# Quantum Computing

A Tutorial at the

2005 Genetic and Evolutionary Computation Conference

(GECCO-2005)

Lee Spector

School of Cognitive Science

Hampshire College

Amherst, MA 01002, USA

lspector@hampshire.edu

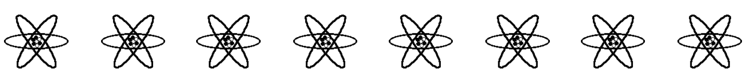
This material is based upon work supported by the United States National Science Foundation under Grant No. 0308540 and Grant No. 0216344, Any opinions, findings, and conclusions or recommendations expressed in this publication are those of the authors and do not necessarily reflect the views of the National Science Foundation.

## Overview

- What is quantum computation?
- Why might it be important?
- How does/might it work?
- Simulating a quantum computer.
- Some quantum algorithms.
- Evolution of new quantum algorithms.
- Sources for more information.

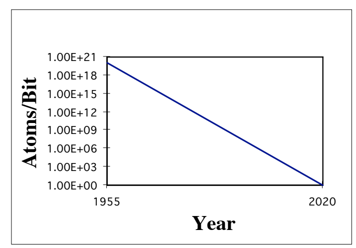
## What is quantum computation?

Computation with coherent atomic-scale dynamics.



The behavior of a quantum computer is governed by the laws of quantum mechanics.

## Why bother with quantum computation?




- Moore’s Law: the amount of information storable on a given amount of silicon has roughly doubled every 18 months. We hit the quantum level 2010 ~ 2020.
- Quantum computation is more powerful than classical computation. More can be computed in less time—the complexity classes are different!

## The power of quantum computation

- In quantum systems ***possibilities count***, even if they never happen!
- Each of exponentially many ***possibilities*** can be used to perform a part of a computation ***at the same time***.

## Nobody understands quantum mechanics

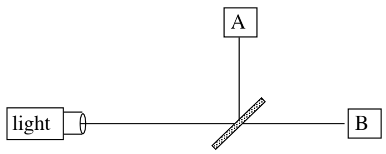
- "Anybody who is not shocked by quantum mechanics hasn't understood it." —Niels Bohr
- "No, you're not going to be able to understand it. ... You see, my physics students don't understand it either. That is because **I** don't understand it. Nobody does. ... The theory of quantum electrodynamics describes Nature as absurd from the point of view of common sense. And it agrees fully with experiment. So I hope you can accept Nature as She is—absurd." —Richard Feynman

## Absurd but taken seriously

(not just quantum mechanics but also quantum computation)

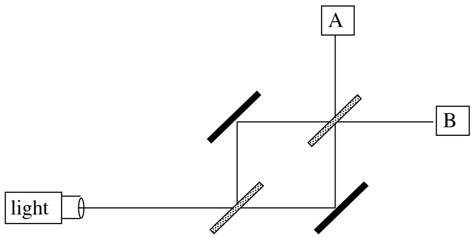
- Under active investigation by many of the top physics labs around the world (including CalTech, MIT, AT&T, Stanford, Los Alamos, UCLA, Oxford, l'Université de Montréal, University of Innsbruck, IBM Research...)
- In the mass media (including *The New York Times*, *The Economist, American Scientist, Scientific American*, ...)
- Here.

## A beam splitter




Half of the photons leaving the light source arrive at detector A; the other half arrive at detector B.

## An interferometer




- Equal path lengths, rigid mirrors.
- Only one photon in the apparatus at a time.
- All of the photons leaving the light source arrive at detector B. WHY?

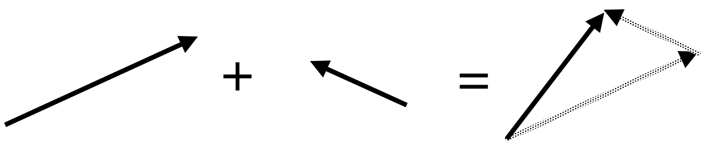
## Possibilities count

- There is an “amplitude” for each possible path that a photon can take.
- The amplitudes can interfere constructively and destructively, even though each photon takes only one path.
- The amplitudes at detector A interfere destructively; those at detector B interfere constructively.

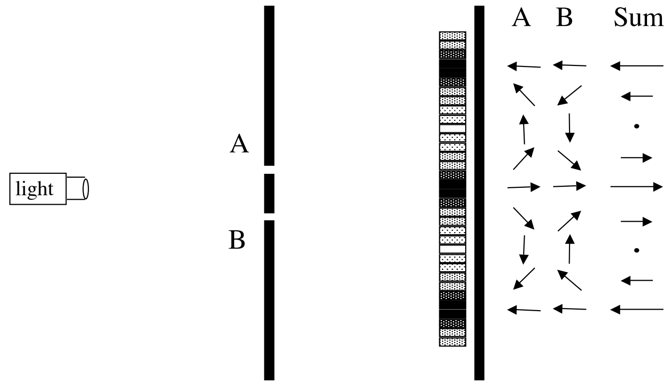
## Calculating interference

- “You will have to brace yourselves for this—not because it is difficult to understand, but because it is absolutely ridiculous: All we do is draw little arrows on a piece of paper—that’s all!” —Richard Feynman
- Arrows for each possibility.
- Arrows rotate; speed depends on frequency.
- Arrows flip 180˚ at mirrors, rotate 90˚ counter-clockwise when reflected from beam splitters.
- Add arrows and square the length of the result to determine the probability for any possibility.

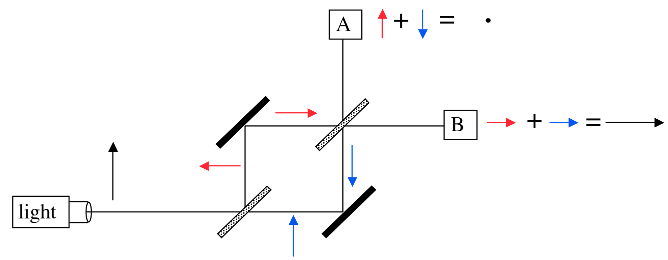
## Adding arrows



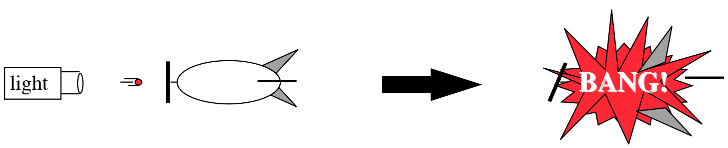
## Double slit interference




## Interference in the interferometer

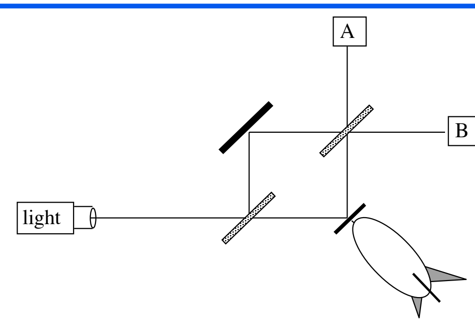



## A photon-triggered bomb




- A mirror is mounted on a plunger on the bomb’s nose.
- A single photon hitting the mirror depresses the plunger and explodes the bomb.
- Some plungers are stuck, producing duds.
- How can you find a good, unexploded bomb?

## Elitzur-Vaidman bomb testing




- Possibilities count!
- Experimentally verified
- Can be enhanced to reduce or eliminate bomb loss
  [Kwiat, Weinfurter and Kasevich]

## Two interesting speedups

- Grover's quantum database search algorithm finds an item in an unsorted list of $n$ items in $O(\sqrt{n})$ steps; classical algorithms require $O(n)$.
- Shor's quantum algorithm finds the prime factors of an $n$-digit number in time $O(n^3)$; the best known classical factoring algorithms require at least time $O(2^{n^{1/3} \log(n)^{2/3}})$.

## Reminder: exponential savings is **very** good!

Factor a 5,000 digit number:

- Classical computer (1ns/instr, ~today's best alg)
  - » **over 5 trillion years**
    (the universe is ~ 10–16 billion years old).
- Quantum computer (1ns/instr, ~Shor's alg)
  - » just over 2 minutes

## Quantum computing and the human brain

- Penrose's argument

  Brains do $X$ (for $X$ uncomputable)

  Classical computers can't do $X$

  ∴ Brains aren't classical computers

  - First premise is false for all proposed $X$. For example, brains don't have knowably sound procedures for mathematical proof.
  - Would imply brains more powerful than quantum computers; new physics.

## Quantum consciousness?

- Relation to consciousness etc. is much discussed, unclear at best. (Bohm, Penrose, Hameroff, others)
- "[Penrose's] argument seemed to be that consciousness is a mystery and quantum gravity is another mystery so they must be related." (Hawking)

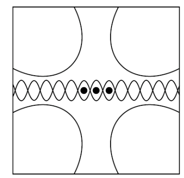
## Quantum information theory

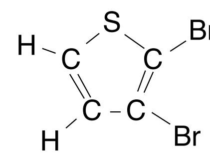
- Quantum cryptography: secure key distribution
- Quantum teleportation
- Quantum data compression
- Quantum error correction

Good introductions to these topics can be found in (Steane, 1998).

## Physical implementation

- Ion traps
- Nuclear spins in NMR devices
- Optical systems
- So far: few qubits, impractical
- A lot of current research






## Languages and notations

- Wave equations
- Wave diagrams
- Matrix mechanics
- Dirac's bra-ket notation ($\langle\phi|\psi\rangle$)
- Particle diagrams
- Amplitude diagrams
- Phasor diagrams
- QGAME programs

## Qubits

- The smallest unit of information in a quantum computer is called a "qubit".
- A qubit may be in the "on" (1) state or in the "off" (0) state or in any superposition of the two!

## State representation, 1 qubit

- The state of a qubit can be represented as:

  $\alpha_0|0\rangle + \alpha_1|1\rangle$

  $\alpha_0$ and $\alpha_1$ are complex numbers that specify the *probability amplitudes* of the corresponding states.
- $|\alpha_0|^2$ gives the probability that you will find the qubit in the "off" (0) state; $|\alpha_1|^2$ gives the probability that you will find the qubit in the "on" (1) state.

## Entanglement

- Qubits in a multi-qubit system are not independent—they can become "entangled." (We'll see some examples.)
- To represent the state of *n* qubits one usually uses $2^n$ complex number amplitudes.

## State representation, 2 qubits

- The state of a two-qubit system can be represented as:

  $\alpha_0|00\rangle + \alpha_1|01\rangle + \alpha_2|10\rangle + \alpha_3|11\rangle$

  $\Sigma\ |\alpha|^2 = 1$
- Measurement will always find the system in some (one) discrete state.

## Measurement at the end of a computation

- $\Sigma|\alpha|^2$, for amplitudes of all states matching the output bit-pattern in question.
- This gives the probability that the particular output will be read upon measurement.
- Example:

  $0.316|00\rangle+0.447|01\rangle+0.548|10\rangle+0.632|11\rangle$

  The probability to read the rightmost bit as 0 is $|0.316|^2+|0.548|^2=0.4$

## Partial measurement during a computation

- One-qubit measurement gates.
- Measurement changes the system.
- In simulation, branch computation for each possible measurement.

## Classical computation in matrix form

A state transition in a 4-bit system:

$$\begin{bmatrix} 0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0 \\ 1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1 \end{bmatrix} * \begin{bmatrix} \alpha_0 \\ \alpha_1 \\ \alpha_2 \\ \alpha_3 \\ \alpha_4 \\ \alpha_5 \\ \alpha_6 \\ \alpha_7 \\ \alpha_8 \\ \alpha_9 \\ \alpha_{10} \\ \alpha_{11} \\ \alpha_{12} \\ \alpha_{13} \\ \alpha_{14} \\ \alpha_{15} \end{bmatrix}$$

## A quantum NOT gate

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

Applied to a qubit:

$$\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} * \begin{pmatrix} \alpha_0 \\ \alpha_1 \end{pmatrix} = \begin{pmatrix} \alpha_1 \\ \alpha_0 \end{pmatrix}$$

$$\alpha_0|0\rangle + \alpha_1|1\rangle \longrightarrow \alpha_1|0\rangle + \alpha_0|1\rangle$$

## Explicit matrix expansion

To expand gate matrix $G$ for application to an $n$-qubit system:

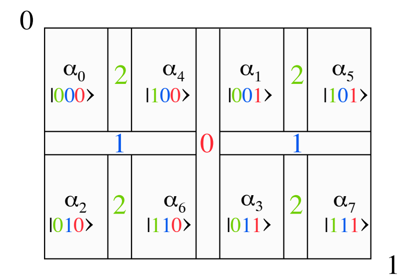
- Create a $2^n$x$2^n$ matrix $M$.
- Let $Q$ be the set of qubits to which the operator is being applied, and $Q'$ be the set of the remaining qubits.
- $M_{ij} = 0$ if $i$ and $j$ differ in positions in $Q'$.
- Otherwise concatenate bits from $i$ in positions $Q$ to produce $i^*$, and bits from $j$ to produce $j^*$. $M_{ij} = G_{i^*j^*}$.

## Implicit matrix expansion

To apply gate matrix $G$ to an $n$-qubit system:

- Let $Q$ be the set of qubits to which the operator is being applied, and $Q'$ be the set of the remaining qubits.
- For every combination $C$ of 1 and 0 for qubits in $Q'$:
  - » Extract the column $A$ of amplitudes that results from holding $C$ constant and varying all qubits in $Q$.
  - » $A' = G \times A$.
  - » Install $A'$ in place of $A$ in the array of amplitudes.

## Amplitude diagrams



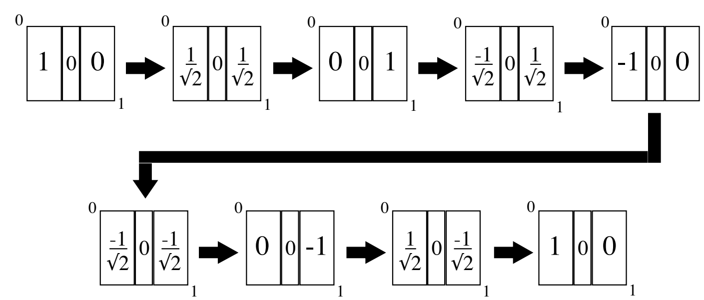

- Help to visualize amplitude distributions
- Scalable, hierarchical
- Can be shuffled to prioritize any qubits

## A square-root-of-NOT (SRN) gate

$$\begin{pmatrix} \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{pmatrix}$$

- Applied once to a classical state, this ~randomizes the value of the qubit.
- Applied twice in a row, this is ~equivalent to NOT: $\begin{bmatrix} \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix} * \begin{bmatrix} \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix} = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}$

## SRN amplitude diagrams



## Other quantum gates

- Rotation ($U\theta$): $\begin{bmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{bmatrix}$
- Hadamard ($H$): $\frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$
- Controlled NOT ($CNOT$): $\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$

There are many small "complete" sets of gates [Barenco et al.].

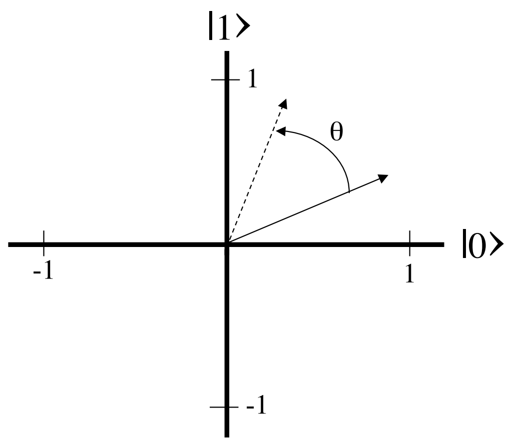
## More quantum gates

- Conditional phase: $\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & e^{i\alpha} \end{bmatrix}$
- $U2$: $\begin{bmatrix} e^{-i\phi} & 0 \\ 0 & e^{i\phi} \end{bmatrix} \times \begin{bmatrix} \cos(\theta) & \sin(-\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix} \times \begin{bmatrix} e^{-i\psi} & 0 \\ 0 & e^{i\psi} \end{bmatrix} \times \begin{bmatrix} e^{i\alpha} & 0 \\ 0 & e^{i\alpha} \end{bmatrix}$

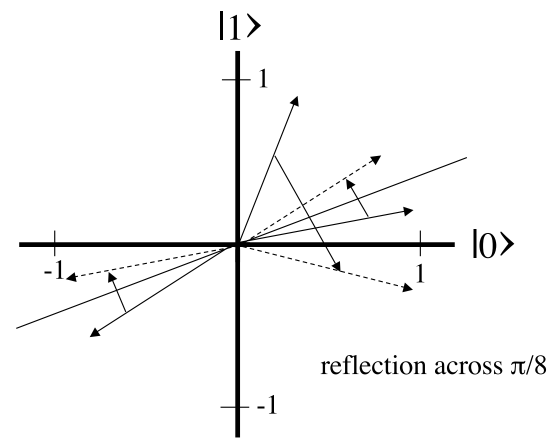
All gates must be unitary: $U^{\dagger}U=UU^{\dagger}=I$,

where $U^{\dagger}$ is the Hermitean adjoint of $U$, obtained by taking the complex conjugate of each element of $U$ and then transposing the matrix.

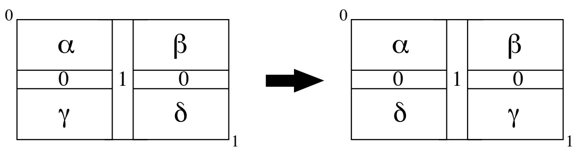
## Rotation polar plot for real vectors




## Hadamard polar plot for real vectors



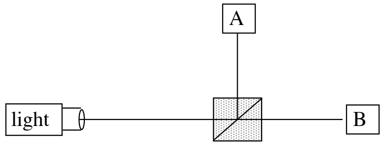
## CNOT amplitude diagrams




CNOT(0 [control], 1 [target])

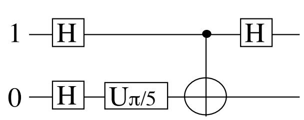
## Polarizing beam-splitter CNOT gate

[Cerf, Adami, and Kwiat]



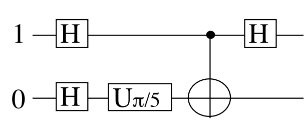

- Two qubits encoded in one photon, one in momentum (direction) and one in polarization.
- Polarization controls change in momentum.
- Cannot be scaled up directly, but demonstrates an implementation of a 2-qubit gate.

## Gate array diagrams

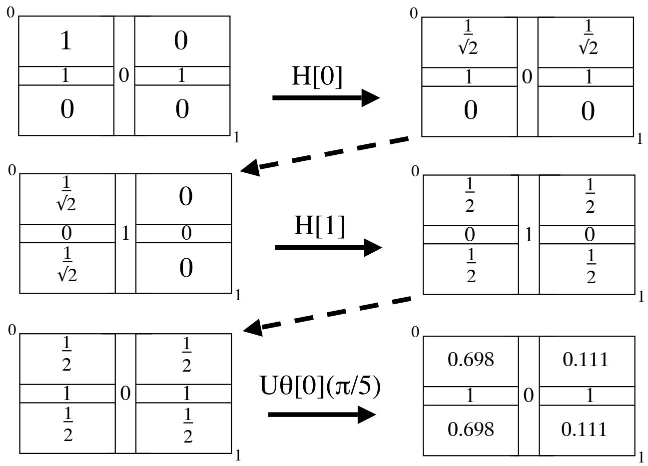



## Example execution trace

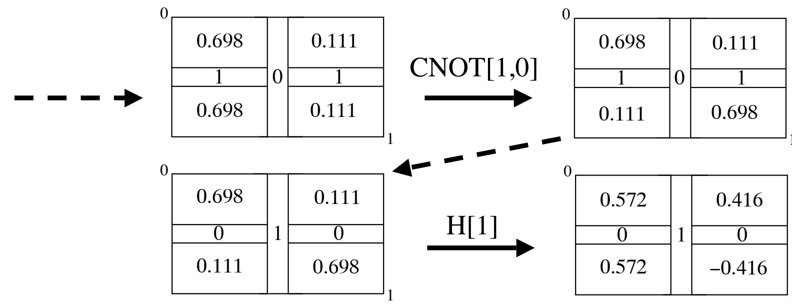
```
Hadamard qubit:0
Hadamard qubit:1
U-theta qubit:0 theta:pi/5
Controlled-not control:1 target:0
Hadamard qubit:1
```



## Trace, cont.




## Trace, cont.




| state | probability |
|---|---|
| \|00› | 0.33 |
| \|01› | 0.33 |
| \|10› | 0.17 |
| \|11› | 0.17 |

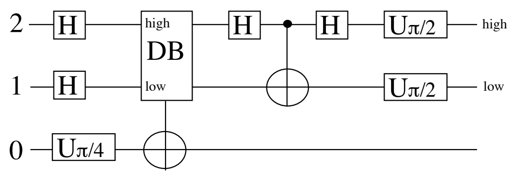
## The database search problem

- Given an unsorted database containing *n* items but only one “marked” item, find the address of the marked item with a minimal number of database calls.
- Lov Grover’s algorithm uses O($\sqrt{n}$ ) calls in general, and only one call for a 4-item database.

## Oracle problems

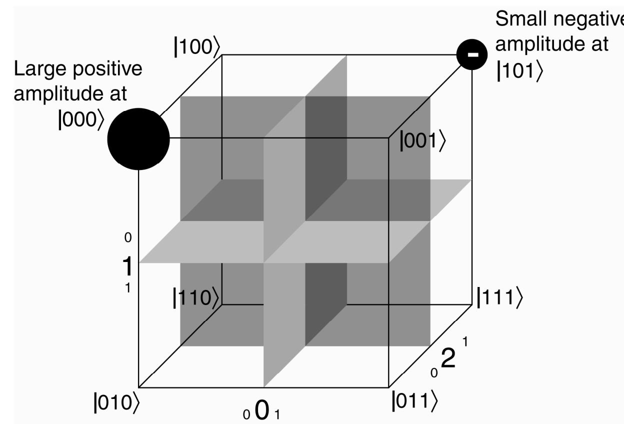
- The database search problem is an example of an “oracle problem.”
- We are given a “black box” or “oracle” function (in this case the database access function) and asked to find out if it has some particular property.
- Many other known quantum algorithms are for oracle problems.
- Often the oracle is “hard” to implement, so complexity is figured from the number of oracle calls.

## Grover's algorithm for a 4-item database

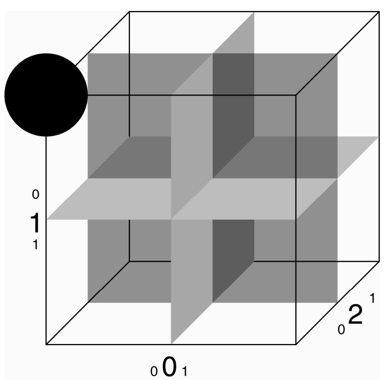



- Start in the state |000>.
- Read answer from qubits 2 and 1.

## Cube diagram for a 3-qubit system

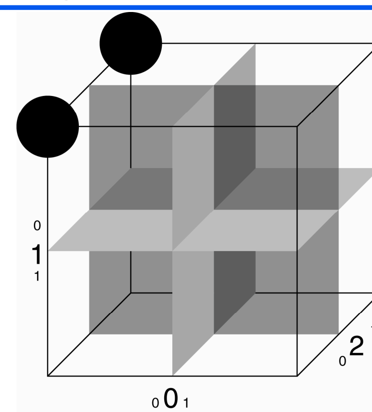



## (0) Grover's algorithm, item at 0,0



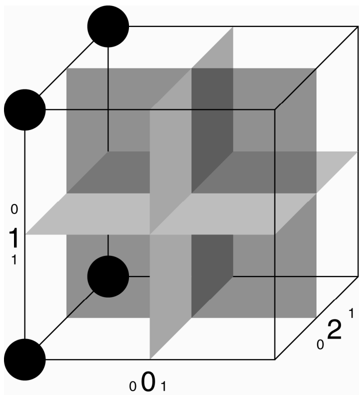

Initial State, |000>

## (1) Grover's algorithm, item at 0,0



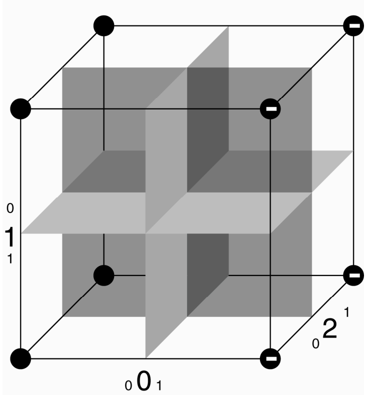

After Hadamard[2]

## (2) Grover's algorithm, item at 0,0




After Hadamard[1]

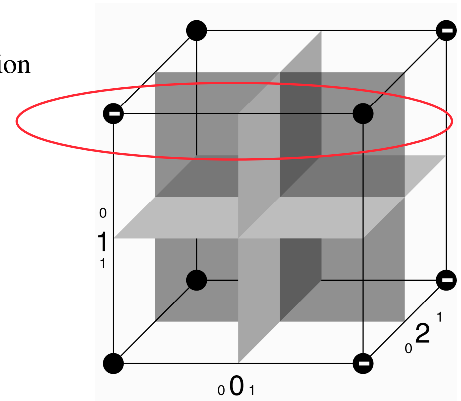
## (3) Grover's algorithm, item at 0,0




After Uθ[0](π/4)

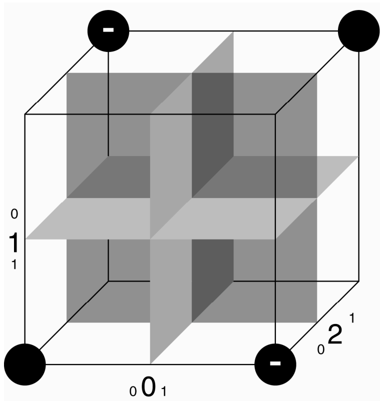
## (4) Grover's algorithm, item at 0,0

Note position
of DB call
effect.



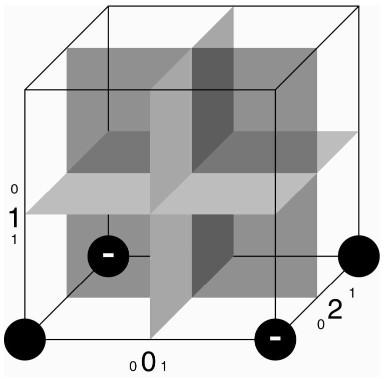

After Database Call [in: 2,1; out:0]

## (5) Grover's algorithm, item at 0,0



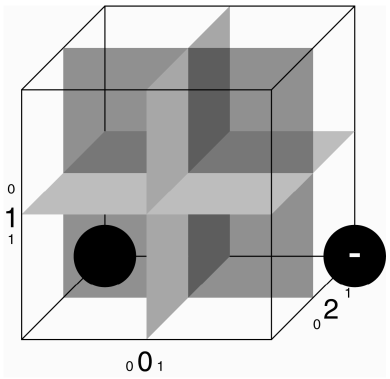

After Hadamard[2]

## (6) Grover's algorithm, item at 0,0



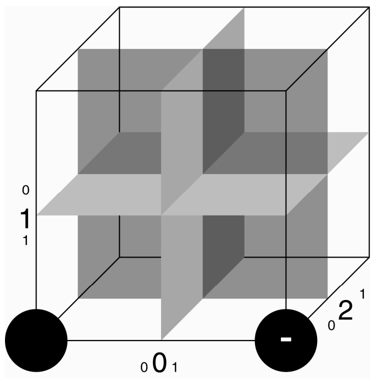

After CNOT [control: 2; target: 1]

## (7) Grover's algorithm, item at 0,0



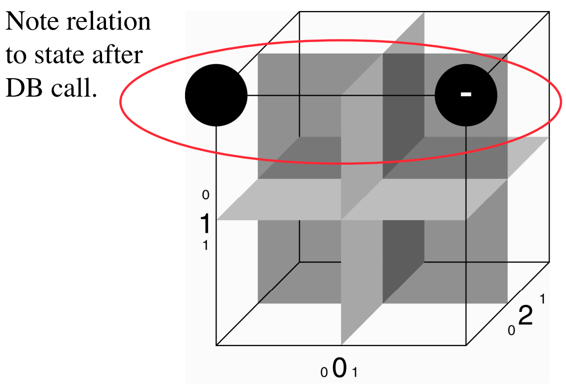

After Hadamard[2]

## (8) Grover's algorithm, item at 0,0



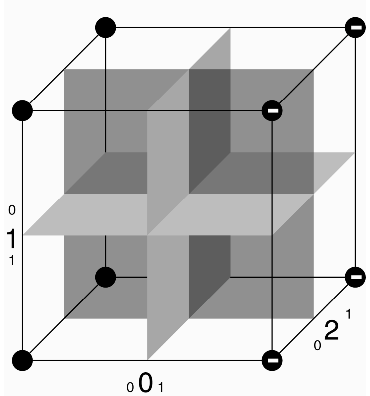

After Uθ[2](π/2)

## (9) Grover's algorithm, item at 0,0



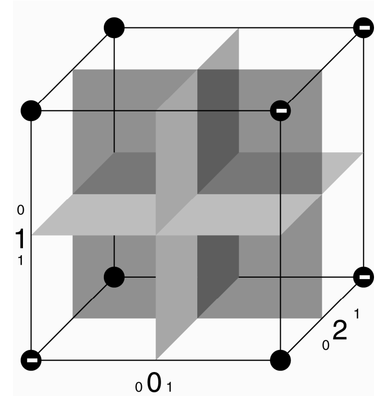

After Uθ[1](π/2), Read output from qubits 2 (high) and 1(low)

## (3) Grover's algorithm, item at 0,1



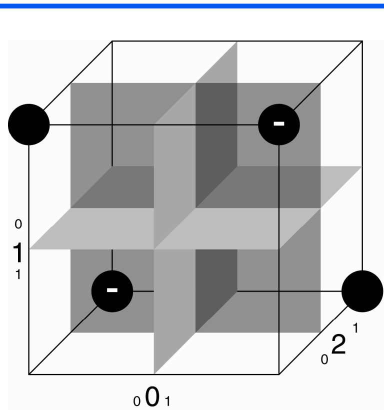

After Uθ[0](π/4)

## (4) Grover's algorithm, item at 0,1



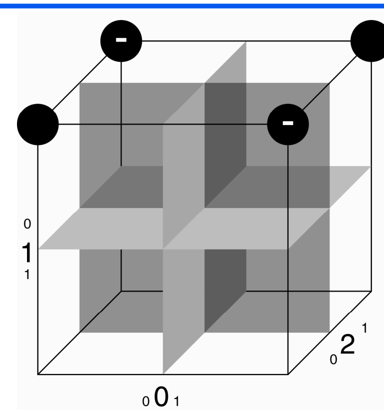

After Database Call [in: 2,1; out:0]

## (5) Grover's algorithm, item at 0,1



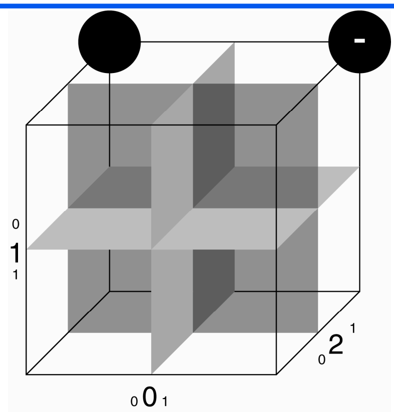

After Hadamard[2]

## (6) Grover's algorithm, item at 0,1



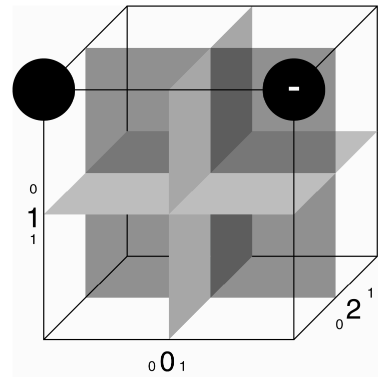

After CNOT [control: 2; target: 1]

## (7) Grover's algorithm, item at 0,1



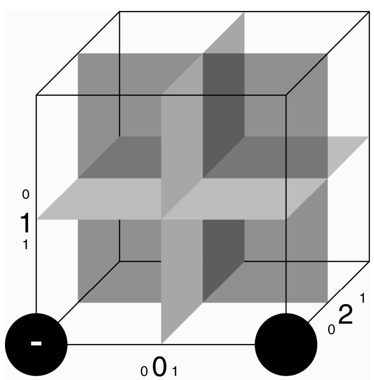

After Hadamard[2]

## (8) Grover's algorithm, item at 0,1



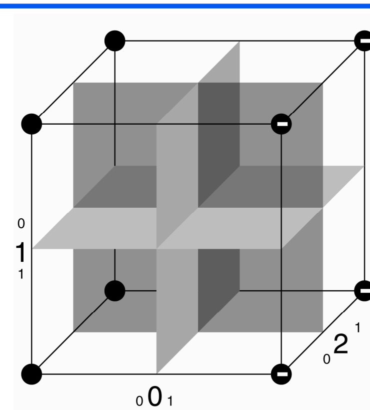

After Uθ[2](π/2)

## (9) Grover's algorithm, item at 0,1



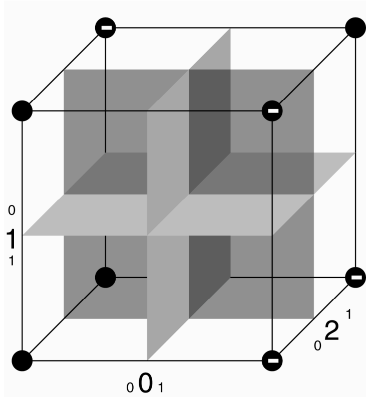

After Uθ[1](π/2), Read output from qubits 2 (high) and 1(low)

## (3) Grover's algorithm, item at 1,0



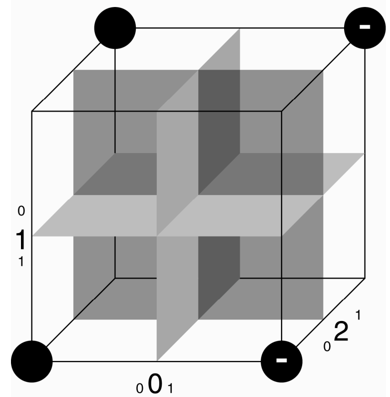

After Uθ[0](π/4)

## (4) Grover's algorithm, item at 1,0



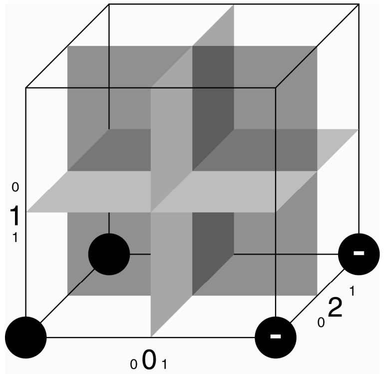
After Database Call [in: 2,1; out:0]

## (5) Grover's algorithm, item at 1,0



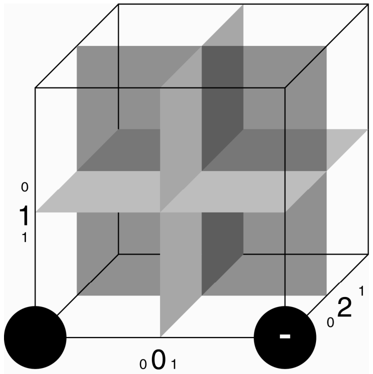
After Hadamard[2]

## (6) Grover's algorithm, item at 1,0



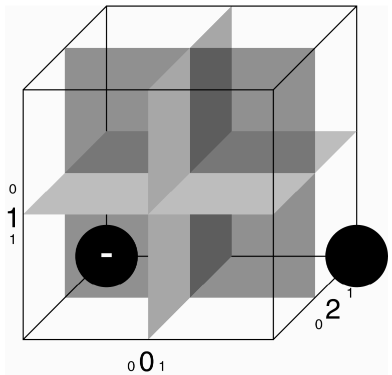
After CNOT [control: 2; target: 1]

## (7) Grover's algorithm, item at 1,0



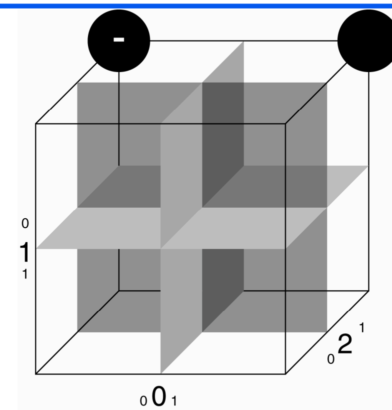
After Hadamard[2]

## (8) Grover's algorithm, item at 1,0



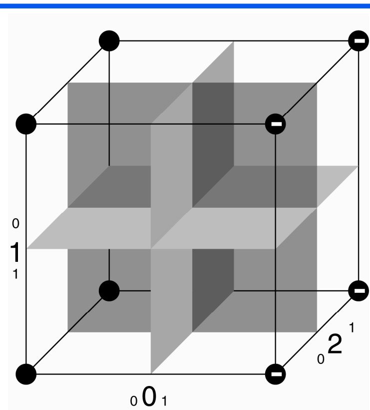

After Uθ[2](π/2)

## (9) Grover's algorithm, item at 1,0



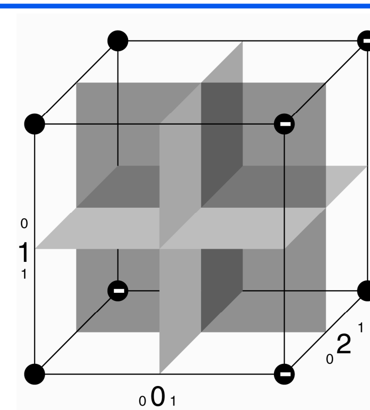

After Uθ[1](π/2), Read output from qubits 2 (high) and 1(low)

## (3) Grover's algorithm, item at 1,1



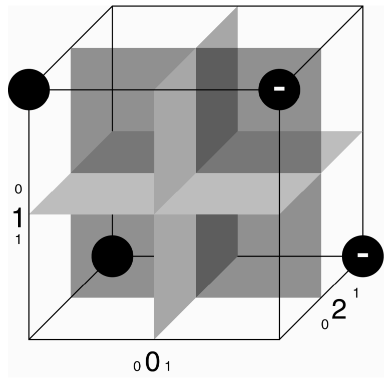

After Uθ[0](π/4)

## (4) Grover's algorithm, item at 1,1



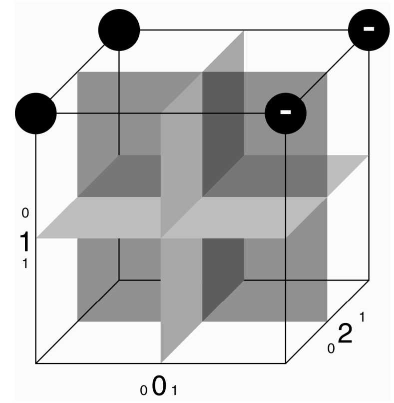

After Database Call [in: 2,1; out:0]

## (5) Grover's algorithm, item at 1,1



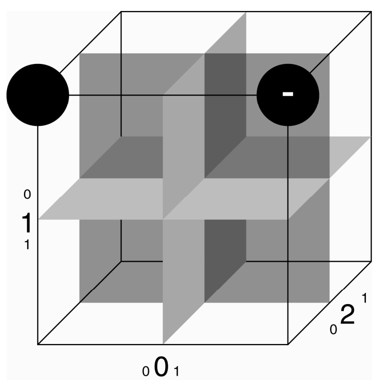

After Hadamard[2]

## (6) Grover's algorithm, item at 1,1



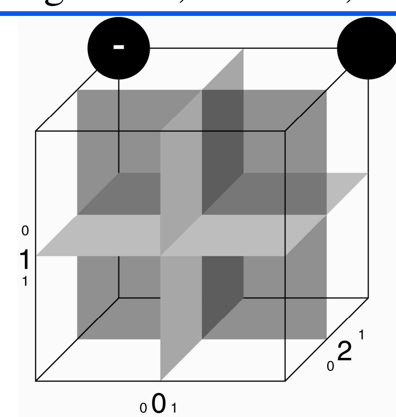

After CNOT [control: 2; target: 1]

## (7) Grover's algorithm, item at 1,1



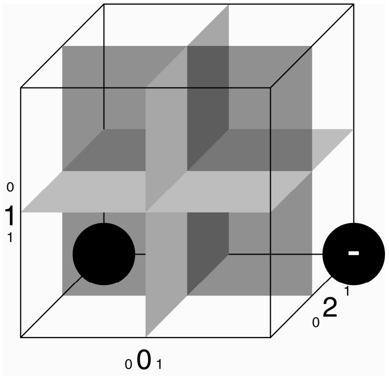

After Hadamard[2]

## (8) Grover's algorithm, item at 1,1




After Uθ[2](π/2)

## (9) Grover's algorithm, item at 1,1

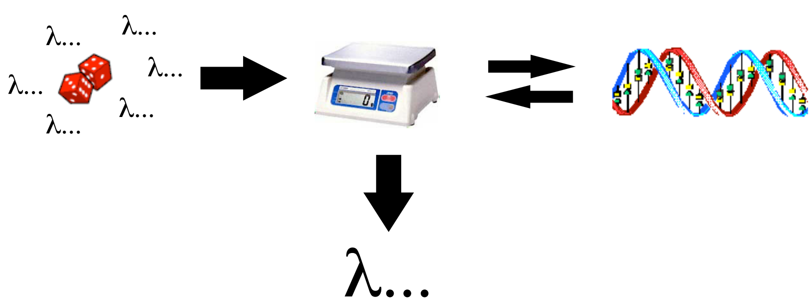



After Uθ[1](π/2), Read output from qubits 2 (high) and 1(low)

## Shor's algorithm

- hybrid algorithm to factor numbers
- quantum component helps to find the period *r* of a sequence $a_1$, $a_2$, ... $a_i$, ... , given an oracle function that maps *i* to $a_i$
- skeleton of the algorithm:
  - create a superposition of all oracle inputs
  - call the oracle function
  - apply a quantum Fourier transform to the input qubits
  - read the input qubits to obtain a random multiple of 1/*r*
  - repeat a small number of times to infer *r*

## Genetic Programming (GP)




## GP for quantum computation

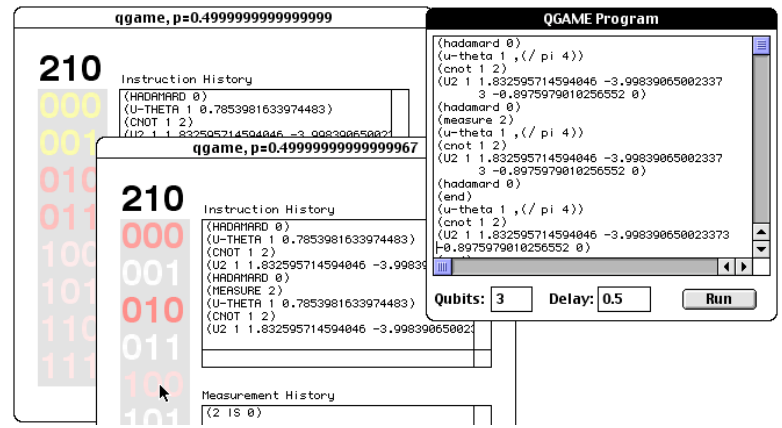
- Evolve:
  - gate arrays
  - programs that produce gate arrays
  - hybrid classical/quantum algorithms
  - input states or parameters
- Genome representation:
  - QGAME program
  - program (in any language) that generates a QGAME program
  - array of numbers

## Fitness

- Assessing the composite matrix
  - the trouble with oracles
- Assessing the results of simulation runs
- Criteria:
  - Error
  - Hits
  - Oracle calls
  - Number of gates

## QGAME Quantum Gate and Measurement Emulator

http://hampshire.edu/lspector/qgame.html



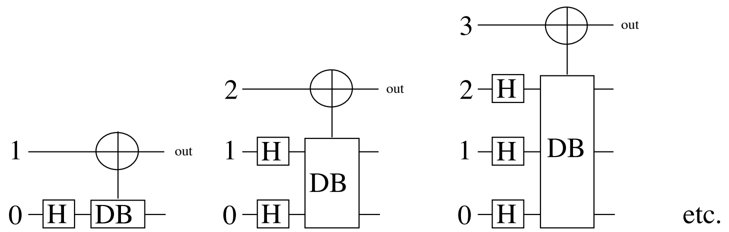

## Primitives; gate-array-producing programs

- Gates: $H$, $U_{\theta}$, $CNOT$, $ORACLE$, ...
- Qubit indices
- Gate parameters (angles)
- Arithmetic operators
- Constants indicating problem size (num-qubits, num-input-qubits, num-output-qubits)
- Iteration structures, recursion, data structures, …

## The scaling majority-on problem

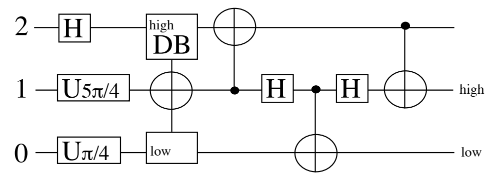
- Does the oracle answer “1” for a majority of inputs?
- Seek program that produces a gate array for any oracle size.

## Evolved scaling majority-on gate arrays

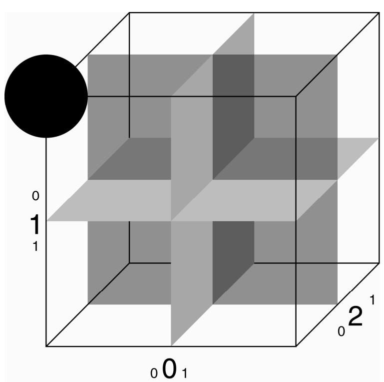



Not better than classical.

## Evolved database search gate array

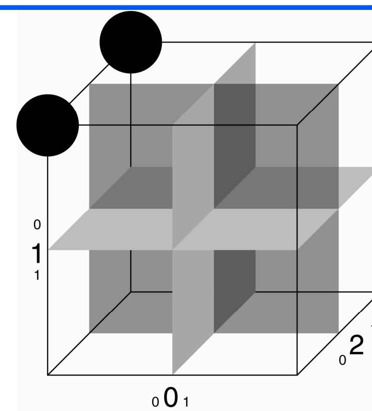



## (0) Evolved quantum database algorithm, item at 0,0



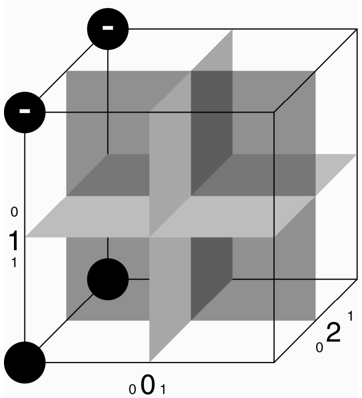

Initial State, |000>

## (1) Evolved quantum database algorithm, item at 0,0



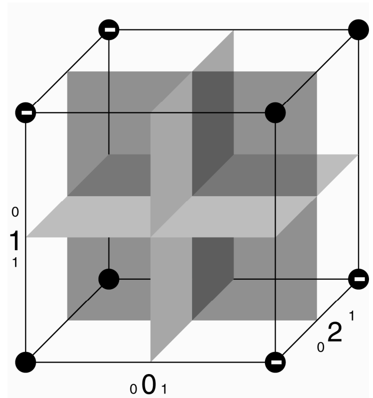

After Hadamard [2]

## (2) Evolved quantum database algorithm, item at 0,0



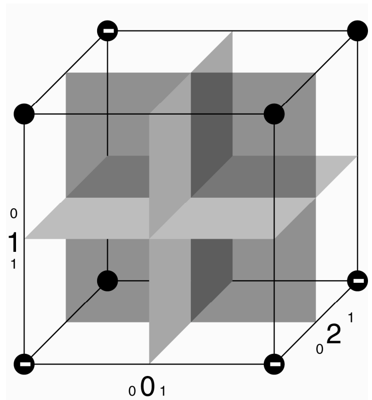

After Uθ [1] (5π/4)

## (3) Evolved quantum database algorithm, item at 0,0



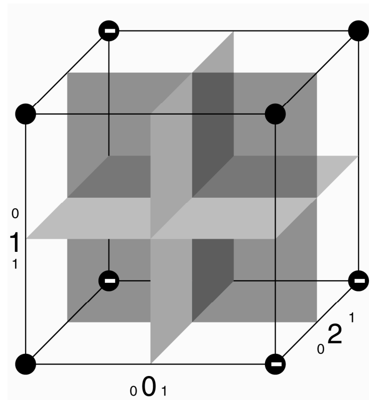

After Uθ [0] (π/4)

## (4) Evolved quantum database algorithm, item at 0,0



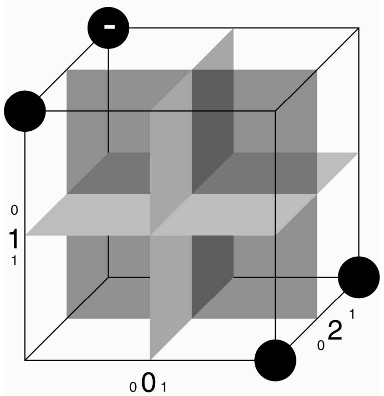

After DB [in:2,0; out:1](item in 0,0)

## (5) Evolved quantum database algorithm, item at 0,0



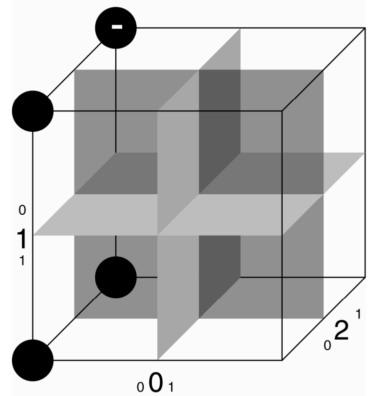

After CNOT [control: 1, target: 2]

## (6) Evolved quantum database algorithm, item at 0,0



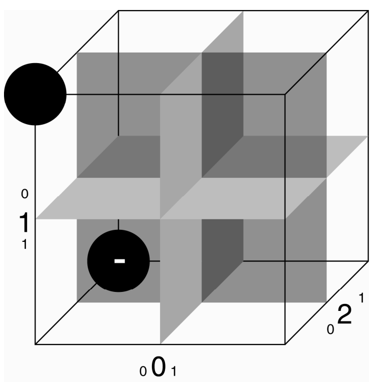

After Hadamard [1]

## (7) Evolved quantum database algorithm, item at 0,0



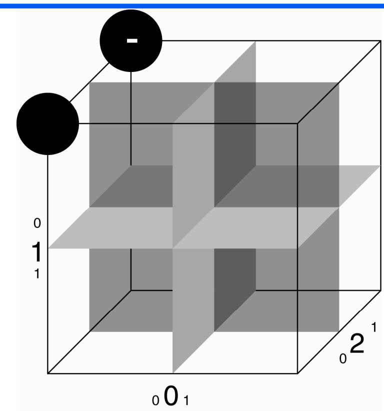

After CNOT [control: 1, target: 0]

## (8) Evolved quantum database algorithm, item at 0,0



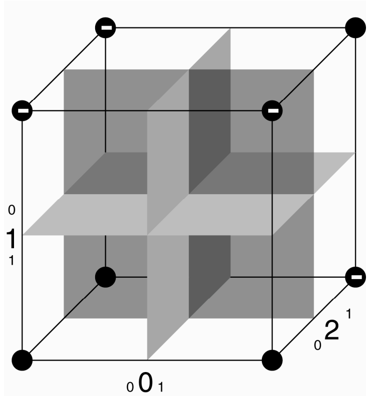

After Hadamard [1]

## (9) Evolved quantum database algorithm, item at 0,0



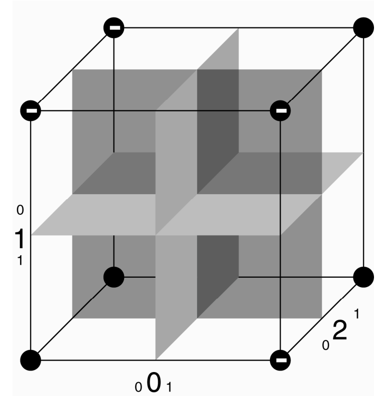

After CNOT [control: 2, target: 1]
Read output from qubits 1 (high) and 0(low)

## (4) Evolved quantum database algorithm, item at 0,1



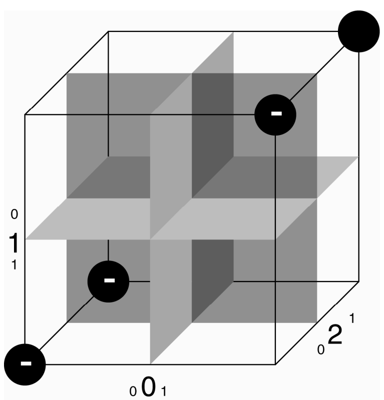

After DB [in:2,0; out:1](item in 0,1)

## (5) Evolved quantum database algorithm, item at 0,1



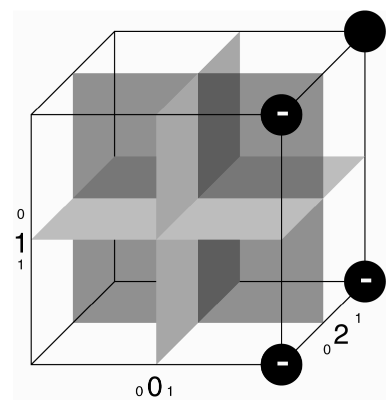

After CNOT [control: 1, target: 2]

## (6) Evolved quantum database algorithm, item at 0,1



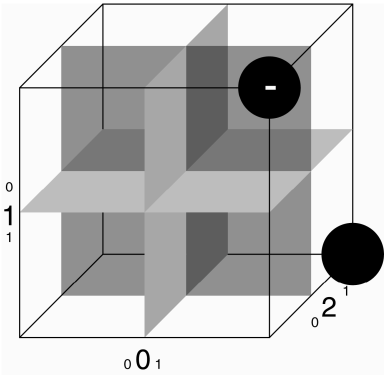

After Hadamard [1]

## (7) Evolved quantum database algorithm, item at 0,1



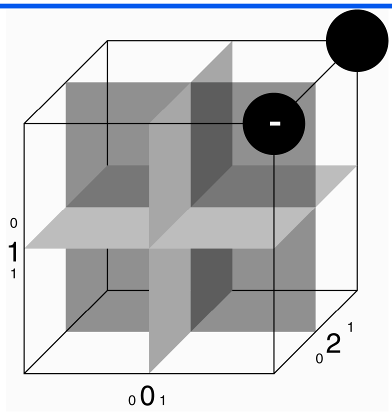

After CNOT [control: 1, target: 0]

## (8) Evolved quantum database algorithm, item at 0,1



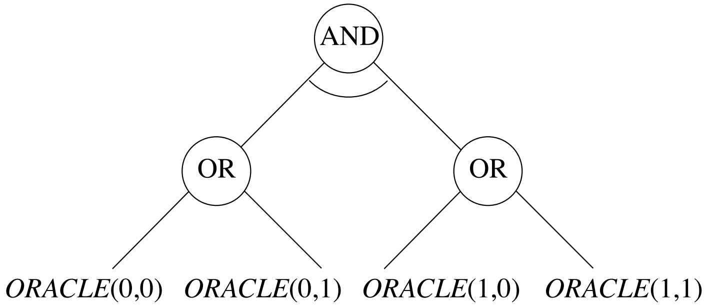

After Hadamard [1]

## (9) Evolved quantum database algorithm, item at 0,1

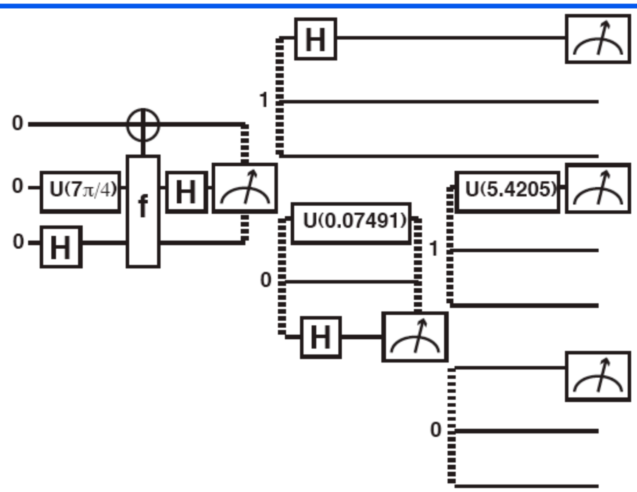



After CNOT [control: 2, target: 1]
Read output from qubits 1 (high) and 0(low)

## The and-or tree problem




## Evolved and-or gate array



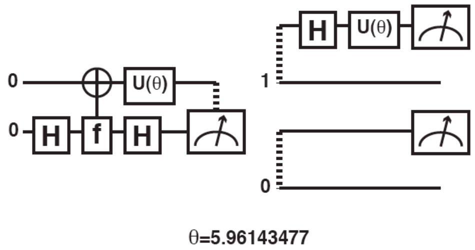
## Error/complexity measures

- *Las Vegas* ≡ always correct, but may answer "don't know" with some probability
- *Monte Carlo* ≡ may err, with some probability
- $p^e_{max}$ ≡ worst case probability of error
- $q^e_{max}$ ≡ worst case expected queries
- *Exact* ≡ $p^e_{max} = 0$

## Complexity of 2-bit AND/OR

- Classical Las Vegas: $q^e_{max}=3$
  - derived from [Saks and Wigderson 1986]
- Classical Monte Carlo: for $q^e_{max}=1$, $p^e_{max}\geq 1/3$
  - derived from [Santha 1991]
- Evolved Quantum Monte Carlo: $p^e_{max} = 0.28732$

## Derived better-than-classical OR

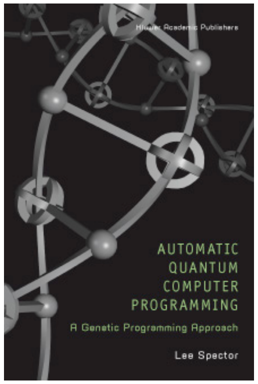



- Classical Monte Carlo: for $q^e_{max}=1$, $p^e_{max}\geq 1/6$ [Jozsa 1991, Beals 1998]
- Evolved algorithm $q^e_{max}=1$, $p^e_{max}=1/10$

## GP/QC research directions

- Application to additional problems with incompletely understood quantum complexity
- Exploration of communication capacity of quantum gates
- Evolution of hybrid quantum/classical algorithms.
- Evolution guided by ease of physical implementation.
- QC applications in AI
  - general AI search?
  - and-or trees and Prolog: quantum logic machine?
  - Bayesian networks?
- Genetic programming *on* quantum computers.

# Book




*Automatic Quantum Computer Programming: A Genetic Programming Approach*

Lee Spector. 2004.

Boston: Kluwer Academic Publishers.
ISBN 1-4020-7894-3.

http://hampshire.edu/lspector/aqcp/

# Sources: selected articles


- A. Steane, 1998. “Quantum Computing,” *Reports on Progress in Physics,* vol. 61, pp. 117-173. http://xxx.lanl.gov/abs/quant-ph/9708022
- P. Shor, 1998. “Quantum Computing,” *Documenta Mathematica,* vol. Extra Volume ICM, pp. 467–486. http://east.camel.math.ca/EMIS/journals/DMJDMV/xvol-icm/00/Shor.MAN.ps.gz
- J. Preskill, 1997. “Quantum Computing: Pro and Con,” Tech. Rep. CALT-68-2113, California Institute of Technology. http://xxx.lanl.gov/abs/quant-ph/9705032
- A. Barenco, C. H. Bennett, R. Cleve, D. P. DiVincenzo, N. Margolus, P. Shor, T. Sleator, J. Smolin, H. Weinfurter, 1995. “Elementary Gates for Quantum Computation,” submitted to *Physical Review A*. http://xxx.lanl.gov/abs/quant-ph/9503016
- N.J. Cerf, C. Adami, P.G. Kwiat, 1998. “Optical Simulation of Quantum Logic,” *Phys. Rev. A* 57, 1477. http://xxx.lanl.gov/abs/quant-ph/9706022
- L. Spector and H.J. Bernstein. 2003. “Communication Capacities of Some Quantum Gates, Discovered in Part through Genetic Programming,” in *Proc. of the Sixth Intl. Conf. on Quantum Communication, Measurement, and Computing*, edited by J.H. Shapiro and O. Hirota. Princeton, NJ: Rinton Press, Inc. pp. 500–503. http://hampshire.edu/lspector/pubs/spector-QCMC-prepress.pdf
- H. Barnum, H.J. Bernstein, and L. Spector. 2000. Quantum circuits for OR and AND of ORs. Journal of Physics A: Mathematical and General, Vol. 33 No. 45 (17 November 2000), pp. 8047–8057. http://hampshire.edu/lspector/pubs/jpa.pdf
- L. Spector, H. Barnum, H.J. Bernstein, N. Swamy, 1999. “Quantum Computing Applications of Genetic Programming,” in *Advances in Genetic Programming 3*, pp. 135–160, MIT Press.
- L. Spector, H. Barnum, H.J. Bernstein, N. Swamy, 1999. “Finding a Better-Than-Classical Quantum AND/OR Algorithm Using Genetic Programming,” in *Proc. 1999 Congress on Evolutionary Computation*, IEEE Press.
- L. Spector, H. Barnum, H.J. Bernstein, 1998. “Genetic Programming for Quantum Computers,” in *Genetic Programming 1998: Proceedings of the Third Annual Conference*, pp. 365–374, Morgan Kaufmann.


# Sources: selected books


- *Automatic Quantum Computer Programming: A Genetic Programming Approach*. By Lee Spector. Kluwer Academic Publishers. 2004.
- *Quantum Computation and Quantum Information*. By Michael A. Nielsen and Isaac L. Chuang. Cambridge University Press. 2000.
- *Schrödinger’s Machines: The Quantum Technology Reshaping Everyday Life*. By Gerard J. Milburn. W.H. Freeman and Company. 1997.
- *Explorations in Quantum Computing*. By Colin P. Williams and Scott H. Clearwater. Springer-Verlag/Telos. 1997.
- *The Fabric of Reality*. By David Deutsch. Penguin Books. 1997.
- *The Large, the Small and the Human Mind*. By Roger Penrose, with Abner Shimony, Nancy Cartwright, and Stephen Hawking. Cambridge University Press. 1997.
- *QED: The Strange Theory of Light and Matter*. By Richard P. Feynman. Princeton University Press. 1985.


# Sources: selected WWW sites

- Oxford’s Center for Quantum Computation: http://www.qubit.org/
- Stanford-Berkeley-MIT-IBM NMR Quantum Computation Project: http://squint.stanford.edu/
- Quantum Information and Computation (Caltech - MIT - USC): http://theory.caltech.edu/~quic/index.html
- Quantum Computation at ISI/USC: http://www.isi.edu/acal/quantum/quantum_intro.html
- Los Alamos National Laboratory quantum physics e-print archive: http://xxx.lanl.gov/form/quant-ph
- John Preskill’s Physics 229 course web page (many good links): http://www.theory.caltech.edu/people/preskill/ph229/
- Samuel L. Braunstein’s on-line tutorial: http://www.sees.bangor.ac.uk/~schmuel/comp/comp.html
- NIST Ion Storage Group: http://www.bldrdoc.gov/timefreq/ion/index.htm
- QGAME, Quantum Gate And Measurement Emulator: http://hampshire.edu/lspector/qgame.html