High-precision quantum algorithms

Andrew Childs

COMPUTER SCIENCE
UNIVERSITY OF MARYLAND

UMIACS
University of Maryland
Institute for Advanced
Computer Studies

Joint Center for
Quantum Information
and Computer Science

What can we do with a quantum computer?

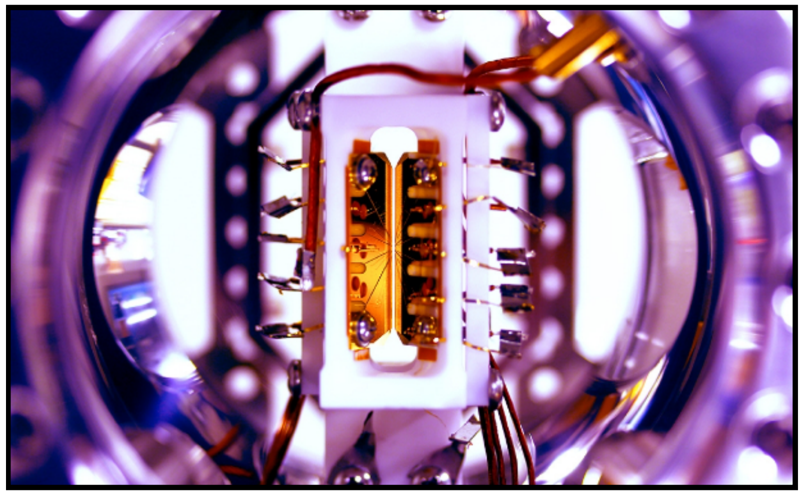
- Factoring
- Many problems with polynomial speedup (combinatorial search, collision finding, graph properties, Boolean formula evaluation, …)
- Simulating quantum mechanics
- Linear systems ($\rightarrow$ quantum machine learning?)
- And more: computing discrete logarithms, decomposing abelian groups, computing properties of algebraic number fields, approximating Gauss sums, counting points on algebraic curves, approximating topological invariants, finding isogenies between elliptic curves…

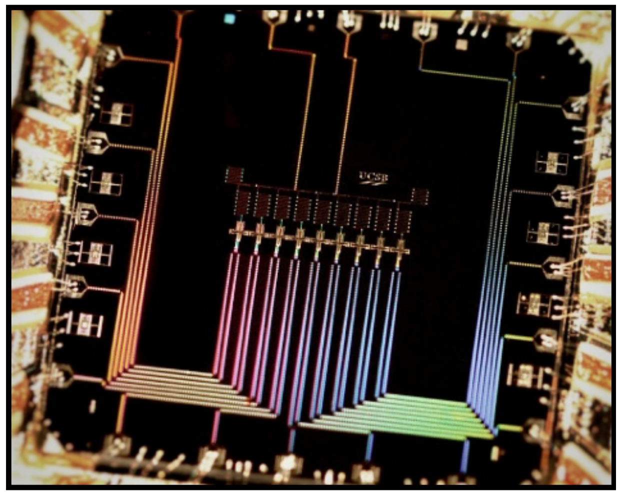
Quantum algorithm zoo: math.nist.gov/quantum/zoo/

When can I have one?

State of the art: well-characterized qubits with well-controlled interactions and long coherence times.

Leading candidate systems:


trapped ions


superconducting qubits

Several large experimental groups (Maryland, UCSB/Google, IBM, Delft/Intel, …) have serious efforts underway to consider scaling up to a larger device—a major engineering challenge!

Why else should you care?

- Studying the power of quantum computers addresses a basic scientific question: what can be computed efficiently in the real world?

- To design cryptosystems that are secure against quantum attacks, we have to understand what kinds of problems quantum computers can solve.

- Ideas from quantum information provide new tools for thinking about physics (e.g., the black hole information loss problem) and computer science (e.g., quantum algorithm for evaluating Boolean formulas → upper bound on polynomial threshold function degree → new subexponential (classical!) algorithms in computational learning theory)

My research

Quantum walk

- Example of exponential speedup (not based on Fourier transform)
- Algorithms for searching spatial regions
- Models for universal computation (single- & multi-particle)
- Nearly optimal algorithm for evaluating Boolean formulas

Algebraic problems

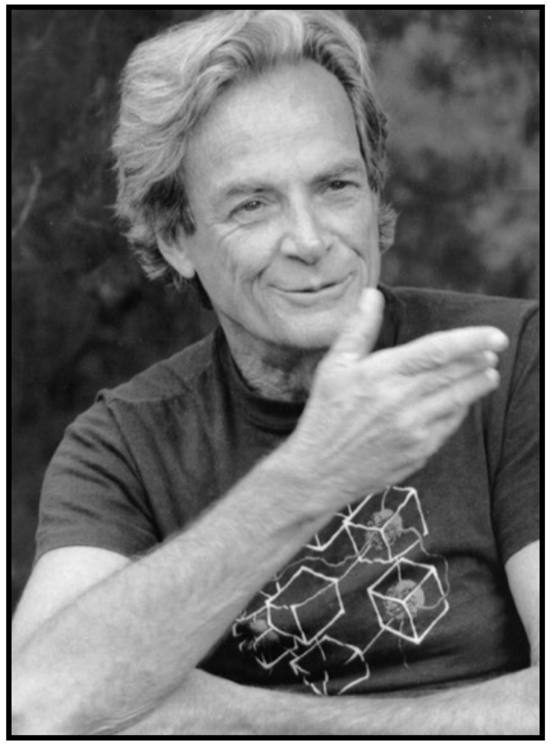
- Efficient algorithms for the hidden subgroup problem in some nonabelian groups (e.g., Heisenberg)
- Hidden nonlinear structures
- Constructing elliptic curve isogenies
- Discrete log in semigroups
- Polynomial interpolation

Quantum simulation

- Algorithms for simulating sparse Hamiltonians
- Simulation with complexity linear in the evolution time
- Simulation & linear systems with complexity logarithmic in the inverse error (this talk)

Plus…

- Quantum query complexity
- Quantum property testing
- Secure delegated quantum computation
- Hamiltonian complexity theory
- Computational power of nonlinear quantum dynamics
- Quantifying quantum nonlocality



"… 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
Simulating physics with computers (1981)

Why simulate quantum mechanics?

Computational chemistry/physics

- chemical reactions (e.g., nitrogen fixation)
- properties of materials

Implementing quantum algorithms

- continuous-time quantum walk (e.g., for formula evaluation)
- adiabatic quantum computation (e.g., for optimization)
- linear/differential equations

Quantum dynamics

The dynamics of a quantum system are determined by its *Hamiltonian*.

$$i\frac{\mathrm{d}}{\mathrm{d}t}|\psi(t)\rangle = H|\psi(t)\rangle$$
$$\Downarrow$$
$$|\psi(t)\rangle = e^{-iHt}|\psi(0)\rangle$$

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 by performing measurements on the state, it can answer questions that (apparently) a classical computer cannot

Local and sparse Hamiltonians

Local Hamiltonians [Lloyd 96]

$H = \sum_{j=1}^{m} H_j$ where each H_j acts on $k = O(1)$ qubits

Sparse Hamiltonians [Aharonov, Ta-Shma 03]

At most d nonzero entries per row, $d = \mathrm{poly}(\log N)$
(where H is $N \times N$)

In any given row, the location of the jth nonzero entry and its value can be computed efficiently (or is given by a black box)

Note: A k-local Hamiltonian with m terms is d-sparse with $d = 2^k m$

High-precision computing

Suppose we want to perform a computation that must be accurate to within ϵ. What is the cost as a function of ϵ?

Computing n digits of π: $O(n \operatorname{poly}(\log n))$

→ precision ϵ in $O(\log(1/\epsilon) \operatorname{poly}(\log\log(1/\epsilon))$

Boosting success probability of a randomized algorithm:

Suppose we can solve a decision problem with bounded success probability (say, 51%)

To get higher accuracy, repeat many times and take a majority vote

For error ϵ, need $O(\log(1/\epsilon))$ repetitions

Quantum circuit synthesis: cost of implementing a one-qubit gate with precision ϵ is $\operatorname{poly}(\log(1/\epsilon))$ [Solovay-Kitaev]

What about quantum simulation?

Product formula simulation

Suppose we want to simulate $H = \sum_{i=1}^{m} H_i$

Combine individual simulations with the Lie product formula:

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

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

To ensure error at most ϵ, take $r = O\big((\|H\|t)^2/\epsilon\big)$

[Lloyd 96]

High-order product formulas

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

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

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

$$\vdots$$

Systematic expansions to arbitrary order are known [Suzuki 92]

Using the kth order expansion, the number of exponentials required for an approximation with error at most ϵ is at most

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

[Berry, Ahokas, Cleve, Sanders 07]

High-precision simulation

We have recently developed a novel approach that directly implements the Taylor series of the evolution operator

New tools:

- Implementing linear combinations of unitary operations
- Oblivious amplitude amplification

Dependence on simulation error is $\mathrm{poly}(\log(1/\epsilon))$, an exponential improvement over previous work

Algorithms are also simpler, with less overhead

[Berry, Childs, Cleve, Kothari, Somma STOC 2014 & PRL 2015]

Linear combinations of unitaries

LCU Lemma: Given the ability to perform unitaries V_j with unit complexity, one can perform the operation $U = \sum_j \beta_j V_j$ with complexity $O(\sum_j |\beta_j|)$. Furthermore, if U is (nearly) unitary then this implementation can be made (nearly) deterministic.

Main ideas:

- Using controlled-V_j operations, implement U with some amplitude:

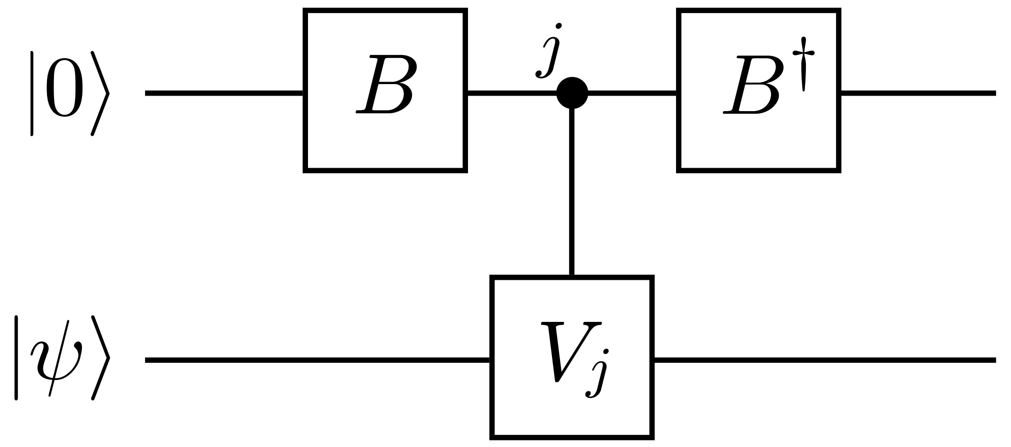
$$|0\rangle|\psi\rangle \mapsto \sin\theta|0\rangle U|\psi\rangle + \cos\theta|\Phi\rangle$$

- Boost the amplitude for success by oblivious amplitude amplification

Implementing U with some amplitude

$$U = \sum_j \beta_j V_j \qquad \text{(WLOG } \beta_j > 0\text{)}$$

Ancilla state: $B|0\rangle = \frac{1}{\sqrt{s}} \sum_j \sqrt{\beta_j}|j\rangle \qquad s := \sum_j \beta_j$



$$\frac{1}{s}|0\rangle\, U|\psi\rangle + \sqrt{1 - \frac{1}{s^2}}\,|\Phi\rangle$$

with $\langle 0|\Phi\rangle = 0$

Oblivious amplitude amplification

Suppose W implements U with amplitude $\sin\theta$:

$$W|0\rangle|\psi\rangle = \sin\theta|0\rangle U|\psi\rangle + \cos\theta|\Phi\rangle$$

To perform U with amplitude close to 1: use amplitude amplification?

But the input state is unknown!

Using ideas from [Marriott, Watrous 05], we can show that a $|\psi\rangle$-independent reflection suffices to do effective amplitude amplification.

With this *oblivious amplitude amplification*, we can perform the ideal evolution with only about $1/\sin\theta$ steps.

We also give a robust version that works even when U is not exactly unitary.

Simulating the Taylor series

Taylor series of the dynamics generated by H:

$$e^{-iHt} = \sum_{k=0}^{\infty} \frac{(-iHt)^k}{k!}$$

$$\approx \sum_{k=0}^{K} \frac{(-iHt)^k}{k!}$$

Write $H = \sum_\ell \alpha_\ell H_\ell$ where each H_ℓ is unitary

Then $e^{-iHt} \approx \sum_{k=0}^{K} \sum_{\ell_1,\ldots,l_k} \frac{(-it)^k}{k!} \alpha_{\ell_1} \cdots \alpha_{\ell_k} H_{\ell_1} \cdots H_{\ell_k}$

is a linear combination of unitaries

Why $\mathrm{poly}(\log(1/\epsilon))$?

Lowest-order product formula:

$$(e^{-iA/r}e^{-iB/r})^r = e^{-i(A+B)} + O(1/r)$$

so we must take $r = O(1/\epsilon)$ to achieve error at most ϵ

Higher-order formulas exist, but they only improve the power of ϵ

The approximation $e^{-iHt} \approx \sum_{k=0}^{K} \frac{(-iHt)^k}{k!}$ has error ϵ provided

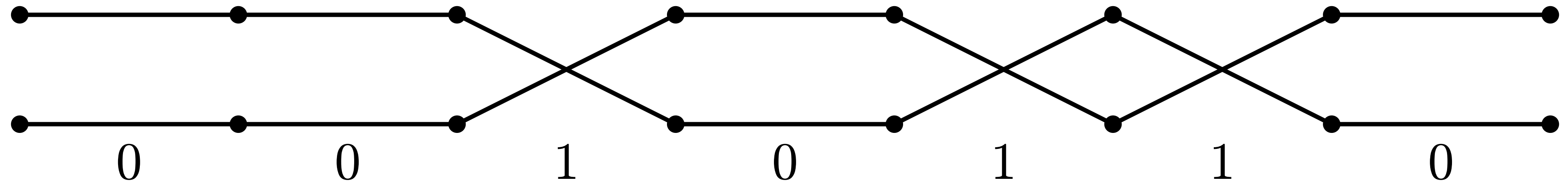
$$K = O\left(\frac{\log(1/\epsilon)}{\log\log(1/\epsilon)}\right)$$

Lower bounds

No-fast-forwarding theorem [BACS 07]: $\Omega(t)$

Main idea:

- Query complexity of computing the parity of n bits is $\Omega(n)$.
- There is a Hamiltonian that can compute parity by running for time $O(n)$.



New lower bound: $\Omega(\frac{\log(1/\epsilon)}{\log\log(1/\epsilon)})$

Main idea:

- Query complexity of parity is $\Omega(n)$ even for *unbounded error*.
- The same Hamiltonian as above computes parity with unbounded error by running for any positive time. Running for constant time gives the parity with probability $\Theta(1/n!)$.

Linear combination of quantum walk steps

Suppose the Hamiltonian is d-sparse.

Query complexity of the Taylor series approach is quadratic in d.

Another approach:
- Define a quantum walk related to the Hamiltonian
- Express the evolution operator as a linear combination of walk steps
- Implement this with the LCU Lemma

Query complexity: $O\left(\tau \frac{\log(\tau/\epsilon)}{\log\log(\tau/\epsilon)}\right)$

$$\tau := d\|H\|_{\max} t$$

[Berry, Childs, Kothari FOCS 2015]

Tradeoff between t and ϵ

Combining known lower bounds on the complexity of simulation as a function of t and ϵ gives

$$\Omega\left(t + \frac{\log \frac{1}{\epsilon}}{\log\log \frac{1}{\epsilon}}\right) \quad \text{vs. upper bound of} \quad O\left(t\,\frac{\log \frac{t}{\epsilon}}{\log\log \frac{t}{\epsilon}}\right)$$

Very recent work [Low, Chuang 2016], using an alternative implementation of linear combinations of quantum walk steps, gives an optimal tradeoff.

Quantum algorithm for linear systems

Consider an $N \times N$ linear system $Ax = b$.

Classical (or quantum!) algorithms need time $\mathrm{poly}(N)$ to determine x.

What if we change the model?

- A is sparse (at most $\mathrm{poly}(\log N)$ nonzeros in any row or column)
- We have a black box that specifies the nonzero entries in any given row or column
- Can efficiently prepare a quantum state $|b\rangle$

Then we can prepare a quantum state ϵ-close to $|x\rangle \propto A^{-1}|b\rangle$ in time $\mathrm{poly}(\log N, 1/\epsilon)$ [Harrow, Hassidim, Lloyd 09].

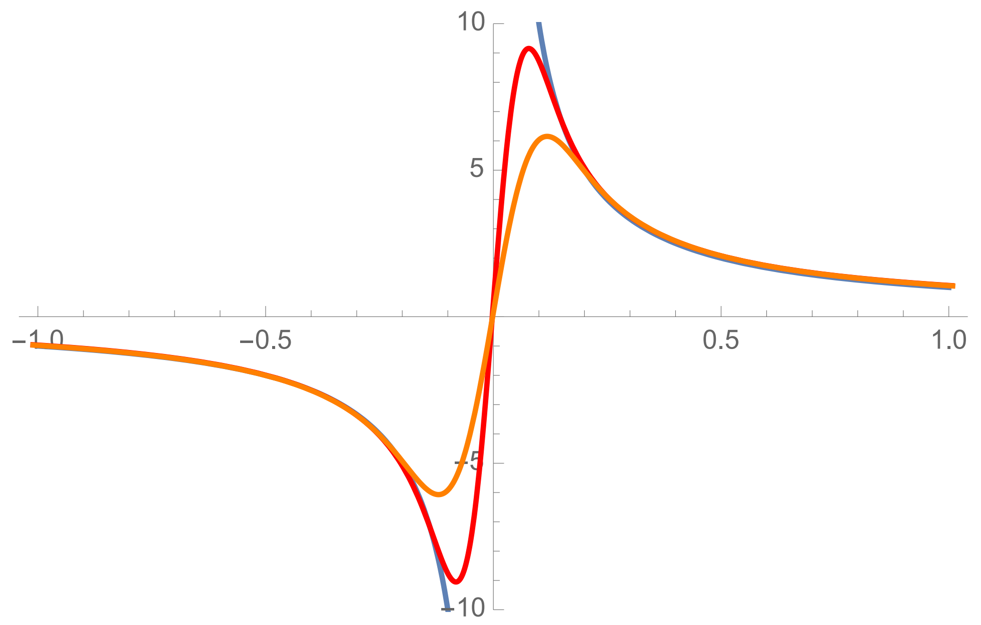
Possible applications: radar scattering cross sections [Clader, Jacobs, Sprouse 13], effective resistance [Wang 13], machine learning (?), …

High-precision quantum linear systems

We give an improved quantum algorithm for linear systems with running time $\mathrm{poly}(\log N, \log(1/\epsilon))$, an exponential improvement.

Main idea: Write $\frac{1}{A} \approx \sum_t c_t e^{-iAt}$ and use the LCU Lemma.

(or an analogous Chebyshev expansion, using a quantum walk related to A)



only need a good approximation over $[-1,-\frac{1}{\kappa}] \cup [\frac{1}{\kappa},1]$

[Childs, Kothari, Somma 2016]

Applications

Faster simulations of quantum mechanics (quantum chemistry, condensed matter, quantum field theory…)

Faster algorithms for scattering cross sections, effective resistance, machine learning (?), …

Quantum circuit synthesis: “Repeat-Until-Success” decompositions of quantum gates using oblivious amplitude amplification [Paetznick, Svore 14], [Wiebe, Roetteler 14]

Complexity theory: $\mathrm{PreciseQMA} = \mathrm{PSPACE}$ [Fefferman, Lin 16]

Ongoing work

Quantum simulation on a small quantum computer
[with Dmitri Maslov, Yunseong Nam, Julien Ross, Yuan Su]

What is the smallest instance of a practical quantum computation that outperforms classical computers?

Simulating open quantum systems
[with Tongyang Li]

Can we efficiently simulate noisy quantum systems?

Quantum algorithms for differential equations
[with Dominic Berry, Aaron Ostrander, Guoming Wang]

A quantum computers can prepare a state proportional to the solution of linear differential equation. Can we do this with complexity $\mathrm{poly}(\log(1/\epsilon))$?

Thank you!

CIFAR
CANADIAN INSTITUTE FOR ADVANCED RESEARCH

NIST
National Institute of Standards and Technology
U.S. Department of Commerce