Introduction to Quantum Information Processing

CS 467 / CS 667
Phys 667 / Phys 767
C&O 481 / C&O 681

Lecture 4 (2008)

Richard Cleve
DC 2117
cleve@cs.uwaterloo.ca

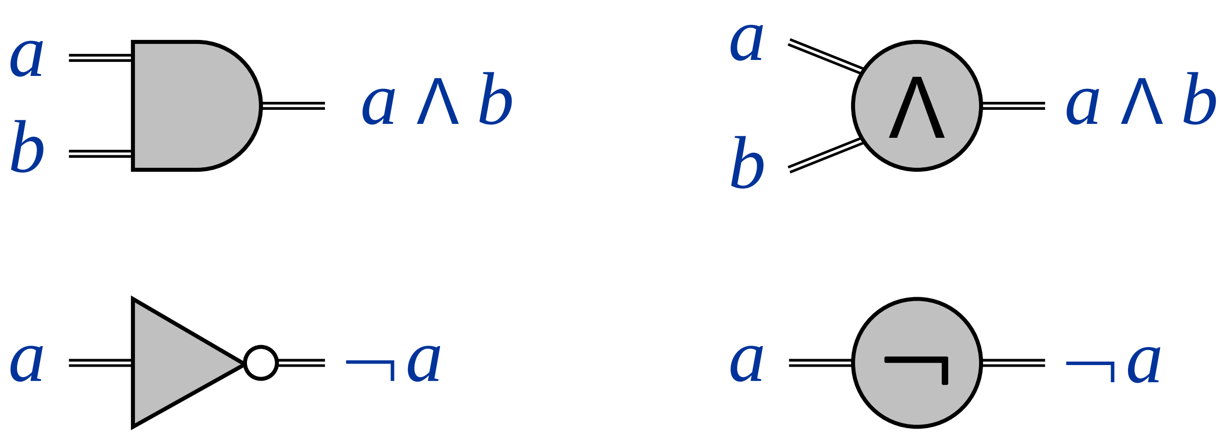
Classical computations as circuits

Classical (boolean logic) gates

“old” notation — “new” notation

AND gate

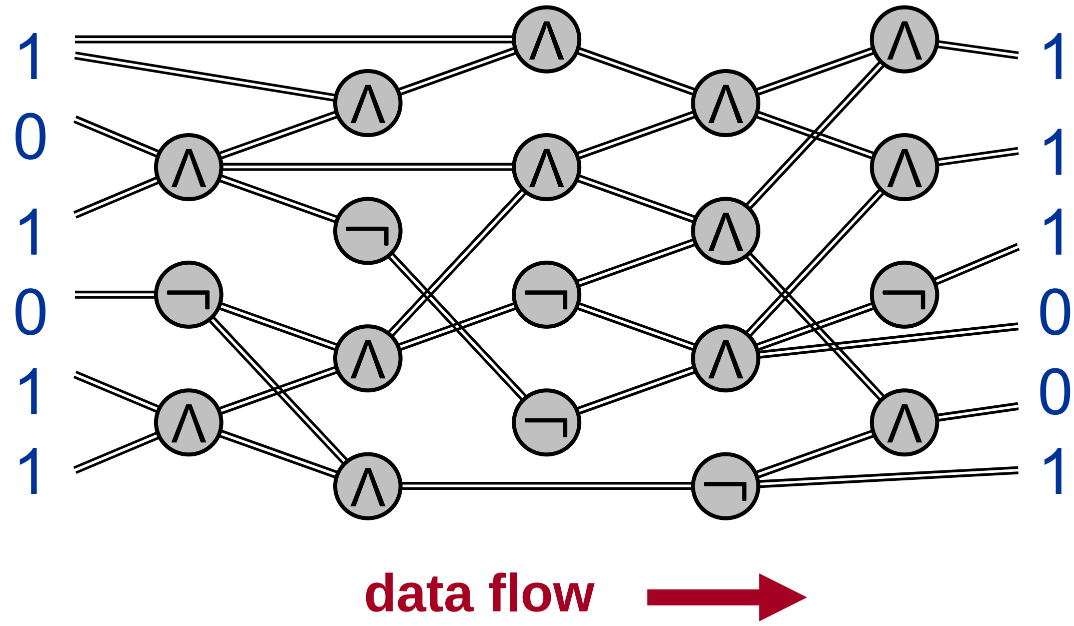
NOT gate



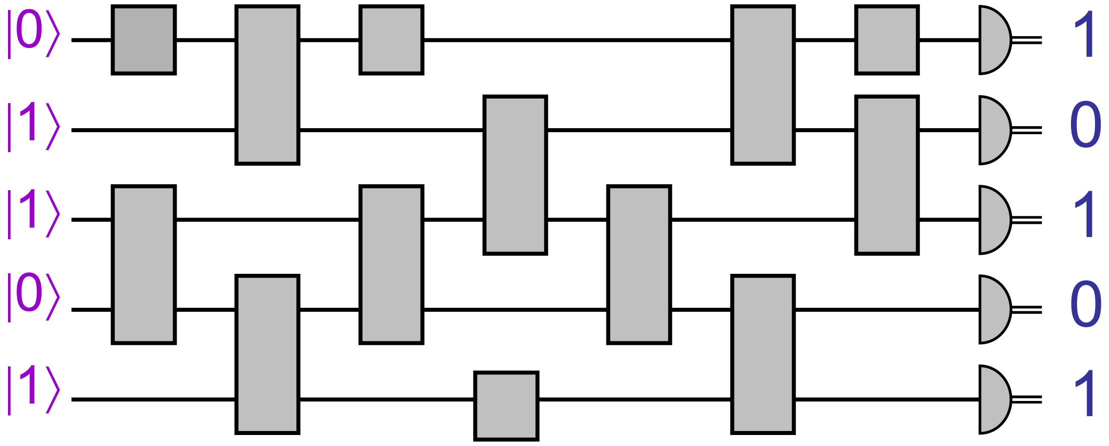
Note: an **OR** gate can be simulated by one **AND** gate and three **NOT** gates (since $a \vee b = \neg(\neg a \wedge \neg b)$)

Models of computation

Classical circuits:



Quantum circuits:



Multiplication problem

Input: two n-bit numbers (e.g. 101 and 111)

Output: their product (e.g. 100011)

- "Grade school" algorithm costs $O(n^2)$
- Best currently-known ***classical*** algorithm costs $O(n \log n \operatorname{loglog} n)$
- Best currently-known ***quantum*** method: same

Factoring problem

Input: an n-bit number (e.g. 100011)

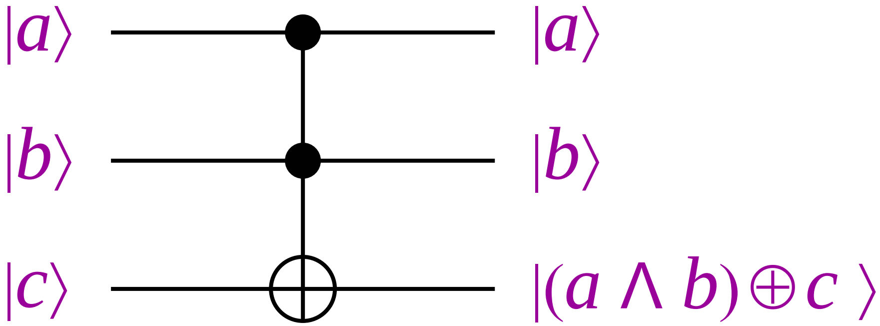
Output: their product (e.g. 101, 111)

- Trial division costs $\approx 2^{n/2}$
- Best currently-known ***classical*** algorithm costs $\approx 2^{n^{1/3}}$
- Hardness of factoring is the basis of the security of many cryptosystems (e.g. RSA)
- Shor's ***quantum*** algorithm costs $\approx n^2$
- Implementation would break RSA and many other cryptosystems

Simulating ***classical*** circuits with ***quantum*** circuits

Toffoli gate

(Sometimes called a "controlled-controlled-NOT" gate)



Matrix representation:

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

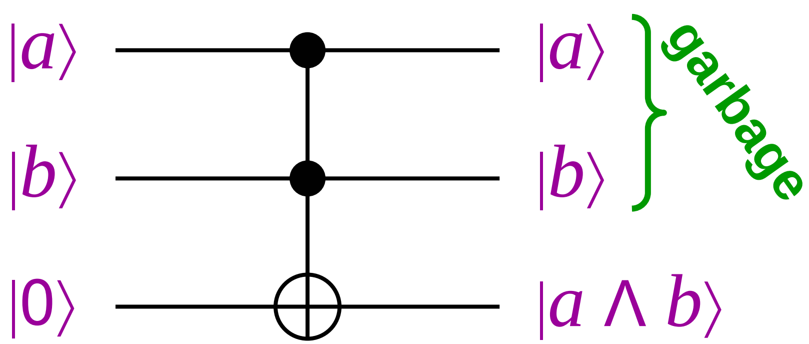
In the computational basis, it negates the third qubit iff the first two qubits are both $|0\rangle$

Quantum simulation of classical

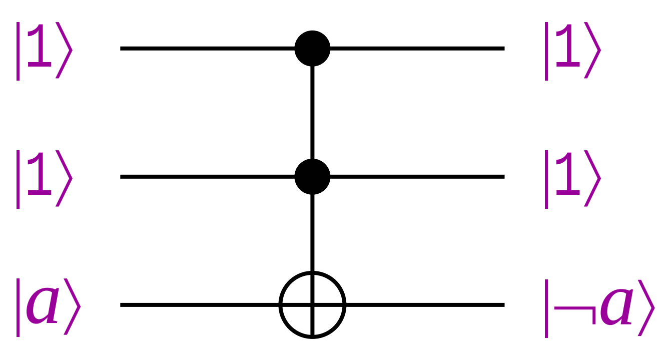
Theorem: a classical circuit of size s can be simulated by a quantum circuit of size $O(s)$

Idea: using Toffoli gates, one can simulate:

AND gates



NOT gates

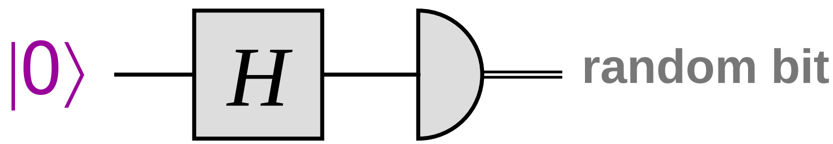


This garbage will have to be reckoned with later on …

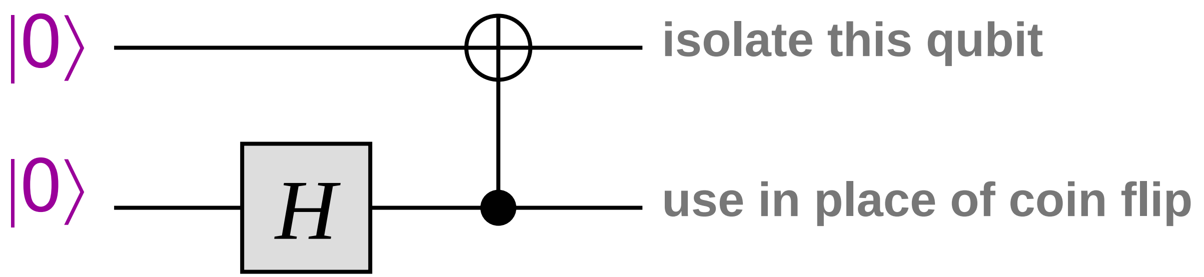
Simulating probabilistic algorithms

Since quantum gates can simulate **AND** and **NOT**, the outstanding issue is how to simulate randomness

To simulate "coin flips", one can use the circuit:



It can also be done without intermediate measurements:



Exercise: prove that this works

Simulating ***quantum*** circuits with ***classical*** circuits

Classical simulation of quantum

Theorem: a quantum circuit of size s acting on n qubits can be simulated by a classical circuit of size $O(sn^2 2^n) = O(2^{cn})$

Idea: to simulate an n-qubit state, use an array of size 2^n containing values of all 2^n amplitudes within precision 2^{-n}

α_{000}
α_{001}
α_{010}
α_{011}
:
α_{111}

Can adjust this state vector whenever a unitary operation is performed at cost $O(n^2 2^n)$

From the final amplitudes, can determine how to set each output bit

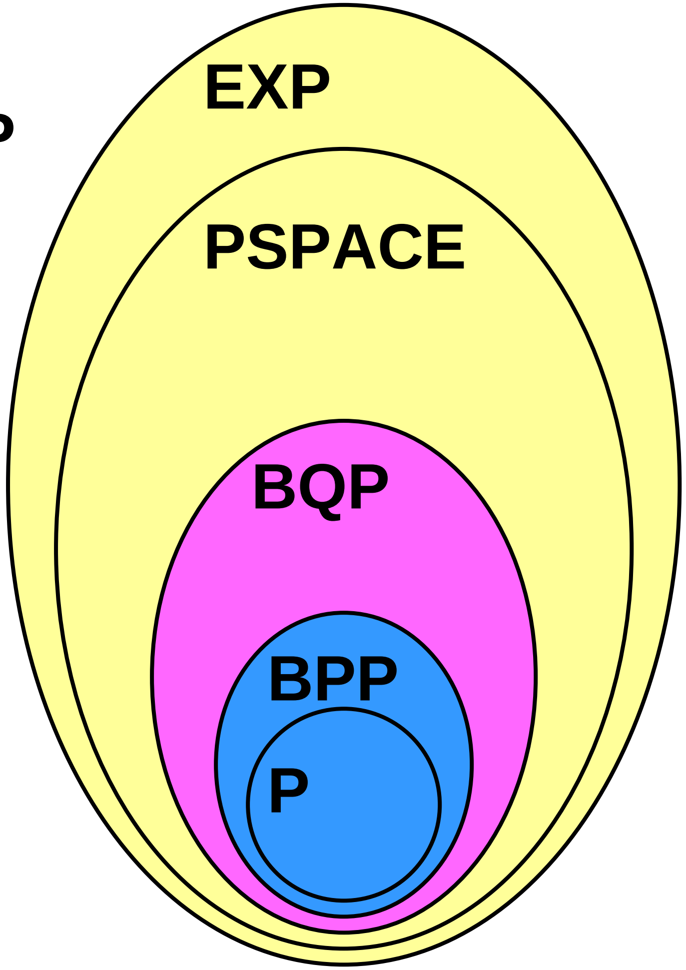
Exercise: show how to do the simulation using only a polynomial amount of ***space*** (memory)

Some complexity classes

- **P (polynomial time):** problems solved by $O(n^c)$-size classical circuits (decision problems and uniform circuit families)
- **BPP (bounded error probabilistic polynomial time):** problems solved by $O(n^c)$-size ***probabilistic*** circuits that err with probability $\leq$ ¼
- **BQP (bounded error quantum polynomial time):** problems solved by $O(n^c)$-size ***quantum*** circuits that err with probability $\leq$ ¼
- **EXP (exponential time):** problems solved by $O(2^{n^c})$-size circuits.

Summary of basic containments

P $\subseteq$ BPP $\subseteq$ BQP $\subseteq$ PSPACE $\subseteq$ EXP

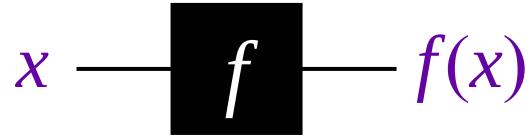


This picture will be fleshed out more later on

Simple quantum algorithms in the query scenario

Query scenario

Input: a function f, given as a black box (a.k.a. oracle)



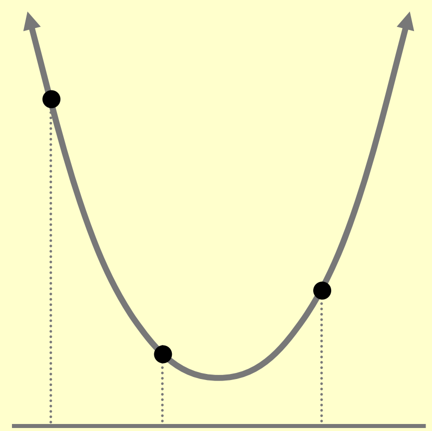
Goal: determine some information about f making as few queries to f (and other operations) as possible

Example: polynomial interpolation

Let: $f(x) = c_0 + c_1 x + c_2 x^2 + \ldots + c_d\, x^d$

Goal: determine $c_0\,, c_1\,, c_2\,, \ldots\,, c_d$

Question: How many f-queries does one require for this?

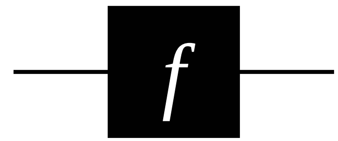


Answer: $d+1$

Deutsch's problem

Deutsch's problem

Let $f : \{0,1\} \rightarrow \{0,1\}$



There are ***four*** possibilities:

x	$f_1(x)$
0	0
1	0

x	$f_2(x)$
0	1
1	1

x	$f_3(x)$
0	0
1	1

x	$f_4(x)$
0	1
1	0

Goal: determine whether or not $f(0) = f(1)$ (i.e. $f(0) \oplus f(1)$)

Any classical method requires ***two*** queries

What about a quantum method?

To be continued …

Introduction to Quantum Information Processing

CS 467 / CS 667
Phys 667 / Phys 767
C&O 481 / C&O 681

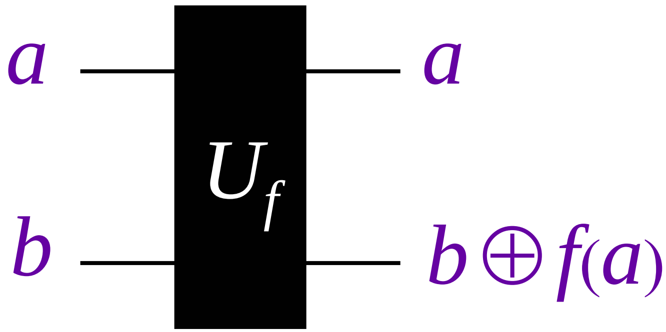
Lecture 5 (2008)

Richard Cleve
DC 2117
cleve@cs.uwaterloo.ca

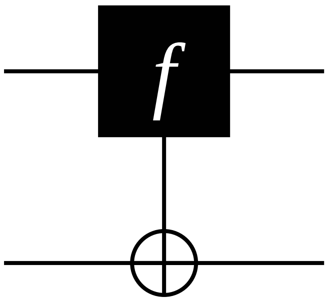
Deutsch’s problem

(continued)

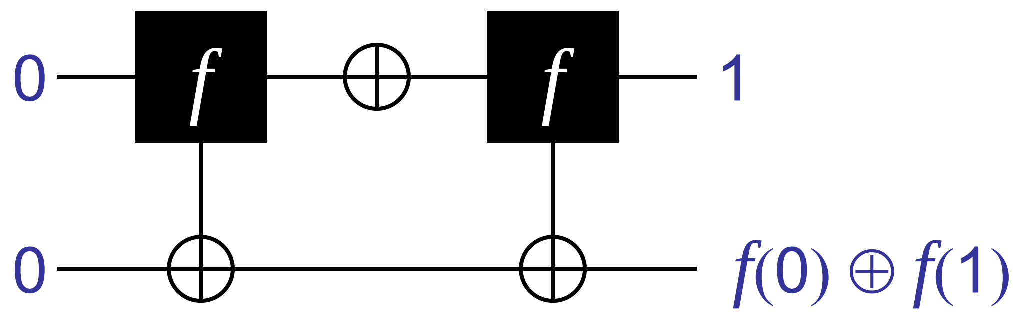
Reversible black box for *f*



alternate notation:

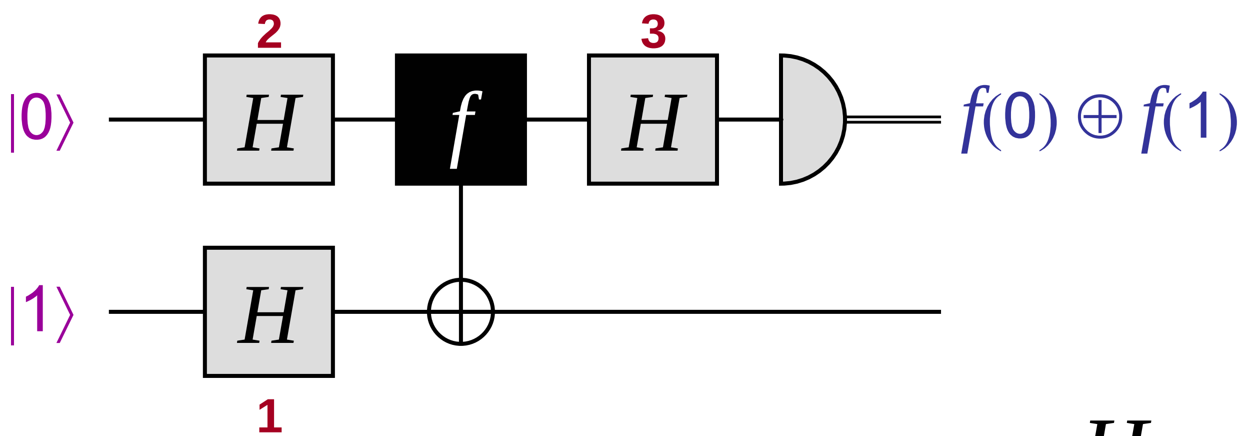


A classical algorithm:
(still requires 2 queries)



2 queries + **1** auxiliary operation

Quantum algorithm for Deutsch



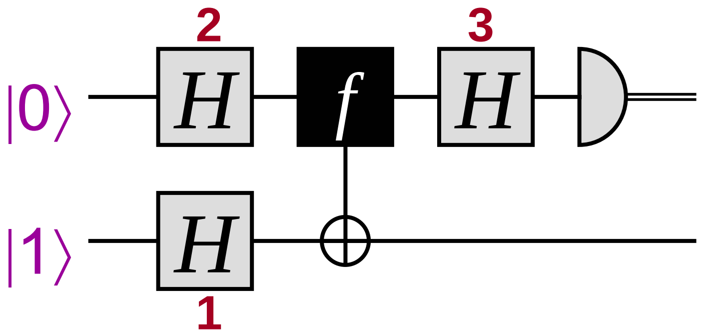
1 query + **4** auxiliary operations

$$H = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

How does this algorithm work?

Each of the three H operations can be seen as playing a different role ...

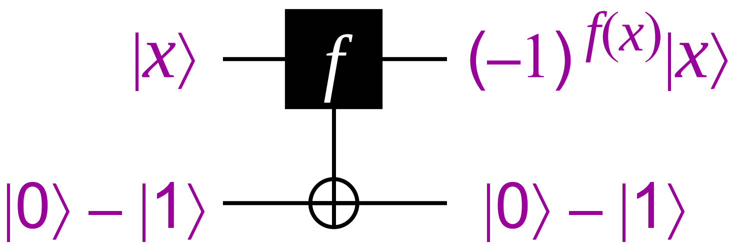
Quantum algorithm (1)



1. Creates the state $|0\rangle - |1\rangle$, which is an eigenvector of

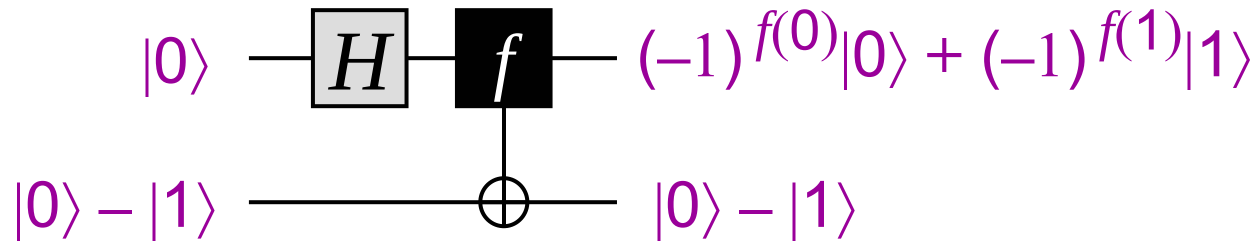
$$\begin{cases} \textbf{NOT} \text{ with eigenvalue } -1 \\ I \quad \text{with eigenvalue } +1 \end{cases}$$

This causes f to induce a ***phase shift*** of $(-1)^{f(x)}$ to $|x\rangle$



Quantum algorithm (2)

2. Causes f to be queried ***in superposition*** (at $|0\rangle + |1\rangle$)



x	$f_1(x)$
0	0
1	0

x	$f_2(x)$
0	1
1	1

f_1, f_2: $\pm(|0\rangle + |1\rangle)$

x	$f_3(x)$
0	0
1	1

x	$f_4(x)$
0	1
1	0

f_3, f_4: $\pm(|0\rangle - |1\rangle)$

Quantum algorithm (3)

3. Distinguishes between $\pm(|0\rangle + |1\rangle)$ and $\pm(|0\rangle - |1\rangle)$

$$\pm(|0\rangle + |1\rangle) \overset{H}{\longleftrightarrow} \pm|0\rangle$$

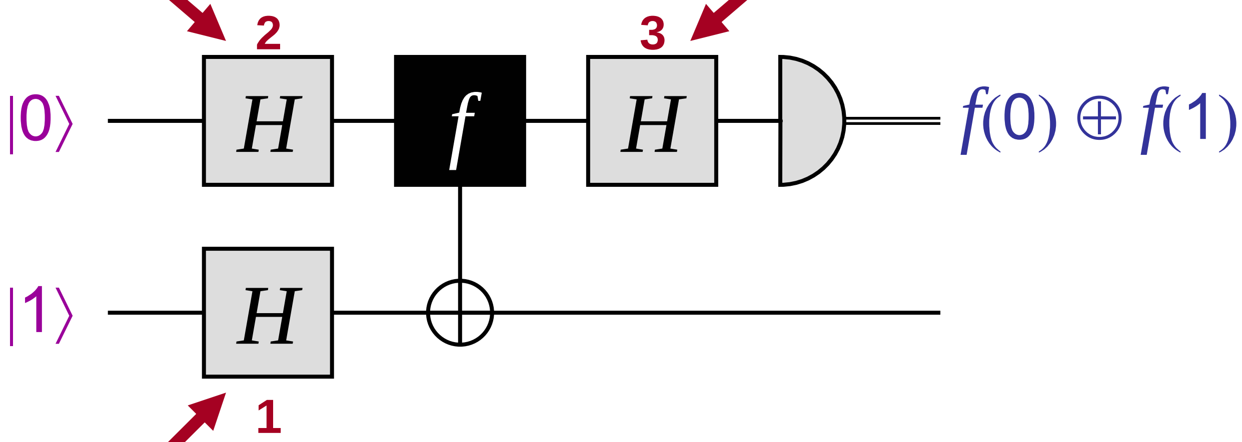
$$\pm(|0\rangle - |1\rangle) \overset{H}{\longleftrightarrow} \pm|1\rangle$$

Summary of Deutsch's algorithm

Makes only one query, whereas two are needed classically

produces superpositions of inputs to f: $|0\rangle + |1\rangle$

extracts phase differences from $(-1)^{f(0)}|0\rangle + (-1)^{f(1)}|1\rangle$



constructs eigenvector so f-queries induce phases: $|x\rangle \rightarrow (-1)^{f(x)}|x\rangle$

One-out-of-four search

One-out-of-four search

Let $f : \{0,1\}^2 \rightarrow \{0,1\}$ have the property that there is exactly one $x \in \{0,1\}^2$ for which $f(x) = 1$

Four possibilities:

x	$f_{00}(x)$
00	1
01	0
10	0
11	0

x	$f_{01}(x)$
00	0
01	1
10	0
11	0

x	$f_{10}(x)$
00	0
01	0
10	1
11	0

x	$f_{11}(x)$
00	0
01	0
10	0
11	1

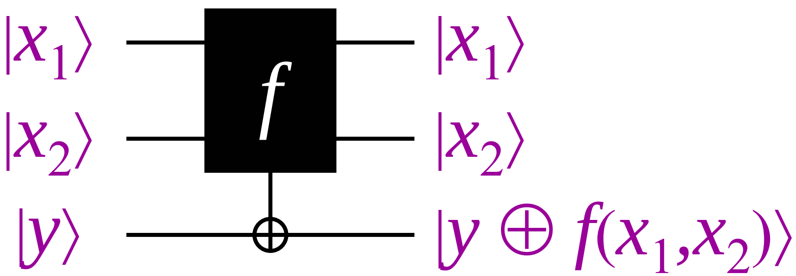
Goal: find $x \in \{0,1\}^2$ for which $f(x) = 1$

What is the minimum number of queries ***classically?*** ____

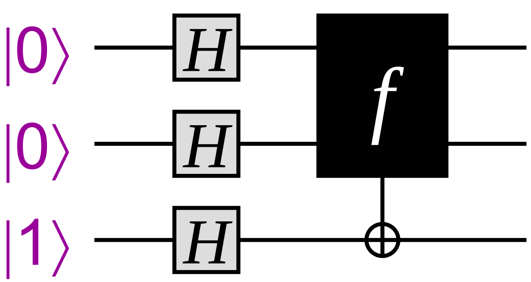
Quantumly? ____

Quantum algorithm (I)

Black box for 1-4 search:



Start by creating phases in superposition of all inputs to f:



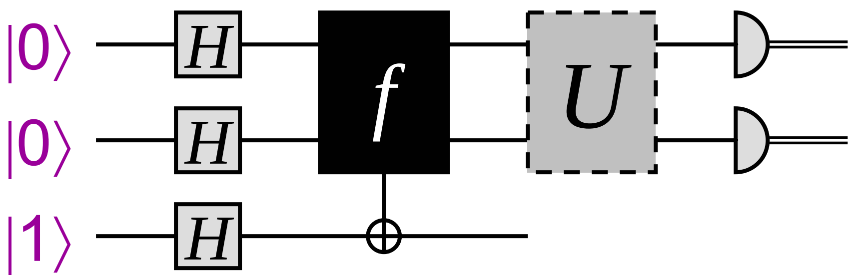
Input state to query?

$(|00\rangle + |01\rangle + |10\rangle + |11\rangle)(|0\rangle - |1\rangle)$

Output state of query?

$((-1)^{f(00)}|00\rangle + (-1)^{f(01)}|01\rangle + (-1)^{f(10)}|10\rangle + (-1)^{f(11)}|11\rangle)(|0\rangle - |1\rangle)$

Quantum algorithm (II)



← Apply the U that maps
← $|\psi_{00}\rangle, |\psi_{01}\rangle, |\psi_{10}\rangle, |\psi_{11}\rangle$ to
← $|00\rangle, |01\rangle, |10\rangle, |11\rangle$ (resp.)

Output state of the first two qubits in the four cases:

Case of f_{00}? $|\psi_{00}\rangle = -|00\rangle + |01\rangle + |10\rangle + |11\rangle$

Case of f_{01}? $|\psi_{01}\rangle = +|00\rangle - |01\rangle + |10\rangle + |11\rangle$

Case of f_{10}? $|\psi_{10}\rangle = +|00\rangle + |01\rangle - |10\rangle + |11\rangle$

Case of f_{11}? $|\psi_{11}\rangle = +|00\rangle + |01\rangle + |10\rangle - |11\rangle$

What noteworthy property do these states have? **Orthogonal!**

Challenge Exercise: simulate the above U in terms of H, Toffoli, and NOT gates

one-out-of-N search?

Natural question: what about search problems in spaces larger than ***four*** (and without uniqueness conditions)?

For spaces of size ***eight*** (say), the previous method breaks down—the state vectors will not be orthogonal

Later on, we'll see how to search a space of size N with $O(\sqrt{N})$ queries ...

Constant vs. balanced

Constant vs. balanced

Let $f : \{0,1\}^n \rightarrow \{0,1\}$ be either constant or balanced, where

- ***constant*** means $f(x) = 0$ for all x, or $f(x) = 1$ for all x
- ***balanced*** means $\Sigma_x \; f(x) = 2^{n-1}$

Goal: determine whether f is constant or balanced

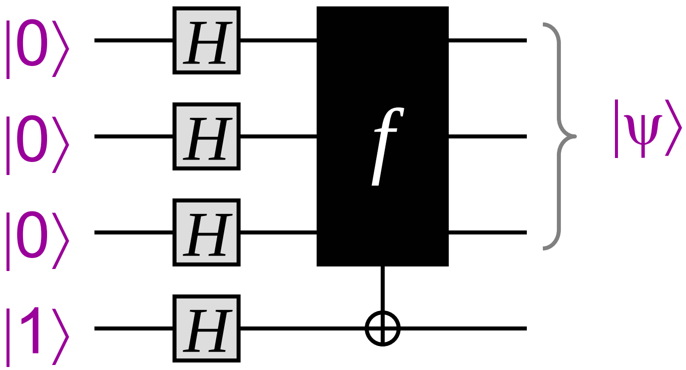
How many queries are there needed ***classically?*** ____

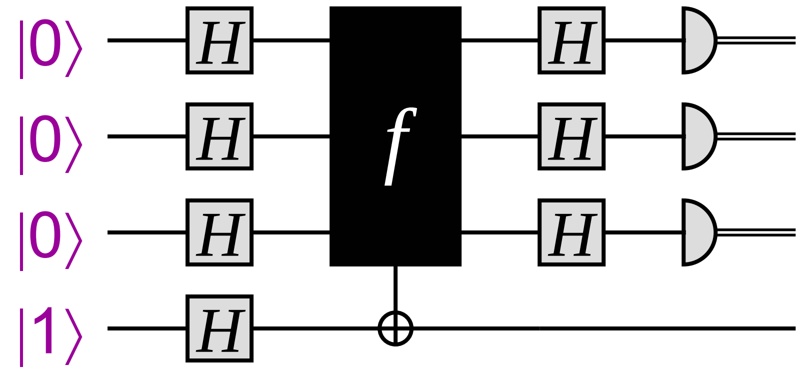
Example: if $f(0000) = f(0001) = f(0010) = \ldots = f(0111) = 0$ then it still could be either

Quantumly? ____

[Deutsch & Jozsa, 1992]

Quantum algorithm





Constant case: $|\psi\rangle = \pm \sum_x |x\rangle$ ***Why?***

Balanced case: $|\psi\rangle$ is ***orthogonal*** to $\pm \sum_x |x\rangle$ ***Why?***

How to distinguish between the cases? What is $H^{\otimes n}|\psi\rangle$?

Constant case: $H^{\otimes n}|\psi\rangle = \pm\, |00\ldots0\rangle$

Balanced case: $H^{\otimes n}\, |\psi\rangle$ is orthogonal to $|0\ldots00\rangle$

Last step of the algorithm: if the measured result is 000 then output “constant”, otherwise output “balanced”

Probabilistic *classical* algorithm solving constant vs balanced

But here's a classical procedure that makes only **2** queries and performs fairly well probabilistically:

1. pick $x_1, x_2 \in \{0,1\}^n$ randomly
2. **if** $f(x_1) \neq f(x_2)$ **then** output balanced **else** output constant

What happens if f is constant? The algorithm always succeeds

What happens if f is balanced? Succeeds with probability ½

By repeating the above procedure k times:
$2k$ queries and one-sided error probability $(½)^k$

Therefore, for large n, $<< 2^n$ queries are likely sufficient

Introduction to Quantum Information Processing

CS 467 / CS 667
Phys 667 / Phys 767
C&O 481 / C&O 681

Lecture 6 (2008)

Richard Cleve
DC 2117
cleve@cs.uwaterloo.ca

$$H \otimes H \otimes \ldots \otimes H$$

About $H\otimes H\otimes \dots \otimes H = H^{\otimes n}$

Theorem: for $x \in \{0,1\}^n$, $H^{\otimes n}|x\rangle = \frac{1}{2^{n/2}} \sum_{y\in\{0,1\}^n} (-1)^{x\cdot y}|y\rangle$

where $x\cdot y = x_1y_1 \oplus \dots \oplus x_ny_n$

Example: $H\otimes H = \frac{1}{2}\begin{bmatrix} +1 & +1 & +1 & +1 \\ +1 & -1 & +1 & -1 \\ +1 & +1 & -1 & -1 \\ +1 & -1 & -1 & +1 \end{bmatrix}$

Pf: For all $x \in \{0,1\}^n$, $H|x\rangle = |0\rangle + (-1)^x|1\rangle = \Sigma_y (-1)^{xy}|y\rangle$

Thus, $H^{\otimes n}|x_1 \dots x_n\rangle = \left(\Sigma_{y_1} (-1)^{x_1y_1}|y_1\rangle\right) \dots \left(\Sigma_{y_n} (-1)^{x_ny_n}|y_n\rangle\right)$

$= \Sigma_y (-1)^{x_1y_1 \oplus \dots \oplus x_ny_n}|y_1 \dots y_n\rangle$ ∎

Simon’s problem

Quantum vs. classical separations

black-box problem	quantum	classical	
constant vs. balanced	**1** (query)	**2** (queries)	
1-out-of-4 search	**1**	**3**	
constant vs. balanced	**1**	**½ 2^n + 1**	**(only for exact)**
Simon's problem			**(probabilistic)**

Simon's problem

Let $f : \{\mathbf{0,1}\}^n \rightarrow \{\mathbf{0,1}\}^n$ have the property that there exists an $r \in \{\mathbf{0,1}\}^n$ such that $f(x) = f(y)$ iff $x \oplus y = r$ or $x = y$

Example:

x	$f(x)$
000	**011**
001	**101**
010	**000**
011	**010**
100	**101**
101	**011**
110	**010**
111	**000**

What is r is this case? ________

Answer: r = **101**

A classical algorithm for Simon

Search for a ***collision***, an $x \neq y$ such that $f(x) = f(y)$

1. Choose $x_1, x_2, \ldots, x_k \in \{0,1\}^n$ randomly (independently)
2. For all $i \neq j$, if $f(x_i) = f(x_j)$ then output $x_i \oplus x_j$ and halt

A hard case is where r is chosen randomly from $\{\mathbf{0,1}\}^n - \{\mathbf{0}^n\}$ and then the "table" for f is filled out randomly subject to the structure implied by r

How big does k have to be for the probability of a collision to be a constant, such as ¾?

Answer: order $2^{n/2}$ (each $(x_i\,, x_j)$ collides with prob. $O(2^{-n})$)

Classical lower bound

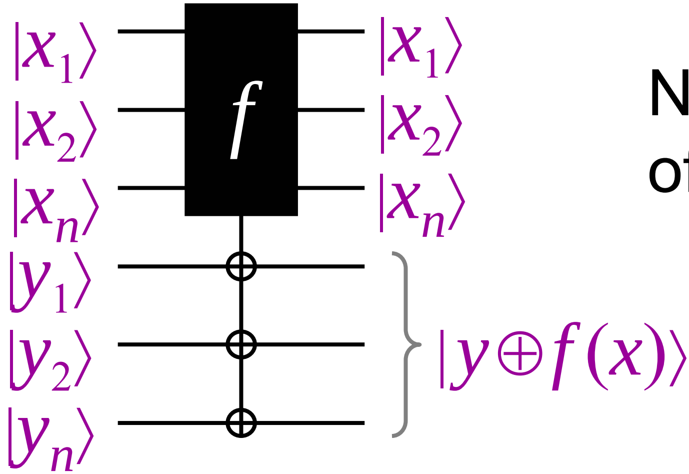
Theorem: ***any*** classical algorithm solving Simon's problem must make $\Omega(2^{n/2})$ queries

Proof is omitted here—note that the performance analysis of the previous algorithm does ***not*** imply the theorem

… how can we know that there isn't a ***different*** algorithm that performs better?

A *quantum* algorithm for Simon I

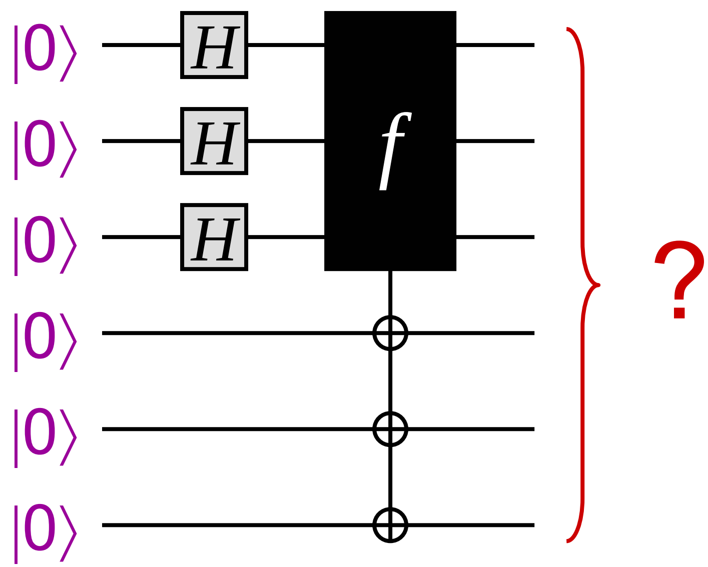
Queries:



Not clear what ***eigenvector*** of target registers is ...

Proposed start of quantum algorithm: query all values of f in superposition

What is the output state of this circuit?



A quantum algorithm for Simon II

Answer: the output state is $\sum_{x\in\{0,1\}^n} |x\rangle|f(x)\rangle$

Let $T \subseteq \{\mathbf{0},\mathbf{1}\}^n$ be such that ***one*** element from each matched pair is in T (assume $r \neq$ **00...0**)

Example: could take $T = \{$**000**, **001**, **011**, **111**$\}$

Then the output state can be written as:

$$\sum_{x\in T} |x\rangle|f(x)\rangle + |x\oplus r\rangle|f(x\oplus r)\rangle$$

$$= \sum_{x\in T} \big(|x\rangle + |x\oplus r\rangle\big)|f(x)\rangle$$

x	$f(x)$
000	**011**
001	**101**
010	**000**
011	**010**
100	**101**
101	**011**
110	**010**
111	**000**

A quantum algorithm for Simon III

Measuring the second register yields $|x\rangle + |x \oplus r\rangle$ in the first register, for a random $x \in T$

How can we use this to obtain ***some*** information about r?

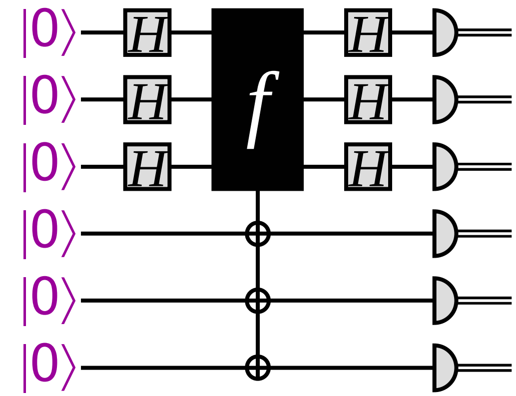
Try applying $H^{\otimes n}$ to the state, yielding:

$$\sum_{y \in \{0,1\}^n} (-1)^{x \bullet y} |y\rangle + \sum_{y \in \{0,1\}^n} (-1)^{(x \oplus r) \bullet y} |y\rangle$$

$$= \sum_{y \in \{0,1\}^n} (-1)^{x \bullet y} \left(1 + (-1)^{r \bullet y}\right) |y\rangle$$

Measuring this state yields y with prob. $\begin{cases} (1/2)^{n-1} & \text{if } r \cdot y = 0 \\ 0 & \text{if } r \cdot y \neq 0 \end{cases}$

A quantum algorithm for Simon IV



Executing this algorithm $k = O(n)$ times yields random $y_1, y_2, \ldots, y_k \in \{0,1\}^n$ such that $r \cdot y_1 = r \cdot y_2 = \ldots = r \cdot y_n = 0$

How does this help?

This is a system of k linear equations:

$$\begin{bmatrix} y_{11} & y_{12} & \cdots & y_{1n} \\ y_{21} & y_{22} & \cdots & y_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ y_{k1} & y_{k2} & \cdots & y_{kn} \end{bmatrix} \begin{bmatrix} r_1 \\ r_2 \\ \vdots \\ r_n \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

With high probability, there is a unique non-zero solution that is r (which can be efficiently found by linear algebra)

Conclusion of Simon's algorithm

- Any classical algorithm has to query the black box $\Omega(2^{n/2})$ times, even to succeed with probability ¾

- There is a quantum algorithm that queries the black box only $O(n)$ times, performs only $O(n^3)$ auxiliary operations (for the Hadamards, measurements, and linear algebra), and succeeds with probability ¾