Exponential algorithmic speedup by quantum walk

Andrew Childs  
MIT Center for Theoretical Physics  

joint work with  
Richard Cleve  
Enrico Deotto  
Eddie Farhi  
Sam Gutmann  
Dan Spielman  

quant-ph/0209131
Motivation

• Find new kinds of quantum algorithms

• Classical random walks → quantum walks
• Construct an (oracular) problem that is naturally suited to quantum walks
• Show that the problem can be solved efficiently using quantum walks
  – Walk finds the solution fast
  – Walk can be implemented
• Show that the problem cannot be solved efficiently using a classical computer
Black box computation

• **Standard computation:** compute a function of some data
  Example: Factoring.
  Input: An integer $j$
  Output: Integers $k, l$ such that $j = kl$

• **Black box computation** (oracular computation)
  Input: A black box for a function $f(x)$
  Output: Some property of $f(x)$

Running time: count queries to $f(x)$.
Easier to obtain bounds.
Black box computation

(Physical perspective)

- Physics experiment
  
  Input: Apparatus with Hamiltonian $H$
  
  Output: Parameters $c_1, c_2, \ldots$

  $$|\psi_{\text{in}}\rangle \xrightarrow{H} |\psi_{\text{out}}\rangle = e^{-iHt} |\psi_{\text{in}}\rangle$$

  Determine $c_1, c_2, \ldots$ as fast as possible.
History of quantum algorithms

Deutsch 85
One quantum query vs. two classical queries

Deutsch/Josza 92
Exact quantum solution exponentially faster than exact classical solution

Bernstein/Vazirani 93
Superpolynomial quantum-classical separation

Simon 94
Exponential quantum-classical separation

Shor 94
Factoring/discrete log

Kitaev 95
Abelian hidden subgroup problem

van Dam/Hallgren 00
Quadratic character problems

Watrous 01
Algorithms for solvable groups

Hallgren 02
Pell’s equation

Grover 96
Quadratic speedup of search

CCDFGS 02
Quantum walks
Quantum walk

Classical random walk

Differential equation
\[ \frac{dp_a(t)}{dt} = \sum_{a'} K_{aa'} p_{a'}(t) \]

Generator
\[ K_{aa'} = \begin{cases} 
\gamma & a \neq a', \; aa' \in G \\
0 & a \neq a', \; aa' \notin G \\
-d(a)\gamma & a = a'. \end{cases} \]

Probability conservation
\[ \frac{d}{dt} \sum_a p_a(t) = 0 \]

Quantum walk

Differential equation (Schrödinger)
\[ i \frac{d}{dt} \langle a|\psi(t)\rangle = \sum_{a'} \langle a|H|a'\rangle \langle a'|\psi(t)\rangle \]

Generator
\[ \langle a|H|a'\rangle = K_{aa'} \]
\[ \langle a|H|a'\rangle = \begin{cases} 
\gamma & a \neq a', \; aa' \in G \\
0 & \text{otherwise.} \end{cases} \]

Probability conservation
\[ \frac{d}{dt} \sum_a |\langle a|\psi(t)\rangle|^2 = 0 \]

Black box graph traversal problem

Names of vertices: random $2n$-bit strings ($n = \lceil \log N \rceil$)
Name of ENTRANCE is known
Oracle outputs the names of adjacent vertices
$v_c(a) = c$th neighbor of $a$

Examples:
$v_1(\text{ENTRANCE}) = 0110101$ $v_1(0110101) = 1001101$
$v_2(\text{ENTRANCE}) = 1110100$ $v_2(0110101) = \text{ENTRANCE}$
$v_3(\text{ENTRANCE}) = 1111111$ $v_3(0110101) = 1110100$
$v_4(\text{ENTRANCE}) = 1111111$ $v_4(0110101) = 1111111$
Example: $G_n$

**Classical random walk**
Time to reach EXIT is exponential in $n$.

**Quantum walk**
Time to reach EXIT is linear in $n$.

But there is a (non-random walk) classical algorithm that finds EXIT in polynomial time!

$O(2^n)$ vertices

Childs, Farhi, Gutmann, QIP 1, 35 (2002).
A harder graph: $G_n'$

(Actually a distribution on graphs)
Connection: a random cycle that alternates sides
Reduction of $G_{n'}$ to a line

Column subspace

$$|\text{col } j\rangle = \frac{1}{\sqrt{N_j}} \sum_{a \in \text{column } j} |a\rangle$$

where

$$N_j = \begin{cases} 2^j & 0 \leq j \leq n \\ 2^{2n+1-j} & n + 1 \leq j \leq 2n + 1 \end{cases}$$

Reduced Hamiltonian

$$\langle \text{col } j|H|\text{col}(j + 1)\rangle = \begin{cases} \sqrt{2}\gamma & 0 \leq j \leq n - 1, \\ 2\gamma & j = n \end{cases}$$

$$\gamma = \frac{1}{\sqrt{2}}$$
Quantum walk on $G_{250}^j$
Implementing the quantum walk 1

General edge-colored graph

\[ v_c(a) = \text{neighbor of } a \text{ connected by an edge of color } c \]

\[ v_c(v_c(a)) = a \text{ for } v_c(a) \in G \]

Hilbert space: states \( |a, b, r\rangle \)

Vertex states: \( |a, 0, 0\rangle \)

Using the oracle, can compute

\[ V_c |a, b, r\rangle = |a, b \oplus v_c(a), r \oplus f_c(a)\rangle \]

\[ f_c(a) = \begin{cases} 
0 & v_c(a) \in G \\
1 & v_c(a) \notin G 
\end{cases} \]
Implementing the quantum walk 2

Tools for simulating Hamiltonians

• Linear combination
  \[ e^{-i(H_1 + \cdots + H_k) t} = \left( e^{-iH_1 t / j} \cdots e^{-iH_k t / j} \right)^j + O(k \| [H_p, H_q] \| t^2 / j) \]

• Unitary conjugation
  \[ U e^{-iHt} U^\dagger = e^{-iUHU^\dagger t} \]

A simple Hamiltonian

\[ T |a, b, 0\rangle = |b, a, 0\rangle \]
\[ T |a, b, 1\rangle = 0 \]

Graph Hamiltonian

\[ H = \sum_c V_c^\dagger TV_c \]

Proof:

\[ H |a, 0, 0\rangle = \sum_c V_c T |a, v_c(a), f_c(a)\rangle \]
\[ = \sum_c \delta_{0, f_c(a)} V_c |v_c(a), a, 0\rangle \]
\[ = \sum_{c: v_c(a) \in G} |v_c(a), a \oplus v_c(v_c(a)), f_c(v_c(a))\rangle \]
\[ = \sum_{c: v_c(a) \in G} |v_c(a), 0, 0\rangle \]
Simulating $T$

Simulate

\[ T|a, b, 0\rangle = |b, a, 0\rangle \]
\[ T|a, b, 1\rangle = 0 \]

i.e.,

\[ T = \left( \bigotimes_{l=1}^{2n} S(l,2n+l) \right) \otimes |0\rangle\langle 0| \]

SWAP:

\[ S|z_1 z_2\rangle = |z_2 z_1\rangle \]

Diagonalize:

\[ W|00\rangle = |00\rangle \]
\[ W \frac{1}{\sqrt{2}} (|01\rangle + |10\rangle) = |01\rangle \]
\[ W \frac{1}{\sqrt{2}} (|01\rangle - |10\rangle) = |10\rangle \]
\[ W|11\rangle = |11\rangle \]
Coloring $G_n'$

- General oracle: $v_c(v_c(a)) \neq a$
- Construct an oracle that has this property
- $G_n'$ is bipartite. Define a parity bit:
  0 if vertex is in an even column
  1 if vertex is in an odd column
- Parity of ENTRANCE is 0, and we can easily modify the oracle to keep track of parity
- Color of an edge depends on parity:
  - Parity 0 \[ c = (c_{\text{in}}, c_{\text{out}}) \]
  - Parity 1 \[ c = (c_{\text{out}}, c_{\text{in}}) \]
Quantum algorithm

- Start in the state $|\text{ENTRANCE}, 0, 0\rangle$
- Simulate the quantum walk for a time $t = \text{poly}(n)$
- Measure in the computational basis
- If \text{EXIT} is found, stop; otherwise repeat

- **Theorem:** If $t$ is chosen uniformly in $[0, n^4]$ then the probability of finding \text{EXIT} is greater than $1/4n$.

⇒ Quantum walk algorithm finds the \text{EXIT} with high probability using a polynomial number of gates and oracle queries.
Classical lower bound 1

Consider the set of vertices visited by the algorithm.

Restrict to a connected subgraph

- Number of vertices in $G'_n$: $O(2^n)$
- Number of possible names: $2^{2n}$
- Probability of guessing a valid name at random: $O(2^{-n})$

Restrict to a subtree

- Allow the algorithm to win if it finds the EXIT or if it finds a cycle
Classical lower bound 2

Restrict to random embeddings of rooted binary trees

• If the algorithm does not find a cycle, then it is simply tracing out a rooted subtree of $G_n'$

![Diagram of a binary tree with ENTRANCE at the top and a branch leading to an EXIT at the bottom]

• Consider an arbitrary rooted binary tree with $t$ vertices. What is the probability that a random embedding into $G_n'$ produces a cycle or finds the EXIT?
Classical lower bound 3

- Answer: if $t < 2^{n/6}$, then the probability is less than $3 \cdot 2^{-n/6}$.

- Putting it all together, we have

  **Theorem:** Any classical algorithm that makes at most $2^{n/6}$ queries to the oracle finds the EXIT with probability at most $4 \cdot 2^{-n/6}$. 
Remarks

- Provably exponential quantum-classical separation using quantum walks

- Q: Why does the algorithm work?
  A: Quantum interference!

- Find the **EXIT** without finding a path from **ENTRANCE** to **EXIT**

- Could put the coloring in the classical lower bound

- Easy to formulate as a decision problem
Open problems

• Is it possible to implement the walk for a general graph with no restriction on the initial state?

• Are there *interesting* computational problems that can be solved using quantum walks?