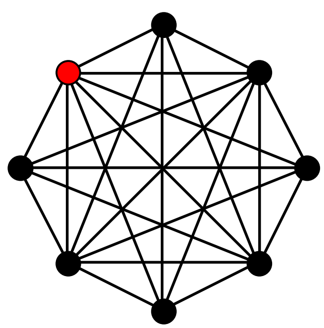
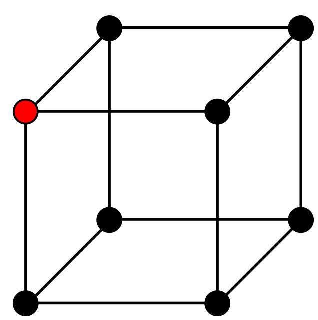
Spatial search by quantum walk

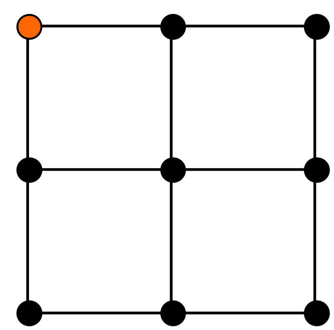
Andrew Childs

Jeffrey Goldstone

MIT Center for Theoretical Physics







quant-ph/0306054

Unstructured search

- N items $\{1,2,\ldots,N\}$
- One "marked item" w
- Query: "is $w=x$?"
  I.e., black box function $f(x) = \begin{cases} 0 & x \neq w \\ 1 & x = w \end{cases}$

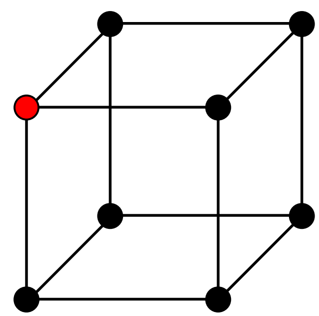
- Classical: $\Theta(N)$
- Grover 1996: $O(N^{1/2})$ quantum algorithm
- BBBV 1996: This is optimal

Combinatorial search vs. spatial search

- Combinatorial search: $f(x)$ is an efficiently computable function

- Spatial search: N items distributed in space (e.g., a physical database)

Model: N-vertex graph G



Algorithm must be *local* with respect to this graph.

Grover's algorithm in *d* dimensions

- One dimension: no speedup; $\Theta(N)$
- Benioff 00: searching a *d*-dimensional grid with a "quantum robot"
  - Each iteration takes $O(N^{1/d})$ steps to traverse the grid
  - $N^{1/2}$ Grover iterations $\Rightarrow$ $O(N^{1/2+1/d})$ algorithm
- Can we do better?

Aaronson-Ambainis algorithm

- Recursive search of subcubes with amplitude amplification

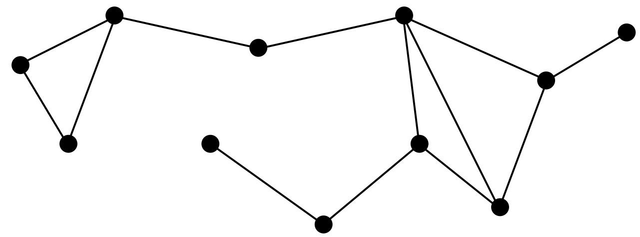
- Results
  - $d>2$: $O(N^{1/2})$ algorithm
  - $d=2$: $O(N^{1/2} \log^2 N)$ algorithm

Quantum walk search algorithm

- Simple Hamiltonian dynamics
- Applicable to any graph G
- Results
  - Complete graph="analog analogue" [FG96]; run time $O(N^{1/2})$
  - Hypercube: $O(N^{1/2})$ by previous results
  - d-dimensional lattice
    - $d>4$: $O(N^{1/2})$
    - $d=4$: $O(N^{1/2} \log^{3/2} N)$
    - $d<4$: no speedup

Graphs and matrices

- Undirected graph *G* with no self loops



- Adjacency matrix: $A_{jk} = \begin{cases} 1 & (j,k) \in G \\ 0 & \text{otherwise} \end{cases}$

- Laplacian: $L = A - D$
  D diagonal, $D_{jj} = \deg(j)$

Random walk	Quantum walk
State space	
N vertices j=1,…,N p_j = probability of being at vertex j	N basis states $\|1\rangle$,…, $\|N\rangle$ $q_j = \langle j\|\psi\rangle$ = amplitude to be at vertex j
Differential equation	
$\frac{\mathrm{d}p_j}{\mathrm{d}t} = \gamma \sum_k L_{jk}\, p_k$	$i\frac{\mathrm{d}q_j}{\mathrm{d}t} = \sum_k H_{jk}\, q_k$
Generator	
L = Laplacian of G	Can choose $H = -\gamma L$
Probability conservation	
$\sum_j L_{jk} = 0 \Rightarrow \frac{\mathrm{d}}{\mathrm{d}t}\sum_j p_j = 0$	$H = H^\dagger \Rightarrow \frac{\mathrm{d}}{\mathrm{d}t}\sum_j \|q_j\|^2 = 0$

Quantum walk search algorithm

- Marked state identified by "oracle Hamiltonian" $H_w = -|w\rangle\langle w|$

Algorithm

- Start in state $|s\rangle = \frac{1}{\sqrt{N}} \sum_j |j\rangle$
- Schrödinger evolve for time T using Hamiltonian $H = -\gamma L + H_w$
- Measure position

- Goal: Choose γ, T so that $|\langle w|e^{-i H T}|s\rangle|^2$ is as close to 1 as possible (for T not too big)

Why might this work?

$$H = -\gamma L - |w\rangle\langle w|$$

critical γ

ground state $\sim |s\rangle + |w\rangle$

first excited state $\sim |s\rangle - |w\rangle$

time $\sim 1/(E_1 - E_0)$

$\gamma \to 0$

$H \sim -|w\rangle\langle w|$

ground state $\sim |w\rangle$

first excited state $\sim |s\rangle$

$\gamma \to \infty$

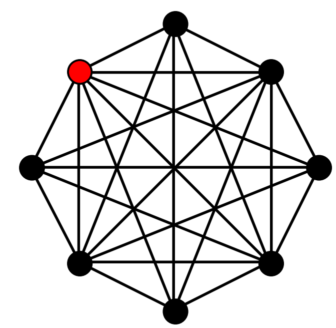
$H \sim -\gamma L$

ground state $\sim |s\rangle$

γ

Complete graph

$$L + NI = N|s\rangle\langle s| = \begin{pmatrix} 1 & \cdots & 1 \\ \vdots & \ddots & \vdots \\ 1 & \cdots & 1 \end{pmatrix}$$



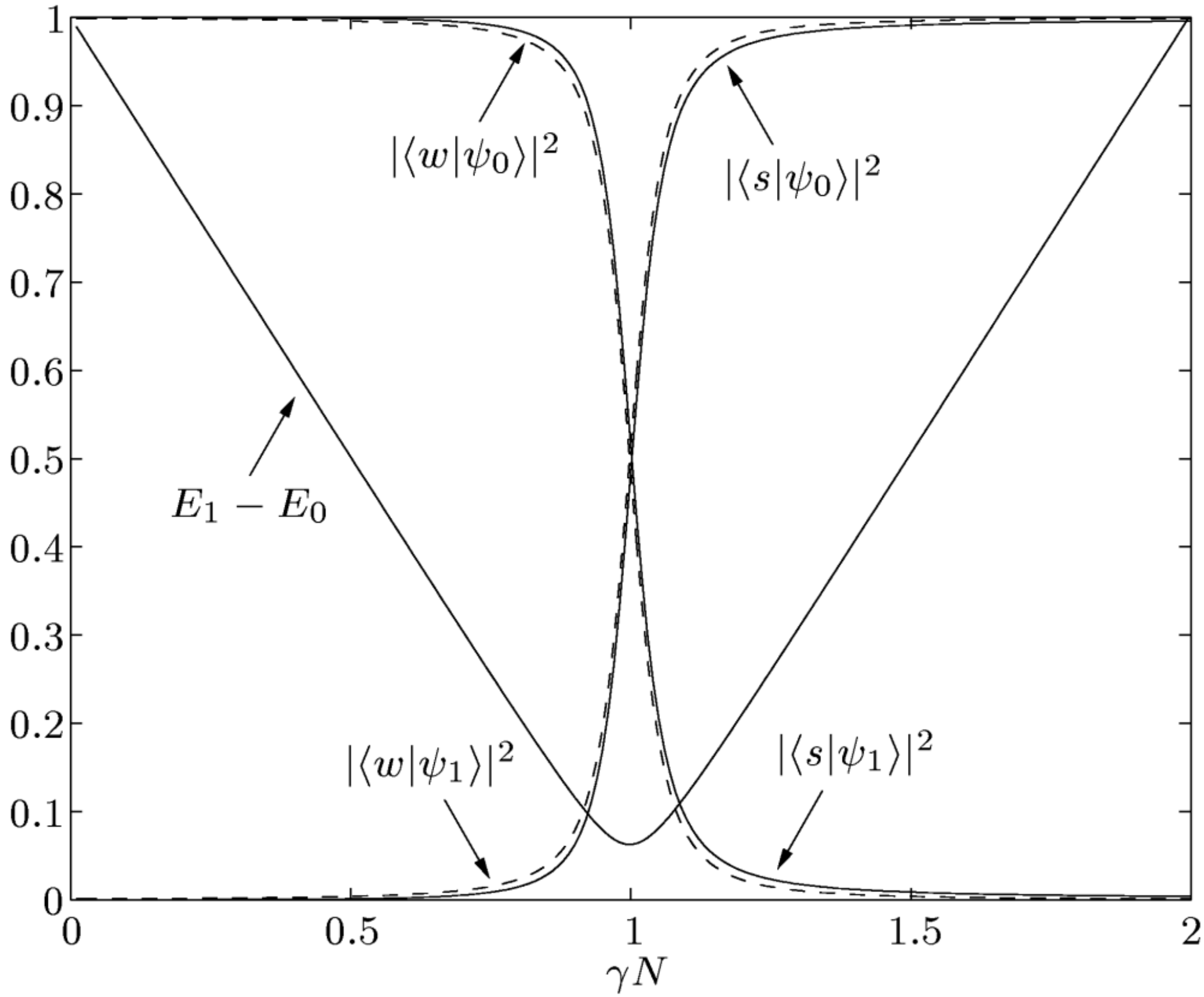
$$H = -\gamma N|s\rangle\langle s| - |w\rangle\langle w|$$

$\gamma N = 1$ is the “analog analogue” of Grover’s algorithm

Eigenstates ~ $|s\rangle \pm |w\rangle$

Gap $2N^{-1/2}$

Complete graph

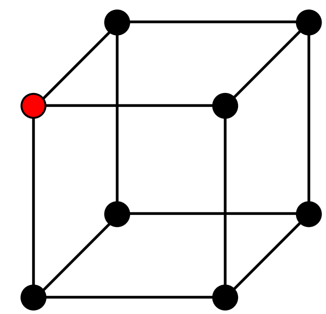


N=1024

Hypercube

Vertices labelled by *n*-bit strings

$N=2^n$

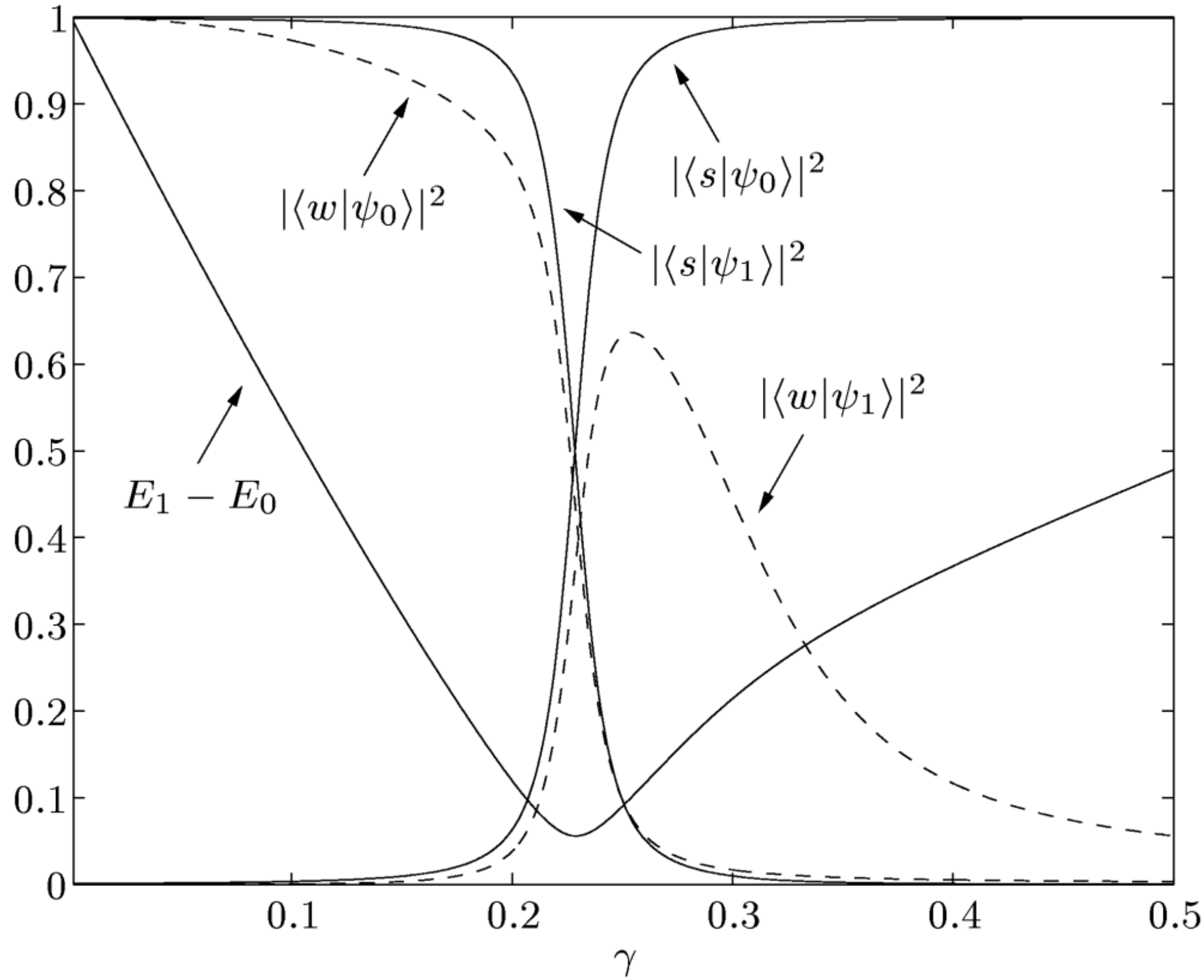


Adjacency matrix: $A = \sum_{j=1}^{n} \sigma_x^{(j)}$

Hamiltonian: $H = -\gamma A - |w\rangle\langle w|$

Analyze using total spin operators [FGGS00]

Hypercube



$N=2^{10}=1024$

d-dimensional lattice

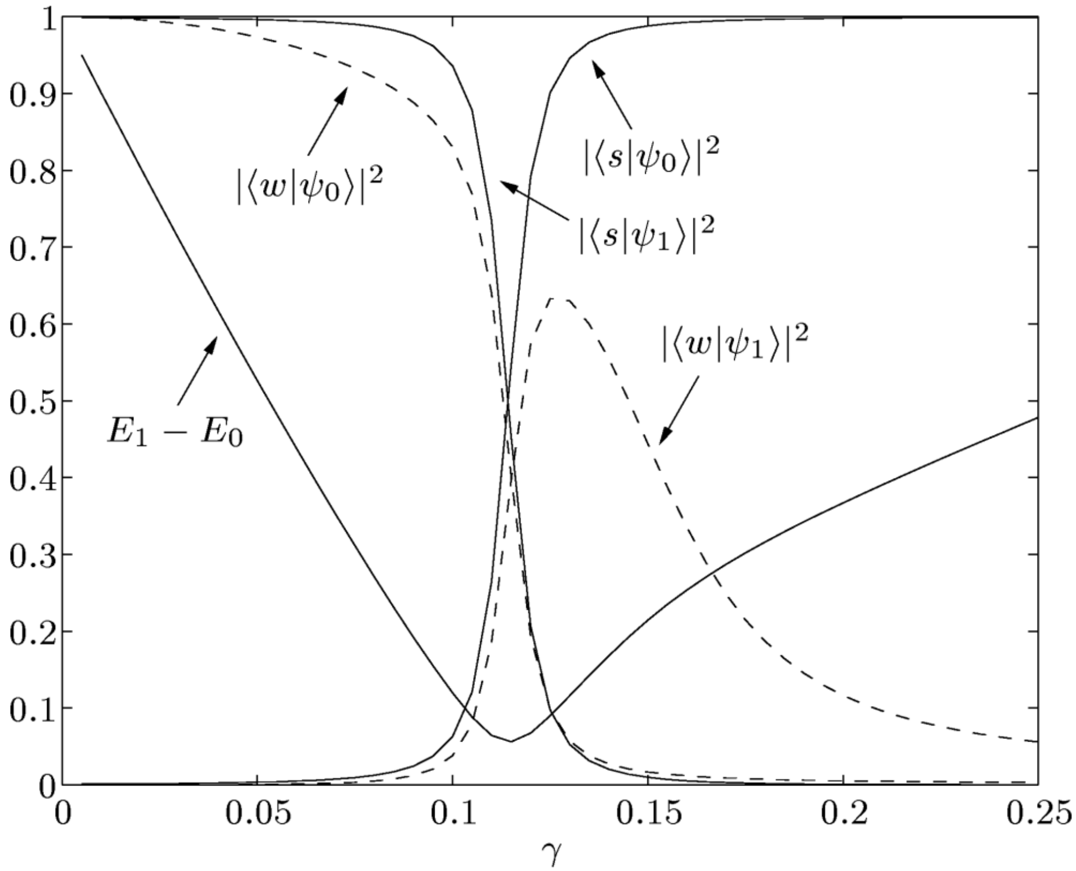
- Periodic cubic lattice with *N* sites, size $N^{1/d}$ in each dimension
- Exact eigenstates/eigenvalues of –*L*

$$|\phi(\vec{k})\rangle = \frac{1}{\sqrt{N}} \sum_{\vec{x}} e^{i\vec{k}\cdot\vec{x}} |\vec{x}\rangle$$

$$\mathcal{E}(\vec{k}) = 2\left(d - \sum_{j=1}^{d} \cos k_j\right)$$

$$k_j = \frac{2\pi m_j}{N^{1/d}}, \quad m_j = 0, 1, \ldots, N^{1/d} - 1$$

(d>4)-dimensional lattice



d=5
N=4^5=1024

Critical region

$\gamma = \gamma^* \pm O(N^{-1/2})$

$\gamma < \gamma^*$

$|s\rangle \sim |\psi_1\rangle$

$\gamma > \gamma^*$

$|s\rangle \sim |\psi_0\rangle$

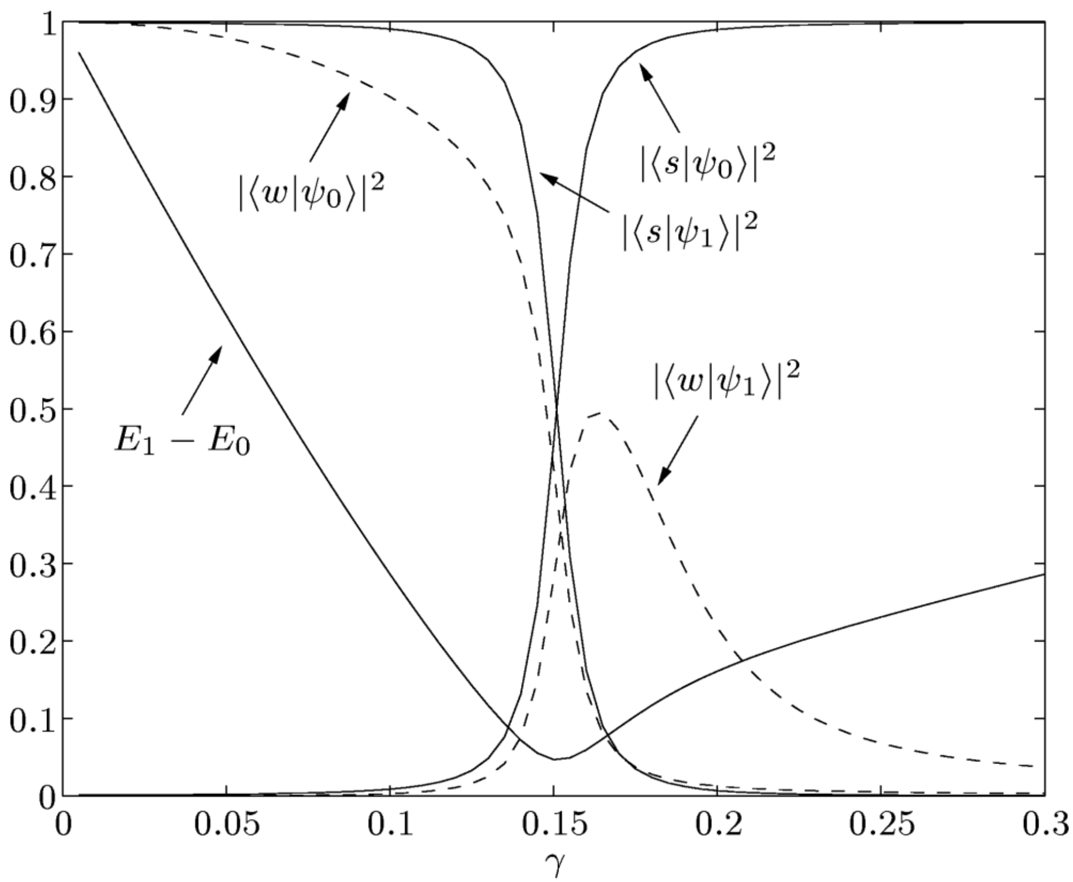
$\gamma \sim \gamma^*$

$E_1 - E_0 = O(N^{-1/2})$

$|\psi_{0,1}\rangle \sim |s\rangle \pm O(1)\, |w\rangle$

Run time $O(N^{1/2})$

4-dimensional lattice



d=4
N=6^4=1296

Critical region

$$\gamma = \gamma^* \pm O\left(\sqrt{\frac{\log N}{N}}\right)$$

$\gamma < \gamma^*$

$|s\rangle \sim |\psi_1\rangle$

$\gamma > \gamma^*$

$|s\rangle \sim |\psi_0\rangle$

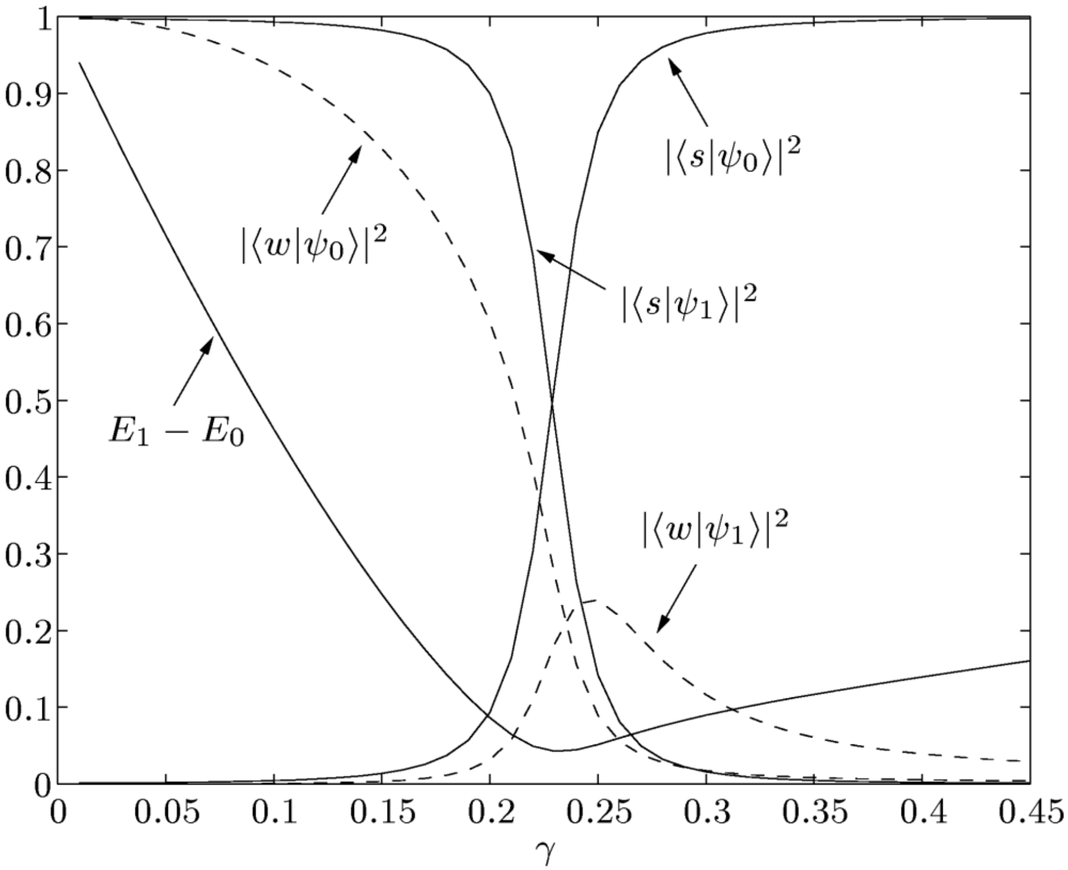
$\gamma \sim \gamma^*$

$$E_1 - E_0 = O\left(\frac{1}{\sqrt{N \log N}}\right)$$

$$|\psi_{0,1}\rangle \sim |s\rangle \pm O\left(\frac{1}{\sqrt{\log N}}\right)|w\rangle$$

Run time $O(\sqrt{N}\ \log^{3/2} N)$

3-dimensional lattice



d=3

$N=10^3=1000$

Critical region

$\gamma = \gamma^* \pm O\left(\frac{1}{N^{1/3}}\right)$

$\gamma < \gamma^*$

$|s\rangle \sim |\psi_1\rangle$

$\gamma > \gamma^*$

$|s\rangle \sim |\psi_0\rangle$

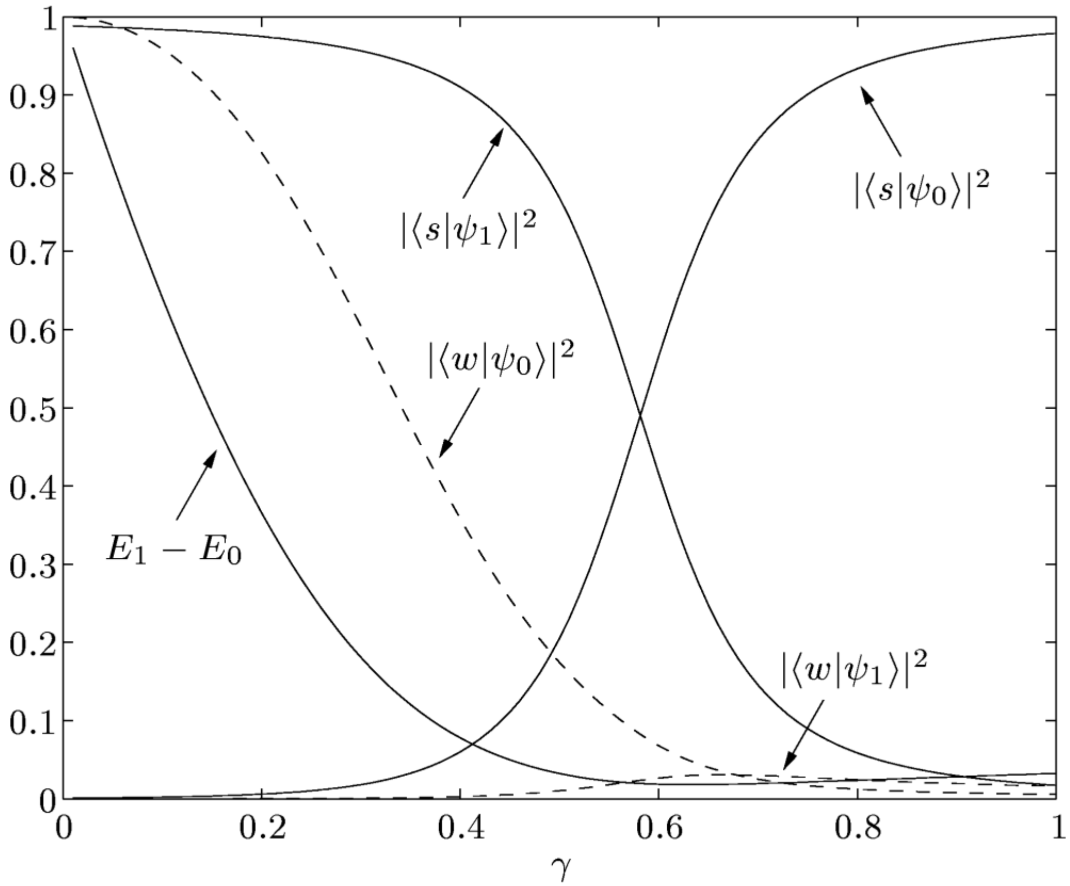
$\gamma \sim \gamma^*$

$E_1 - E_0 = O\left(\frac{1}{N^{2/3}}\right)$

$|\psi_{0,1}\rangle \sim |s\rangle \pm O\left(\frac{1}{N^{1/6}}\right) |w\rangle$

Run time $O(N)$

2-dimensional lattice



d=2
N=32²=1024

Critical region

$\gamma = \gamma^* \log N \pm O(1)$

$\gamma < \gamma^* \log N$

$|s\rangle \sim |\psi_1\rangle$

$\gamma > \gamma^* \log N$

$|s\rangle \sim |\psi_0\rangle$

$\gamma \sim \gamma^* \log N$

$E_1 - E_0 = O\left(\frac{\log N}{N}\right)$

$|\psi_{0,1}\rangle \sim |s\rangle \pm O\left(\sqrt{\frac{\log N}{N}}\right)|w\rangle$

Run time $O(N^2/\log^2 N)$

Related algorithms

- Shenvi, Kempe, Whaley 02: discrete time quantum walk search algorithm on hypercube, $O(N^{1/2})$

  Behavior in finite dimensions?

- Adiabatic evolution [RC01, vDMV01]

- Measurement [CDFGGS02]

Open questions

- Find more applications of quantum walks
- What is the actual complexity of the search problem in d=2?