Quantum Computation: a computational perspective

Rod Canfield

erc@cs.uga.edu

Department of Computer Science

University of Georgia

UGA Physics Department Colloquium

April 14, 2011

Outline of Talk

What is a quantum computer

Aspects of parallelism

The role of probability

The killer ap

The complexity zoo

Definition

A quantum computer is a peripheral device (black box) attached to your (non-quantum) computer which is capable of executing four commands:

1. Reset
2. Apply U on bit i
3. $\langle$ to-be-revealed $\rangle$
4. Read output (used once only, at the end of the computation)

Mental picture

Conceptually, the black-box stores an array of $N = 2^n$ complex numbers

$$a_0, a_1, \ldots, a_{N-1}$$

Array

index	contents
000	a_0
001	a_1
010	a_2
011	a_3
100	a_4
101	a_5
110	a_6
111	a_7

n is the number of bits in an address; $N = 2^n$ is the size of the array
Each stored value a_i is a complex number

Array – Pairing 0

index	contents	pairing 0
000	a_0	a_0, a_1
001	a_1	
010	a_2	a_2, a_3
011	a_3	
100	a_4	a_4, a_5
101	a_5	
110	a_6	a_6, a_7
111	a_7	

Array – Pairing 1

index	contents	pairing 1
000	a_0	a_0, a_2
001	a_1	a_1, a_3
010	a_2	
011	a_3	
100	a_4	a_4, a_6
101	a_5	a_5, a_7
110	a_6	
111	a_7	

Array – Pairing 2

index	contents	pairing 2
000	a_0	a_0, a_4
001	a_1	a_1, a_5
010	a_2	a_2, a_6
011	a_3	a_3, a_7
100	a_4	
101	a_5	
110	a_6	
111	a_7	

Meaning of Command 1

“Reset” means to initialize the array:

$$a_0 := 1, \quad \text{and} \quad a_i := 0, \ \text{for } 0 < i < N$$

Meaning of Command 2

“Apply U on bit i” means to apply the 2×2 matrix U to each of the pairs

$$\begin{bmatrix} a_0 \\ a_{2^i} \end{bmatrix}, \quad \begin{bmatrix} a_1 \\ a_{2^i+1} \end{bmatrix} \quad \cdots$$

For example, when $n = 3$, “Apply U on bit 1” causes

$$\begin{bmatrix} a_0 \\ a_2 \end{bmatrix} \leftarrow \begin{bmatrix} \alpha & \beta \\ \gamma & \delta \end{bmatrix} \begin{bmatrix} a_0 \\ a_2 \end{bmatrix} \qquad \begin{bmatrix} a_1 \\ a_3 \end{bmatrix} \leftarrow \begin{bmatrix} \alpha & \beta \\ \gamma & \delta \end{bmatrix} \begin{bmatrix} a_1 \\ a_3 \end{bmatrix}$$

$$\begin{bmatrix} a_4 \\ a_6 \end{bmatrix} \leftarrow \begin{bmatrix} \alpha & \beta \\ \gamma & \delta \end{bmatrix} \begin{bmatrix} a_4 \\ a_6 \end{bmatrix} \qquad \begin{bmatrix} a_5 \\ a_7 \end{bmatrix} \leftarrow \begin{bmatrix} \alpha & \beta \\ \gamma & \delta \end{bmatrix} \begin{bmatrix} a_5 \\ a_7 \end{bmatrix}$$

Meaning of Command 4

“Read output” means for the black box to return to the master computer an integer i in the range $0 \leq i < N$,

(that is, an index)

chosen according to the probabilities

$$|a_0|^2, |a_1|^2, \ldots |a_{N-1}|^2$$

The importance of being unitary

When the “Read output” command is executed, we would like

$$\sum_{i=0}^{N-1} |a_i|^2 \;=\; 1.$$

This can be assured by requiring each 2×2 matrix U used in Command 2 to be a unitary matrix

What is a_i ?

We never “see” any of the complex numbers a_i

The device is“storing” them for the ultimate purpose of making a random choice

Quantum Pretender

If we stick with only commands 1, 2, and 4 then fairly large arrays can be simulated efficiently.

Namely, for each i in the range $0 \le i < N$ we keep up with

$$p_i = \mathrm{Prob}\{\mathrm{bit}\ i = 1\}$$

and, when asked to make a final report by “Read output” we return the bit string $b_{n-1} \cdots b_1 b_0$ with probability

$$\phi_{n-1} \times \cdots \times \phi_1 \times \phi_0$$

where

$$\phi_i = \begin{cases} p_i & \text{if } b_i = 1 \\ 1 - p_i & \text{if } b_i = 0 \end{cases}$$

The Pretender's Method

Maintain $2n$ pairs of complex numbers

$$(w_0, z_0), \ldots, (w_{n-1}, z_{n-1})$$

On "Reset", set (w_i, z_i) to $(1, 0)$ for all i

On "Apply U on bit i",

$$\begin{bmatrix} w_i \\ z_i \end{bmatrix} \leftarrow \begin{bmatrix} \alpha & \beta \\ \gamma & \delta \end{bmatrix} \begin{bmatrix} w_i \\ z_i \end{bmatrix}$$

On "Read output", generate b_i independently using

$$p_i = |z_i|^2$$

Why it Works

As long as only commands of type 1, 2, or 4 are used, the 2^n complex numbers $a_0, a_1, \ldots, a_{N-1}$ can be remembered by symbolic expansion of the product

$$\left(w_{n-1}|0\rangle + z_{n-1}|1\rangle\right) \otimes \cdots \otimes \left(w_0|0\rangle + z_0|1\rangle\right)$$

Why it Works

As long as only commands of type 1, 2, or 4 are used, the 2^n complex numbers $a_0, a_1, \ldots, a_{N-1}$ can be remembered by symbolic expansion of the product

$$(w_{n-1}|0\rangle + z_{n-1}|1\rangle) \otimes \cdots \otimes (w_0|0\rangle + z_0|1\rangle)$$

$$= \quad \cdots \quad + \; w_{n-1}z_{n-2}\cdots z_1 w_0 \, |0\rangle \otimes |1\rangle \otimes |1\rangle \otimes |0\rangle \; + \quad \cdots$$

Why it Works

As long as only commands of type 1, 2, or 4 are used, the 2^n complex numbers $a_0, a_1, \ldots, a_{N-1}$ can be remembered by symbolic expansion of the product

$$(w_{n-1}|0\rangle + z_{n-1}|1\rangle) \otimes \cdots \otimes (w_0|0\rangle + z_0|1\rangle)$$

$$= \quad \cdots \quad + \; w_{n-1}z_{n-2}\cdots z_1 w_0 \,|0\rangle \otimes |1\rangle \otimes |1\rangle \otimes |0\rangle \; + \quad \cdots$$

$$= \quad \cdots \quad + \; w_{n-1}z_{n-2}\cdots z_1 w_0 \,|01\cdots 10\rangle \; + \quad \cdots$$

Command 3

Apply NOT on bit i under control of bit j

$0 \leq i, j < n, i \neq j$

NOT is the 2×2 unitary

$$\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} w \\ z \end{bmatrix} = \begin{bmatrix} z \\ w \end{bmatrix}$$

This command is called "Controlled NOT" denoted CNOT

Command 3, continued

Specifying two bits, i and j, partitions the 2^n complex numbers into $2^n/4$ quadruples. For example, $n = 4$, $\{i, j\} = \{1, 2\}$

binary	quadruple
$0{*}{*}0$	a_0, a_2, a_4, a_6
$0{*}{*}1$	a_1, a_3, a_5, a_7
$1{*}{*}0$	$a_8, a_{10}, a_{12}, a_{14}$
$1{*}{*}1$	$a_9, a_{11}, a_{13}, a_{15}$

Command 3, continued

Within each quadruple, swap the two numbers whose control bit is 1. $n = 4$ target bit 2, control bit 1:

$$\begin{bmatrix} a_0 \\ a_2 \\ a_4 \\ a_6 \end{bmatrix} \leftarrow \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} a_0 \\ a_2 \\ a_4 \\ a_6 \end{bmatrix} = \begin{bmatrix} a_0 \\ a_6 \\ a_4 \\ a_2 \end{bmatrix},$$

$$\begin{bmatrix} a_1 \\ a_3 \\ a_5 \\ a_7 \end{bmatrix} \leftarrow \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} a_1 \\ a_3 \\ a_5 \\ a_7 \end{bmatrix} = \begin{bmatrix} a_1 \\ a_7 \\ a_5 \\ a_3 \end{bmatrix},$$

etc.

Net effect

For $n = 4$, target bit 2, control bit 1:

$$\begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \\ a_6 \\ a_7 \\ a_8 \\ a_9 \\ a_{10} \\ a_{11} \\ a_{12} \\ a_{13} \\ a_{14} \end{bmatrix} \mapsto \begin{bmatrix} a_0 \\ a_1 \\ a_6 \\ a_7 \\ a_4 \\ a_5 \\ a_2 \\ a_3 \\ a_8 \\ a_9 \\ a_{14} \\ a_{15} \\ a_{12} \\ a_{13} \\ a_{10} \end{bmatrix}$$

Quantum Program

Reset, sequence of type 2 & type 3 commands, read result

The effect of the interior sequence is a $2^n \times 2^n$ unitary matrix

Can every unitary matrix be fabricated ?

Fourier Transform

The $N \times N$ F.T.

$$\begin{bmatrix} A_0 \\ \vdots \\ A_{N-1} \end{bmatrix} = \frac{1}{N^{1/2}} \begin{bmatrix} & \cdots & \\ \vdots & \exp(2\pi ijk/N) & \vdots \\ & \cdots & \end{bmatrix} \begin{bmatrix} a_0 \\ \vdots \\ a_{N-1} \end{bmatrix}$$

$$|x\rangle \mapsto N^{-1/2} \sum_{y \in \{0,1\}^n} e|y\rangle$$

1805, Carl Friedrich Gauss, asteroids Pallas and Juno
1965, J. W. Cooley and John W. Tukey, reinvention & computerization

Boolean Functions

$$f : \{0,1\}^n \rightarrow \{0,1\}$$

$$y = f(x_0, \ldots, x_{n-1})$$

Example

$$MAJ(x_0, \ldots, x_{n-1}) = \begin{cases} 1 & \text{if \# ones} \geq \text{\# zeros} \\ 0 & \text{otherwise} \end{cases}$$

Fabrication

Every Boolean function can be fabricated from the elementary operations AND, OR, NOT

Associated Unitary Operator

Let $f : \{0,1\}^n \rightarrow \{0,1\}$
be a Boolean function. The operator U_f acts on the computational basis by the rule

$$U_f|x_{n-1}\cdots x_1x_0, y\rangle \stackrel{\text{def}}{=} |x_{n-1}\cdots x_1x_0, y \oplus f(x_0,\ldots,x_{n-1})\rangle$$

U_f is a $2^{n+1} \times 2^{n+1}$ matrix

Operator U_f can be fabricated by a number of quantum gates that is proportional to the number of AND's, OR's, NOT's needed for f

Parallelism

* Familiar hypercube architecture (see Graphic 1)

* Obtain state

$$\frac{1}{2^{n/2}} \sum_{x \in \{0,1\}^n} |x_{n-1} \cdots x_1 x_0, f(x_0, \ldots, x_{n-1})\rangle$$

in time $n + \mathrm{BitComplexity}(f)$.

Hypercube obsolete?

SICORTEX at Argonne
Our system has: 972 nodes
* - 6 cores per node
* - 4 GB/memory per node
* - 1300 MB/s interconnect bandwidth per node
* - 1 us of latency
* - the system has a novel network topology, described at:
http://en.wikipedia.org/wiki/Kautz_graph

Student Comment/Question

“Why doesn’t this stuff look anything like what
I did in my quantum mechanics class last semester ?”

Probabilistic Aspects

Randomized algorithms can be compared to exponential-sized search spaces with good “odds”

Very popular example within Computer Science is primality testing (see Graphic 2)

Solovay, Robert M. and Strassen, Volker
"A fast Monte-Carlo test for primality"
SIAM Journal on Computing (1977).

Derandomization

Very active subfield of theory of computation

Major success recently: Lovasz Local Lemma

In the case of primes:
Manindra Agrawal, Neeraj Kayal, Nitin Saxena
"PRIMES is in P"
Annals of Mathematics (2004)

Probabilistic Algorithms

1940s, physicists in Los Alamos

Buffon's needle problem
Georges-Louis Leclerc, Comte de Buffon
Essai d'arithmétique morale
Vol. 4 of the Supplément ã l'Histoire Naturelle (1777)

Artificial Randomness

Pseudo-random number generators

Any one who considers arithmetical methods of producing random digits is, of course, in a state of sin.
- v. Neumann

A.M. Ferrenberg, D.P. Landau and K. Binder,
"Statistical and Systematic Errors
in Monte Carlo Simulations,"
J. Stat. Phys. (1991).

Probability in Q. Algorithms

By nature, each program is a probabilistic algorithm

With a quantum computer, we have a “true” random number generator

Killer Ap

Factoring

The basis of the algorithm is number theoretic

There is some non-trivial classical computing

There is an essential quantum core

The Order

Factor: 1007

Need base a and even exponent r such that

$$1007 \text{ divides } (a^r - 1)$$

and no smaller positive exponent works.

It's Not Even

Factor: 1007

Try $a = 16$

$$1007 \text{ divides } (16^{117} - 1)$$

and no smaller positive exponent works.

OK, Even, but $\cdots$

Factor: 1007

Try $a = 29$

$$1007 \text{ divides } (29^{234} - 1)$$

and no smaller positive exponent works.

$$\mathrm{GCD}(1007, 29^{117} - 1) = 1$$

JACKPOT!!

Try $a = 11$

$$1007 \text{ divides } 11^{78} - 1$$

and no smaller positive exponent works.

$$\mathrm{GCD}(1007, 11^{39} - 1) = 19$$

$$1007 = 19 \times 53$$

The Quantum Core

The quantum computer enables us to find the order

Example: 15

Using 11 qubits

x	0	1	2	3	4	$\cdots$	15
7^x	1	7	4	13	1	$\cdots$	13

$$|0\rangle|1\rangle \; + \; |1\rangle|7\rangle \; + \; |2\rangle|4\rangle \; + \; \cdots$$

$$|2\rangle|4\rangle \; + \; |6\rangle|4\rangle \; + \; |10\rangle|4\rangle \; + \; \cdots$$

FFT: $|0\rangle - |512\rangle + |1024\rangle - |1536\rangle$

Factoring 15

(For real)
Letters to Nature
Nature 414, 883-887 (20 December 2001)
Experimental realization of Shor's quantum factoring algorithm using nuclear magnetic resonance
Vandersypen, Steffen, Breyta, Yannoni, Sherwood, & Chuang

Why Factoring?

Secure data transmission requires that the two parties share a key (Example: AES)

Diffie-Hellman key exchange 1976
Rivest-Shamir-Adelman 1978

Bob and Alice choose prime p, base g
Alice to Bob: g^a mod p, a secret
Bob to Alice: g^b mod p, b secret
Now they share g^{ab} mod p, listeners bewildered

Turing Machines

Provides a way to define complexity

(see Graphic 3)

Two complexity classes: P and PSPACE

Non-deterministic Computation

In the TM's program, for a given state, symbol pair, there are some finite number of moves

Models two important concepts

- parallelism
- it's easier to check than to find

One more complexity class: NP

Complexity Zoo

http://qwiki.stanford.edu/index.php/Complexity_Zoo

494 classes

originally established by Scott Aaronson,
2004 doctoral thesis:
Limits on Efficient Computation in the Physical World

Known Containments

$$P \subseteq NP$$

Is the inclusion proper, P $\stackrel{?}{=}$ NP
one of the Clay institute's Millennium Prize Problems

Known Containments

$$P \subseteq NP \subseteq PSPACE$$

$$P \subseteq BPP \subseteq QPP \subseteq PSPACE$$

Quantum vs NP-c

No NP-complete problem has a known polynomial-time quantum algorithm (presently)

Complexity of Factoring

December 12, 2009
T. Kleinjung, K. Aoki, J. Franke, A. K. Lenstra, E. Thomé, J. W. Bos, P. Gaudry, A. Kruppa, P. L. Montgomery, D. A. Osvik, H. te Riele, A. Timofeev, P. Zimmermann plus researchers from the CWI, the EPFL, INRIA and NTT factored RSA-768, a 232-digit semiprime using the equivalent of almost 2000 years of computing on a single core 2.2 GHz AMD Opteron.

b-bit number:

$$\exp\left((1+o(1))\left(\frac{64}{9}b\right)^{\frac{1}{3}}(\log b)^{\frac{2}{3}}\right)$$

GNFS, Generalized number field sieve

Summary

- Based on reliable quantum-mechanical principles, we can envision a model of quantum computation
- The envisioned model can factor integers surprisingly fast as measured by complexity theory and practice
- The integer factorisation problem lies at the heart of a number of secure communication protocols

Acknowledgments

Students who have taken CSCI/MATH/PHYS 4612/6612

Semmy Purewal (2007) and Shahab Razavi (2009)

Special ack: Bob Anderson