum, lattice problems, TSP, set cover, ...)

Some of these algorithms introduce interesting new tools:

- quantum backtracking using quantum walk [Montanaro 16]
- quantum methods for dynamic programming [Ambainis, Balodis, Iraids, Kokainis, Prusis, Vihrovs 18]

# Continuous optimization

## Linear/semidefinite programming

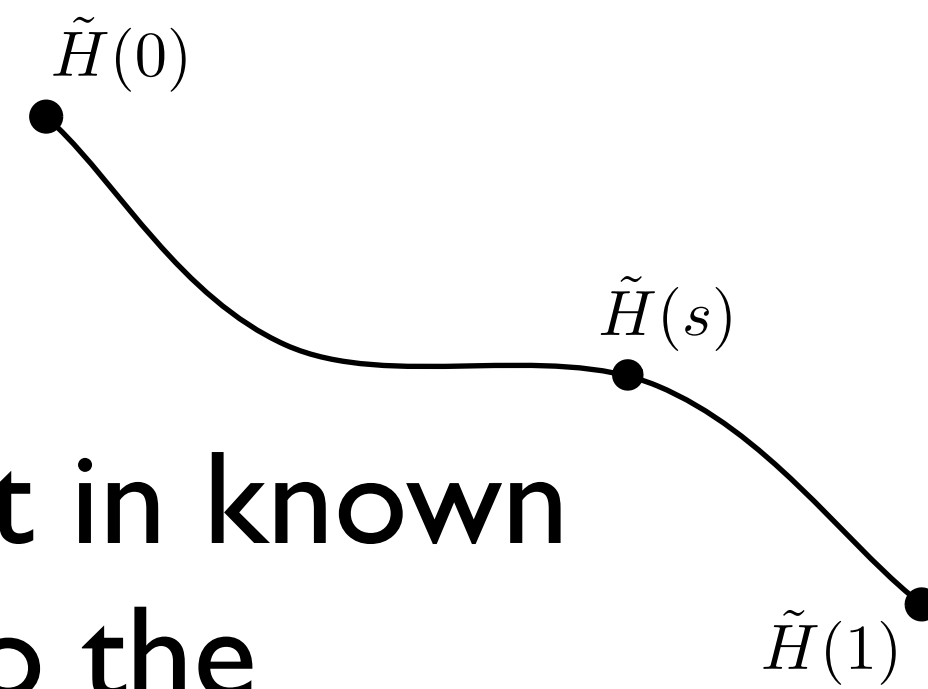
- polynomial speedups based on Gibbs sampling [Brandão, Svore 17; van Apeldoorn, Gilyén 19]
- faster algorithms in a stronger input model [Brandão, Kalev, Li, Lin, Svore, Wu 19]

## Gradient-based algorithms

- Fast algorithm for computing gradients [Jordan 05]
- Minimization using gradient descent [Rebentrost, Schuld, Wossnig, Petruccione, Lloyd 19; Kerenidis, Prakash 20]
- Quantum query speedup for convex optimization with membership and evaluation oracles [van Apeldoorn, Gilyén, Gribling, de Wolf 20; Chakrabarti, Childs, Li, Wu 20]
- For high-dimensional non-smooth convex optimization with a gradient oracle, cannot achieve a quantum speedup as a function of the allowed error [Garg, Kothari, Netrapalli, Sherif 20]

# Adiabatic optimization and QAOA

Strategy: encode a constraint problem with a diagonal Hamiltonian. Start in known ground state of a simple, non-diagonal Hamiltonian. Slowly interpolate to the problem Hamiltonian to produce its ground state. [Farhi, Goldstone, Gutmann, Sipser 00]



Complexity depends on the minimum spectral gap, but this is hard to estimate.

Often this is done with a Hamiltonian that has all negative off-diagonal entries (“stoquastic”). Then we can in principle apply quantum Monte Carlo (a classical algorithm), but its efficiency is also unclear.

Related strategy: quantum approximate optimization algorithm (QAOA). Alternate between diagonal & off-diagonal evolutions with optimized parameters. [Farhi, Goldstone, Gutmann 14]

# 6. Machine learning



# Quantum machine learning

A challenge: much of the impressive success of classical machine learning is empirical

Quantum algorithms for some ML tasks have been proposed, e.g., recommendation systems [Kerenidis, Prakash 17]

Data structures that enable coherent quantum access can be exploited classically [Tang 19]

Other proposed algorithms for principal component analysis, clustering, etc. Potential for quantum speedup is unclear.

Another direction: computational learning theory [survey:Arunachalam, de Wolf 17]

Learn a concept given the ability to interact with it quantumly

- query access to a concept  $c: \{0, 1\}^n \rightarrow \{0, 1\}$
- quantum examples  $\sum_x \sqrt{p_x} |x, c(x)\rangle$

**Conclusion**

# Outlook

Finding quantum algorithms is hard!

- Quantum mechanics is nonintuitive
- Classical algorithms are powerful
- We have limited quantum techniques

But we have come a long way in the 30 years since Shor's algorithm

- New exponential speedups
- New techniques
- Much better understanding of quantum query complexity

Large-scale quantum computers could dramatically change our understanding of quantum algorithms

# Further reading

Quantum Algorithm Zoo: [quantumalgorithmzoo.org](http://quantumalgorithmzoo.org)

Lecture notes: [cs.umd.edu/~amchilds/qa/](http://cs.umd.edu/~amchilds/qa/)

Montanaro survey: [arXiv:1511.04206](https://arxiv.org/abs/1511.04206)

QIP tutorials:

- András Gilyén (2020): [www.koushare.com/video/videodetail/4073](http://www.koushare.com/video/videodetail/4073)
- Andrew Childs (2021, longer version of this talk): [youtu.be/M0e5gkf7QSQ](https://youtu.be/M0e5gkf7QSQ)
- Robin Kothari (2024): [youtu.be/4jjswyS9ieg](https://youtu.be/4jjswyS9ieg)

Topical surveys:

- quantum walk search (Santha): [arXiv:0808.0059](https://arxiv.org/abs/0808.0059)
- quantum walk (Reitzner, Nagaj, Buzek): [arXiv:1207.7283](https://arxiv.org/abs/1207.7283)
- algebraic problems (Childs, van Dam): [arXiv:0812.0380](https://arxiv.org/abs/0812.0380)
- optimization (de Wolf): [youtu.be/1-2LlopvNlk](https://youtu.be/1-2LlopvNlk)
- computational learning theory (Arunachalam, de Wolf): [arXiv:1701.06806](https://arxiv.org/abs/1701.06806)