Quantum walks on graphs

Andrew Childs
California Institute of Technology

Information is physical.

Outline

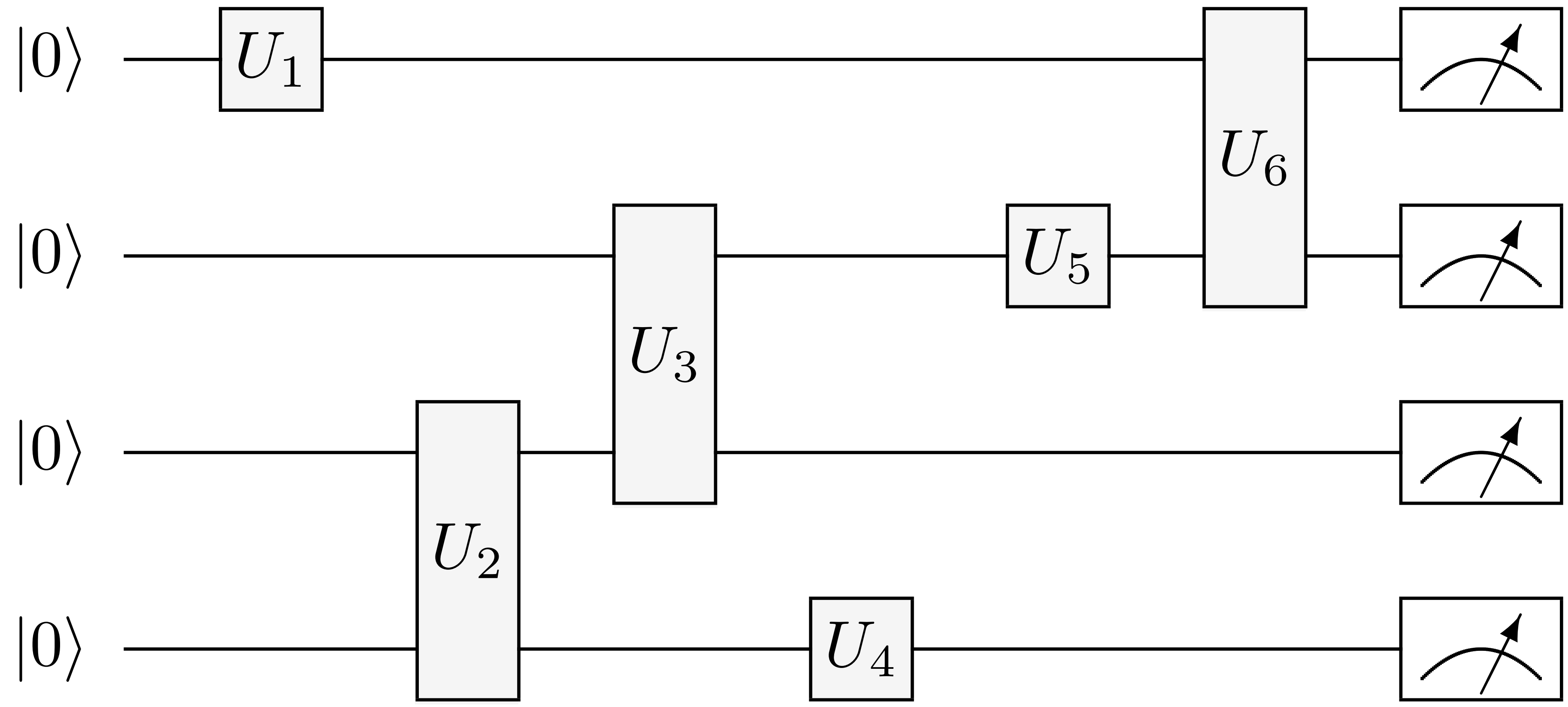
1. Introduction: Quantum systems as information processing machines
2. Exponential speedup by quantum walk
3. Spatial search by quantum walk
4. Evaluating Boolean formulas

1. Introduction

The universal quantum computer

(The ultimate quantum physics lab!)

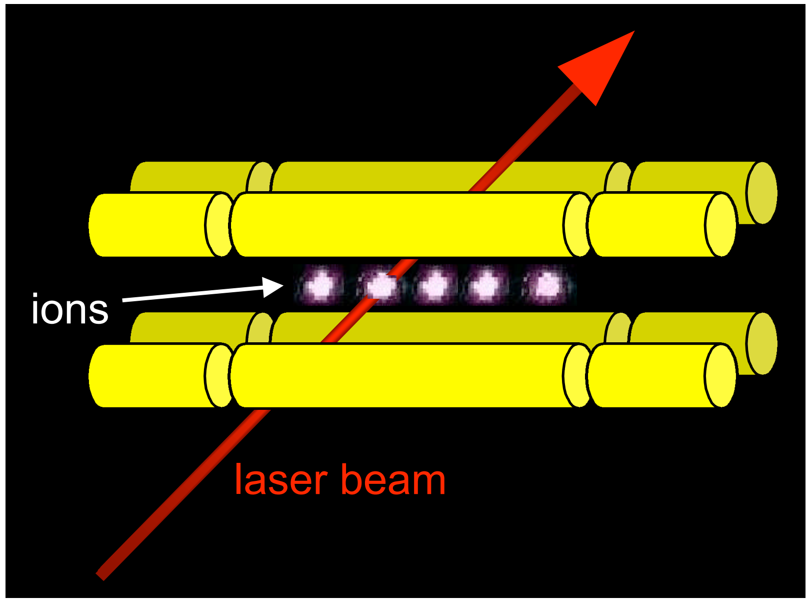
- Prepare n qubits in the state $|00\ldots0\rangle$
- Apply a sequence of unitary operations acting on one or two qubits at a time
- Perform a measurement to get the result



Note: Many equivalent models exist (Hamiltonian dynamics of coupled spins, braiding of nonabelian anyons, quantum cellular automata, ...).

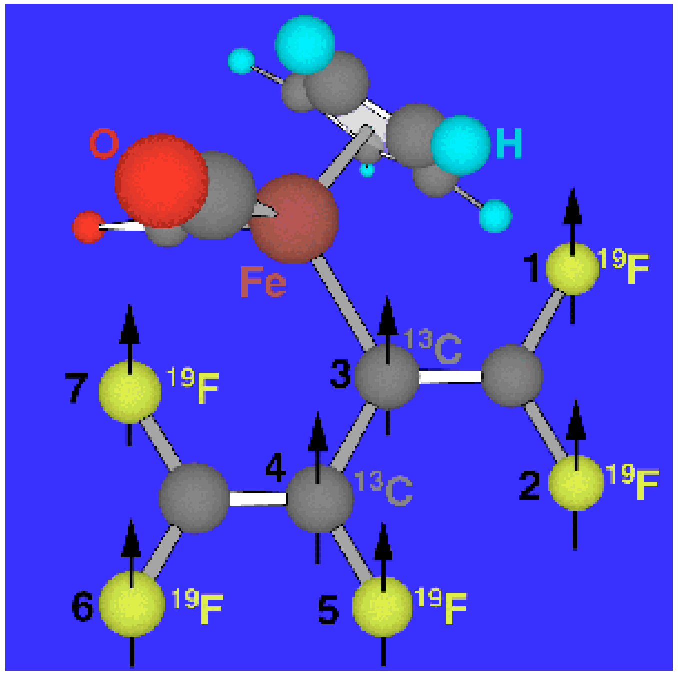
Implementations of quantum computers

Trapped ions



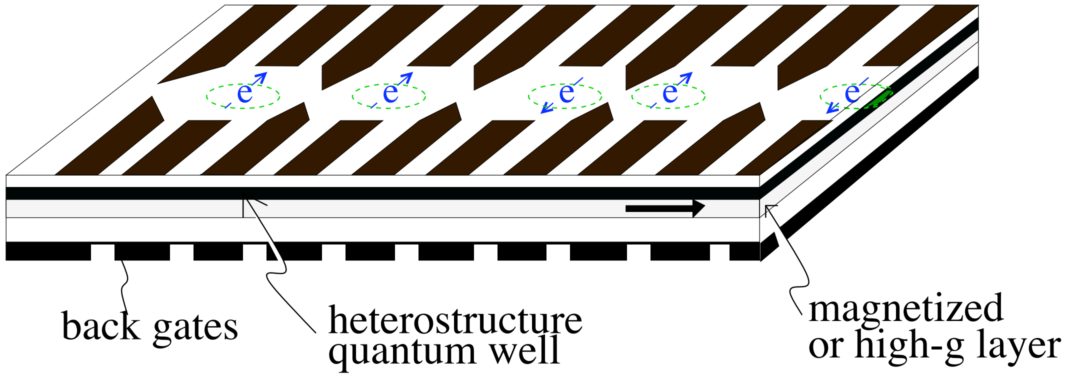
(Monroe & Wineland)

Nuclear spins



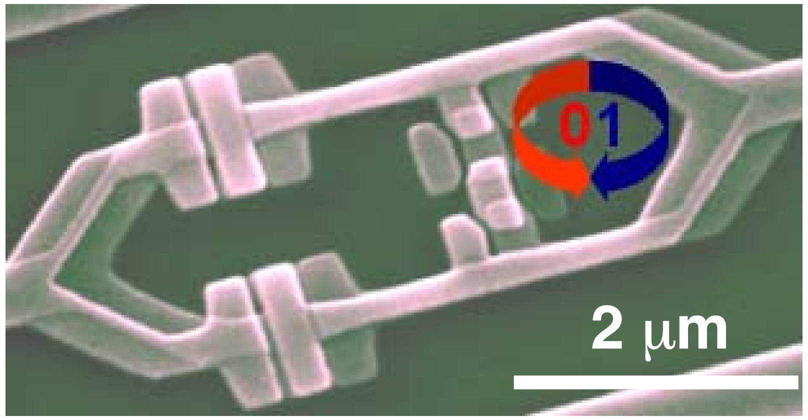
(Chuang et al.)

Quantum dots



(Loss & DiVincenzo)

Superconducting circuits



(Nakamura et al.)

... and many others!

The threshold theorem

Realistic quantum systems are subject to noise. Is quantum computation still possible?

The threshold theorem

Realistic quantum systems are subject to noise. Is quantum computation still possible?

von Neumann 1956 ("Probabilistic logics and the synthesis of reliable organisms from unreliable components"): Provided the noise level is below some threshold value, any computation can be performed with success probability arbitrarily close to 1, by encoding the computation redundantly (incurring only logarithmic overhead).

The threshold theorem

Realistic quantum systems are subject to noise. Is quantum computation still possible?

von Neumann 1956 ("Probabilistic logics and the synthesis of reliable organisms from unreliable components"): Provided the noise level is below some threshold value, any computation can be performed with success probability arbitrarily close to 1, by encoding the computation redundantly (incurring only logarithmic overhead).

Shor 1996, Aharonov and Ben-Or 1997, Kitaev 1997, et al.: Similar result for quantum computers. Threshold estimates $\approx 10^{-3}$ to 10^{-5}.

The threshold theorem

Realistic quantum systems are subject to noise. Is quantum computation still possible?

von Neumann 1956 ("Probabilistic logics and the synthesis of reliable organisms from unreliable components"): Provided the noise level is below some threshold value, any computation can be performed with success probability arbitrarily close to 1, by encoding the computation redundantly (incurring only logarithmic overhead).

Shor 1996, Aharonov and Ben-Or 1997, Kitaev 1997, et al.: Similar result for quantum computers. Threshold estimates $\approx 10^{-3}$ to 10^{-5}.

In the rest of this talk, we assume a perfectly functioning quantum computer.

Interference as a resource

Interference as a resource

Toy problem [Deutsch 1985]

Given a black box for $f : \{0,1\} \to \{0,1\}$, determine $f(0) \oplus f(1)$.

Interference as a resource

Toy problem [Deutsch 1985]

Given a black box for $f : \{0,1\} \to \{0,1\}$, determine $f(0) \oplus f(1)$.

Classically: Two queries required.

Interference as a resource

Toy problem [Deutsch 1985]

Given a black box for $f : \{0,1\} \to \{0,1\}$, determine $f(0) \oplus f(1)$.

Classically: Two queries required.

Quantumly: One query sufficient!

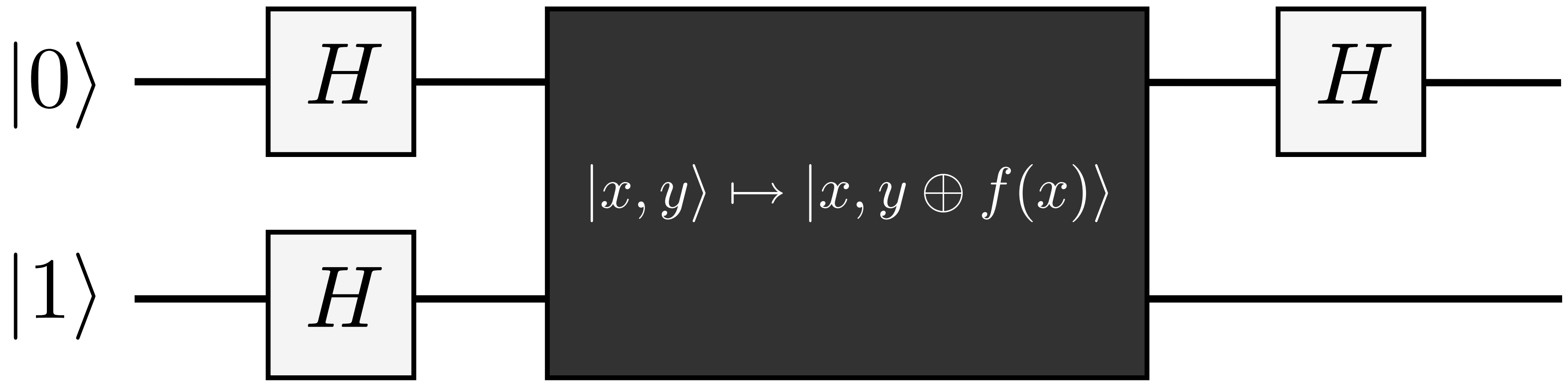
Interference as a resource

Toy problem [Deutsch 1985]

Given a black box for $f: \{0,1\} \to \{0,1\}$, determine $f(0) \oplus f(1)$.

Classically: Two queries required.

Quantumly: One query sufficient!



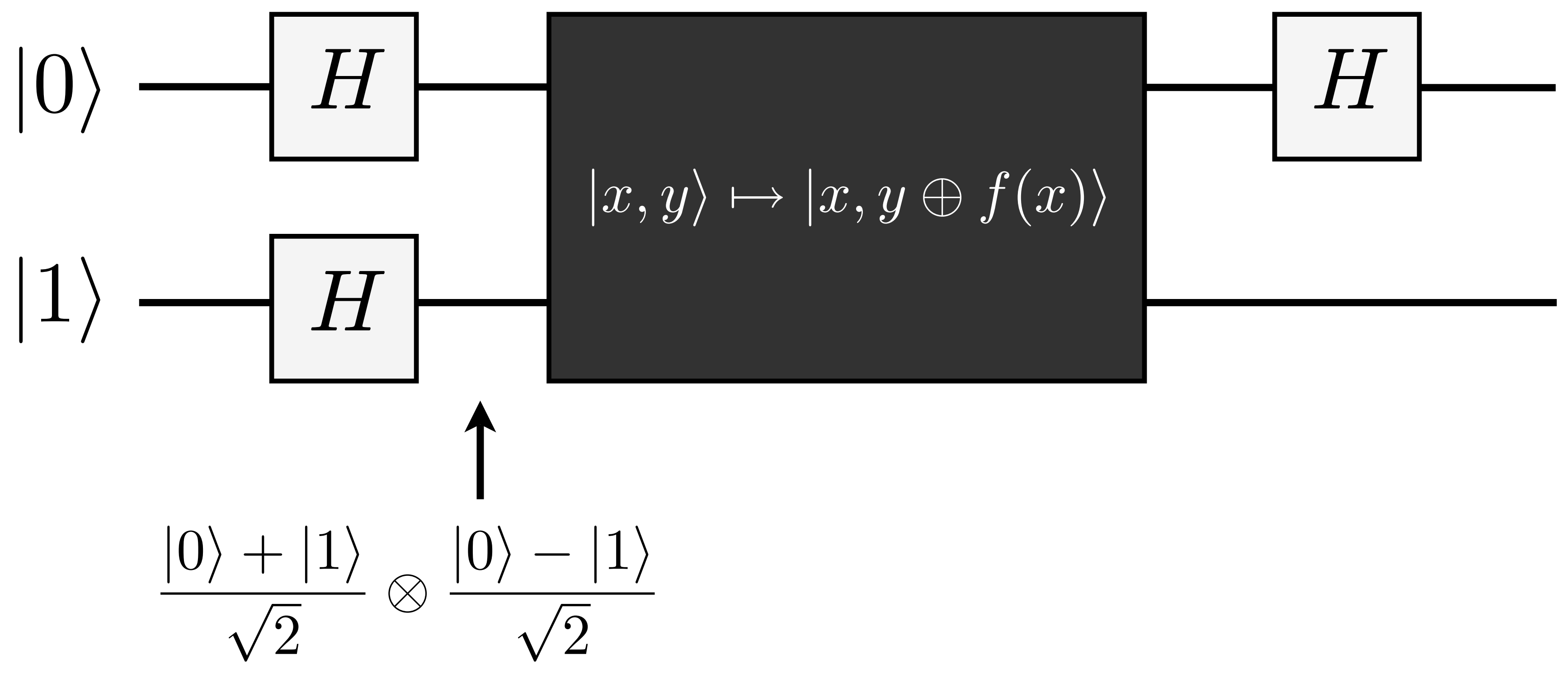
Interference as a resource

Toy problem [Deutsch 1985]

Given a black box for $f : \{0,1\} \to \{0,1\}$, determine $f(0) \oplus f(1)$.

Classically: Two queries required.

Quantumly: One query sufficient!



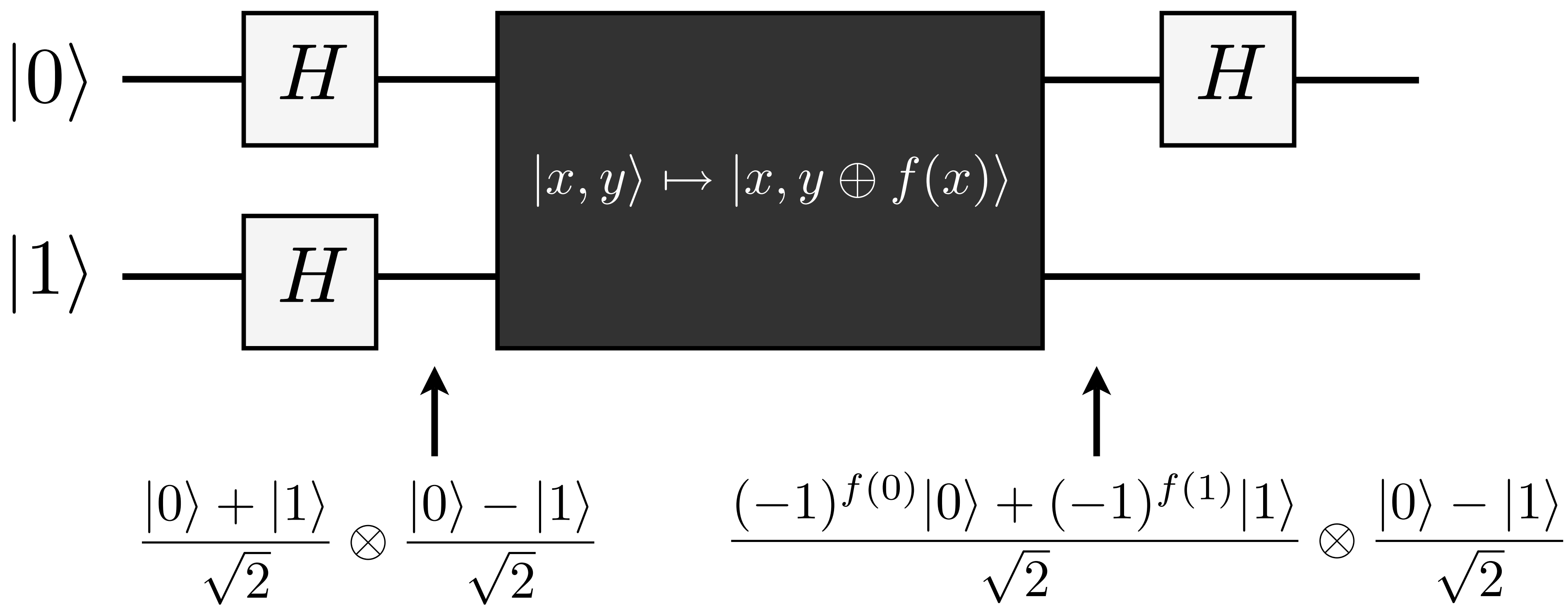
Interference as a resource

Toy problem [Deutsch 1985]

Given a black box for $f\colon \{0,1\} \to \{0,1\}$, determine $f(0) \oplus f(1)$.

Classically: Two queries required.

Quantumly: One query sufficient!



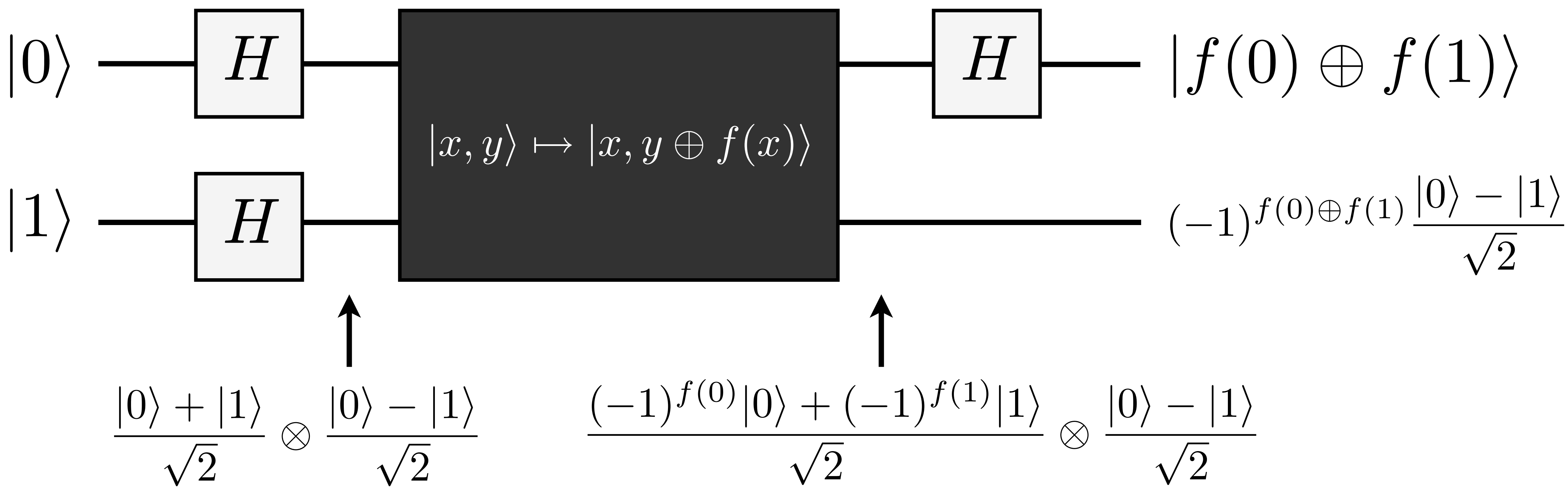
Interference as a resource

Toy problem [Deutsch 1985]

Given a black box for $f : \{0,1\} \to \{0,1\}$, determine $f(0) \oplus f(1)$.

Classically: Two queries required.

Quantumly: One query sufficient!



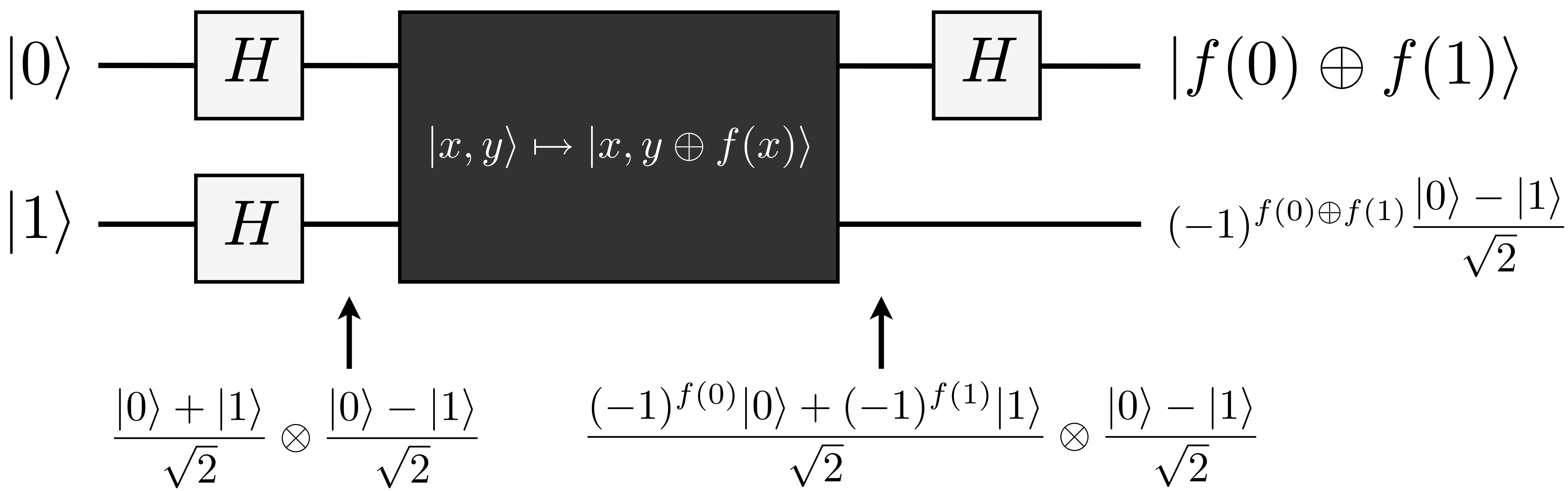
Interference as a resource

Toy problem [Deutsch 1985]

Given a black box for $f : \{0,1\} \to \{0,1\}$, determine $f(0) \oplus f(1)$.

Classically: Two queries required.

Quantumly: One query sufficient!



But don't classical systems exhibit interference too?

Hilbert space is a big place

n classical bits: 2^n discrete values

Hilbert space is a big place

n classical bits: 2^n discrete values

n quantum bits: 2^n complex numbers

$$|\psi\rangle = \sum_{x=0}^{2^n-1} \alpha_x |x\rangle$$

Hilbert space is a big place

n classical bits: 2^n discrete values

n quantum bits: 2^n complex numbers

$$|\psi\rangle = \sum_{x=0}^{2^n-1} \alpha_x |x\rangle$$

n random bits: 2^n real numbers (probabilities)

Hilbert space is a big place

n classical bits: 2^n discrete values

n quantum bits: 2^n complex numbers $$|\psi\rangle = \sum_{x=0}^{2^n-1} \alpha_x |x\rangle$$

n random bits: 2^n real numbers (probabilities)

But probabilities do not exhibit interference!

Hilbert space is a big place

n classical bits: 2^n discrete values

n quantum bits: 2^n complex numbers $$|\psi\rangle = \sum_{x=0}^{2^n-1} \alpha_x |x\rangle$$

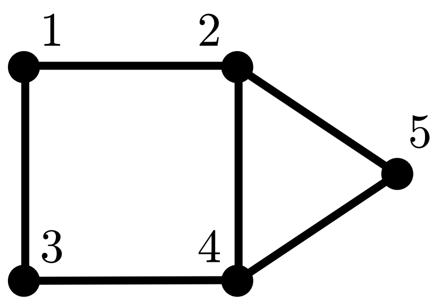
n random bits: 2^n real numbers (probabilities)

But probabilities do not exhibit interference!

How can we exploit the efficient representation of interference phenomena to perform fast computations?

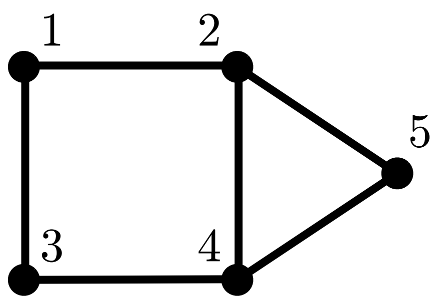
From random walk to quantum walk

Graph G:



From random walk to quantum walk

Graph G:

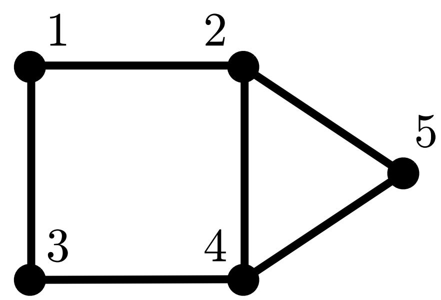


$$A = \begin{pmatrix} 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 \end{pmatrix}$$

adjacency matrix

From random walk to quantum walk

Graph G:

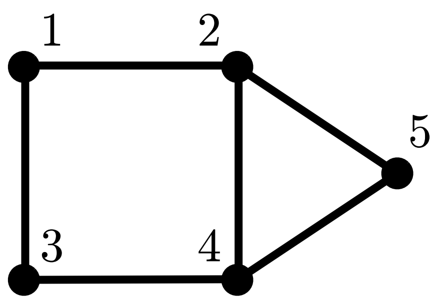


$$A = \begin{pmatrix} 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 \end{pmatrix} \qquad L = \begin{pmatrix} -2 & 1 & 1 & 0 & 0 \\ 1 & -3 & 0 & 1 & 1 \\ 1 & 0 & -2 & 1 & 0 \\ 0 & 1 & 1 & -3 & 1 \\ 0 & 1 & 0 & 1 & -2 \end{pmatrix}$$

adjacency matrix *Laplacian*

From random walk to 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*

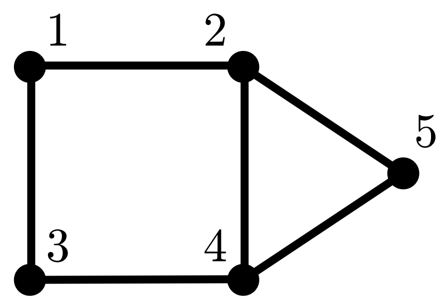
Random walk on G

State: Probability $p_j(t)$ of being at vertex j at time t

Dynamics: $\frac{\mathrm{d}}{\mathrm{d}t}\vec{p} = -\gamma L \vec{p}$

From random walk to quantum walk

Graph G:



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

adjacency matrix

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

Laplacian

Random walk on G

State: Probability $p_j(t)$ of being at vertex j at time t

Dynamics: $$\frac{\mathrm{d}}{\mathrm{d}t}\vec{p} = -\gamma L\vec{p}$$

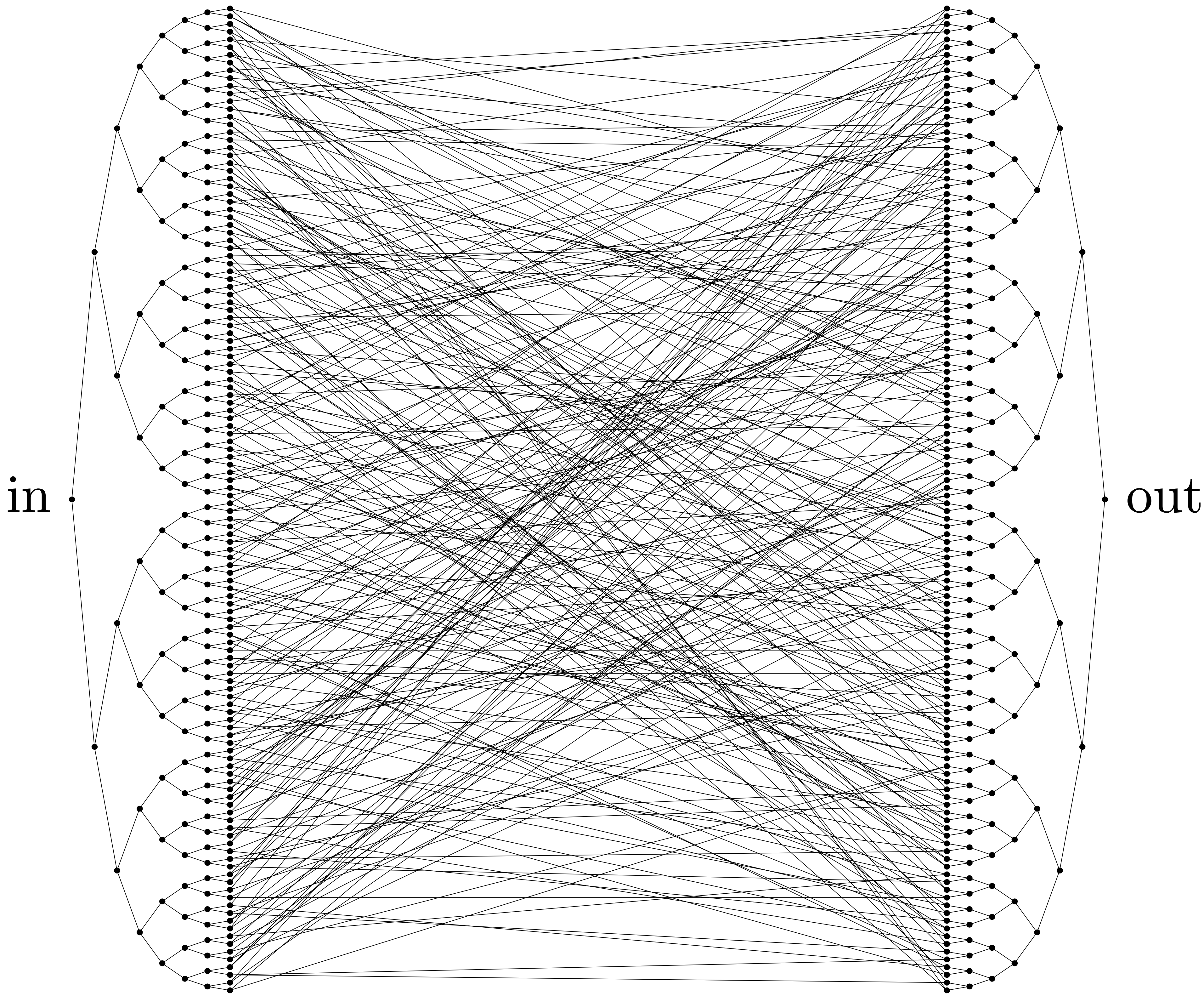
Quantum walk on G

State: Amplitude $q_j(t)$ to be at vertex j at time t

Dynamics: $$i\frac{\mathrm{d}}{\mathrm{d}t}\vec{q} = -\gamma L\vec{q}$$

2. Exponential speedup by quantum walk

- Childs, Farhi, and Gutmann, Quantum Inf. Proc. **1**, 35-43 (2002).
- Childs, Cleve, Deotto, Farhi, Gutmann, and Spielman, in Proc. STOC (2003), pp. 59-68.



in
out

Black box graph traversal problem

Names of vertices are random bit strings (length $2\log|G|$).
Name of in vertex is known.
Black box outputs the names of adjacent vertices.
Name of out vertex is unknown; find it!

Black box graph traversal problem

Names of vertices are random bit strings (length $2\log|G|$).
Name of in vertex is known.
Black box outputs the names of adjacent vertices.
Name of out vertex is unknown; find it!

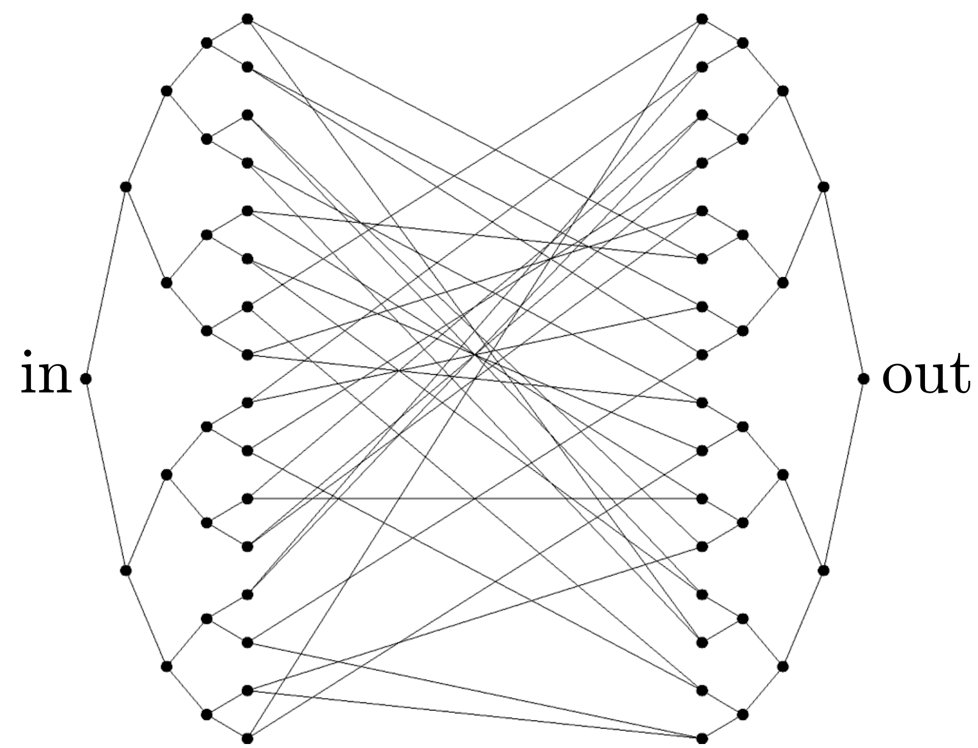
Claim: There is a family of graphs G_n (with designated in and out vertices) for which a quantum walk starting at the in vertex finds the out vertex in time $\mathrm{poly}(n)$, but any classical algorithm using $\mathrm{poly}(n)$ queries finds the out vertex with exponentially small probability.

Black box graph traversal problem

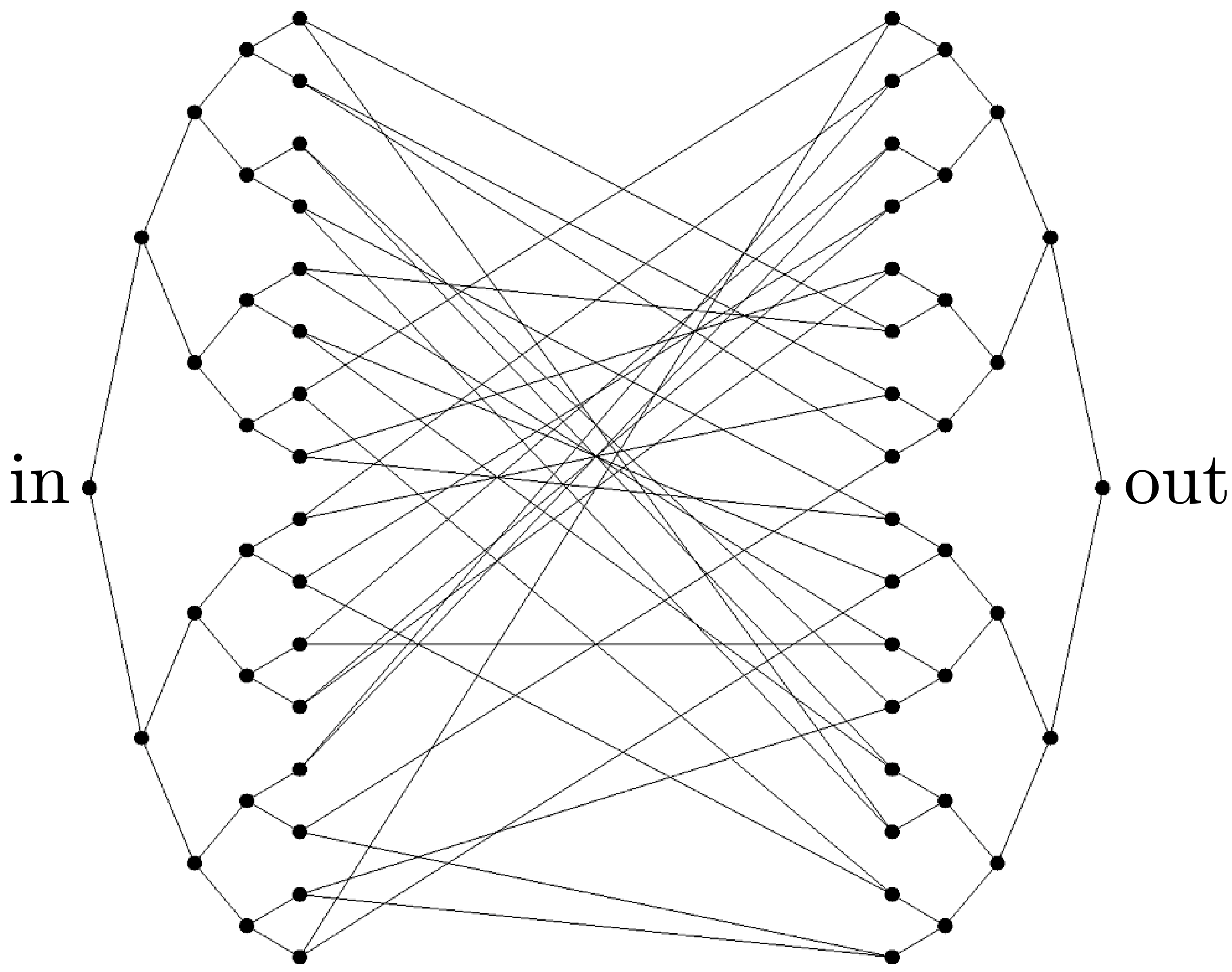
Names of vertices are random bit strings (length $2\log|G|$).
Name of in vertex is known.
Black box outputs the names of adjacent vertices.
Name of out vertex is unknown; find it!

Claim: There is a family of graphs G_n (with designated in and out vertices) for which a quantum walk starting at the in vertex finds the out vertex in time $\mathrm{poly}(n)$, but any classical algorithm using $\mathrm{poly}(n)$ queries finds the out vertex with exponentially small probability.

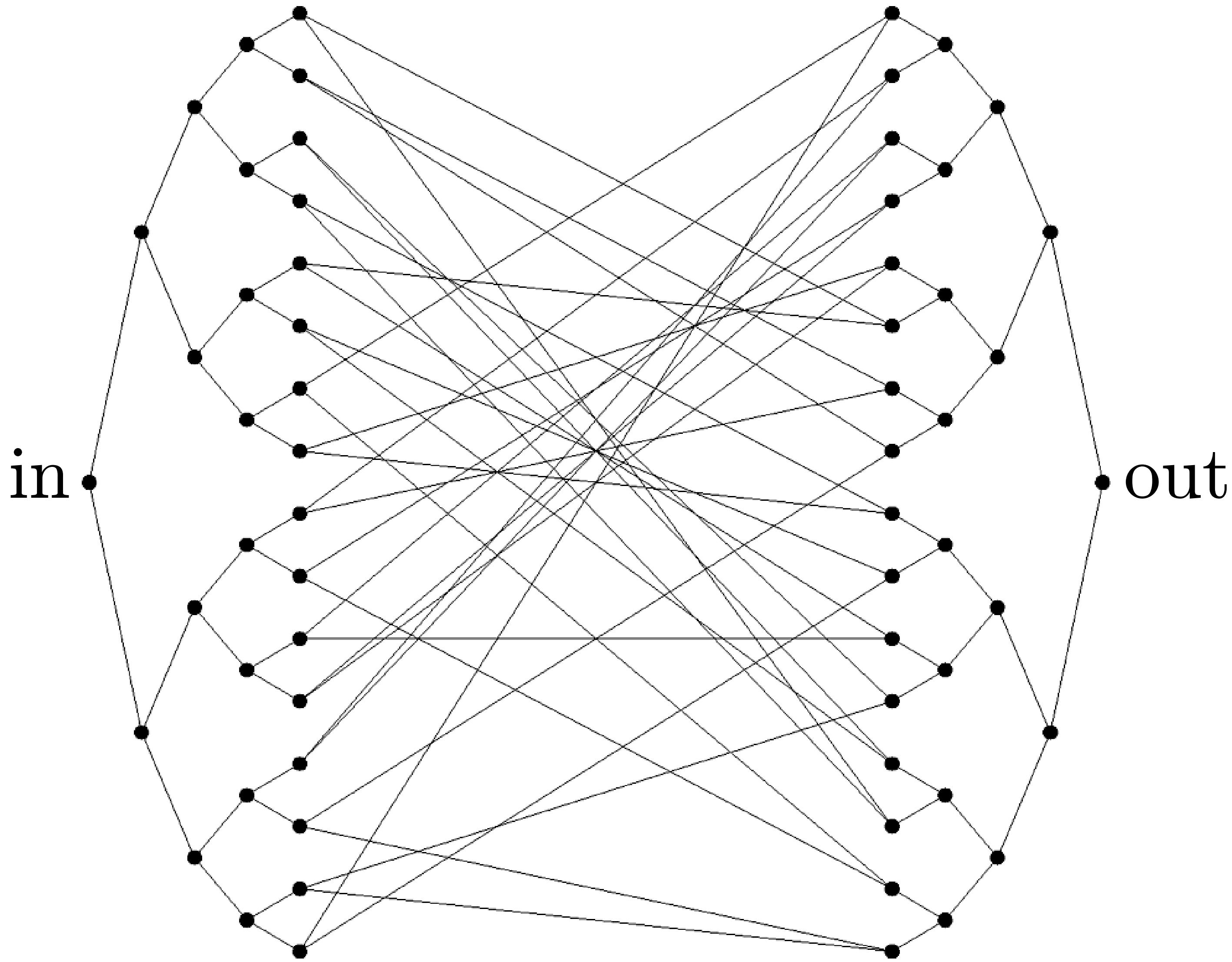
Typical G_4:



Reduction of the quantum walk



Reduction of the quantum walk



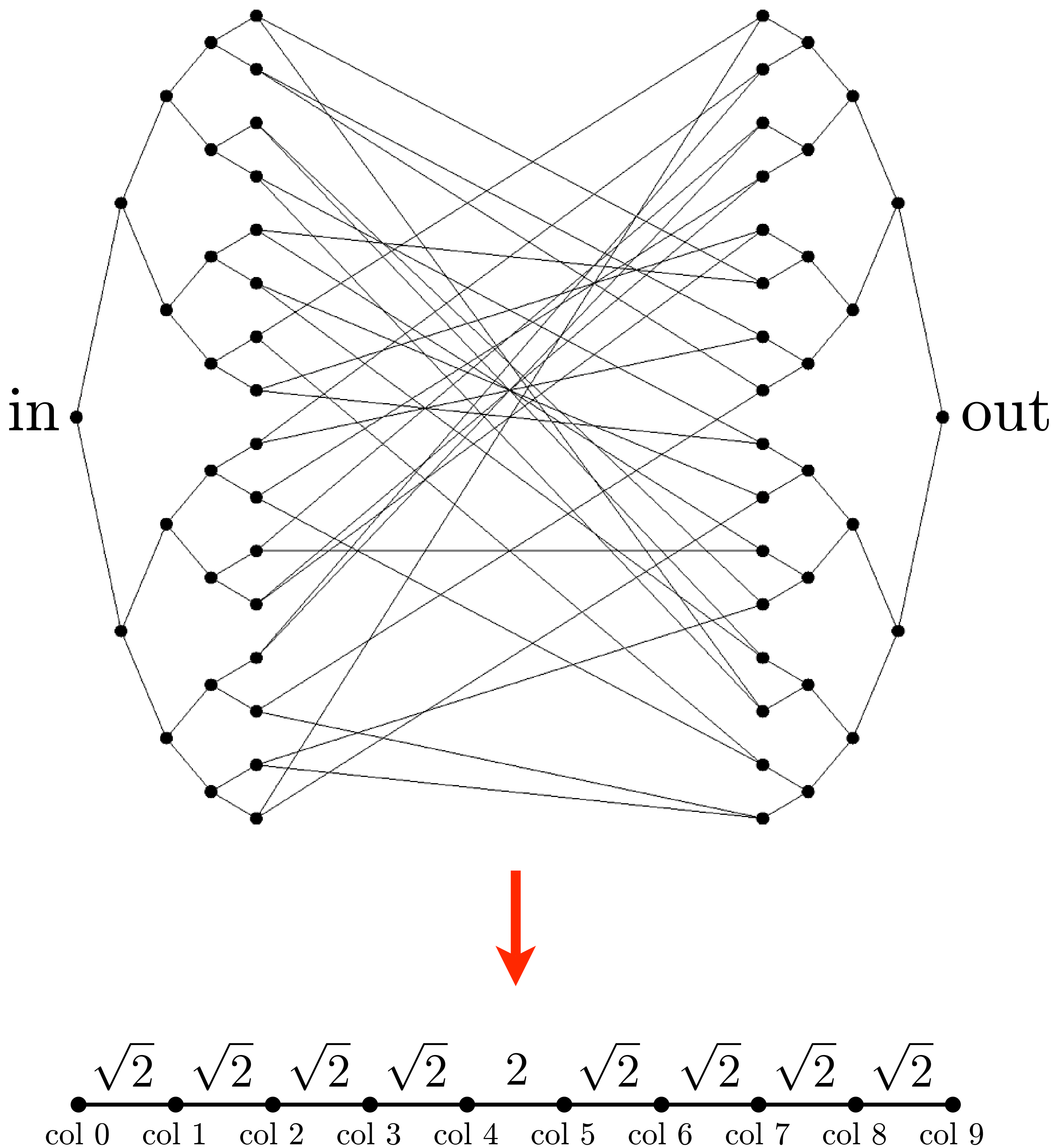
Column subspace

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

where

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

Reduction of the quantum walk



Column subspace

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

where

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

Reduced adjacency matrix

$$\langle \text{col } j|A|\text{col } j+1\rangle$$

$$= \begin{cases} \sqrt{2} & 0 \le j \le n-1\,, \\ & n+1 \le j \le 2n \\ 2 & j = n \end{cases}$$

Implementing the walk

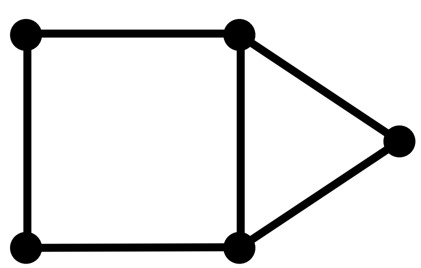
Problem: Given a black box for G, implement the quantum walk on G, i.e., simulate the unitary time evolution e^{-iHt} where $H{=}L$ (or A).
(Cf. implementing a random walk, which is easy.)

Main idea: Color the graph. Then the simulation breaks into small pieces that are easy to handle.

Implementing the walk

Problem: Given a black box for G, implement the quantum walk on G, i.e., simulate the unitary time evolution e^{-iHt} where $H=L$ (or A).
(Cf. implementing a random walk, which is easy.)

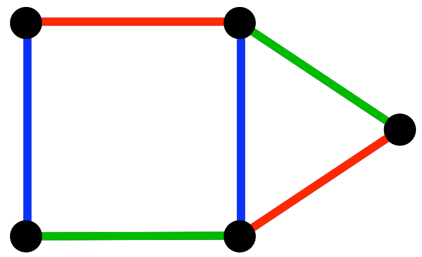
Main idea: Color the graph. Then the simulation breaks into small pieces that are easy to handle.



Implementing the walk

Problem: Given a black box for G, implement the quantum walk on G, i.e., simulate the unitary time evolution e^{-iHt} where $H=L$ (or A).
(Cf. implementing a random walk, which is easy.)

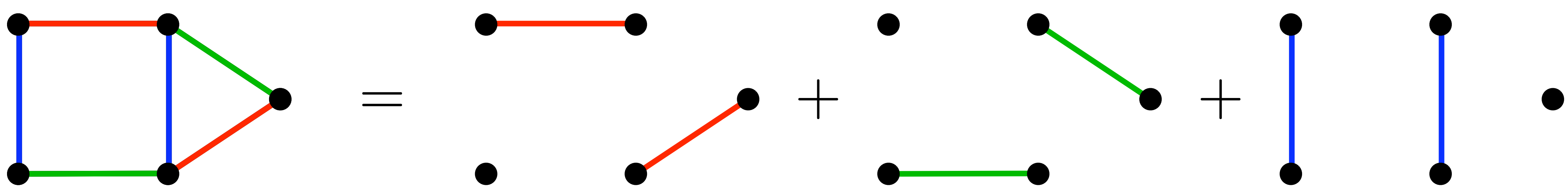
Main idea: Color the graph. Then the simulation breaks into small pieces that are easy to handle.



Implementing the walk

Problem: Given a black box for G, implement the quantum walk on G, i.e., simulate the unitary time evolution e^{-iHt} where $H = L$ (or A).
(Cf. implementing a random walk, which is easy.)

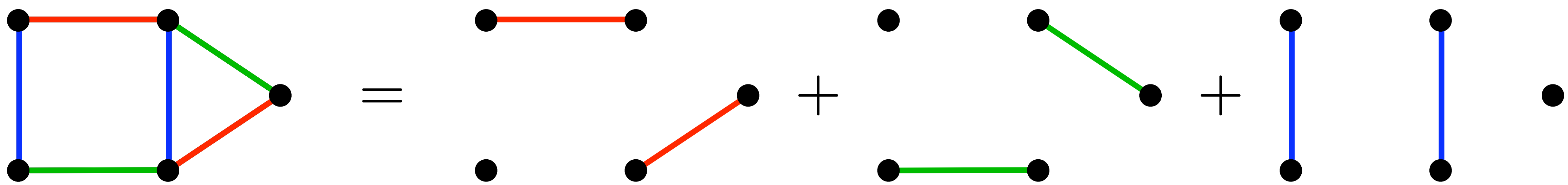
Main idea: Color the graph. Then the simulation breaks into small pieces that are easy to handle.



Implementing the walk

Problem: Given a black box for G, implement the quantum walk on G, i.e., simulate the unitary time evolution e^{-iHt} where $H=L$ (or A).
(Cf. implementing a random walk, which is easy.)

Main idea: Color the graph. Then the simulation breaks into small pieces that are easy to handle.

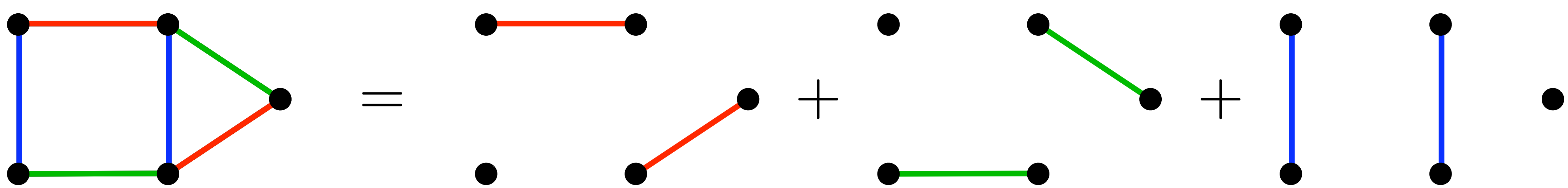


Any sufficiently sparse graph can be efficiently colored using only local information (Linial 1987).

Implementing the walk

Problem: Given a black box for G, implement the quantum walk on G, i.e., simulate the unitary time evolution e^{-iHt} where $H=L$ (or A).
(Cf. implementing a random walk, which is easy.)

Main idea: Color the graph. Then the simulation breaks into small pieces that are easy to handle.



Any sufficiently sparse graph can be efficiently colored using only local information (Linial 1987).

This idea is useful for simulating Hamiltonians whenever the graph of nonzero matrix elements is sufficiently sparse.

Classical lower bound

Theorem: Any classical algorithm that makes at most $2^{n/6}$ queries to the black box finds the out vertex with probability at most $4 \cdot 2^{-n/6}$.

Classical lower bound

Theorem: Any classical algorithm that makes at most $2^{n/6}$ queries to the black box finds the out vertex with probability at most $4 \cdot 2^{-n/6}$.

Proof idea:

Classical lower bound

Theorem: Any classical algorithm that makes at most $2^{n/6}$ queries to the black box finds the out vertex with probability at most $4 \cdot 2^{-n/6}$.

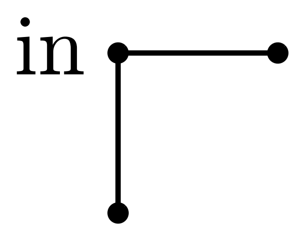
Proof idea:

$\mathrm{in} \bullet$

Classical lower bound

Theorem: Any classical algorithm that makes at most $2^{n/6}$ queries to the black box finds the out vertex with probability at most $4 \cdot 2^{-n/6}$.

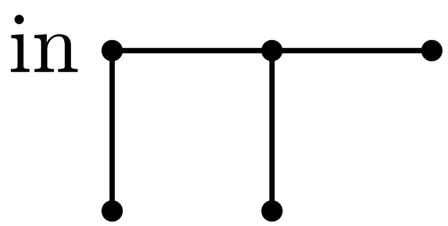
Proof idea:



Classical lower bound

Theorem: Any classical algorithm that makes at most $2^{n/6}$ queries to the black box finds the out vertex with probability at most $4 \cdot 2^{-n/6}$.

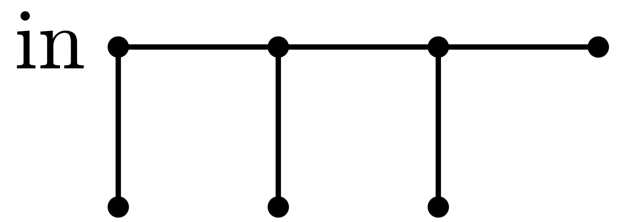
Proof idea:



Classical lower bound

Theorem: Any classical algorithm that makes at most $2^{n/6}$ queries to the black box finds the out vertex with probability at most $4 \cdot 2^{-n/6}$.

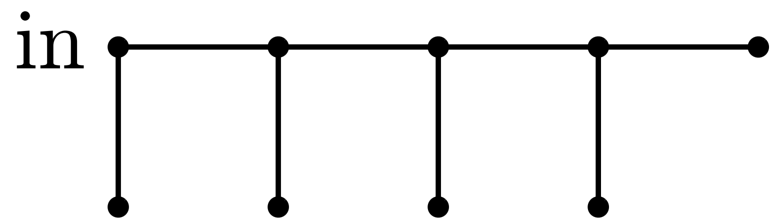
Proof idea:



Classical lower bound

Theorem: Any classical algorithm that makes at most $2^{n/6}$ queries to the black box finds the out vertex with probability at most $4 \cdot 2^{-n/6}$.

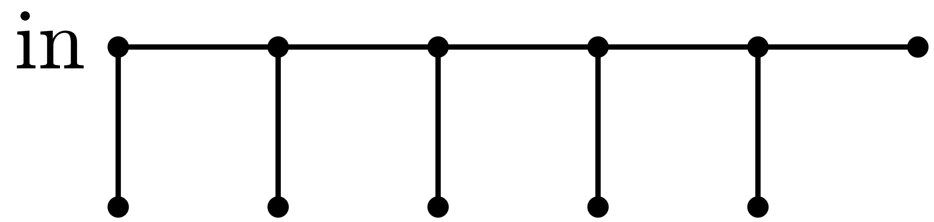
Proof idea:



Classical lower bound

Theorem: Any classical algorithm that makes at most $2^{n/6}$ queries to the black box finds the out vertex with probability at most $4 \cdot 2^{-n/6}$.

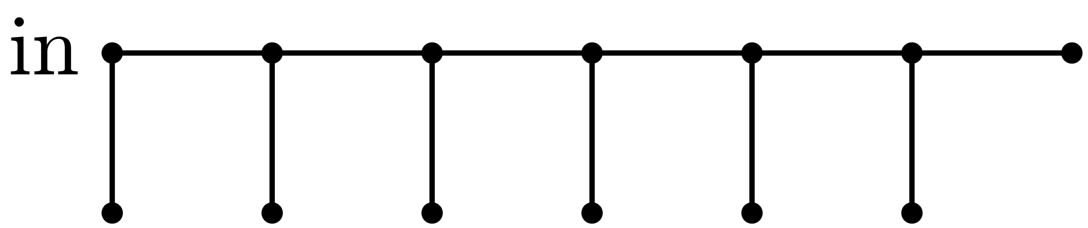
Proof idea:



Classical lower bound

Theorem: Any classical algorithm that makes at most $2^{n/6}$ queries to the black box finds the out vertex with probability at most $4 \cdot 2^{-n/6}$.

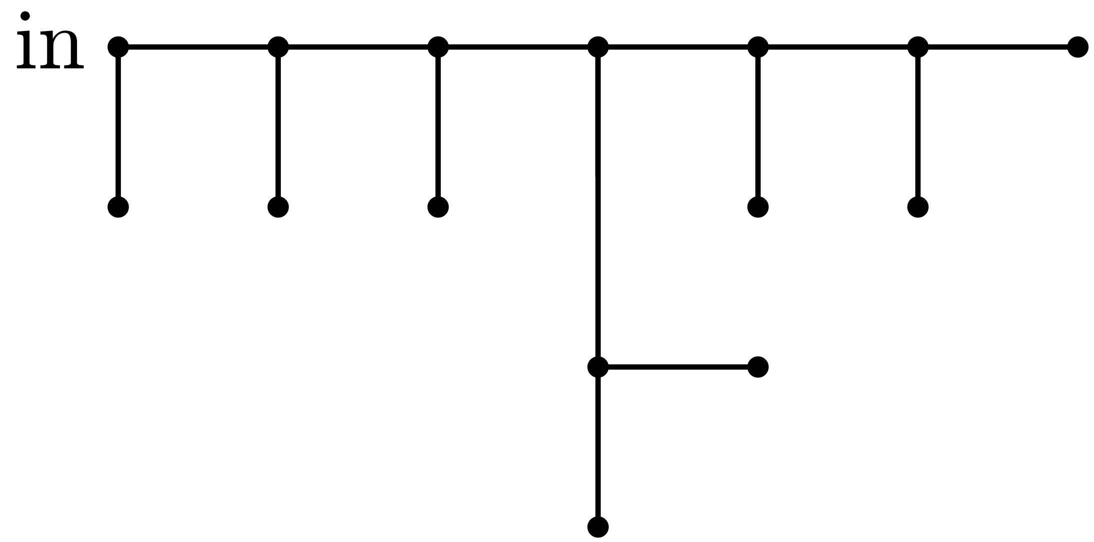
Proof idea:



Classical lower bound

Theorem: Any classical algorithm that makes at most $2^{n/6}$ queries to the black box finds the out vertex with probability at most $4 \cdot 2^{-n/6}$.

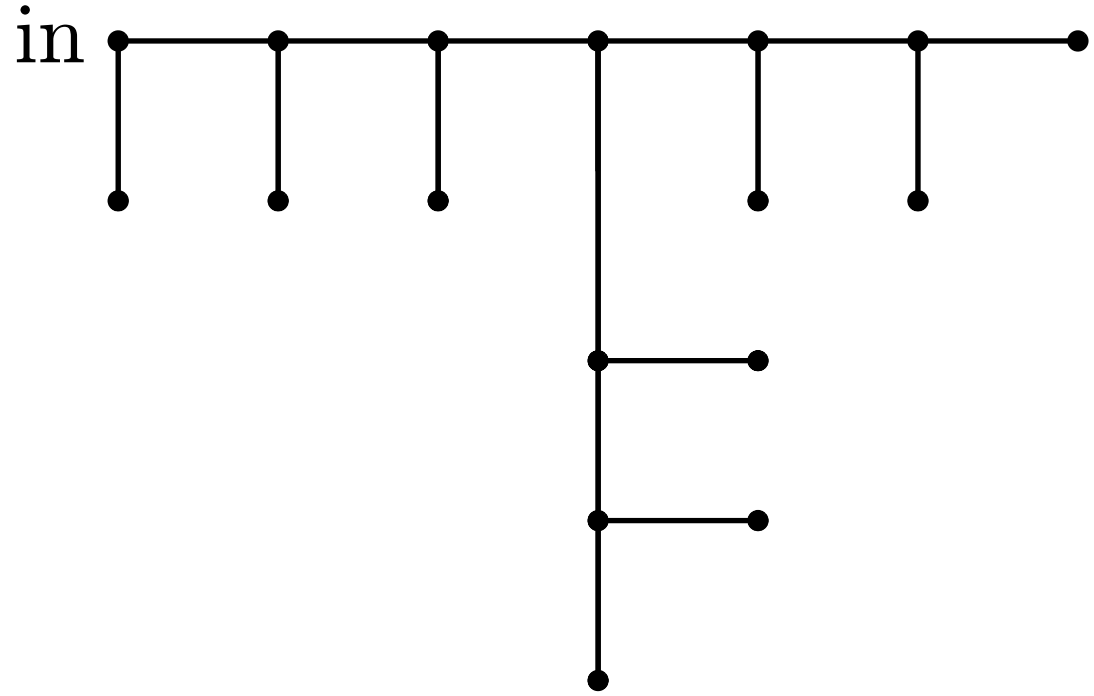
Proof idea:



Classical lower bound

Theorem: Any classical algorithm that makes at most $2^{n/6}$ queries to the black box finds the out vertex with probability at most $4 \cdot 2^{-n/6}$.

Proof idea:



3. Spatial search by quantum walk

- Childs and Goldstone, Phys. Rev. A **70**, 022314 (2004).
- Childs and Goldstone, Phys. Rev. A **70**, 042312 (2004).

Unstructured search

Search space: N items, $x \in \{1, 2, \ldots, N\}$

Goal: Find one *marked item*, w

Query: "is $w = x$?"

i.e., black box function $f(x) = \begin{cases} 0 & x \neq w \\ 1 & x = w \end{cases}$

Unstructured search

Search space: N items, $x \in \{1, 2, \ldots, N\}$

Goal: Find 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 complexity: $\Theta(N)$

Unstructured search

Search space: N items, $x \in \{1, 2, \ldots, N\}$

Goal: Find 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 complexity: $\Theta(N)$

Quantum algorithm [Grover 1996]: $O(\sqrt{N})$

Unstructured search

Search space: N items, $x \in \{1, 2, \ldots, N\}$

Goal: Find 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 complexity: $\Theta(N)$

Quantum algorithm [Grover 1996]: $O(\sqrt{N})$

Quantum lower bound [BBBV 1996]: $\Omega(\sqrt{N})$

Unstructured search

Search space: N items, $x \in \{1, 2, \ldots, N\}$

Goal: Find 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 complexity: $\Theta(N)$

Quantum algorithm [Grover 1996]: $O(\sqrt{N})$

Quantum lower bound [BBBV 1996]: $\Omega(\sqrt{N})$

Grover searching can be applied to many computational problems.
But can it be used to search a physical database, in which the N items are distributed in space?

Spatial search by quantum walk

Quantum walk approach: $H = -\gamma L - |w\rangle\langle w|$

$$L_{ab} = \begin{cases} 1 & ab \in G \\ -\deg(a) & a = b \\ 0 & \text{otherwise} \end{cases}$$

$$L \approx \nabla^2$$

Spatial search by quantum walk

Quantum walk approach: $H = -\gamma L - |w\rangle\langle w|$

$$L_{ab} = \begin{cases} 1 & ab \in G \\ -\deg(a) & a = b \\ 0 & \text{otherwise} \end{cases}$$

$$L \approx \nabla^2$$

Start in the state $|s\rangle = \dfrac{1}{\sqrt{N}} \sum_{x \in G} |x\rangle$.

Spatial search by quantum walk

Quantum walk approach: $H = -\gamma L - |w\rangle\langle w|$

$$L_{ab} = \begin{cases} 1 & ab \in G \\ -\deg(a) & a = b \\ 0 & \text{otherwise} \end{cases}$$

$$L \approx \nabla^2$$

Start in the state $|s\rangle = \frac{1}{\sqrt{N}} \sum_{x \in G} |x\rangle$.

Choose γ so that for as small a T as possible, $|\langle w|e^{-iHT}|s\rangle|^2$ is large.

Spatial search by quantum walk

Quantum walk approach: $H = -\gamma L - |w\rangle\langle w|$

$$L_{ab} = \begin{cases} 1 & ab \in G \\ -\deg(a) & a = b \\ 0 & \text{otherwise} \end{cases}$$

$$L \approx \nabla^2$$

Start in the state $|s\rangle = \dfrac{1}{\sqrt{N}} \sum_{x \in G} |x\rangle$.

Choose γ so that for as small a T as possible, $|\langle w|e^{-iHT}|s\rangle|^2$ is large.

γ

Spatial search by quantum walk

Quantum walk approach: $H = -\gamma L - |w\rangle\langle w|$

$$L_{ab} = \begin{cases} 1 & ab \in G \\ -\deg(a) & a = b \\ 0 & \text{otherwise} \end{cases}$$

$$L \approx \nabla^2$$

Start in the state $|s\rangle = \frac{1}{\sqrt{N}} \sum_{x \in G} |x\rangle$.

Choose γ so that for as small a T as possible, $|\langle w|e^{-iHT}|s\rangle|^2$ is large.

$$\gamma \to 0$$

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

ground state $\approx |w\rangle$

first excited state $\approx |s\rangle$

Spatial search by quantum walk

Quantum walk approach: $H = -\gamma L - |w\rangle\langle w|$

$$L_{ab} = \begin{cases} 1 & ab \in G \\ -\deg(a) & a = b \\ 0 & \text{otherwise} \end{cases}$$

$$L \approx \nabla^2$$

Start in the state $|s\rangle = \frac{1}{\sqrt{N}} \sum_{x \in G} |x\rangle$.

Choose γ so that for as small a T as possible, $|\langle w|e^{-iHT}|s\rangle|^2$ is large.

$$\gamma \to 0$$

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

ground state $\approx |w\rangle$

first excited state $\approx |s\rangle$

$$\gamma \to \infty$$

$$H \approx -\gamma L$$

ground state $\approx |s\rangle$

Spatial search by quantum walk

Quantum walk approach: $H = -\gamma L - |w\rangle\langle w|$

$$L_{ab} = \begin{cases} 1 & ab \in G \\ -\deg(a) & a = b \\ 0 & \text{otherwise} \end{cases}$$

$$L \approx \nabla^2$$

Start in the state $|s\rangle = \frac{1}{\sqrt{N}} \sum_{x \in G} |x\rangle$.

Choose γ so that for as small a T as possible, $|\langle w|e^{-iHT}|s\rangle|^2$ is large.

$\gamma \to 0$

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

ground state $\approx |w\rangle$

first excited state $\approx |s\rangle$

critical γ

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

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

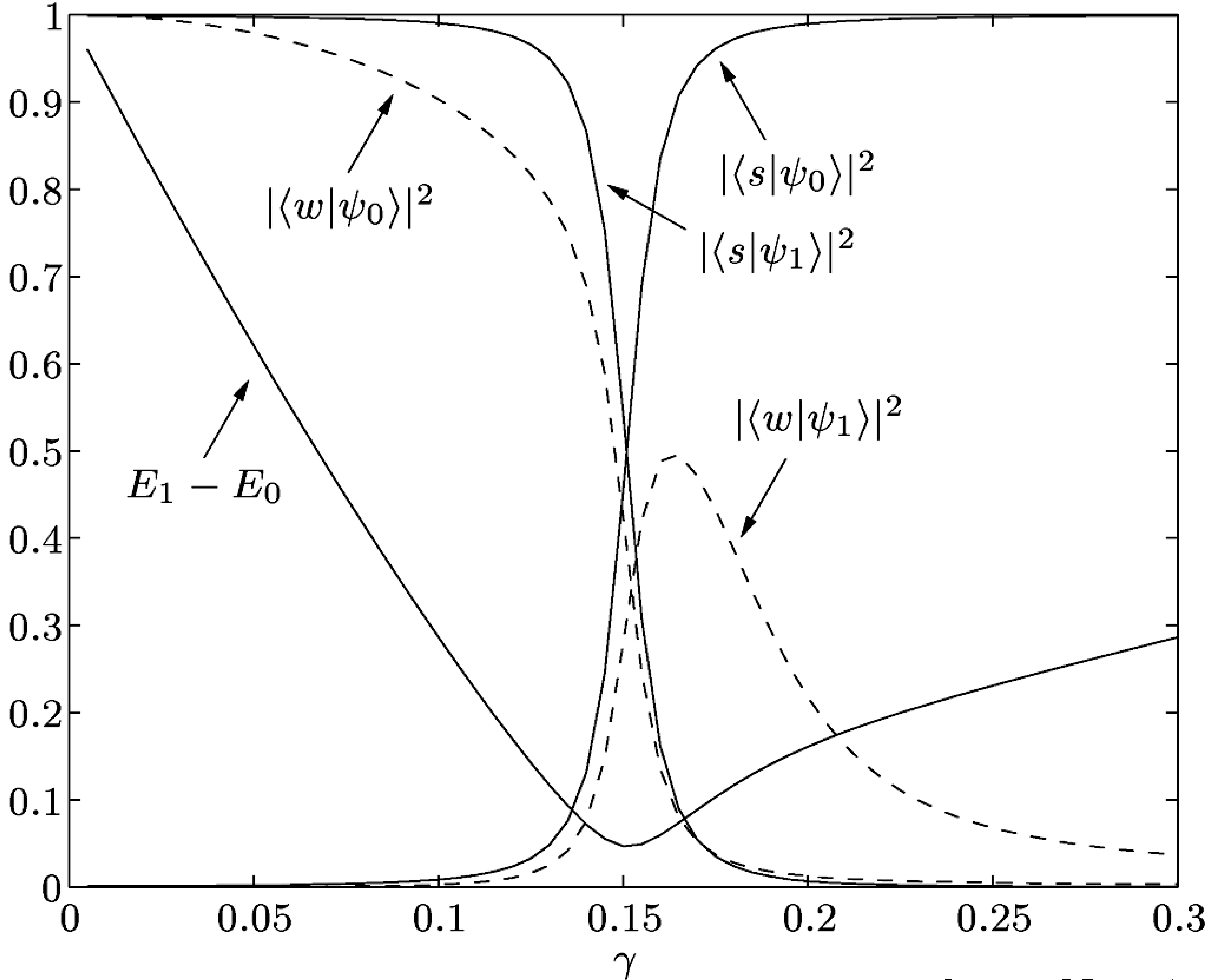
time $\approx \frac{1}{E_1 - E_0}$

$\gamma \to \infty$

$H \approx -\gamma L$

ground state $\approx |s\rangle$

γ



1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
0.05
0.1
0.15
0.2
0.25
0.3
γ
$|\langle w|\psi_0\rangle|^2$
$|\langle s|\psi_0\rangle|^2$
$|\langle s|\psi_1\rangle|^2$
$|\langle w|\psi_1\rangle|^2$
$E_1 - E_0$

$d = 4,\ N = 6^4 = 1296$

Results

Graph	Success amplitude	Run time
Complete	$1-o(1)$	$O(N^{1/2})$
Hypercube	$1-o(1)$	$O(N^{1/2})$
Lattice, $d>4$	$O(1)$	$O(N^{1/2})$
Lattice, $d=4$	$O(1/\log^{1/2} N)$	$O((N \log N)^{1/2})$
Lattice, $d=3$	$O(N^{-1/6})$	$O(N^{2/3})$
Lattice, $d=2$	$O((\log N/N)^{1/2})$	$O(N/\log N)$

Behavior for $d>4$

Critical γ: $\gamma_* = I_{1,d}$

Run time: $T = \dfrac{\pi\sqrt{I_{2,d}N}}{2I_{1,d}}$

Success probability: $|\langle w|e^{-iHT}|s\rangle|^2 = \dfrac{I_{1,d}^2}{I_{2,d}}$

where

$$I_{j,d} = \frac{1}{(2\pi)^d}\int_{-\pi}^{\pi}\frac{\mathrm{d}^d\vec{k}}{[\mathcal{E}(\vec{k})]^j}$$

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

Behavior for $d>4$

Critical γ: $\gamma_* = I_{1,d}$

Run time: $T = \dfrac{\pi\sqrt{I_{2,d} N}}{2 I_{1,d}}$

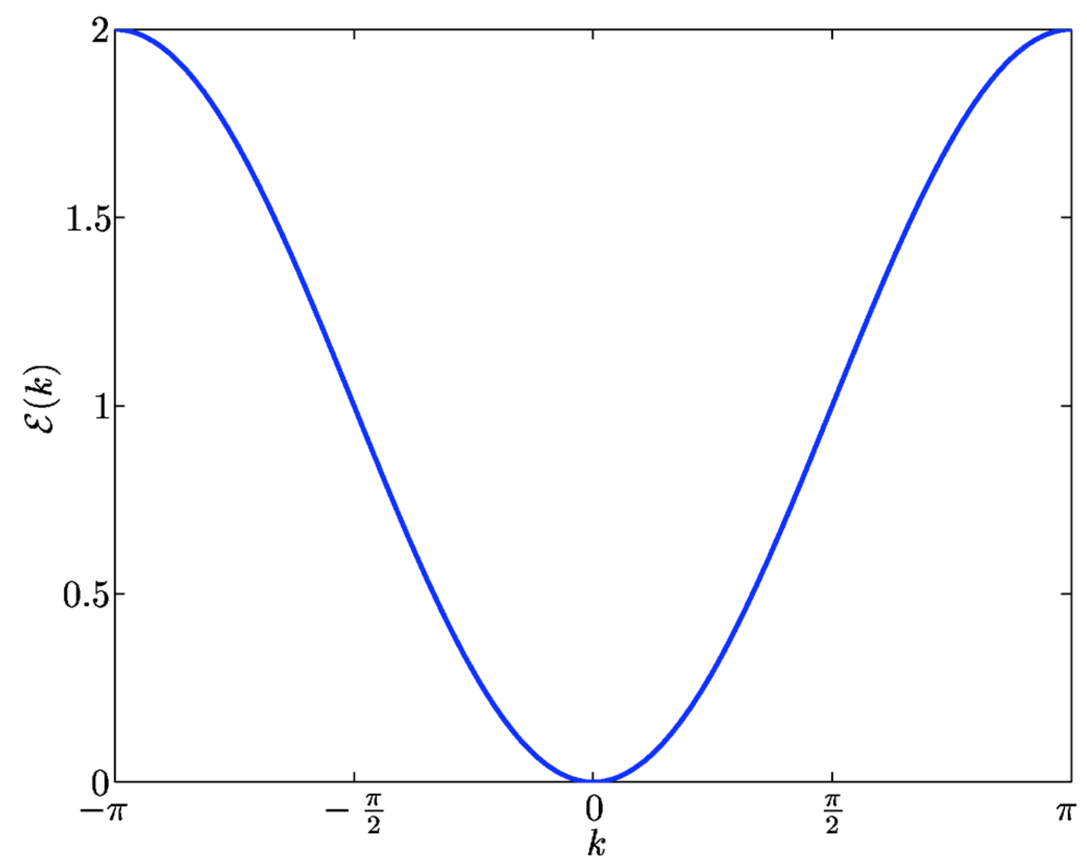
Success probability: $|\langle w|e^{-iHT}|s\rangle|^2 = \dfrac{I_{1,d}^2}{I_{2,d}}$

where

$$I_{j,d} = \frac{1}{(2\pi)^d}\int_{-\pi}^{\pi} \frac{\mathrm{d}^d\vec{k}}{[\mathcal{E}(\vec{k})]^j}$$

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

dispersion relation



Behavior for $d > 4$

Critical γ: $\gamma_* = I_{1,d}$

Run time: $T = \dfrac{\pi\sqrt{I_{2,d}N}}{2I_{1,d}}$

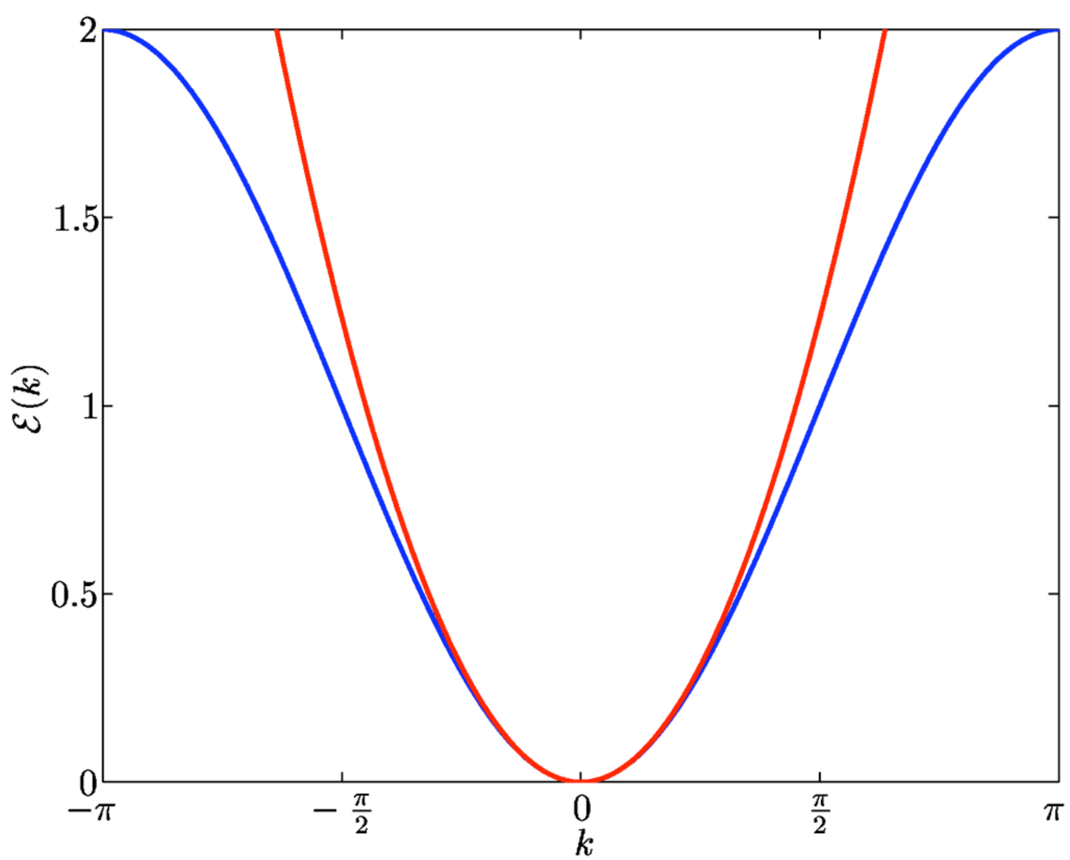
Success probability: $|\langle w|e^{-iHT}|s\rangle|^2 = \dfrac{I_{1,d}^2}{I_{2,d}}$

where

$$I_{j,d} = \frac{1}{(2\pi)^d}\int_{-\pi}^{\pi}\frac{\mathrm{d}^d\vec{k}}{[\mathcal{E}(\vec{k})]^j}$$

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

dispersion relation



Behavior for $d>4$

Critical γ: $\gamma_* = I_{1,d}$

Run time: $T = \dfrac{\pi\sqrt{I_{2,d}N}}{2I_{1,d}}$

Success probability: $|\langle w|e^{-iHT}|s\rangle|^2 = \dfrac{I_{1,d}^2}{I_{2,d}}$

where

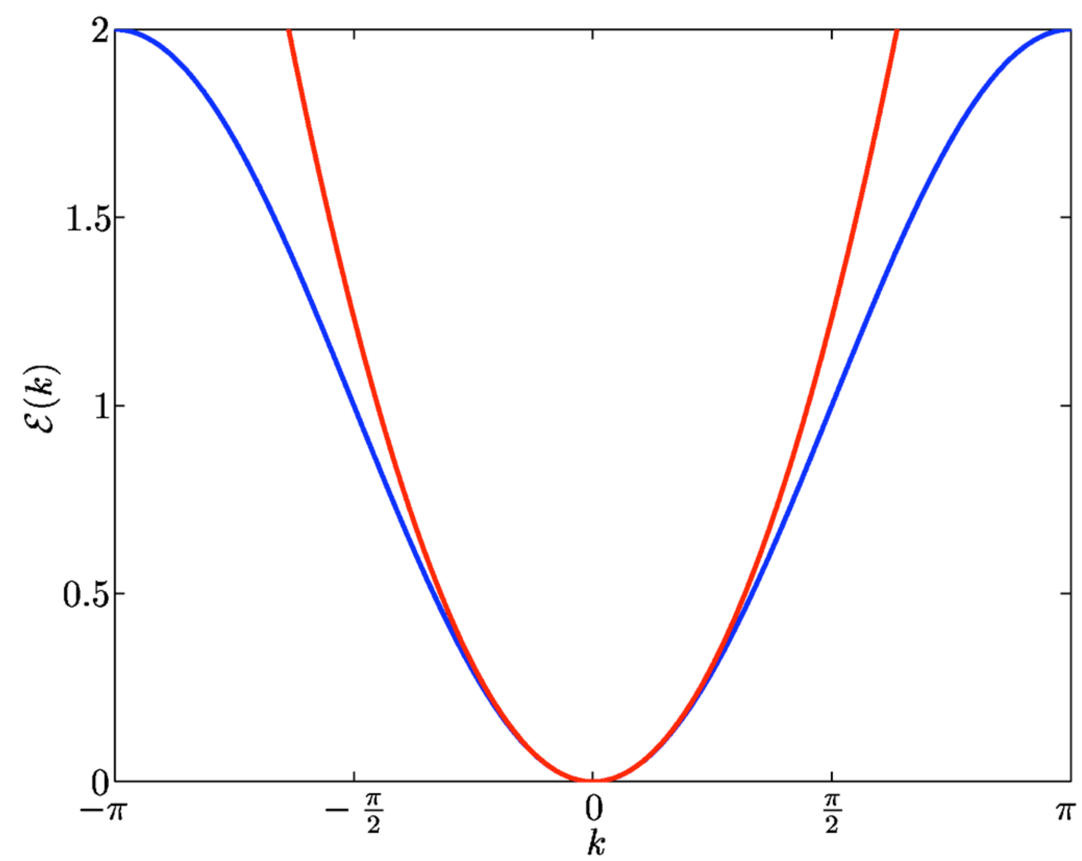
$$I_{j,d} = \frac{1}{(2\pi)^d}\int_{-\pi}^{\pi}\frac{\mathrm{d}^d\vec{k}}{[\mathcal{E}(\vec{k})]^j}$$

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

Note: $\displaystyle\int_0 \frac{\mathrm{d}^d\vec{k}}{|\vec{k}|^p} \sim \int_0 \frac{k^{d-1}\mathrm{d}k}{|\vec{k}|^p}$

converges for $d>p$.

dispersion relation



The Dirac equation: Faster search for $d=2,3,4$

Dirac Hamiltonian: $H_{\rm Dirac} = \sum_{j=1}^{d} \alpha_j p_j + \beta m \,, \quad p_j = i\frac{\rm d}{{\rm d}x_j}$

with $\{\alpha_j, \alpha_k\} = 2\delta_{jk} \,, \quad \{\alpha_j, \beta\} = 0 \,, \quad \beta^2 = 1$

The Dirac equation: Faster search for $d = 2, 3, 4$

Dirac Hamiltonian: $H_{\mathrm{Dirac}} = \sum_{j=1}^{d} \alpha_j p_j + \beta m \,, \quad p_j = i \frac{\mathrm{d}}{\mathrm{d}x_j}$

with $\{\alpha_j, \alpha_k\} = 2\delta_{jk} \,, \quad \{\alpha_j, \beta\} = 0 \,, \quad \beta^2 = 1$

Lattice version: $H_0 = \omega \sum_{j=1}^{d} \alpha_j P_j, \quad P_j |\vec{x}\rangle = \frac{i}{2}(|\vec{x} + \hat{e}_j\rangle - |\vec{x} - \hat{e}_j\rangle)$

$$\mathcal{E}(\vec{k}) = \pm\omega \sqrt{\sum_{j=1}^{d} \sin^2 k_j}$$

The Dirac equation: Faster search for $d = 2, 3, 4$

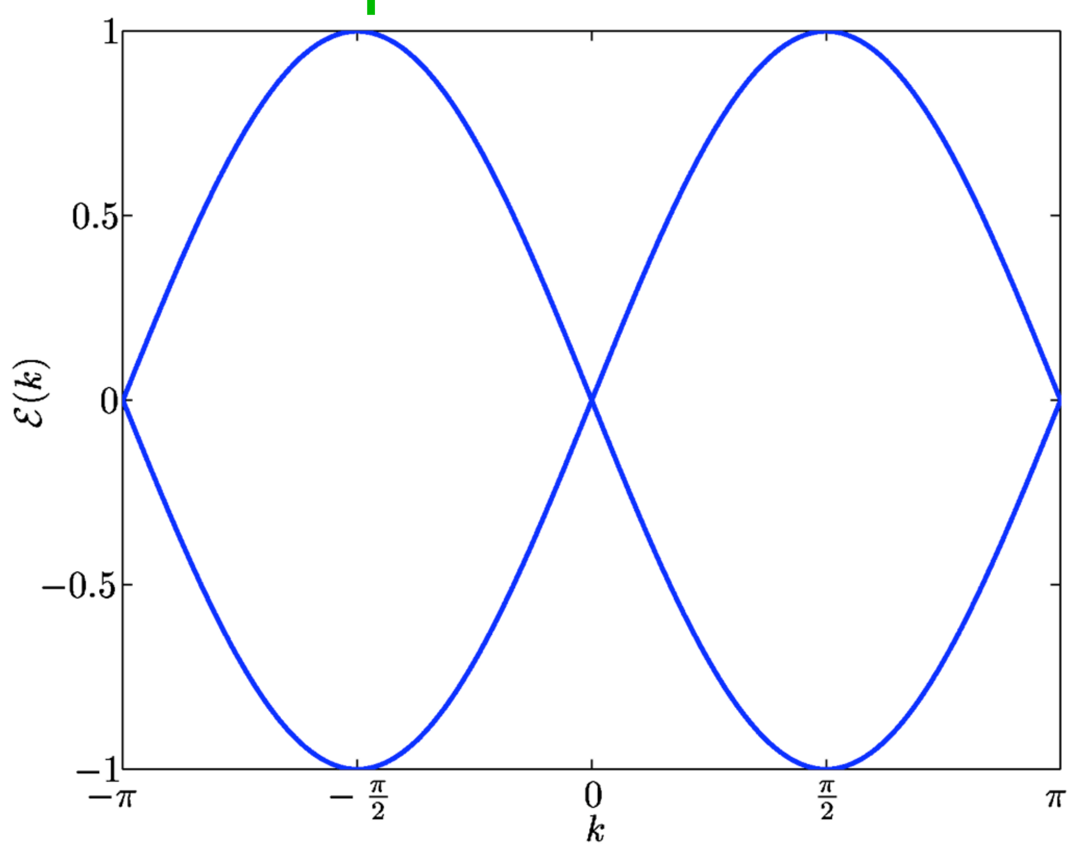
Dirac Hamiltonian: $H_{\mathrm{Dirac}} = \sum_{j=1}^{d} \alpha_j p_j + \beta m\,, \quad p_j = i \frac{\mathrm{d}}{\mathrm{d}x_j}$

with $\{\alpha_j, \alpha_k\} = 2\delta_{jk}\,, \quad \{\alpha_j, \beta\} = 0\,, \quad \beta^2 = 1$

Lattice version: $H_0 = \omega \sum_{j=1}^{d} \alpha_j P_j, \quad P_j |\vec{x}\rangle = \frac{i}{2}(|\vec{x} + \hat{e}_j\rangle - |\vec{x} - \hat{e}_j\rangle)$

$$\mathcal{E}(\vec{k}) = \pm\omega \sqrt{\sum_{j=1}^{d} \sin^2 k_j}$$

dispersion relation



The Dirac equation: Faster search for $d=2,3,4$

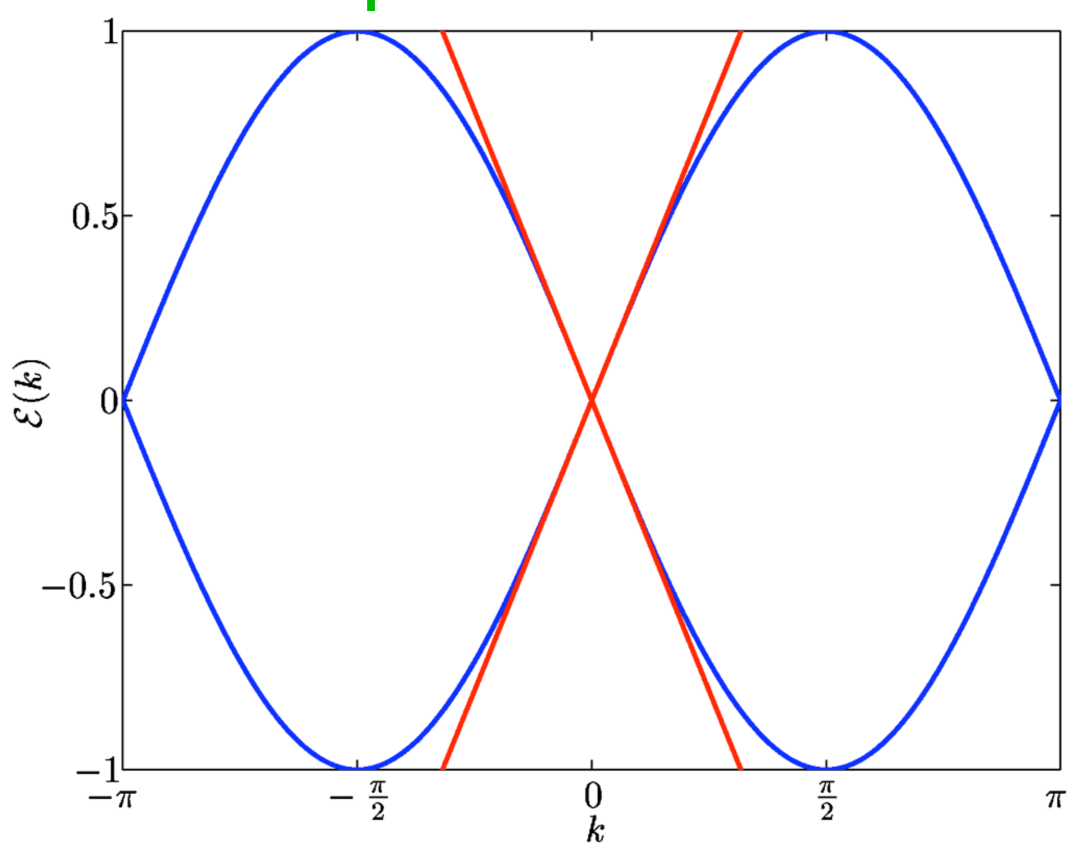
Dirac Hamiltonian: $H_{\mathrm{Dirac}} = \sum_{j=1}^{d} \alpha_j p_j + \beta m\,, \quad p_j = i\frac{\mathrm{d}}{\mathrm{d}x_j}$

with $\{\alpha_j, \alpha_k\} = 2\delta_{jk}\,, \quad \{\alpha_j, \beta\} = 0\,, \quad \beta^2 = 1$

Lattice version: $H_0 = \omega \sum_{j=1}^{d} \alpha_j P_j, \quad P_j|\vec{x}\rangle = \frac{i}{2}(|\vec{x}+\hat{e}_j\rangle - |\vec{x}-\hat{e}_j\rangle)$

$$\mathcal{E}(\vec{k}) = \pm\omega\sqrt{\sum_{j=1}^{d} \sin^2 k_j}$$

dispersion relation



The Dirac equation: Faster search for $d=2,3,4$

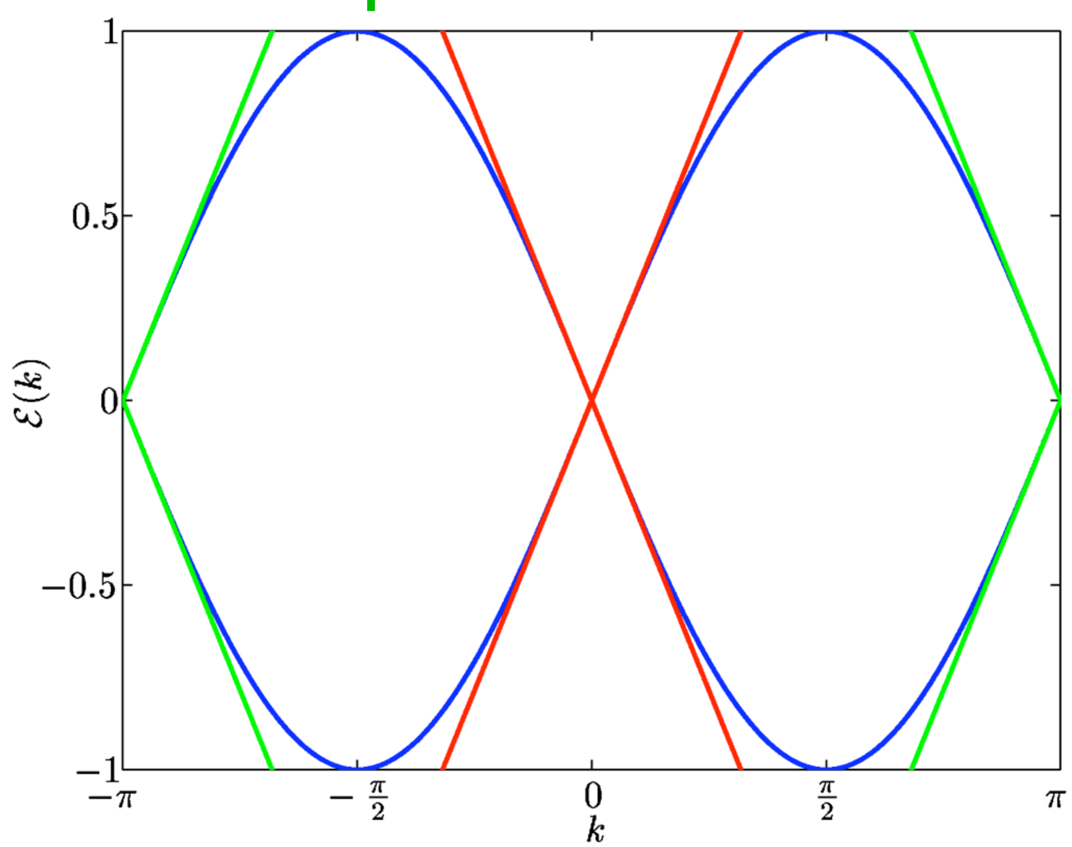
Dirac Hamiltonian: $H_{\rm Dirac} = \sum_{j=1}^{d} \alpha_j p_j + \beta m\,, \quad p_j = i\frac{\mathrm{d}}{\mathrm{d}x_j}$

with $\{\alpha_j, \alpha_k\} = 2\delta_{jk}\,, \quad \{\alpha_j, \beta\} = 0\,, \quad \beta^2 = 1$

Lattice version: $H_0 = \omega \sum_{j=1}^{d} \alpha_j P_j, \quad P_j|\vec{x}\rangle = \frac{i}{2}(|\vec{x}+\hat{e}_j\rangle - |\vec{x}-\hat{e}_j\rangle)$

$$\mathcal{E}(\vec{k}) = \pm\omega\sqrt{\sum_{j=1}^{d} \sin^2 k_j}$$

dispersion relation



The Dirac equation: Faster search for $d=2,3,4$

Dirac Hamiltonian: $H_{\mathrm{Dirac}} = \sum_{j=1}^{d} \alpha_j p_j + \beta m \,, \quad p_j = i \frac{\mathrm{d}}{\mathrm{d}x_j}$

with $\{\alpha_j, \alpha_k\} = 2\delta_{jk} \,, \quad \{\alpha_j, \beta\} = 0 \,, \quad \beta^2 = 1$

Improved: $H_0 = \omega \sum_{j=1}^{d} \alpha_j P_j + \gamma \beta L$

$$\mathcal{E}(\vec{k}) = \pm \sqrt{\omega^2 \sum_{j=1}^{d} \sin^2 k_j + \gamma^2 [2 \sum_{j=1}^{d} (1 - \cos k_j)]^2}$$

The Dirac equation: Faster search for $d = 2, 3, 4$

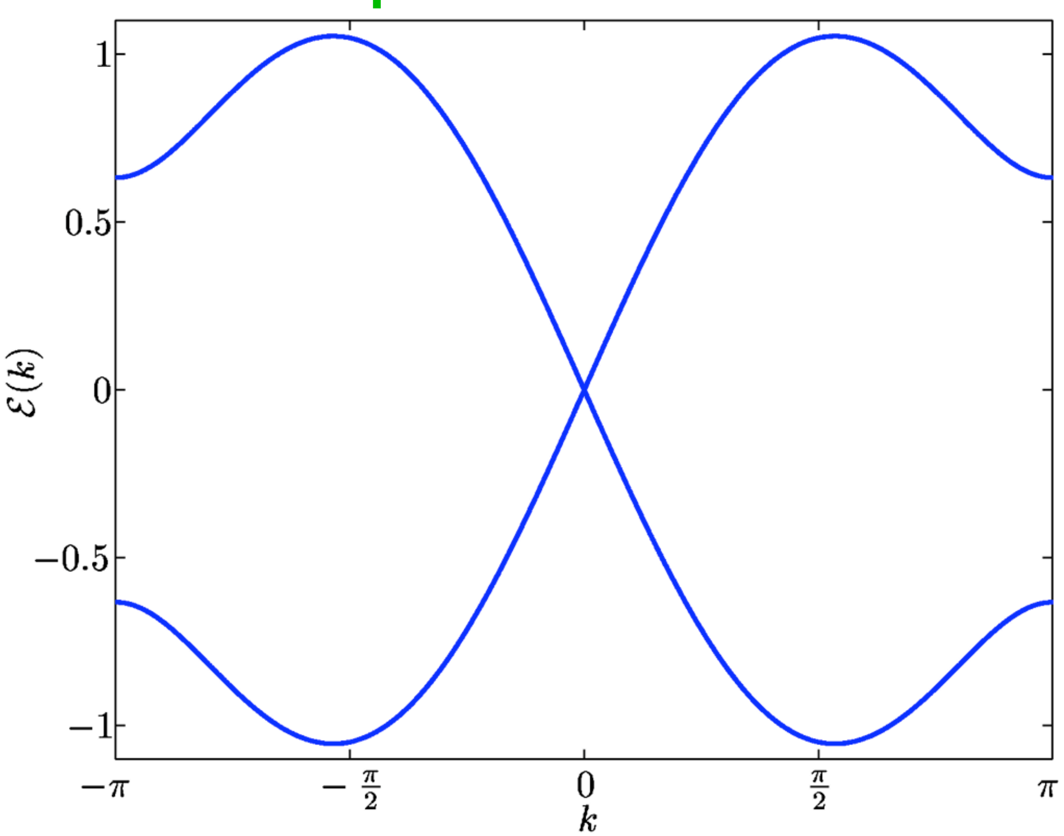
Dirac Hamiltonian: $H_{\mathrm{Dirac}} = \sum_{j=1}^{d} \alpha_j p_j + \beta m \,, \quad p_j = i\frac{\mathrm{d}}{\mathrm{d}x_j}$

with $\{\alpha_j, \alpha_k\} = 2\delta_{jk} \,, \quad \{\alpha_j, \beta\} = 0 \,, \quad \beta^2 = 1$

Improved: $H_0 = \omega \sum_{j=1}^{d} \alpha_j P_j + \gamma \beta L$

$$\mathcal{E}(\vec{k}) = \pm\sqrt{\omega^2 \sum_{j=1}^{d} \sin^2 k_j + \gamma^2 [2 \sum_{j=1}^{d} (1 - \cos k_j)]^2}$$

dispersion relation



The Dirac equation: Faster search for $d=2,3,4$

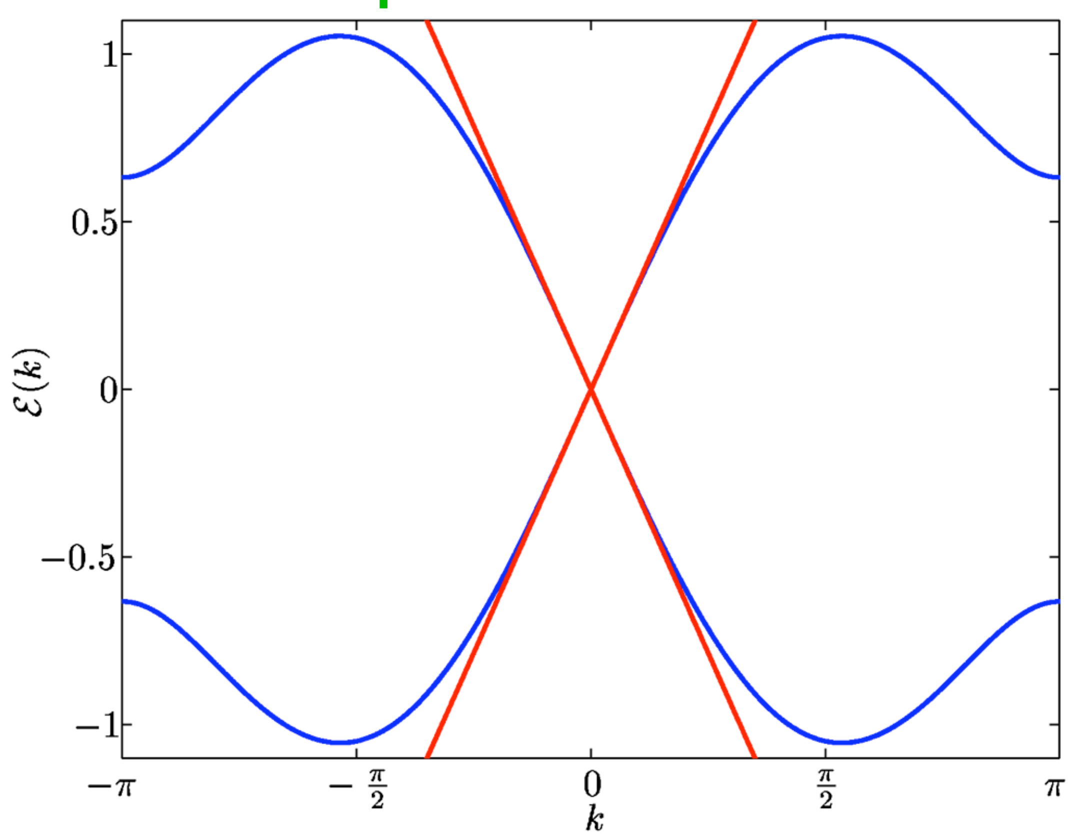
Dirac Hamiltonian: $H_{\mathrm{Dirac}} = \sum_{j=1}^{d} \alpha_j p_j + \beta m\,, \quad p_j = i\frac{\mathrm{d}}{\mathrm{d}x_j}$

with $\{\alpha_j, \alpha_k\} = 2\delta_{jk}\,, \quad \{\alpha_j, \beta\} = 0\,, \quad \beta^2 = 1$

Improved: $H_0 = \omega \sum_{j=1}^{d} \alpha_j P_j + \gamma\beta L$

$$\mathcal{E}(\vec{k}) = \pm\sqrt{\omega^2 \sum_{j=1}^{d} \sin^2 k_j + \gamma^2 [2\sum_{j=1}^{d}(1-\cos k_j)]^2}$$

dispersion relation



The Dirac equation: Faster search for $d = 2, 3, 4$

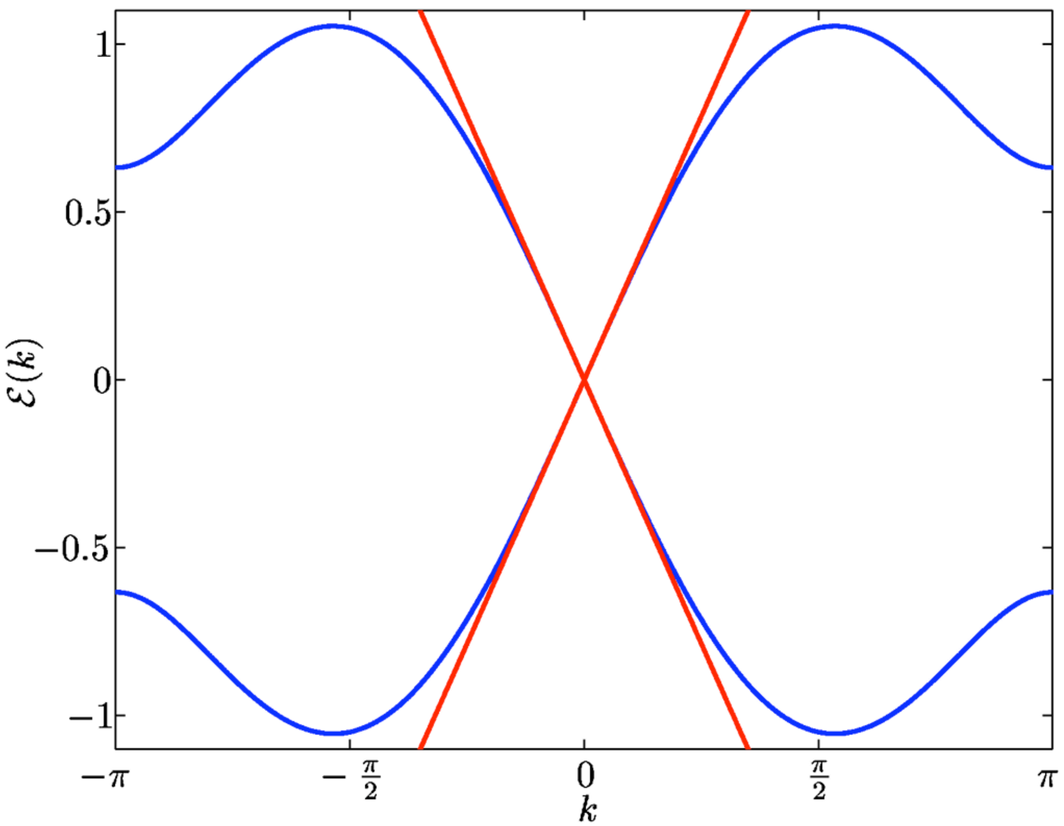
Dirac Hamiltonian: $H_{\text{Dirac}} = \sum_{j=1}^{d} \alpha_j p_j + \beta m\,, \quad p_j = i\frac{\mathrm{d}}{\mathrm{d}x_j}$

with $\{\alpha_j, \alpha_k\} = 2\delta_{jk}\,, \quad \{\alpha_j, \beta\} = 0\,, \quad \beta^2 = 1$

Improved: $H_0 = \omega \sum_{j=1}^{d} \alpha_j P_j + \gamma \beta L$

$$\mathcal{E}(\vec{k}) = \pm\sqrt{\omega^2 \sum_{j=1}^{d} \sin^2 k_j + \gamma^2 [2 \sum_{j=1}^{d} (1 - \cos k_j)]^2}$$

dispersion relation



Algorithm:

Let $H = H_0 - \beta|w\rangle\langle w|$.
Start in $|\eta, s\rangle$.
Choose constants ω, γ so that for as small a T as possible, $|\langle \eta, w|e^{-iHT}|\eta, s\rangle|^2$ is large.

The Dirac equation: Faster search for $d = 2, 3, 4$

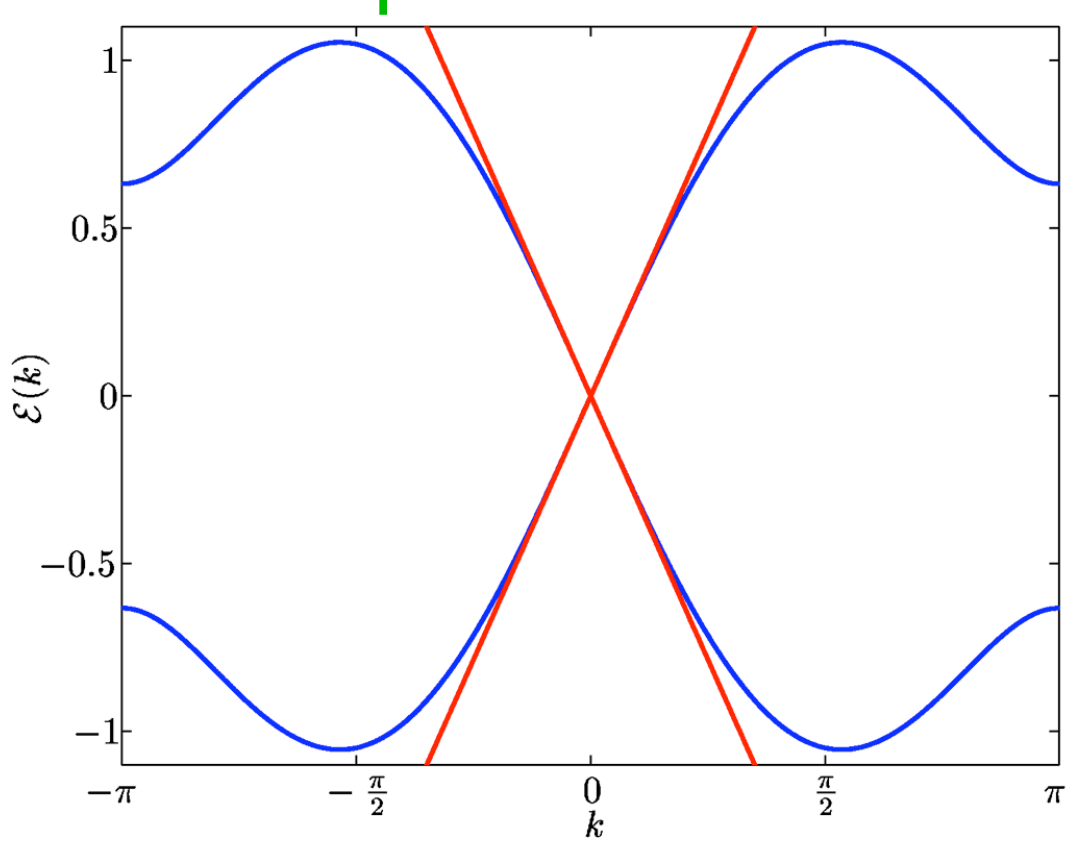
Dirac Hamiltonian: $H_{\mathrm{Dirac}} = \sum_{j=1}^{d} \alpha_j p_j + \beta m\,, \qquad p_j = i\frac{\mathrm{d}}{\mathrm{d}x_j}$

with $\{\alpha_j, \alpha_k\} = 2\delta_{jk}\,, \quad \{\alpha_j, \beta\} = 0\,, \quad \beta^2 = 1$

Improved: $H_0 = \omega \sum_{j=1}^{d} \alpha_j P_j + \gamma \beta L$

$$\mathcal{E}(\vec{k}) = \pm\sqrt{\omega^2 \sum_{j=1}^{d} \sin^2 k_j + \gamma^2 [2 \sum_{j=1}^{d} (1 - \cos k_j)]^2}$$

dispersion relation

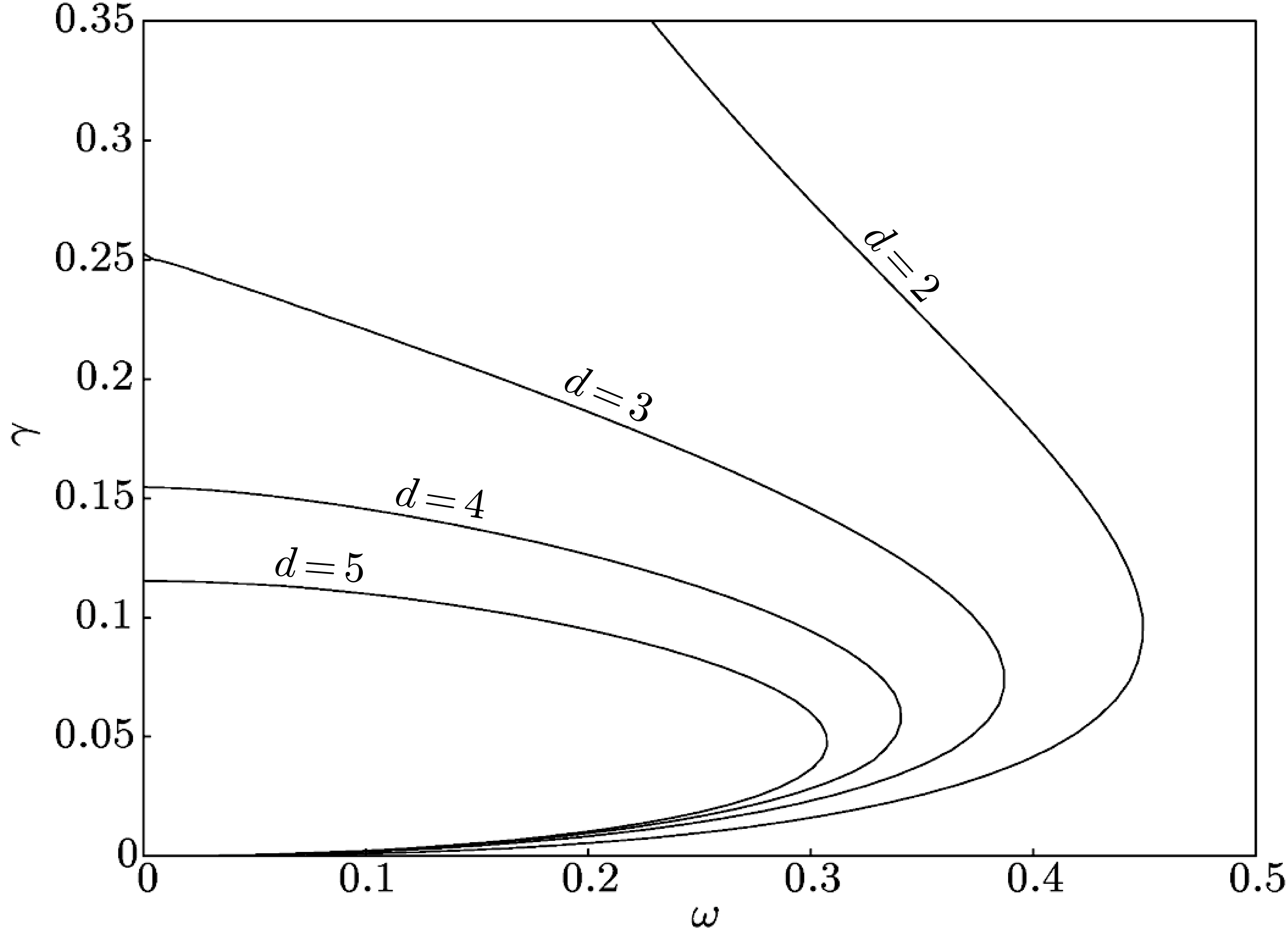


Algorithm:

Let $H = H_0 - \beta|w\rangle\langle w|$.
Start in $|\eta, s\rangle$.
Choose constants ω, γ so that for as small a T as possible, $|\langle \eta, w|e^{-iHT}|\eta, s\rangle|^2$ is large.

Run time:

$O(\sqrt{N})$ in $d > 2$,
$O(\sqrt{N} \log N)$ in $d = 2$.



0.35
0.3
0.25
0.2
0.15
0.1
0.05
0
0
0.1
0.2
0.3
0.4
0.5
γ
ω
$d=2$
$d=3$
$d=4$
$d=5$

4. Evaluating Boolean formulas

- Childs, Cleve, Jordan, and Yeung, quant-ph/0702160.
- Childs, Reichardt, Špalek, and Zhang, quant-ph/0703015.

Evaluating Boolean formulas

Fix a Boolean formula on N variables.

Given a black box for determining how the variables are assigned, how many variables must we query to determine the value of the formula?

Evaluating Boolean formulas

Fix a Boolean formula on N variables.

Given a black box for determining how the variables are assigned, how many variables must we query to determine the value of the formula?

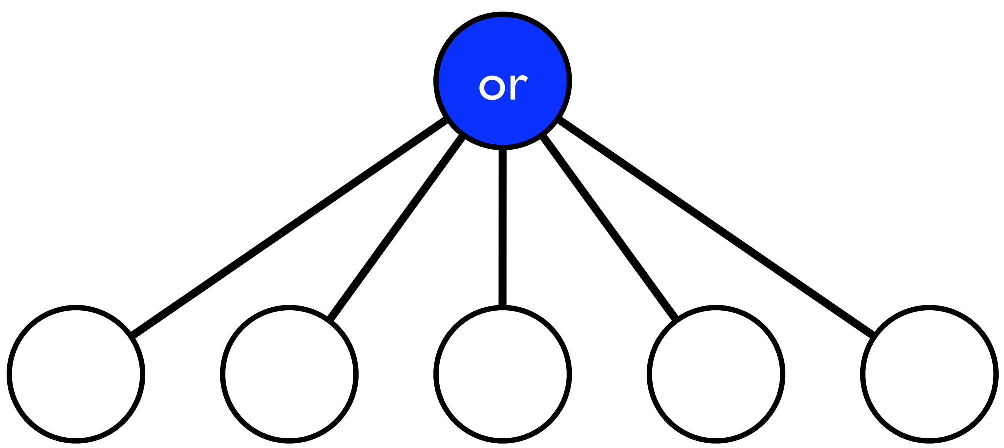
Example: Unstructured search (OR)

Evaluating Boolean formulas

Fix a Boolean formula on N variables.

Given a black box for determining how the variables are assigned, how many variables must we query to determine the value of the formula?

Example: Unstructured search (OR)

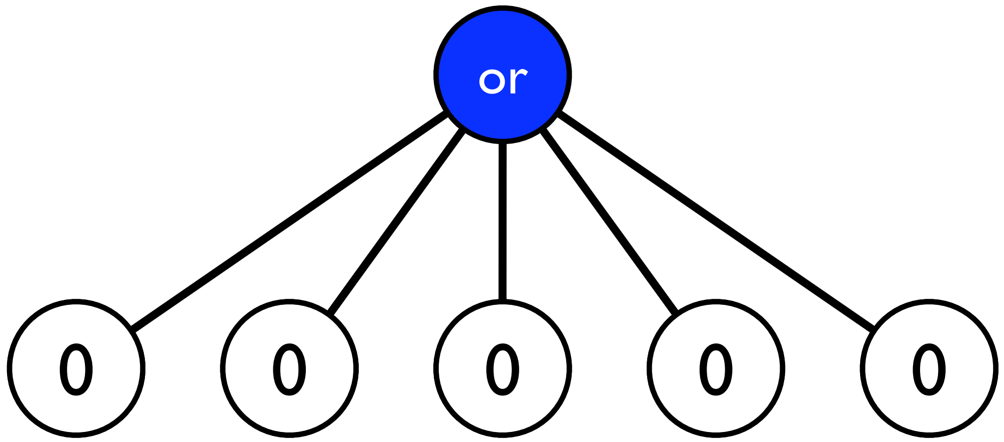


Evaluating Boolean formulas

Fix a Boolean formula on N variables.

Given a black box for determining how the variables are assigned, how many variables must we query to determine the value of the formula?

Example: Unstructured search (OR)

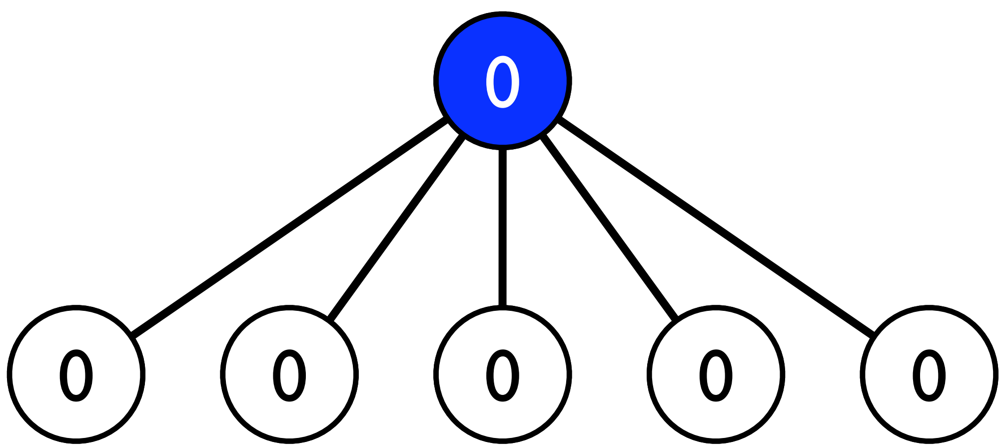


Evaluating Boolean formulas

Fix a Boolean formula on N variables.

Given a black box for determining how the variables are assigned, how many variables must we query to determine the value of the formula?

Example: Unstructured search (OR)

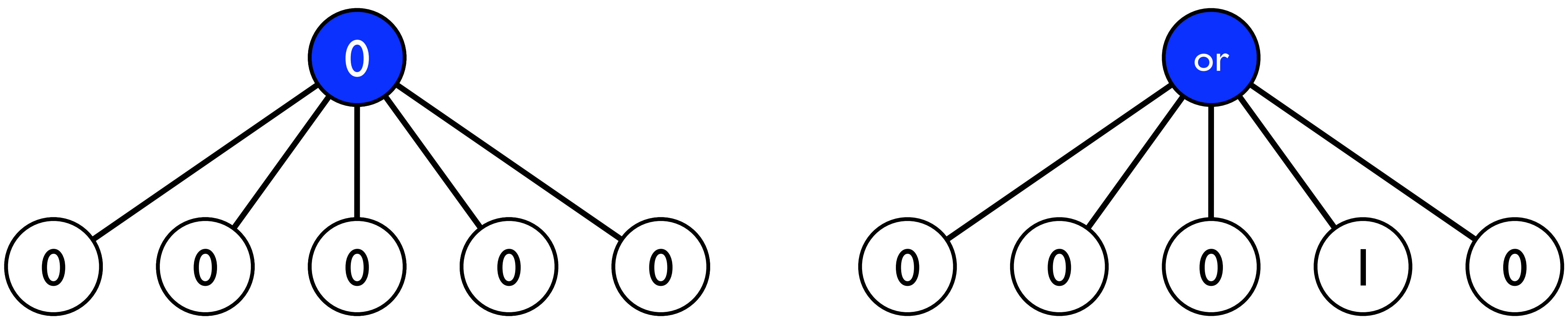


Evaluating Boolean formulas

Fix a Boolean formula on N variables.

Given a black box for determining how the variables are assigned, how many variables must we query to determine the value of the formula?

Example: Unstructured search (OR)

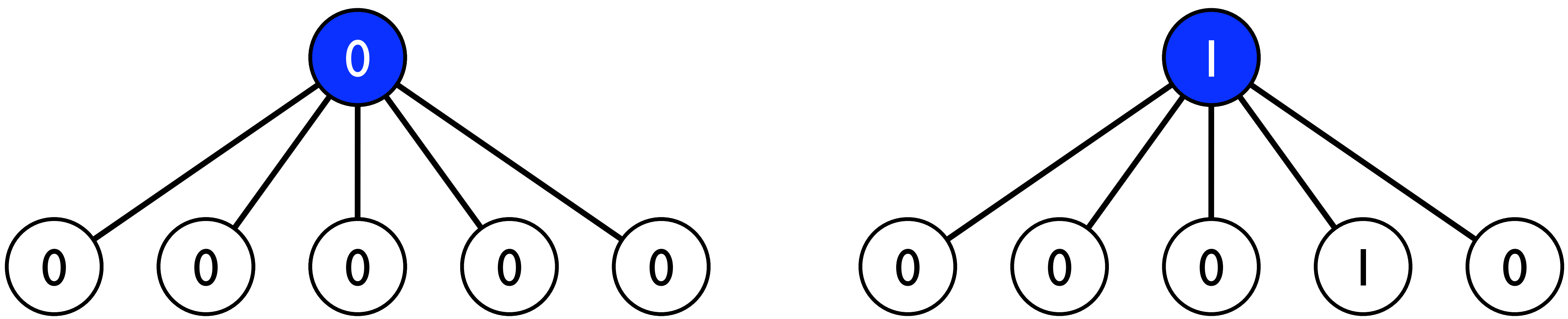


Evaluating Boolean formulas

Fix a Boolean formula on N variables.

Given a black box for determining how the variables are assigned, how many variables must we query to determine the value of the formula?

Example: Unstructured search (OR)

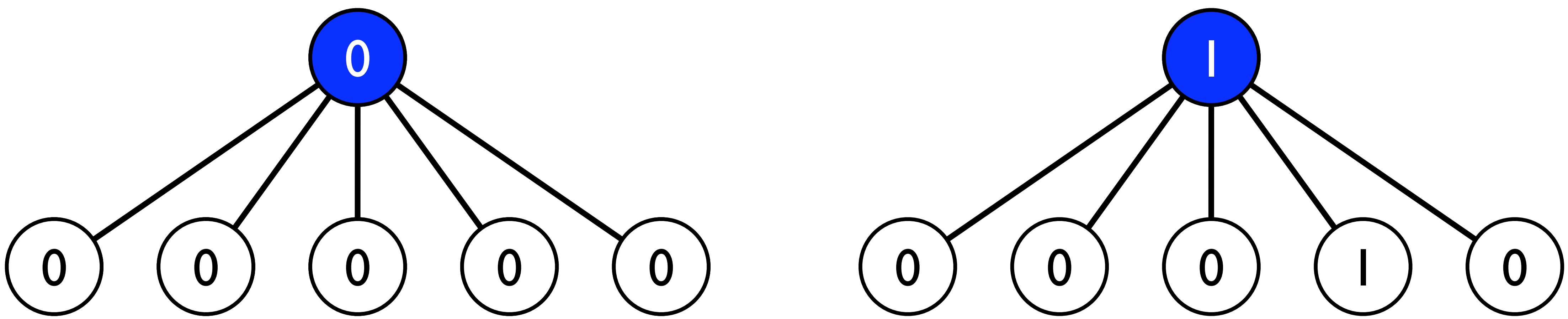


Evaluating Boolean formulas

Fix a Boolean formula on N variables.

Given a black box for determining how the variables are assigned, how many variables must we query to determine the value of the formula?

Example: Unstructured search (OR)

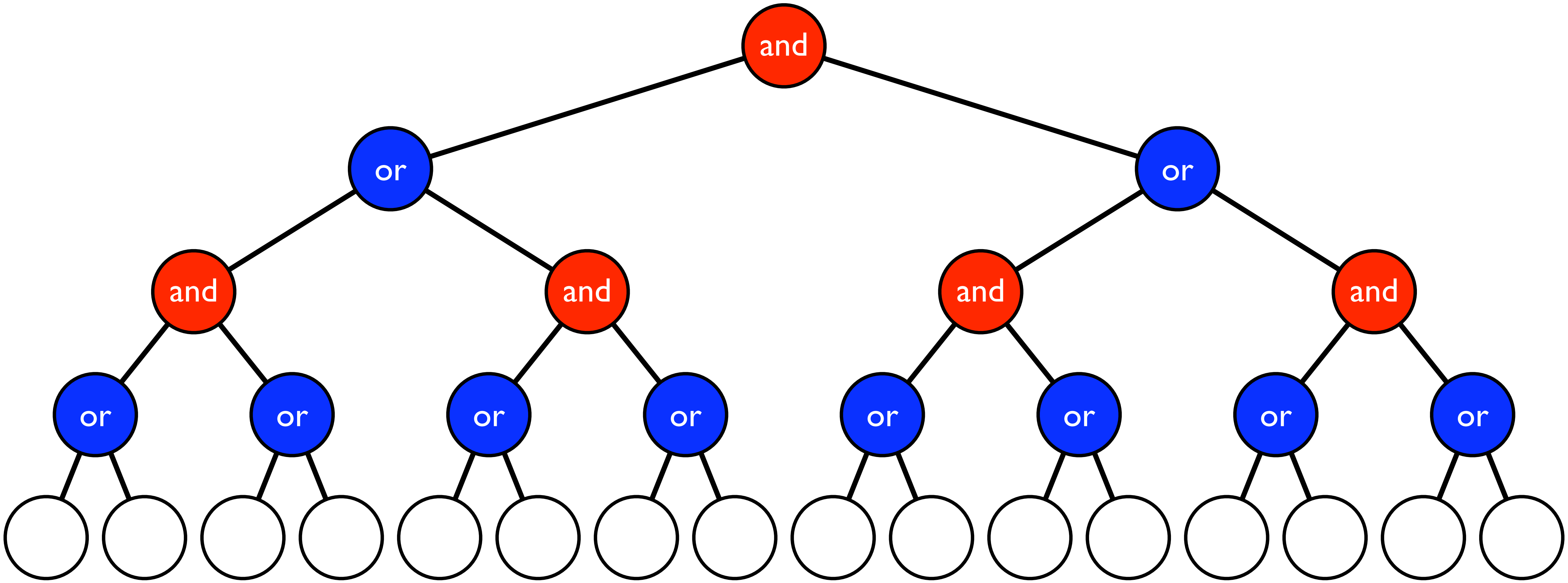


Classical complexity: $\Theta(N)$

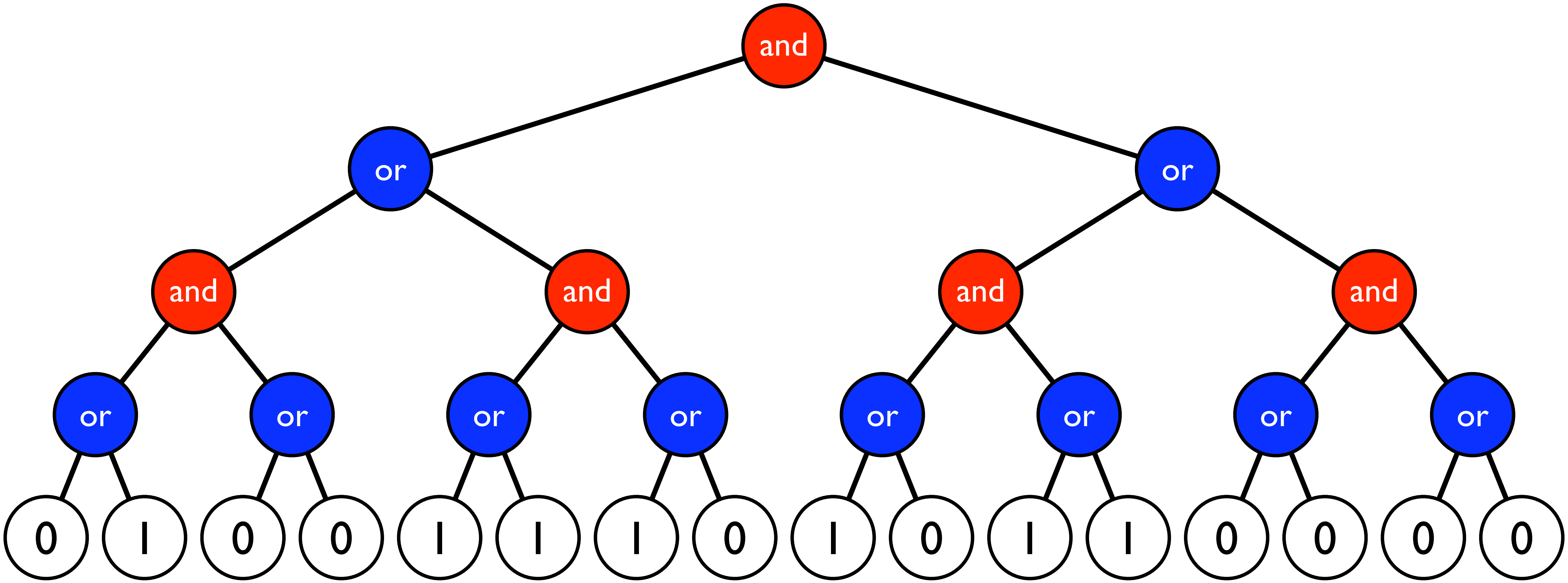
Quantum algorithm [Grover 1996]: $O(\sqrt{N})$

Quantum lower bound [BBBV 1996]: $\Omega(\sqrt{N})$

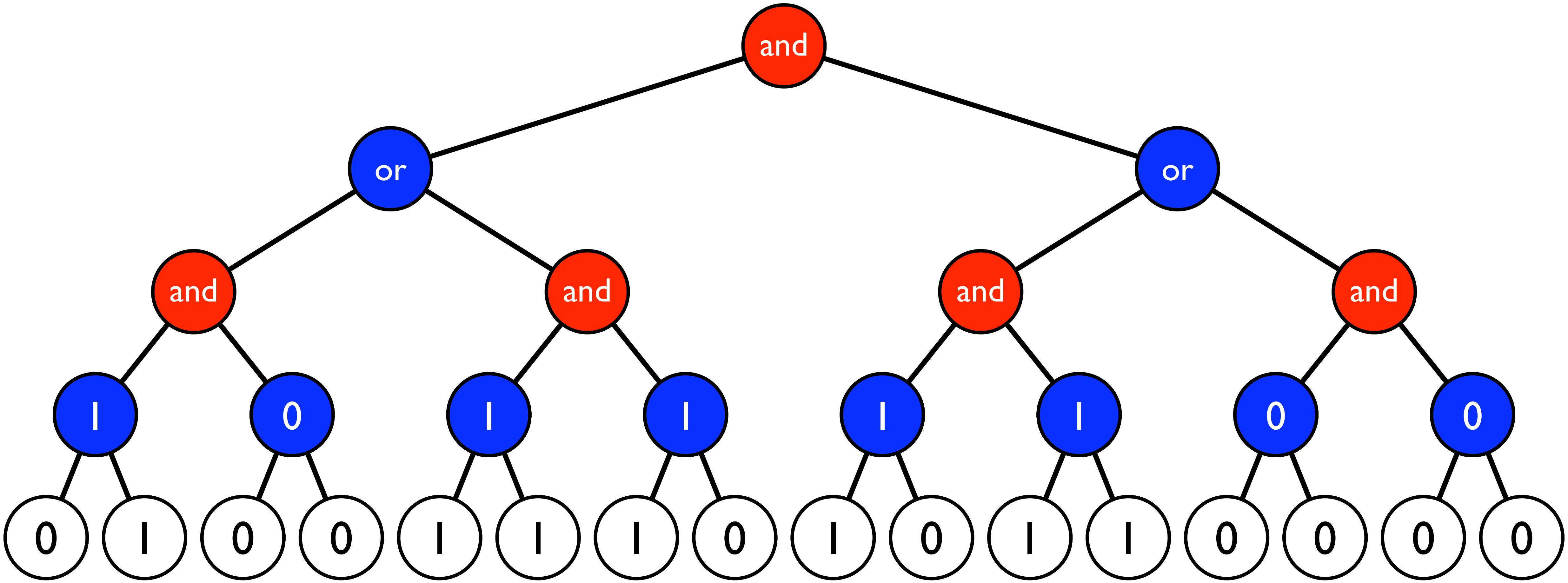
AND-OR trees (aka game trees)



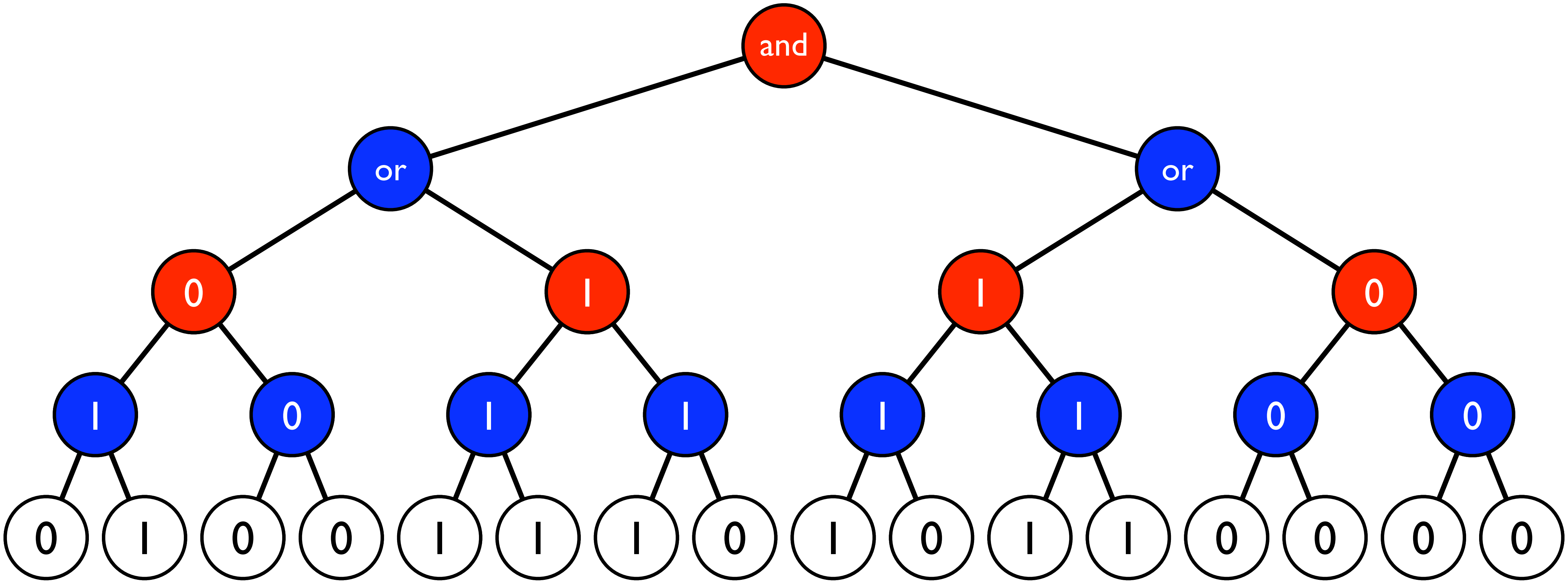
AND-OR trees (aka game trees)



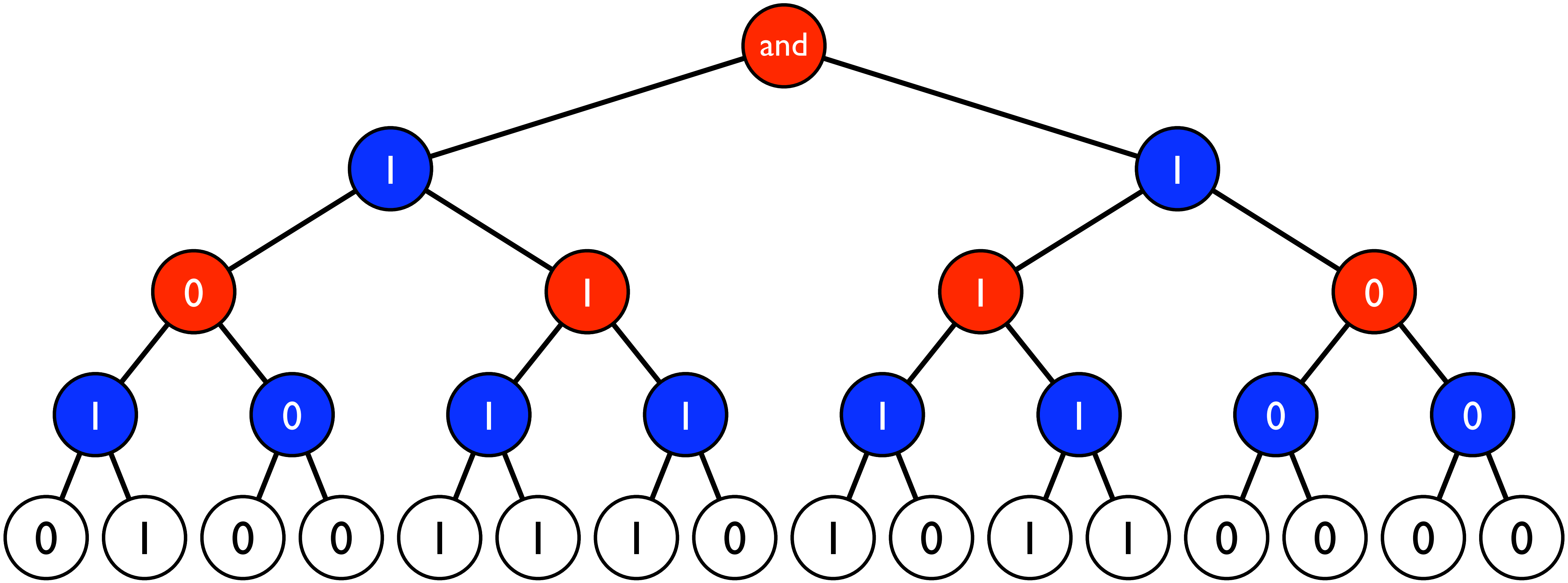
AND-OR trees (aka game trees)



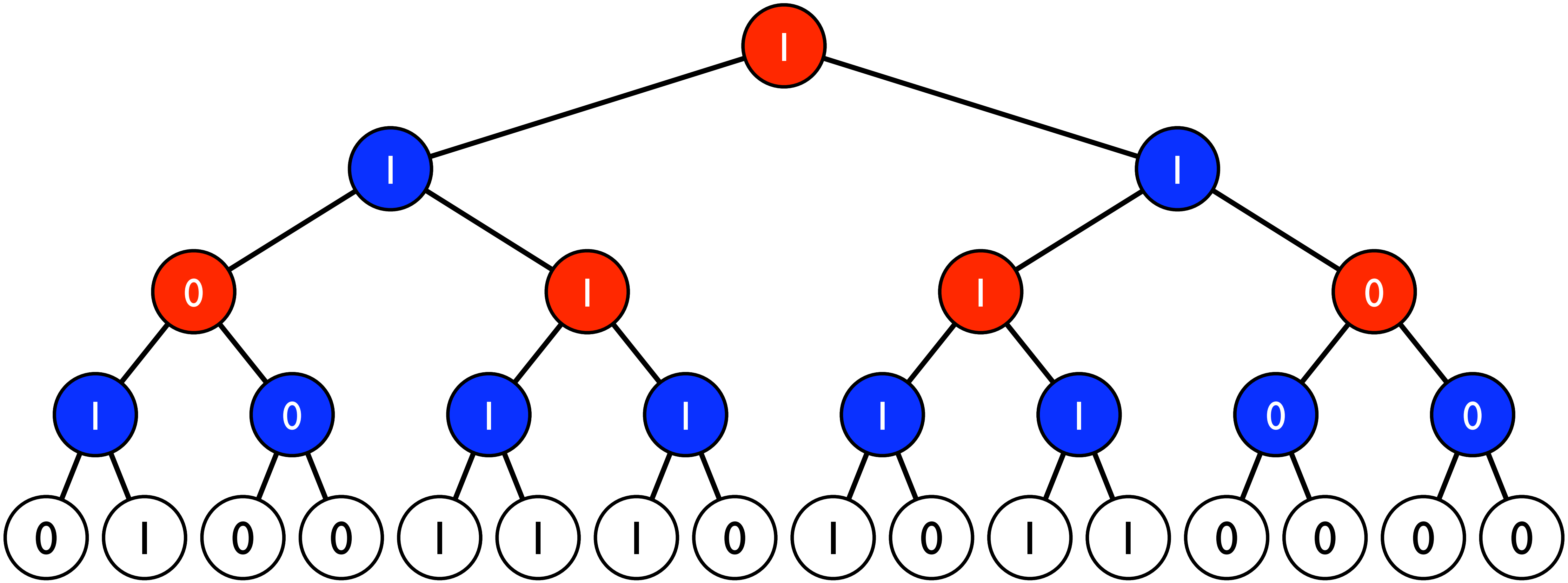
AND-OR trees (aka game trees)



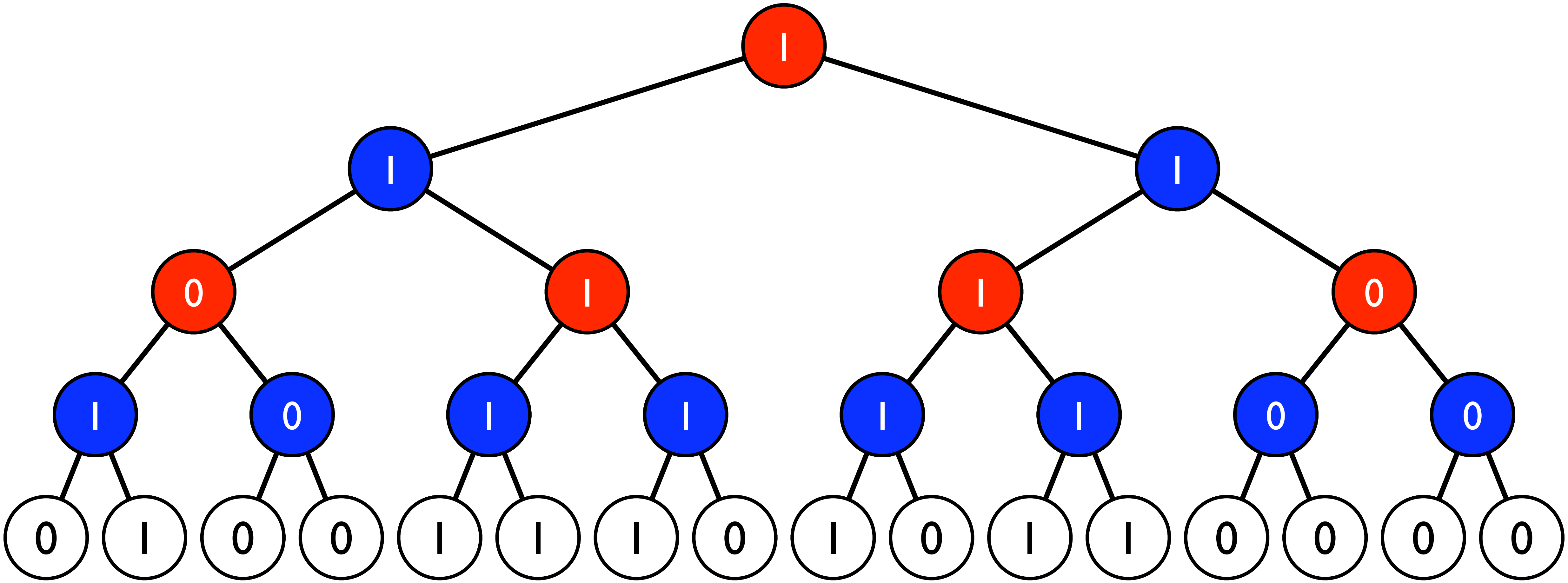
AND-OR trees (aka game trees)



AND-OR trees (aka game trees)

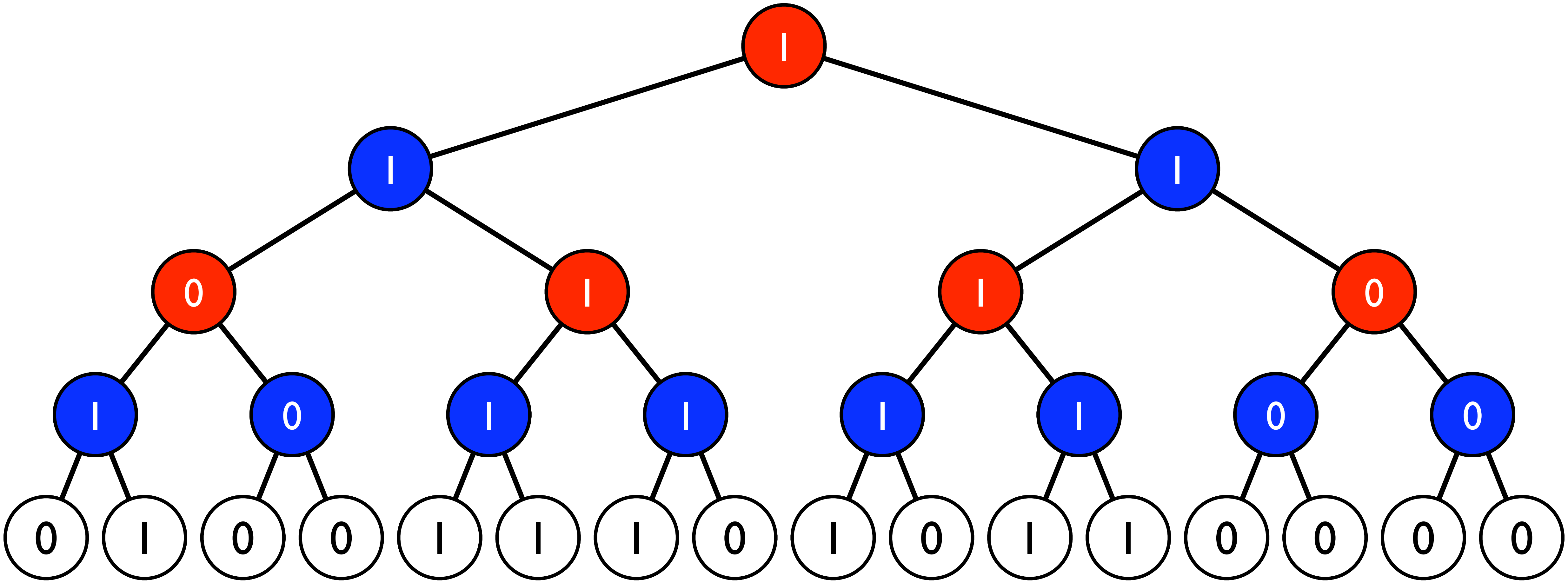


AND-OR trees (aka game trees)



Classical complexity: $\Theta(N^{0.753})$ [Snir 85, Saks-Wigderson 86, Santha 95]

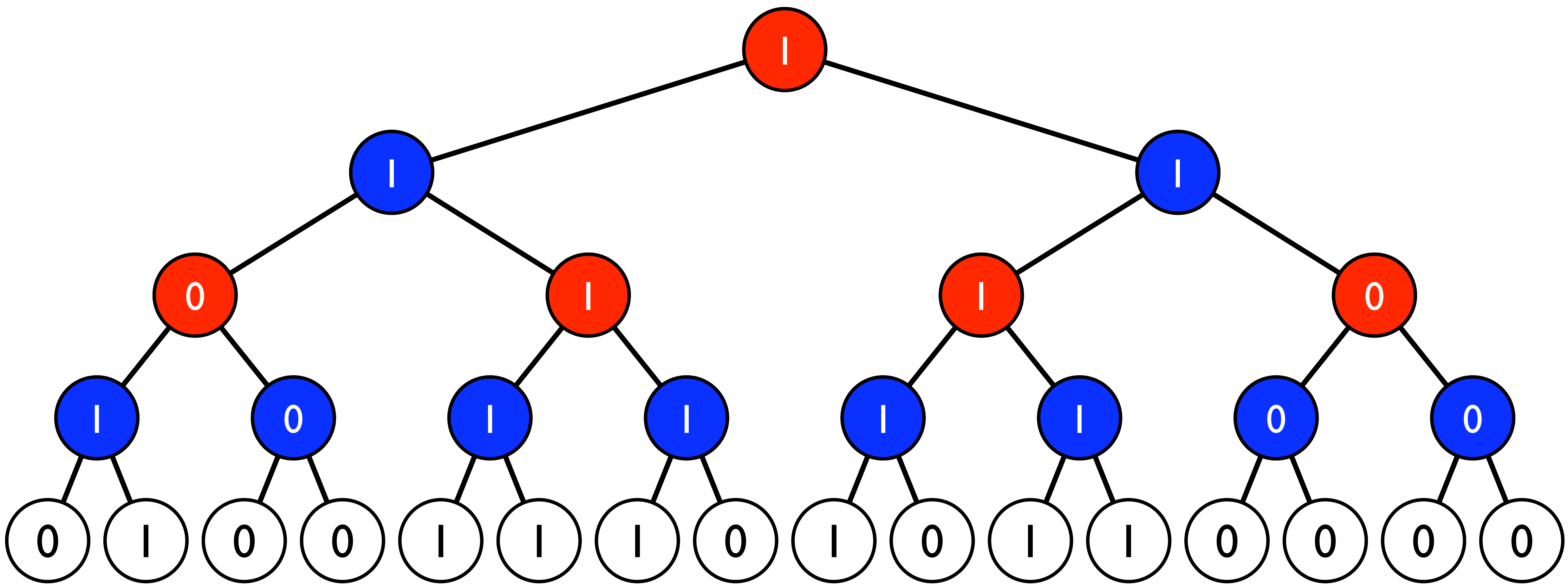
AND-OR trees (aka game trees)



Classical complexity: $\Theta(N^{0.753})$ [Snir 85, Saks-Wigderson 86, Santha 95]

Quantum lower bound [Barnum-Saks 02]: $\Omega(\sqrt{N})$

AND-OR trees (aka game trees)



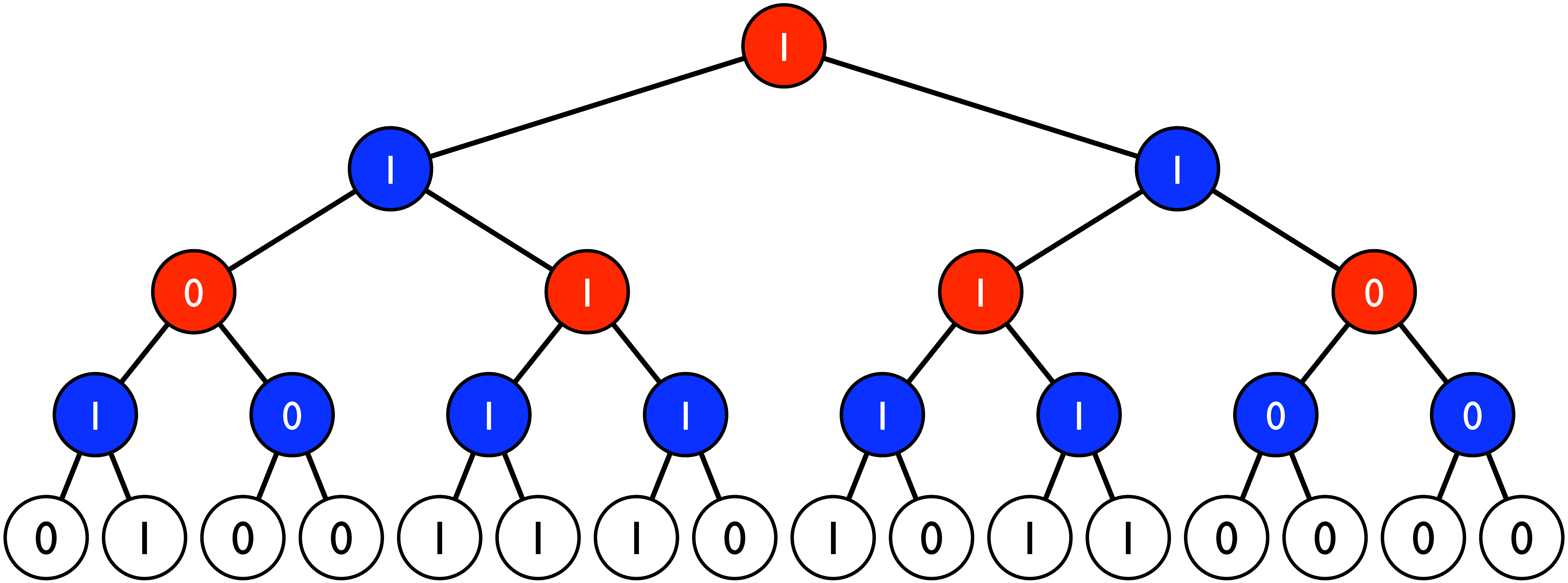
Classical complexity: $\Theta(N^{0.753})$ [Snir 85, Saks-Wigderson 86, Santha 95]

Quantum lower bound [Barnum-Saks 02]: $\Omega(\sqrt{N})$

Quantum walk algorithm [FGG 07]: query time $O(\sqrt{N})$

(in the *Hamiltonian oracle model*)

AND-OR trees (aka game trees)



Classical complexity: $\Theta(N^{0.753})$ [Snir 85, Saks-Wigderson 86, Santha 95]

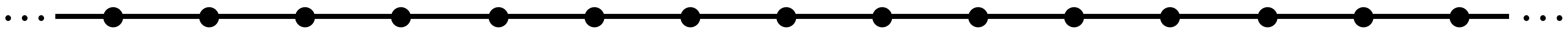
Quantum lower bound [Barnum-Saks 02]: $\Omega(\sqrt{N})$

Quantum walk algorithm [FGG 07]: query time $O(\sqrt{N})$

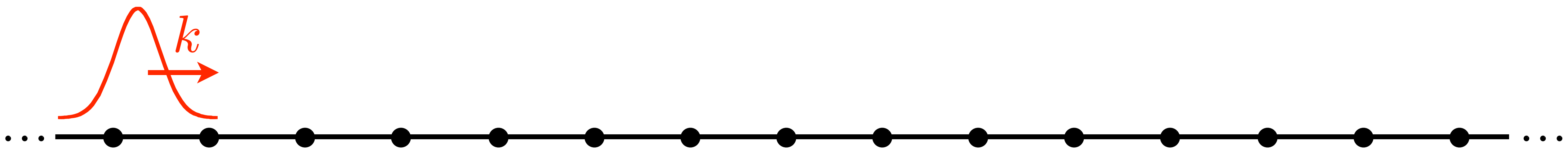
(in the *Hamiltonian oracle model*)

Discretized version [Childs-Cleve-Jordan-Yeung 07]: $O(\sqrt{N^{1+\epsilon}})$

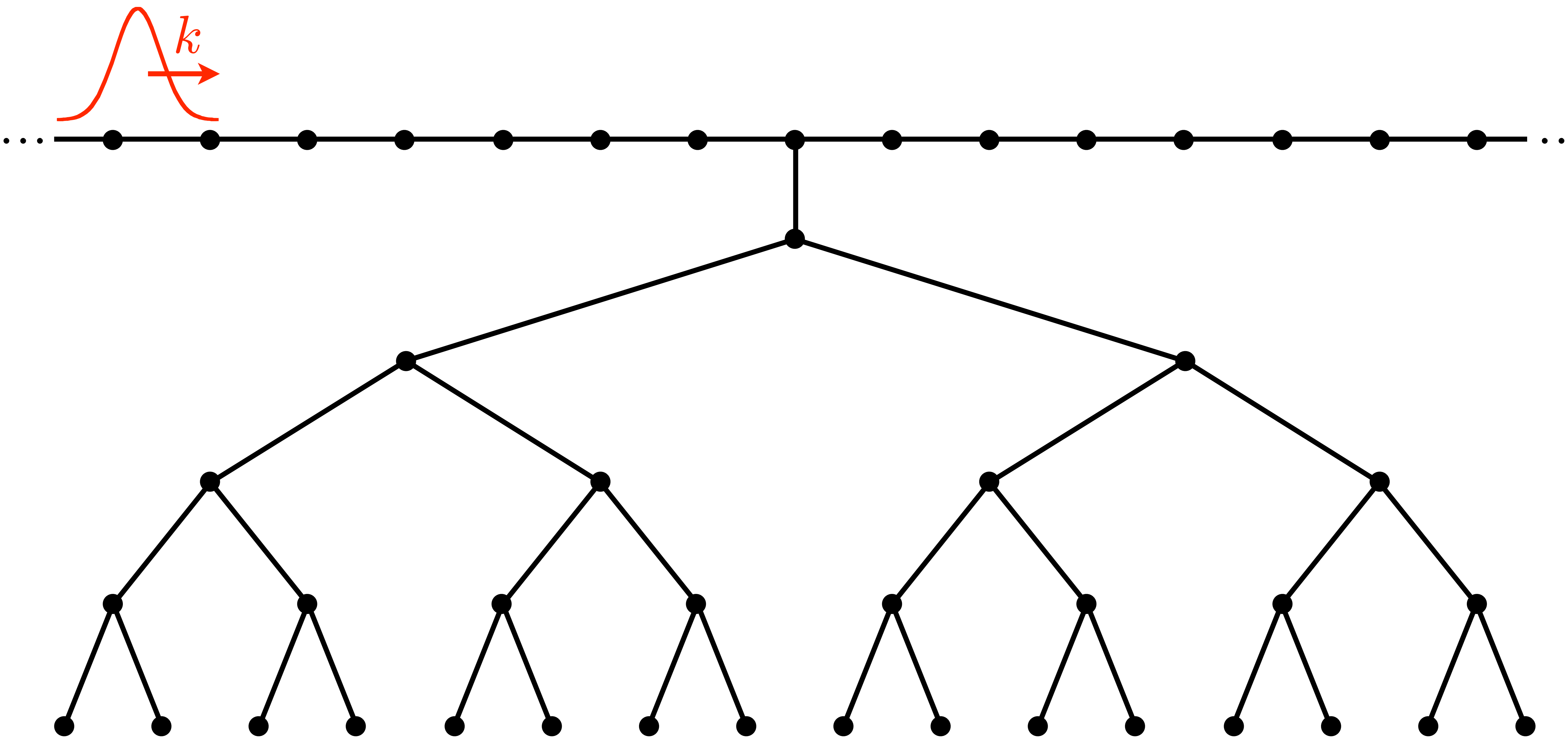
Evaluating AND-OR trees by scattering



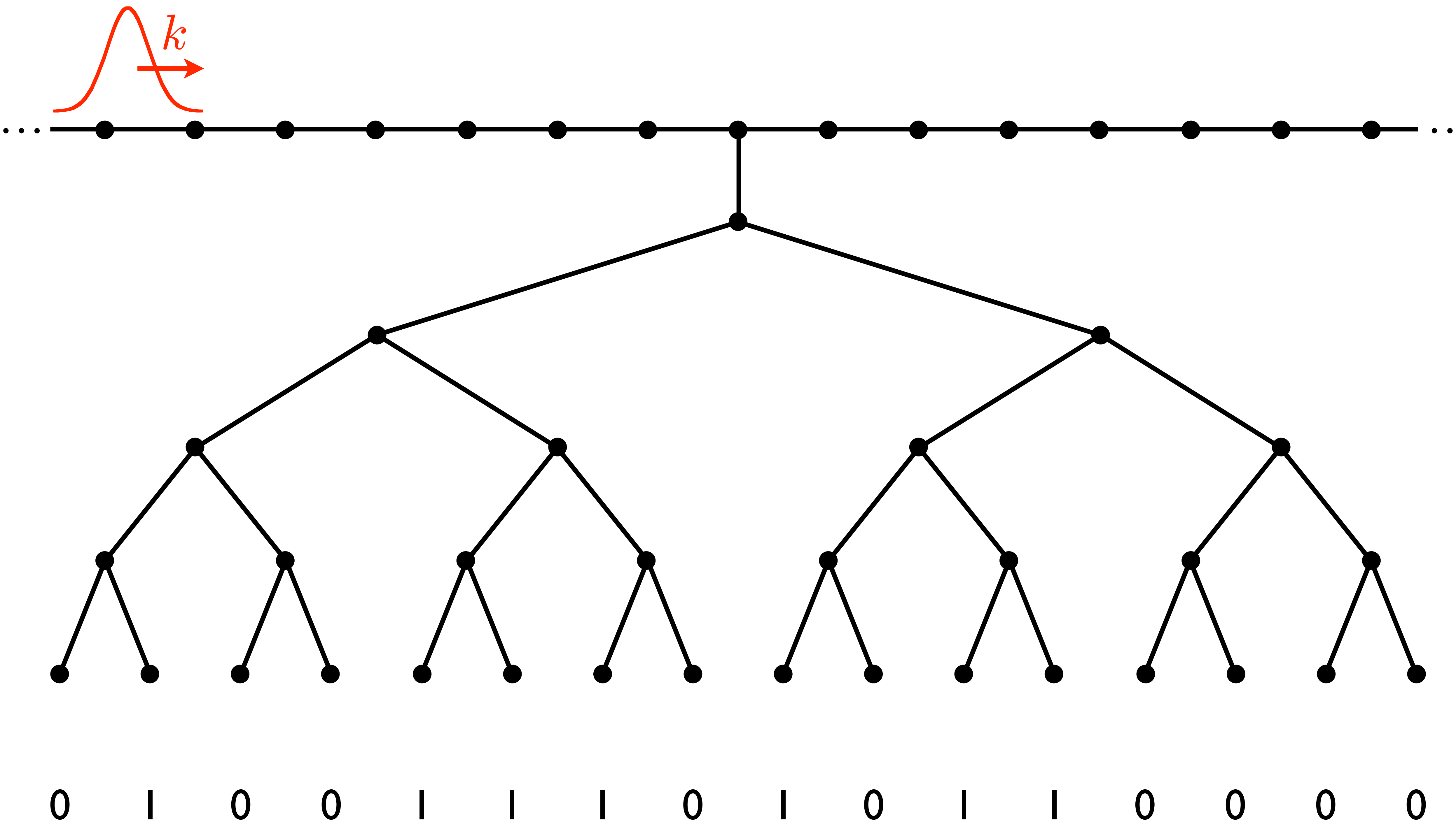
Evaluating AND-OR trees by scattering



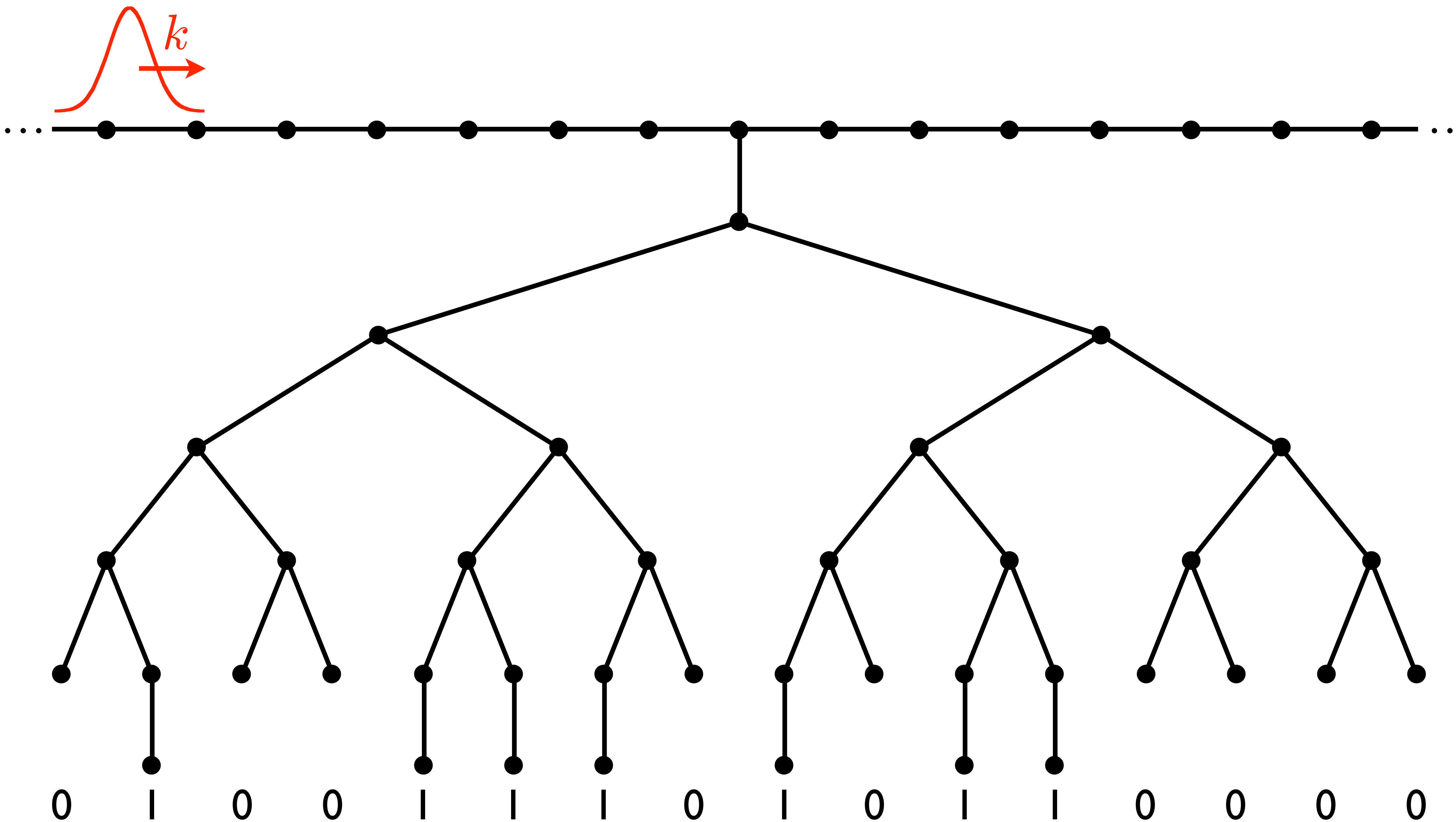
Evaluating AND-OR trees by scattering



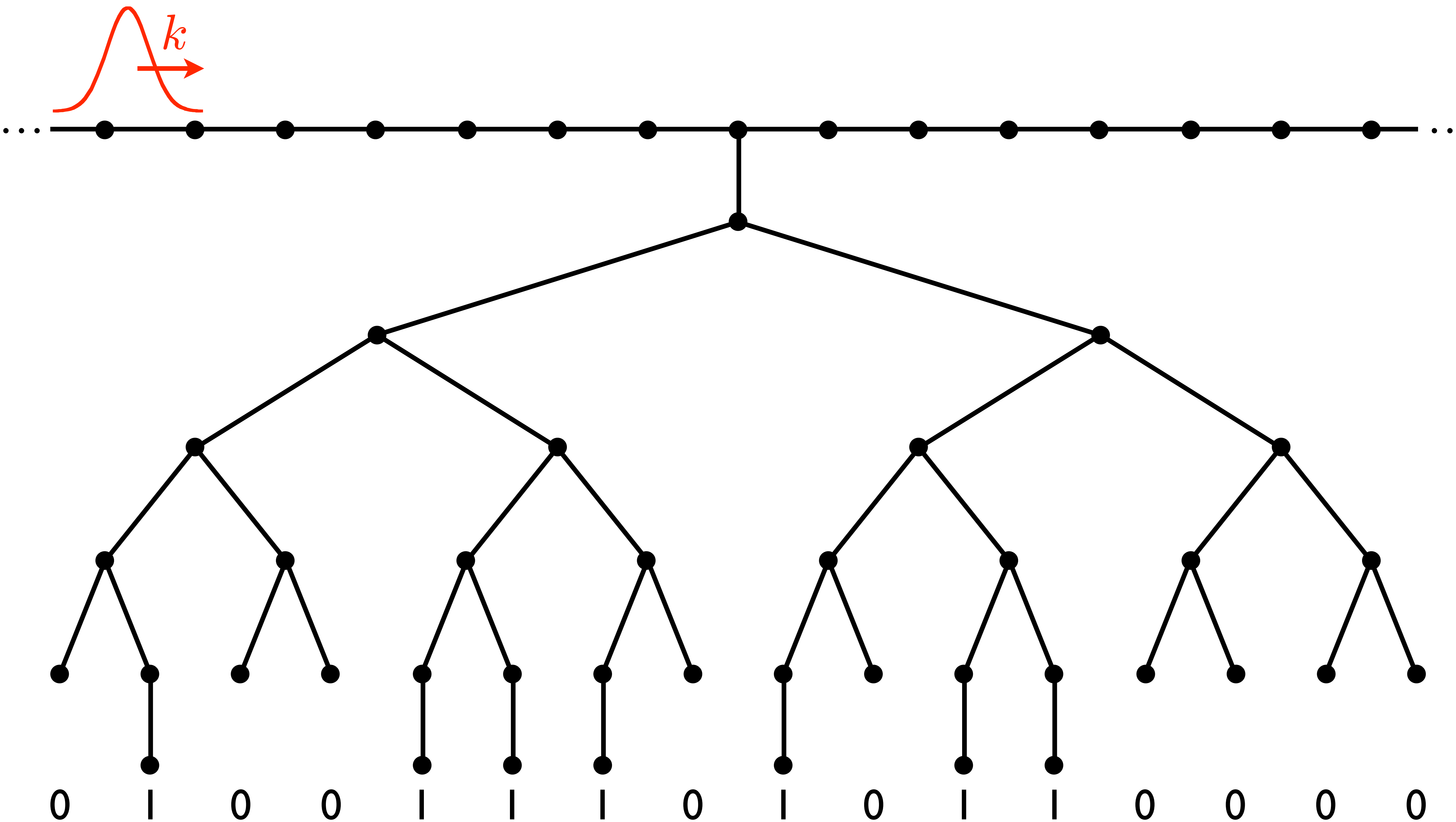
Evaluating AND-OR trees by scattering



Evaluating AND-OR trees by scattering

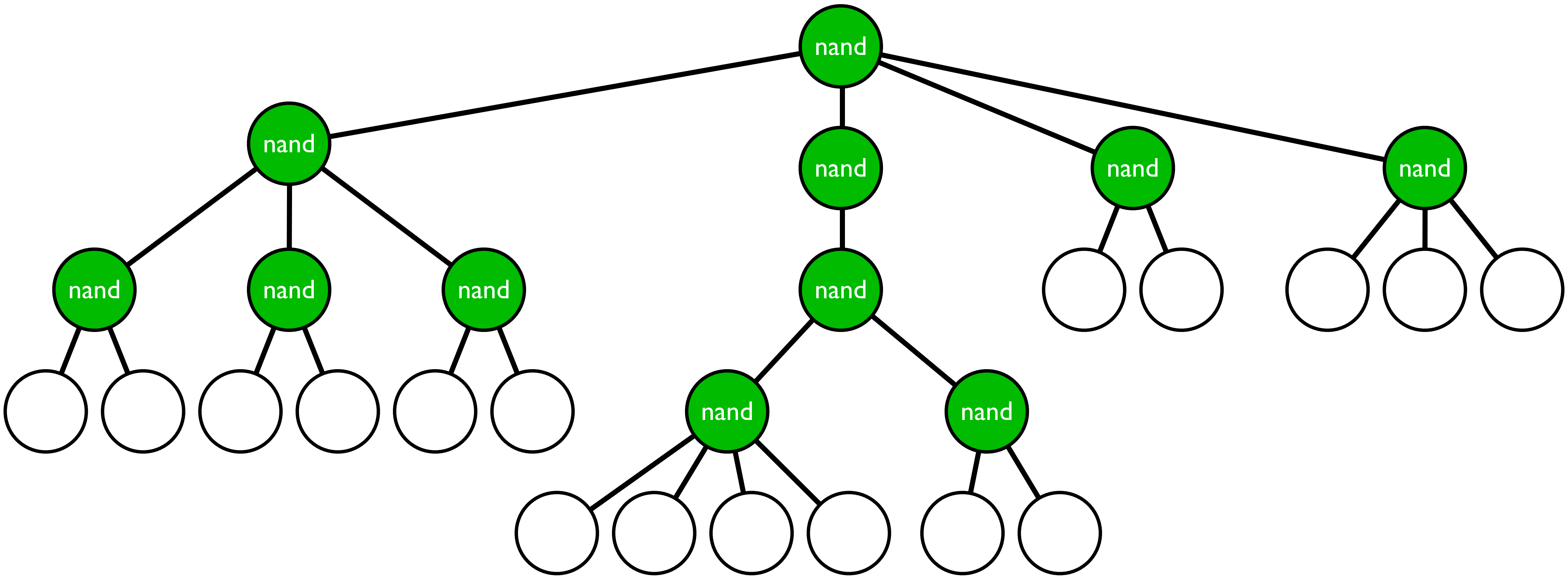


Evaluating AND-OR trees by scattering

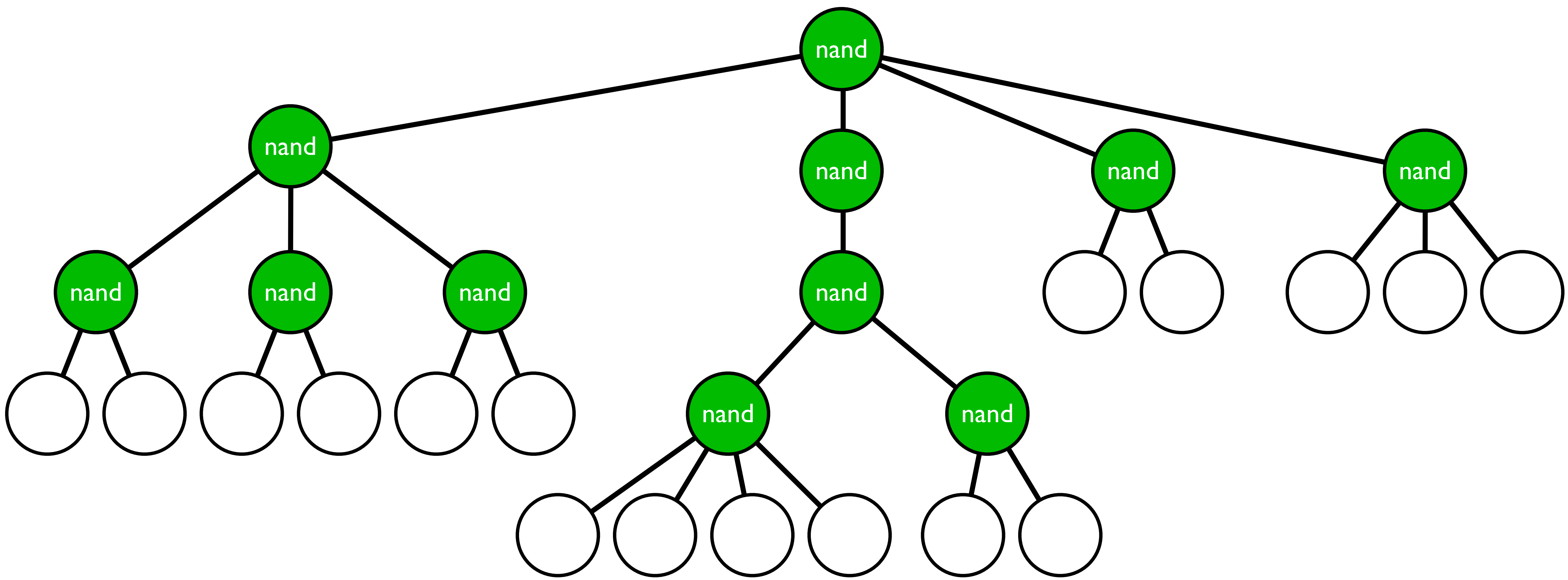


Claim: For small k, the transmission coefficient is large if the formula (translated into NAND gates) evaluates to 0, and small if it evaluates to 1.

General formulas

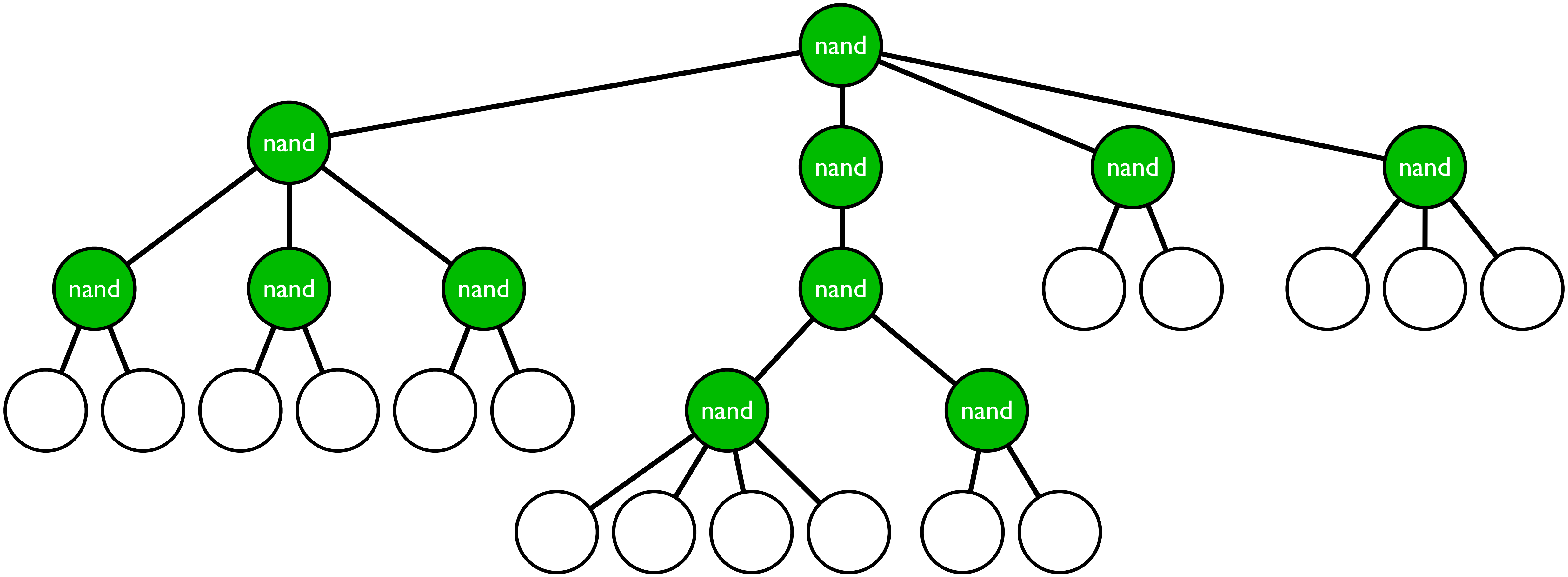


General formulas



Quantum lower bound [Barnum-Saks 02]: $\Omega(\sqrt{N})$

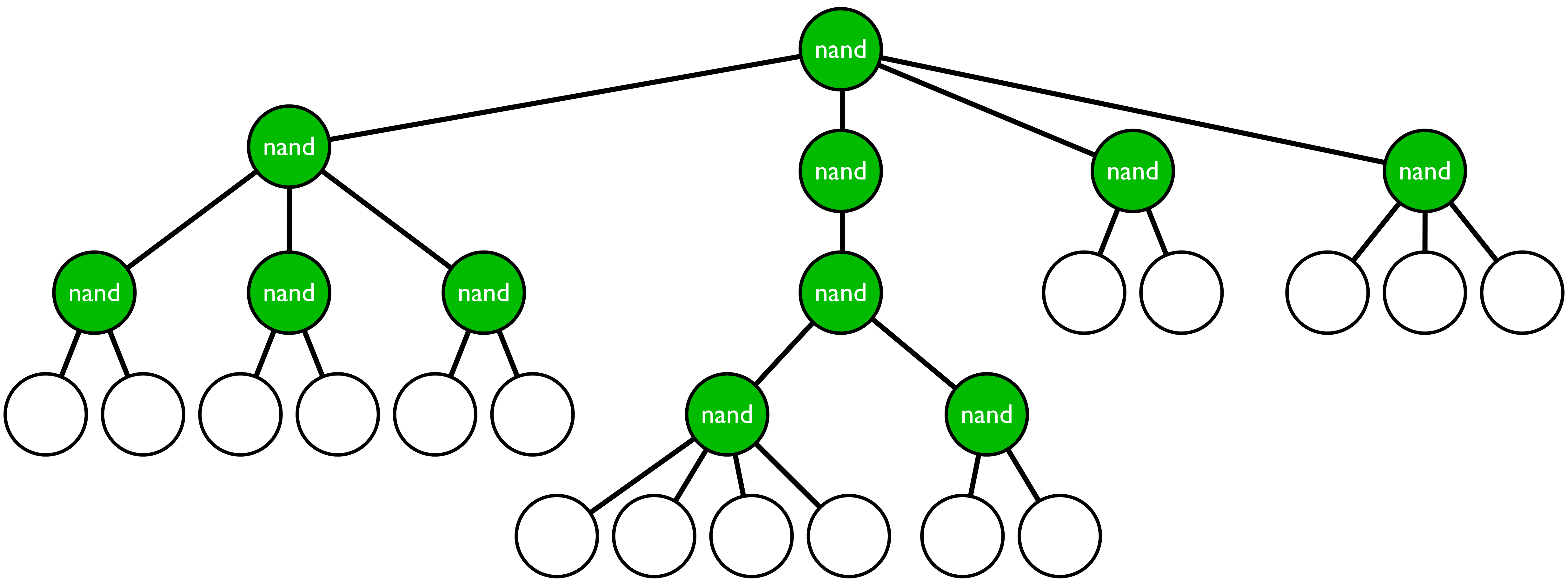
General formulas



Quantum lower bound [Barnum-Saks 02]: $\Omega(\sqrt{N})$

Quantum algorithm [Childs-Reichardt-Špalek-Zhang 07]: $O(\sqrt{N^{1+\epsilon}})$

General formulas



Quantum lower bound [Barnum-Saks 02]: $\Omega(\sqrt{N})$

Quantum algorithm [Childs-Reichardt-Špalek-Zhang 07]: $O(\sqrt{N^{1+\epsilon}})$

Main idea: The quantum walk on the (expanded, appropriately weighted) tree has a zero eigenvalue if the formula evaluates to 1, and a smallest eigenvalue $O(\frac{1}{\sqrt{N}})$ if it evaluates to 0. Use phase estimation.

Summary and outlook

- Physics $\leftrightarrow$ Information
- Quantum systems can encode and process information in a fundamentally non-classical way.
- While we have many examples of this phenomenon, we are far from having a general understanding of the information processing power of quantum mechanics.
- In particular, we would like to develop new ways of exploiting quantum interference to perform information processing tasks.