Quantum walk algorithms

Andrew Childs
Institute for Quantum Computing
University of Waterloo

28 September 2011

Randomized algorithms

Randomness is an important tool in computer science

Black-box problems

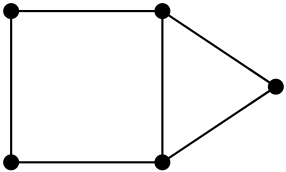
- Huge speedups are possible (Deutsch-Jozsa: $2^{\Omega(n)}$ vs. $O(1)$)
- Polynomial speedup for some total functions (game trees: $\Omega(n)$ vs. $O(n^{0.754})$)

Natural problems

- Majority view is that derandomization should be possible (P=BPP)
- Randomness may give polynomial speedups (Schöning algorithm for k-SAT)
- Can be useful for algorithm design

Random walk

Graph $G = (V, E)$



Two kinds of walks:

- Discrete time
- Continuous time

Random walk algorithms

Undirected s–t connectivity in log space

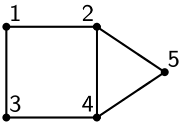
- Problem: given an undirected graph $G = (V, E)$ and $s, t \in V$, is there a path from s to t?
- A random walk from s eventually reaches t iff there is a path
- Taking a random walk only requires log space
- Can be derandomized (Reingold 2004), but this is nontrivial

Markov chain Monte Carlo

- Problem: sample from some probability distribution (uniform distribution over some set of combinatorial objects, thermal equilibrium state of a physical system, etc.)
- Create a Markov chain whose stationary distribution is the desired one
- Run the chain until it converges

Continuous-time quantum walk

Graph G



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

adjacency matrix Laplacian

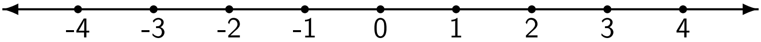
Random walk on G

- State: probability $p_v(t)$ of being at vertex v at time t
- Dynamics: $\frac{\mathrm{d}}{\mathrm{d}t}\vec{p}(t) = -L\vec{p}(t)$

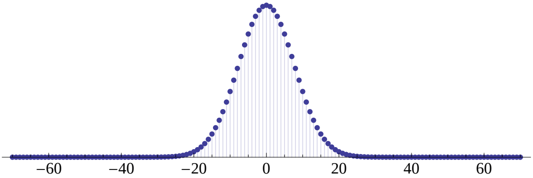
Quantum walk on G

- State: amplitude $q_v(t)$ to be at vertex v at time t (i.e., $|\psi(t)\rangle = \sum_{v \in V} q_v(t)|v\rangle$)
- Dynamics: $\mathrm{i}\frac{\mathrm{d}}{\mathrm{d}t}\vec{q}(t) = -L\vec{q}(t)$

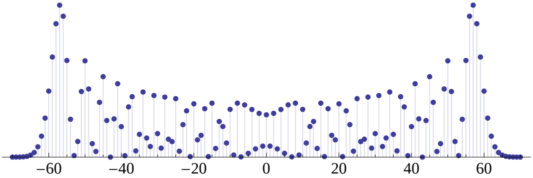
Random vs. quantum walk on the line



Classical:



Quantum:

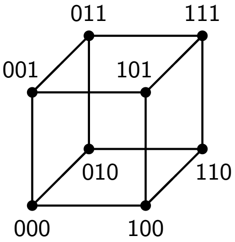


Random vs. quantum walk on the hypercube

$V = \{0,1\}^n$

$E = \{(x,y) \in V \times V :$
x and y differ in
exactly one bit$\}$

$n = 3$:

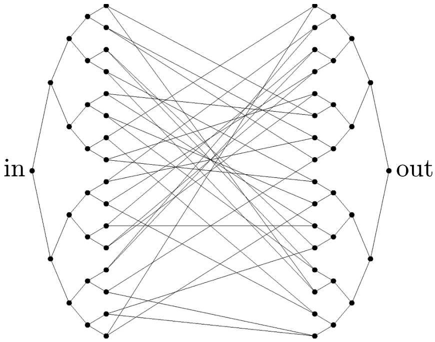


Classical random walk: reaching $11\ldots1$ from $00\ldots0$ is exponentially unlikely

Quantum walk: with $A = \sum_{j=1}^n X_j$,

$$e^{-\mathrm{i}At} = \prod_{j=1}^{n} e^{-\mathrm{i}X_j t} = \bigotimes_{j=1}^{n} \begin{pmatrix} \cos t & -\mathrm{i}\sin t \\ -\mathrm{i}\sin t & \cos t \end{pmatrix}$$

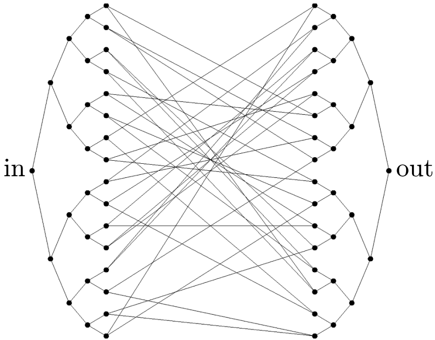
Glued trees problem



Black-box description of a graph

- Vertices have arbitrary labels
- Label of 'in' vertex is known
- Given a vertex label, black box returns labels of its neighbors
- Restricts algorithms to explore the graph locally

Glued trees problem: Classical query complexity

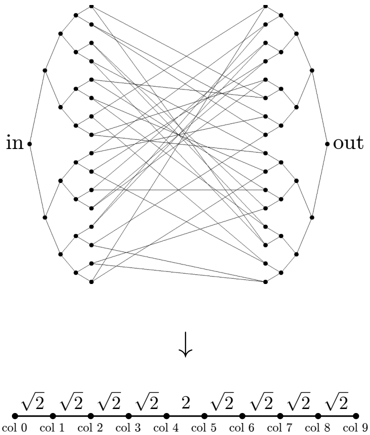


Let n denote the height of one of the binary trees

Classical random walk from 'in': probability of reaching 'out' is $2^{-\Omega(n)}$ at all times

In fact, the classical query complexity is $2^{\Omega(n)}$

Glued trees problem: Exponential speedup



Column subspace

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

$$N_j := \begin{cases} 2^j & \text{if } j \in [0, n] \\ 2^{2n+1-j} & \text{if } j \in [n+1, 2n+1] \end{cases}$$

Reduced adjacency matrix

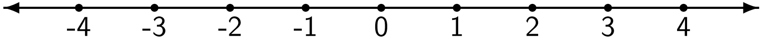
$$\langle \text{col } j | A | \text{col } j+1 \rangle = \begin{cases} \sqrt{2} & \text{if } j \in [0, n-1] \\ \sqrt{2} & \text{if } j \in [n+1, 2n] \\ 2 & \text{if } j = n \end{cases}$$

Discrete-time quantum walk: Need for a coin

Quantum analog of discrete-time random walk?

Unitary matrix $U \in \mathbb{C}^{|V| \times |V|}$ with $U_{vw} \neq 0$ iff $(v, w) \in E$

Consider the line:

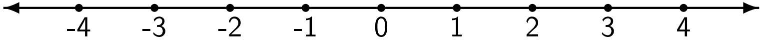


Define walk by $|x\rangle \mapsto \frac{1}{\sqrt{2}}(|x-1\rangle + |x+1\rangle)$?

But then $|x+2\rangle \mapsto \frac{1}{\sqrt{2}}(|x+1\rangle + |x+3\rangle)$, so this is not unitary!

In general, we must enlarge the state space.

Discrete-time quantum walk on a line

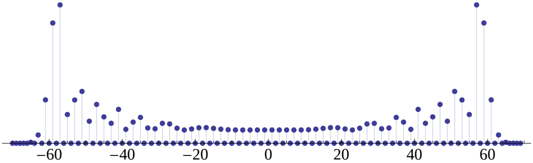


Add a “coin”: state space $\mathrm{span}\{|x\rangle \otimes |\leftarrow\rangle, |x\rangle \otimes |\rightarrow\rangle : x \in \mathbb{Z}\}$

Coin flip: $C := I \otimes H$

Shift: $$\begin{aligned} S|x\rangle \otimes |\leftarrow\rangle &= |x-1\rangle \otimes |\leftarrow\rangle \\ S|x\rangle \otimes |\rightarrow\rangle &= |x+1\rangle \otimes |\rightarrow\rangle \end{aligned}$$

Walk step: SC



The Szegedy walk

State space: $\mathrm{span}\{|v\rangle \otimes |w\rangle, |w\rangle \otimes |v\rangle : (v,w) \in E\}$

Let W be a stochastic matrix (a discrete-time random walk)

$$\text{Define} \quad |\psi_v\rangle := |v\rangle \otimes \sum_{w \in V} \sqrt{W_{wv}} |w\rangle \quad (\text{note } \langle \psi_v | \psi_w \rangle = \delta_{v,w})$$

$$R := 2 \sum_{v \in V} |\psi_v\rangle\langle\psi_v| - I$$

$$S(|v\rangle \otimes |w\rangle) := |w\rangle \otimes |v\rangle$$

Then a step of the walk is the unitary operator $U := SR$

Spectrum of the walk

Let $T := \sum_{v \in V} |\psi_v\rangle\langle v|$, so $R = 2TT^\dagger - I$.

Theorem (Szegedy)

Let W be a stochastic matrix. Suppose the matrix

$$\sum_{v,w} \sqrt{W_{vw} W_{wv}} |w\rangle\langle v|$$

has an eigenvector $|\lambda\rangle$ with eigenvalue λ. Then

$$\frac{I - e^{\pm \mathrm{i} \arccos \lambda} S}{\sqrt{2(1-\lambda^2)}} T|\lambda\rangle$$

are eigenvectors of $U = SR$ with eigenvalues

$$e^{\pm \mathrm{i} \arccos \lambda}.$$

Proof of Szegedy's spectral theorem

Proof sketch.
Straightforward calculations give

$$TT^\dagger = \sum_{v \in V} |\psi_v\rangle\langle\psi_v| \qquad T^\dagger T = I$$

$$T^\dagger S T = \sum_{v,w \in V} \sqrt{W_{vw} W_{wv}} |w\rangle\langle v| = \sum_{\lambda} |\lambda\rangle\langle\lambda|$$

which can be used to show

$$U(T|\lambda\rangle) = ST|\lambda\rangle \qquad U(ST|\lambda\rangle) = 2\lambda ST|\lambda\rangle - T|\lambda\rangle.$$

Diagonalizing within the subspace span$\{T|\lambda\rangle, ST|\lambda\rangle\}$ gives the desired result. $\square$

Exercise. Fill in the details

Random walk search algorithm

Given $G = (V, E)$, let $M \subset V$ be a set of *marked vertices*

Start at a random unmarked vertex

Walk until we reach a marked vertex:

$$W'_{vw} := \begin{cases} 1 & w \in M \text{ and } v = w \\ 0 & w \in M \text{ and } v \neq w \\ W_{vw} & w \notin M. \end{cases}$$

$$= \begin{pmatrix} W_M & 0 \\ V & I \end{pmatrix} \quad (W_M\text{: delete marked rows and columns of } W)$$

Question. How long does it take to reach a marked vertex?

Classical hitting time

Take t steps of the walk:

$$(W')^t = \begin{pmatrix} W_M^t & 0 \\ V(I + W_M + \cdots + W_M^{t-1}) & I \end{pmatrix}$$
$$= \begin{pmatrix} W_M^t & 0 \\ V\frac{I-W_M^t}{I-W_M} & I \end{pmatrix}$$

Convergence time depends on how close $\|W_M\|$ is to 1, which depends on the spectrum of W

Lemma

Let $W = W^T$ be a symmetric Markov chain. Let the second largest eigenvalue of W be $1 - \delta$, and let $\epsilon = |M|/|V|$ (the fraction of marked items). Then the probability of reaching a marked vertex is $\Omega(1)$ after $t = O(1/\delta\epsilon)$ steps of the walk.

Quantum walk search algorithm

Start from the state $\frac{1}{\sqrt{N-|M|}} \sum_{v \notin M} |\psi_v\rangle$

Consider the walk U corresponding to W':

$$\sum_{v,w \in V} \sqrt{W'_{v,w} W'_{w,v}} |w\rangle\langle v| = \begin{pmatrix} W_M & 0 \\ 0 & I \end{pmatrix}$$

Eigenvalues of U are $e^{\pm \mathrm{i} \arccos \lambda}$ where the λ are eigenvalues of W_M

Perform phase estimation on U with precision $O(\sqrt{\delta\epsilon})$

- no marked items $\implies$ estimated phase is 0
- ϵ fraction of marked items $\implies$ nonzero phase with probability $\Omega(1)$

Further refinements give algorithms for *finding* a marked item

Grover's algorithm revisited

Problem

Given a black box $f : X \to \{0,1\}$, is there an x with $f(x) = 1$?

Markov chain on $N = |X|$ vertices:

$$W := \frac{1}{N}\begin{pmatrix} 1 & \cdots & 1 \\ \vdots & \ddots & \vdots \\ 1 & \cdots & 1 \end{pmatrix} = |\psi\rangle\langle\psi|, \quad |\psi\rangle := \frac{1}{\sqrt{N}}\sum_{x\in X}|x\rangle$$

Eigenvalues of W are $0, 1 \implies \delta = 1$

Hard case: one marked vertex, $\epsilon = 1/N$

Hitting times

- Classical: $O(1/\delta\epsilon) = O(N)$
- Quantum: $O(1/\sqrt{\delta\epsilon}) = O(\sqrt{N})$

Element distinctness

Problem

Given a black box $f: X \to Y$, are there distinct x, x' with $f(x) = f(x')$?

Let $N = |X|$; classical query complexity is $\Omega(N)$

Consider a quantum walk on the Hamming graph $H(N, M)$

- Vertices: $\{(x_1, \ldots, x_M) \colon x_i \in X\}$
- Store the values $(f(x_1), \ldots, f(x_M))$ at vertex $(x_1, \ldots, x_M)$
- Edges between vertices that differ in exactly one coordinate

Element distinctness: Analysis

Spectral gap: $\delta = O(1/M)$

Fraction of marked vertices:
$\epsilon \geq \binom{M}{2}(N-2)^{M-2}/N^M = \Theta(M^2/N^2)$

Quantum hitting time: $O(1/\sqrt{\delta\epsilon}) = O(N/\sqrt{M})$

Quantum query complexity:

- M queries to prepare the initial state
- 2 queries for each step of the walk (compute f, uncompute f)
- Overall: $M + O(N/\sqrt{M})$

Choose $M = N^{2/3}$: query complexity is $O(N^{2/3})$ (optimal!)

Quantum walk algorithms

Quantum walk search algorithms

- Spatial search
- Subgraph finding
- Checking matrix multiplication
- Testing if a black-box group is abelian
- Attacking quantum Merkle cryptosystems

Evaluating Boolean formulas

Exponential speedup for a natural problem?

Exercise: Triangle finding (1/2)

The goal of the *triangle problem* is to decide whether an n-vertex graph G contains a triangle (a complete subgraph on 3 vertices). The graph is specified by a black box that, for any pair of vertices of G, returns a bit indicating whether those vertices are connected by an edge in G.

1. What is the classical query complexity of the triangle problem?
2. Say that an edge of G is a *triangle edge* if it is part of a triangle in G. What is the quantum query complexity of deciding whether a particular edge of G is a triangle edge?
3. Now suppose you know the vertices and edges of some m-vertex subgraph of G. Explain how you can decide whether this subgraph contains a triangle edge using $O(m^{2/3}\sqrt{n})$ quantum queries.

Exercise: Triangle finding (2/2)

4. Consider a quantum walk algorithm for the triangle problem. The walk takes place on a graph $\mathcal{G}$ whose vertices correspond to subgraphs of G on m vertices, and whose edges correspond to subgraphs that differ by changing one vertex. A vertex of $\mathcal{G}$ is marked if it contains a triangle edge. How many queries does this algorithm use to decide whether G contains a triangle? (Hint: Be sure to account for the S queries used to initialize the walk, the U queries used to move between neighboring vertices of $\mathcal{G}$, and the C queries used to check whether a given vertex of $\mathcal{G}$ is marked. If the walk has spectral gap δ and an ϵ-fraction of the vertices are marked, it can be shown that there is a quantum walk search algorithm with query complexity $S + \frac{1}{\sqrt{\epsilon}}(\frac{1}{\sqrt{\delta}}U + C)$.)
5. Choose a value of m that minimizes the number of queries used by the algorithm. What is the resulting upper bound on the quantum query complexity of the triangle problem?