Quantum algorithms

Andrew Childs
University of Maryland

COMPUTER SCIENCE
UNIVERSITY OF MARYLAND

UMIACS
University of Maryland
Institute for Advanced
Computer Studies

JOINT CENTER FOR
QUANTUM INFORMATION
AND COMPUTER SCIENCE

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 ≠ 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.

To get significant speedup, quantum computers need to exploit structure

Key question: What kinds of problems have the right structure for quantum computers to give exponential speedup?

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

Outline

1. Quantum walk
2. Hamiltonian simulation
3. Quantum linear algebra

For a broader overview of quantum algorithms, see my QIP 2021 tutorial:

https://www.cs.umd.edu/~amchilds/talks/qip21.pdf

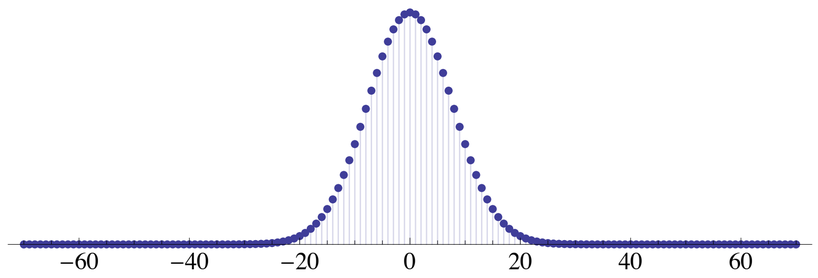
https://youtu.be/M0e5gkf7QSQ

1. Quantum walk

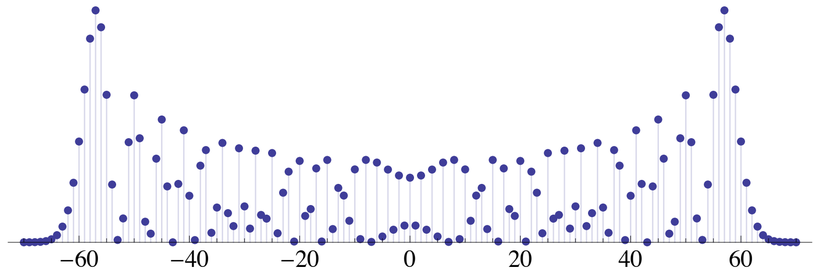
From random to quantum walk

Quantum analog of a random walk on a graph.

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



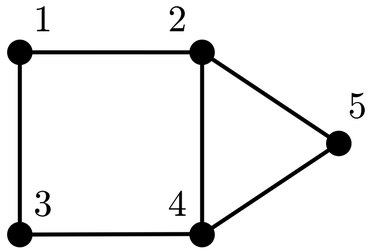
classical



quantum

Continuous-time quantum walk

Graph G:



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

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: $\frac{\mathrm{d}}{\mathrm{d}t}\vec{p} = L\vec{p}$

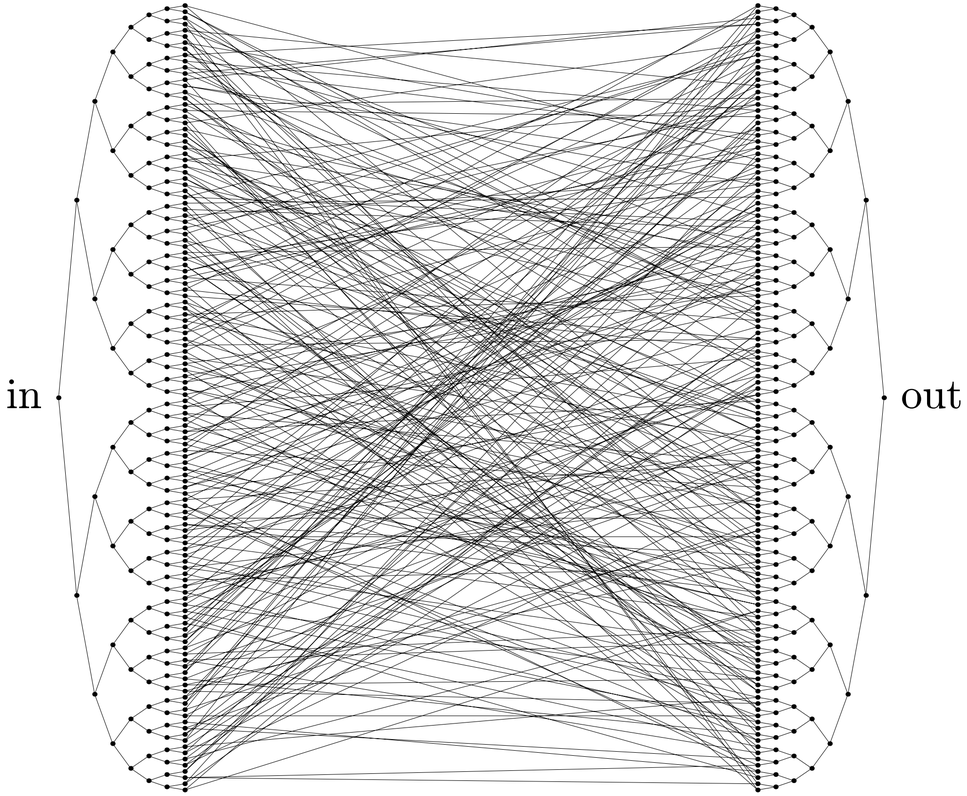
Quantum walk on G

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

Dynamics: $i\frac{\mathrm{d}}{\mathrm{d}t}\vec{a} = L\vec{a}$

$$i\frac{\mathrm{d}}{\mathrm{d}t}\vec{a} = A\vec{a}$$

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

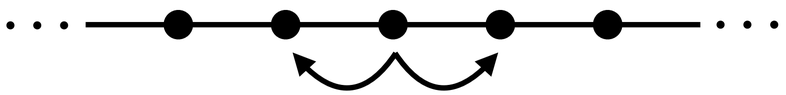
This walk rapidly reaches a state with significant overlap on $|\mathrm{out}\rangle$.

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

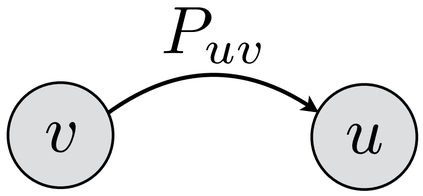
A walk with discrete time steps is a little harder to define.

On a path: $|x\rangle \mapsto \frac{1}{\sqrt{2}}\big(|x-1\rangle + |x+1\rangle\big)$?  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 $\mathrm{span}\{|\psi_v\rangle : v \in V\}$
  where $|\psi_v\rangle := \sum_{u \in V} \sqrt{P_{uv}}|v,u\rangle$
- Swap the edge direction: $S := \sum_{u,v \in V} |u,v\rangle\langle v,u|$

[Szegedy 05]

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

δ = 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 $\quad \delta = \Theta(1) \quad \epsilon = 1/N$

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

Element distinctness: [Ambainis 04]

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

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

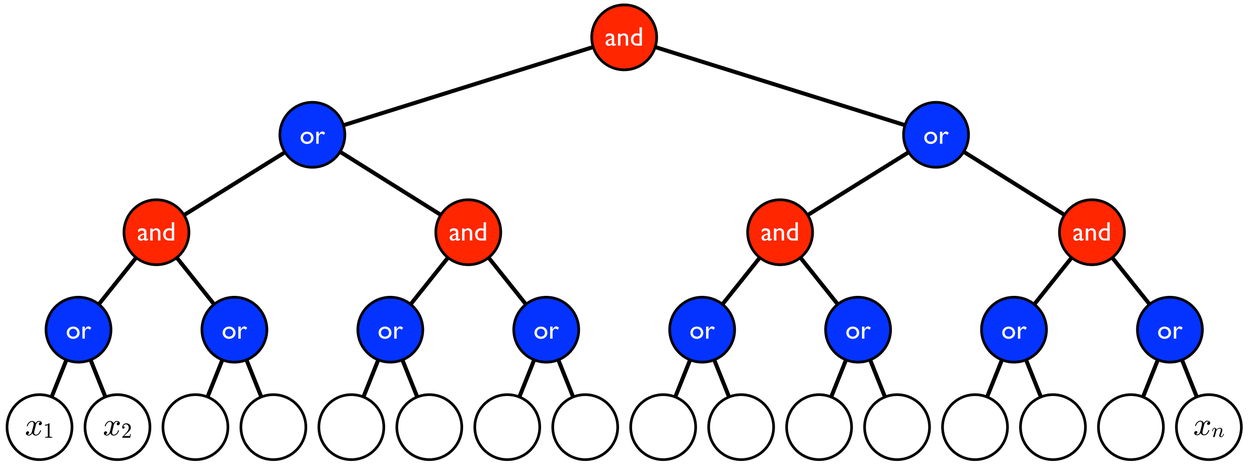
$\delta = \Omega(1/K) \quad \epsilon = \Omega((K/N)^2)$

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

This provides a powerful, general tool for search problems

Formula evaluation

Consider a balanced binary AND-OR tree:

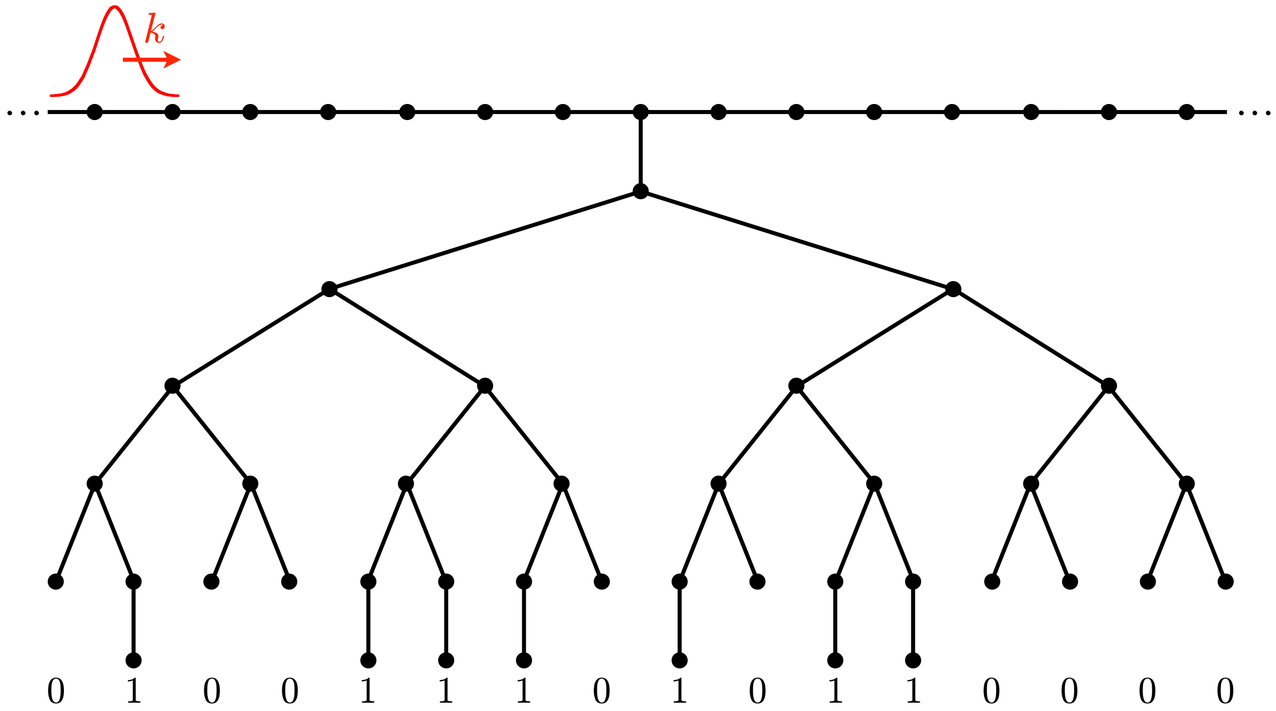


Classical complexity: $\Theta(n^{0.753\ldots})$ [Snir 85; Saks, Wigderson 86; Santha 95]

Quantum lower bound: $\Omega(\sqrt{n})$ [Barnum, Saks 02]

(holds for arbitrary AND-OR formulas)

Formula evaluation by scattering



Claim: For $k = \Theta(1/\sqrt{n})$, the wave is transmitted if the formula (translated into NAND gates) evaluates to 0, and reflected if it evaluates to 1. [Farhi, Goldstone, Gutmann 07]

General formulas and span programs

In fact the quantum query complexity of *any* n-input AND-OR formula is $O(\sqrt{n})$

[Reichardt 10]

One approach: apply phase estimation to a quantum walk on a tree that encodes the formula

Alternative: construct a *span program*, composing span programs for elementary gates

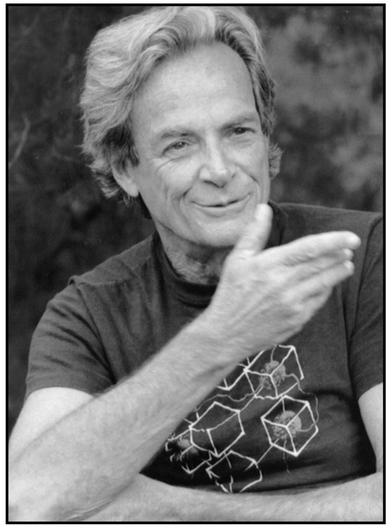
Quantum adversary lower bound: $\mathrm{Adv}(f) = \max_{\Gamma} \frac{\|\Gamma\|}{\max_i \|\Gamma_i\|}$

The dual of this semidefinite program can be used to construct a quantum algorithm for evaluating f with $O(\mathrm{Adv}(f))$ queries (apply phase estimation to a kind of generalized quantum walk)

Useful for understanding general features of query complexity.
In particular: $\mathrm{Adv}(f \circ g) \leq \mathrm{Adv}(f)\,\mathrm{Adv}(g)$

2. 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 ϵ)

A classical computer cannot even represent the state efficiently.

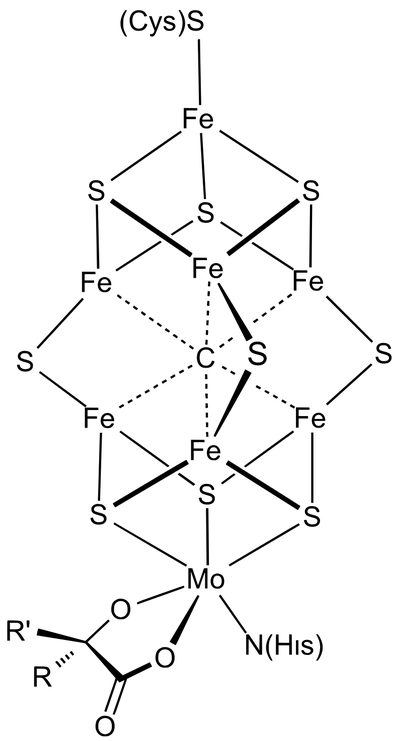
A quantum computer cannot produce a complete description of the state.

But given succinct descriptions of

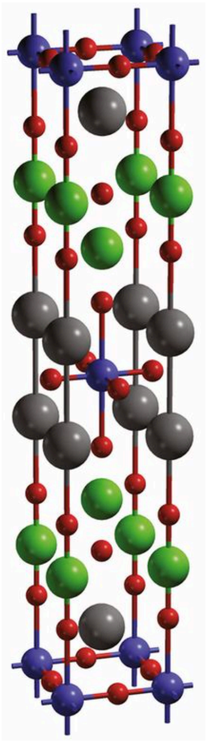
- the initial state (suitable for a quantum computer to prepare it efficiently) and
- a final measurement (say, measurements of the individual qubits in some basis),

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

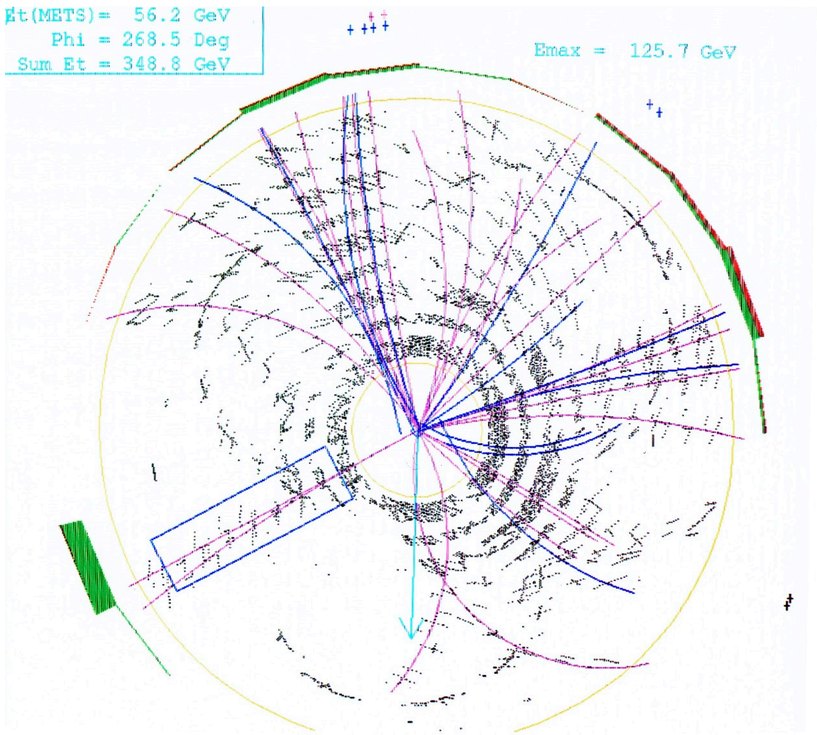
Computational quantum physics



quantum chemistry
(e.g., nitrogen fixation)

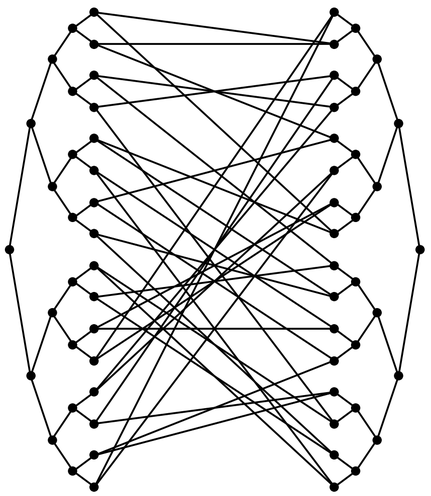


condensed matter physics/
properties of materials

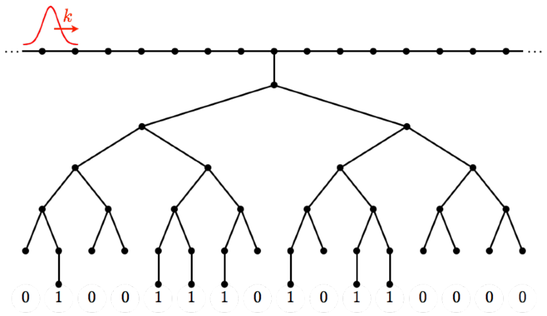


nuclear/particle
physics

Implementing quantum algorithms



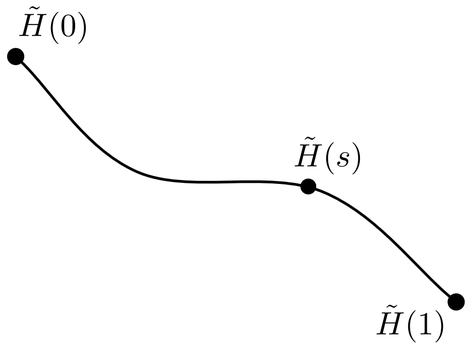
exponential speedup by quantum walk



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:

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

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

To ensure error at most ϵ, take $r = O\left((\|H\|t)^2/\epsilon\right)$ [Lloyd 96]

Gives simulation of d-sparse Hamiltonians with complexity $\mathrm{poly}(d)$ [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 ϵ 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

[Childs10; Berry, Childs 12]

High-precision simulation

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

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

This is the optimal dependence on ϵ

[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

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

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]

Quantum chemistry

Algorithms depend on many choices:

- Often assume nuclei at fixed positions (Born-Oppenheimer approximation)
- Choose a set of electron basis functions (molecular orbitals, plane waves, etc.)

$$H = \sum_{ij} h_{ij} a_i^\dagger a_j + \sum_{ijkl} g_{ijkl} a_i^\dagger a_j^\dagger a_k a_l$$

fermion operators: $\{a_i, a_j\} = 0, \ \{a_i, a_j^\dagger\} = \delta_{ij}$

- Convert to spins using a suitable transformation (Jordan-Wigner, Bravyi-Kitaev, etc.)

- Represent in first (locations of electrons) or second (occupation of modes) quantization

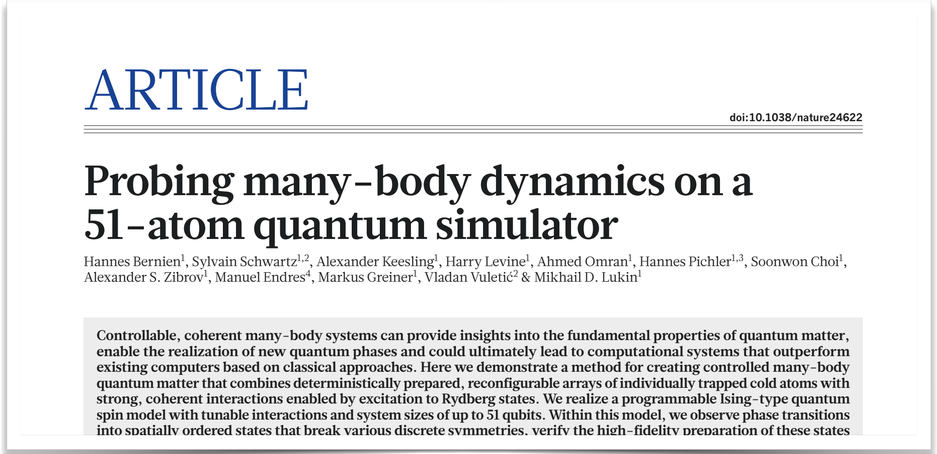
Selected asymptotic complexities (N modes, η electrons):

- [Wecker, Bauer, Clark, Hastings, Troyer 14] (2nd quantization, any basis): $O(N^{10})$
- [Babbush, Berry, Kivlichan, Wei, Love, Aspuru-Guzik 16] (2nd quantization, any basis): $O(N^5)$
- [Low, Wiebe 18] (2nd quantization, plane waves): $O(N^2)$
- [Babbush, Berry, McClean, Neven 18] (1st quantization, plane waves): $O(N^{1/3}\eta^{8/3})$

Analog simulation

Another approach: Construct a system that is described by the Hamiltonian you want to understand, and let it evolve!

Experimental efforts are further along than digital simulators



ARTICLE

doi:10.1038/nature24622

Probing many-body dynamics on a 51-atom quantum simulator

Hannes Bernien[1], Sylvain Schwartz[1,2], Alexander Keesling[1], Harry Levine[1], Ahmed Omran[1], Hannes Pichler[1,3], Soonwon Choi[1], Alexander S. Zibrov[1], Manuel Endres[4], Markus Greiner[1], Vladan Vuletić[2] & Mikhail D. Lukin[1]

Controllable, coherent many-body systems can provide insights into the fundamental properties of quantum matter, enable the realization of new quantum phases and could ultimately lead to computational systems that outperform existing computers based on classical approaches. Here we demonstrate a method for creating controlled many-body quantum matter that combines deterministically prepared, reconfigurable arrays of individually trapped cold atoms with strong, coherent interactions enabled by excitation to Rydberg states. We realize a programmable Ising-type quantum spin model with tunable interactions and system sizes of up to 51 qubits. Within this model, we observe phase transitions into spatially ordered states that break various discrete symmetries, verify the high-fidelity preparation of these states

Key questions:

- What kind of control is needed to realize Hamiltonians of interest?
- How can we be confident in the results?

Cubitt, Montanaro, Piddock 18: Universality result for spin models on lattices

Zhou, Aharonov 21: Universality that does not require spatial locality of the target Hamiltonian

3. 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 $\mathrm{poly}(\log N, 1/\epsilon, \kappa)$ 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 $\kappa\,\mathrm{poly}(\log(1/\epsilon))$ 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 $\frac{\mathrm{d}}{\mathrm{d}t}x = Ax + b$
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.
No compelling end-to-end application with rigorous evidence for speedup.

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 25+ 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

Lecture notes: cs.umd.edu/~amchilds/qa/

Montanaro survey: arXiv:1511.04206

András Gilyén tutorial (QIP 2020): www.koushare.com/video/videodetail/4073

My QIP 2021 tutorial: https://www.cs.umd.edu/~amchilds/talks/qip21.pdf
https://youtu.be/M0e5gkf7QSQ

Quantum walk surveys

- Santha (search): arXiv:0808.0059
- Reitzner, Nagaj, Buzek: arXiv:1207.7283