Simulating Hamiltonian dynamics on a quantum computer

Andrew Childs

MIT Center for Theoretical Physics

27 June 2003

Why quantum computing?

Information is physical.

Physics is quantum mechanical.

In particular: Physical systems are described by vectors in Hilbert space that evolve by unitary transformations and can be measured by projection onto orthogonal subspaces.

The quantum circuit model

- Start in the state j0i
- Apply a sequence of unitary transformations U_1, U_2, …, U_k chosen from a universal gate set, e.g. $\{H, T, \mathrm{CNOT}\}$
- Measure in the computational basis

But time is not really discrete!

Hamiltonian dynamics

Quantum systems evolve according to the Schrödinger equation:

$$\boxed{i\frac{\mathrm{d}}{\mathrm{d}t}|\psi(t)\rangle = H(t)|\psi(t)\rangle}$$

H(*t*) is the **Hamiltonian**.

$H=H^\dagger$ so that the time evolution is *unitary*:

Solution of Schrödinger equation

$$\boxed{i\frac{\mathrm{d}}{\mathrm{d}t}|\psi(t)\rangle = H|\psi(t)\rangle}$$

time independent

Eigenstates of H: $H|\phi_j\rangle = E_j|\phi_j\rangle$

$$H = \sum_j E_j |\phi_j\rangle\langle\phi_j| \qquad E_j \in \mathbb{R} \text{ since } H=H^\dagger$$

Suppose $|\psi(0)\rangle = |\phi_j\rangle$

Then $|\psi(t)\rangle = \exp(-i\,E_j\,t)\,|\phi_j\rangle$

Expand a general initial state: $|\psi(0)\rangle = \sum_j c_j\,|\phi_j\rangle$

$$|\psi(t)\rangle = \sum_j c_j \exp(-i\,E_j\,t)\,|\phi_j\rangle = U(t)\,|\psi(0)\rangle$$

where $U(t) = \exp(-i\,H\,t) = \sum_j (-i\,H\,t)^j / j!$

Solution of Schrödinger equation

Time dependent case

$$|\psi(t)\rangle = U(t)\,|\psi(0)\rangle$$

where $U(t) = \mathrm{T}\exp[-i \int_0^t d\tau\, H(\tau)]$

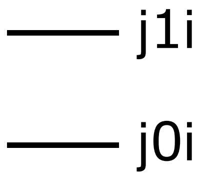
time ordering operator

Special case: Suppose $H(t)$ changes very slowly. Then the evolution is much simpler because of the *Adiabatic Theorem*. More on this later.

Relation to the circuit model

Unitary quantum gates arise from Hamiltonian dynamics.

Simple example: Two level atom.



Apply a laser pulse. $H = \omega \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}$

But perhaps we can use *many-body* Hamiltonians to perform interesting computations.

Why consider Hamiltonians?

Simulating physical systems

New kinds of quantum algorithms

- Quantum walks
- Adiabatic quantum computation

Quantum analogue of the Cook-Levin Theorem (Kitaev: “LOCAL HAMILTONIAN is QMA-complete”)

Efficiently realizable Hamiltonians

How do we know what Hamiltonians are legal for use in computations?

Example:

Let x = instance of a hard problem

Let s_x = solution of x

The Hamiltonian

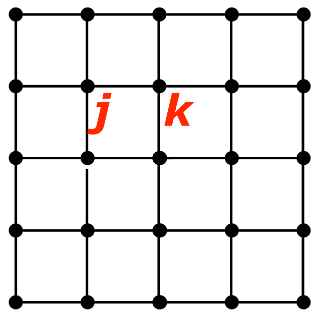
seems hard to implement!

Efficiently realizable Hamiltonians

Two notions of efficient realizability:

1. *H* is the Hamiltonian of a physical system we can “easily” build

   Spins arranged on a 2D lattice



where $a_{jk}=0$ for non-adjacent sites

Efficiently realizable Hamiltonians

Two notions of efficient realizability:

2. H is a Hamiltonian that can be efficiently simulated in the circuit model

Def. A Hamiltonian H acting on n qubits can be *efficiently simulated* if for any $\epsilon>0$, $t>0$ there is a quantum circuit U consisting of poly(n,t,$1/\epsilon$) gates such that $\|U - e^{-iHt}\|<\epsilon$.

Note: This will include the “physically reasonable” Hamiltonians.

Tools for simulating Hamiltonians

Rule 1. Local Hamiltonians.

If H acts on $O(1)$ qubits, it can be efficiently simulated.

Rule 2. Rescaling.

If H can be efficiently simulated, then cH can be efficiently simulated for any c=poly(n).

Tools for simulating Hamiltonians

Rule 3. Linear combination.

If H_j can be efficiently simulated, then $\sum_j H_j$ can be efficiently simulated.

Lemma. Lie product formula.

Let h=max$_j$ kH_jk. Then

Lie product formula

Proof.

By Taylor expansion,

Thus

$$(e^{-iH_1 t/r} \dots e^{-iH_k t/r})^r$$

$\square$

The story so far

Using Rules 1,2,3 we can simulate many "physical" Hamiltonians.

H = sum of terms, each acting on a constant number of qubits

Recall example of a spin glass:

Lloyd 96

Tools for simulating Hamiltonians

Rule 4. Commutation.

If H_1, H_2 can be efficiently simulated, then $i[H_1,H_2]$ can be efficiently simulated.

Rule 5. Unitary conjugation.

If H can be efficiently simulated and the unitary operation U can be efficiently implemented, then $U^\dagger HU$ can be efficiently simulated.

Proof. $e^{-iU^\dagger HUt} = U^\dagger e^{-iHt}U$ $\square$

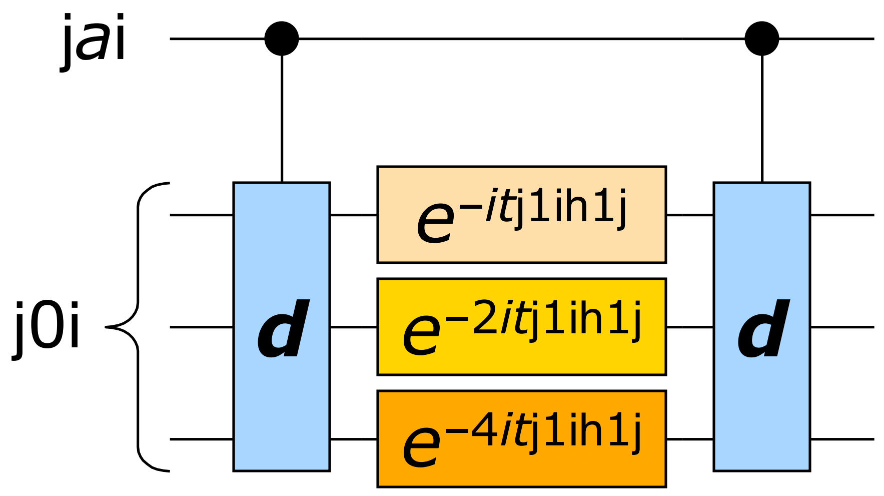
Tools for simulating Hamiltonians

Rule 6. Computable phase shifts.

If *H* is diagonal and the diagonal element *d*(*a*)=haj*H*jai can be efficiently computed for any *a*, then *H* can be efficiently simulated.

Proof.

$$
\begin{aligned}
|a,0\rangle &\to |a,d(a)\rangle \\
&\to e^{-itd(a)}|a,d(a)\rangle \\
&\to e^{-itd(a)}|a,0\rangle
\end{aligned}
$$



□

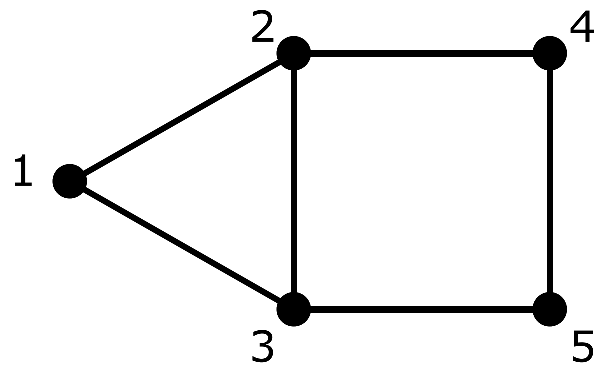
Tools for simulating Hamiltonians

Rule 7. Sparse Hamiltonians.

Suppose that for any *a*, one can efficiently compute all the values of *b* for which $\langle a|H|b\rangle$ is nonzero. Then *H* can be efficiently simulated.

Example:

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



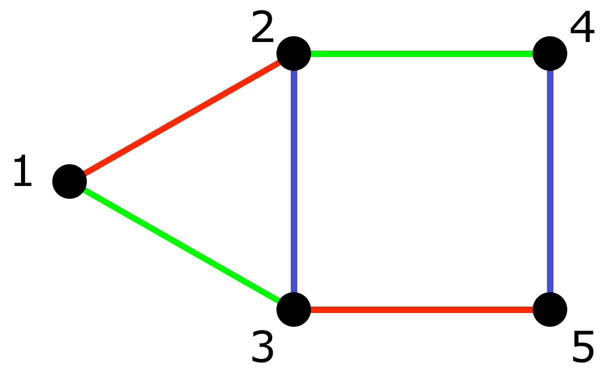
Tools for simulating Hamiltonians

Rule 7. Sparse Hamiltonians.

Suppose that for any *a*, one can efficiently compute all the values of *b* for which h*a*j*H*j*b*i is nonzero. Then *H* can be efficiently simulated.

Example:

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



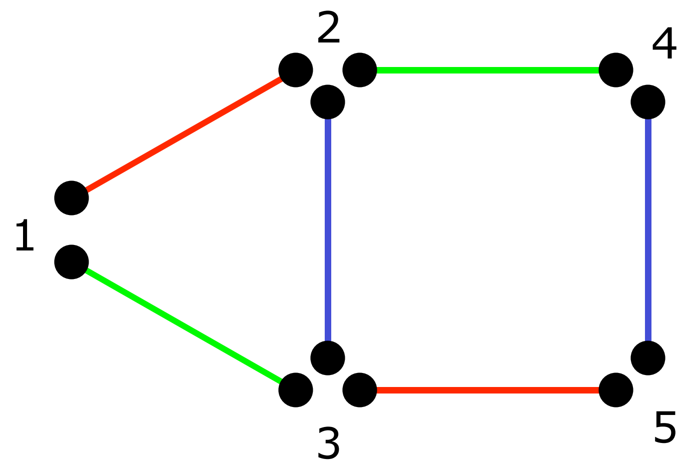
Tools for simulating Hamiltonians

Rule 7. Sparse Hamiltonians.

Suppose that for any *a*, one can efficiently compute all the values of *b* for which h*a*j*H*j*b*i is nonzero. Then *H* can be efficiently simulated.

Example:

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



Coloring a sparse graph

Lemma.

Given an undirected graph G with N vertices and maximum degree d, suppose one can efficiently compute the neighbors of a given vertex. Then there is an efficiently computable function $c(a,b)=c(b,a)$ taking $O(d^2 \log^2 N)$ values such that for all a, $c(a,b)=\mathrm{c}(a,b')$ implies $b=b'$.

Aharonov, Ta-Shma 03

Coloring a sparse graph

Proof.

Let index(a,b) be the index of b in the list of neighbors of a.

Let $k(a,b)$ be the smallest k such that $a \neq b \pmod{k}$.
Note $k(a,b)=k(b,a)$ and $k=O(\log N)$.

For $a<b$, define

$c(a,b) := \big(\boxed{\text{index}(a,b)},\ \text{index}(b,a),\ k(a,b),\ b \bmod k(a,b)\big)$

For $a>b$, define $c(a,b) := c(b,a)$.

Suppose $c(a,b)=c(a,b')$. Four cases:

i. $a<b$, $a<b'$. index(a,b)=index(a,b')) $b=b'$.

Coloring a sparse graph

Proof.

Let index(a,b) be the index of b in the list of neighbors of a.

Let $k(a,b)$ be the smallest k such that $a \neq b$ (mod k).
Note $k(a,b)=k(b,a)$ and $k=O(\log N)$.

For $a<b$, define

$$c(a,b) := \big(\text{index}(a,b), \text{index}(b,a), k(a,b), b \bmod k(a,b)\big)$$

For $a>b$, define $c(a,b) := c(b,a)$.

Suppose $c(a,b)=c(a,b')$. Four cases:

i. $a<b$, $a<b'$. index(a,b)=index(a,b')) $b=b'$.

ii. $a>b$, $a>b'$. index(a,b)=index(a,b')) $b=b'$.

Coloring a sparse graph

Proof.

Let index(a,b) be the index of b in the list of neighbors of a.

Let $k(a,b)$ be the smallest k such that $a \neq b \pmod{k}$.
Note $k(a,b)=k(b,a)$ and $k=O(\log N)$.

For $a<b$, define
$c(a,b) :=$ (index(a,b), index(b,a), $k(a,b)$, $b \bmod k(a,b)$)
For $a>b$, define $c(a,b) := c(b,a)$.

Suppose $c(a,b)=c(a,b')$. Four cases:

i. $a<b$, $a<b'$. index(a,b)=index(a,b')) $b=b'$.

ii. $a>b$, $a>b'$. index(a,b)=index(a,b')) $b=b'$.

iii. $a<b$, $a>b'$. $k(a,b)=k(a,b')$, $a=b \bmod k$, contradiction.

Coloring a sparse graph

Proof.

Let index(a,b) be the index of b in the list of neighbors of a.

Let $k(a,b)$ be the smallest k such that $a \neq b$ (mod k).
Note $k(a,b)=k(b,a)$ and $k=O(\log N)$.

For $a<b$, define
$c(a,b)$:= (index(a,b), index(b,a), $k(a,b)$, b mod $k(a,b)$)
For $a>b$, define $c(a,b)$:= $c(b,a)$.

Suppose $c(a,b)=c(a,b')$. Four cases:

i. $a<b$, $a<b'$. index(a,b)=index(a,b')) $b=b'$.

ii. $a>b$, $a>b'$. index(a,b)=index(a,b')) $b=b'$.

iii. $a<b$, $a>b'$. $k(a,b)=k(a,b')$, $a=b$ mod k, contradiction.

iv. $a>b$, $a<b'$. $k(a,b)=k(a,b')$, $a=b'$ mod k, contradiction.

Coloring a sparse graph

Proof.

Let index(a,b) be the index of b in the list of neighbors of a.

Let $k(a,b)$ be the smallest k such that $a \neq b$ (mod k).
Note $k(a,b)=k(b,a)$ and $k=O(\log N)$.

For $a<b$, define
$$c(a,b) := \big(\text{index}(a,b),\ \text{index}(b,a),\ k(a,b),\ b \bmod k(a,b)\big)$$
For $a>b$, define $c(a,b) := c(b,a)$.

Suppose $c(a,b)=c(a,b')$. Four cases:

- **i.** $a<b$, $a<b'$. index(a,b)=index(a,b')) $b=b'$.
- **ii.** $a>b$, $a>b'$. index(a,b)=index(a,b')) $b=b'$.
- **iii.** $a<b$, $a>b'$. $k(a,b)=k(a,b')$, $a=b$ mod k, contradiction.
- **iv.** $a>b$, $a<b'$. $k(a,b)=k(a,b')$, $a=b'$ mod k, contradiction. □

Simulating a sparse Hamiltonian

Proof. (of Rule 7)

Write H as a diagonal matrix plus a matrix with zeros on the diagonal. The diagonal part can be simulated using Rule 6 and combined with the off-diagonal piece using Rule 3. Thus assume H has zeros on the diagonal WLOG.

Let $v_c(a)$ be the vertex connected to a by an edge of color c.

Let $x_c(a) := \mathrm{Re}\,\langle a|H|v_c(a)\rangle$, $y_c(a) := \mathrm{Im}\,\langle a|H|v_c(a)\rangle$.

Consider the state space $|a,b,z\rangle$ where vertices are $|a,0,0\rangle$.

We can efficiently implement unitary operators

$$V_c\,|a,0,0\rangle = |a, v_c(a), x_c(a)\rangle$$
$$W_c\,|a,0,0\rangle = |a, v_c(a), y_c(a)\rangle.$$

(If no such vertex, $v_c(a)=11\ldots1$, $x_c(a)=y_c(a)=0$.)

We can efficiently simulate the Hamiltonians

$$S\,|a,b,x\rangle = x\,|b,a,x\rangle$$
$$T\,|a,b,y\rangle = i\,y\,|b,a,-y\rangle$$

using Rules 1,5,6.

CCDFGS 02

Simulating a sparse Hamiltonian

Proof. (of Rule 7, continued)

Using Rules 3,5 we can efficiently simulate

This has the proper action on vertices:

$$\begin{aligned}\tilde{H}|a,0,0\rangle &= \sum_c [V_c^\dagger S|a, v_c(a), x_c(a)\rangle \\ &\quad + W_c^\dagger T|a, v_c(a), y_c(a)\rangle] \\ &= \sum_c [x_c(a)\, V_c^\dagger |v_c(a), a, x_c(a)\rangle \\ &\quad + i y_c(a)\, W_c^\dagger |a, v_c(a), -y_c(a)\rangle] \\ &= \sum_c [x_c(a) + i y_c(a)]|v_c(a), 0, 0\rangle \qquad \square\end{aligned}$$

Example: Particle in a potential

Consider the Hamiltonian

Lattice version (lattice spacing *l*):

p^2 is diagonal in the Fourier basis, and the unitary operator corresponding to the Fourier transform is efficiently implementable, so H can be simulated using Rules 3,5,6.

Alternatively, just note that H is sparse and computable, so Rule 7 applies.

Wiesner 96, Zalka 98

Summary

Quantum systems evolve according to the Schrödinger equation $i\frac{\mathrm{d}}{\mathrm{d}t}|\psi(t)\rangle = H(t)|\psi(t)\rangle$

Such systems can be efficiently simulated by a universal quantum computer when *H*

- Is a sum of terms, each acting on at most a constant number of qubits (Rule 1,2,3)
- Is *i* times the commutator of two simulable Hamiltonians (Rule 4)
- Differs from a simulable Hamiltonian by an efficiently implementable unitary transformation (Rule 5)
- Is sparse and efficiently computable (Rules 6,7)

References

Hamiltonian dynamics

- D. J. Griffiths, *Introduction to Quantum Mechanics* (Prentice Hall, 1994).
- R. Liboff, *Introductory Quantum Mechanics* (Addison Wesley, 1998).

Simulating physical systems

- R. Feynman, *Simulating physics with computers*, Int. J. Theor. Phys. **21**, 219 (1982).
- S. Lloyd, *Universal quantum simulators*, Science **273**, 1073 (1996).
- S. Wiesner, *Simulations of many-body quantum systems by a quantum computer*, quant-ph/9603028.
- C. Zalka, *Simulating quantum systems on a quantum computer*, Proc. R. Soc. London A **454**, 313 (1998).
- M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information* (Cambridge University Press, 2000).

Time-independent Hamiltonian dynamics allow universal quantum computation

- R. P. Feynman, *Quantum mechanical computers*, Optics News **11**, 11 (1985).

References

Simulating sparse Hamiltonians

- A. M. Childs, R. Cleve, E. Deotto, E. Farhi, S. Gutmann, and D. Spielman, *Exponential algorithmic speedup by quantum walk*, quant-ph/0209131, STOC 2003.
- D. Aharonov and A. Ta-Shma, *Adiabatic quantum state generation and statistical zero knowledge*, quant-ph/0301040, STOC 2003.

Measuring a Hermitian operator

- C. Zalka, *Simulating quantum systems on a quantum computer*, Proc. R. Soc. London A **454**, 313 (1998).
- D. S. Abrams and S. Lloyd, *A quantum algorithm providing exponential speed increase for finding eigenvalues and eigenvectors*, quant-ph/9807070, Phys. Rev. Lett. **83**, 5162 (1999).
- A. M. Childs, E. Deotto, E. Farhi, J. Goldstone, S. Gutmann, and A. Landahl, Quantum search by measurement, quant-ph/0204013, Phys. Rev. A **66**, 032314 (2002).

Hamiltonian dynamics in quantum algorithms:
See next lecture