Introduction to Quantum Information Processing

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

Lecture 3 (2005)

Richard Cleve
DC 3524
cleve@cs.uwaterloo.ca

Course website

Available at:
http://www.cs.uwaterloo.ca/~cleve

Contents

- Recap: states, unitary operations, measurements
- Classical computations as circuits
- Simulating ***classical*** circuits with ***quantum*** circuits
- Simulating ***quantum*** circuits with ***classical*** circuits
- Simple quantum algorithms in the query scenario

- Recap: states, unitary operations, measurements
- Classical computations as circuits
- Simulating ***classical*** circuits with ***quantum*** circuits
- Simulating ***quantum*** circuits with ***classical*** circuits
- Simple quantum algorithms in the query scenario

Recap (I)

- ***n*-qubit quantum state:** 2^n-dimensional unit vector

- **Unitary op:** $2^n \times 2^n$ linear operation U such that $U^\dagger U = I$ (where $U^\dagger$ denotes the conjugate transpose of U)

$U|0000\rangle$ = the 1st column of U

$U|0001\rangle$ = the 2nd column of U

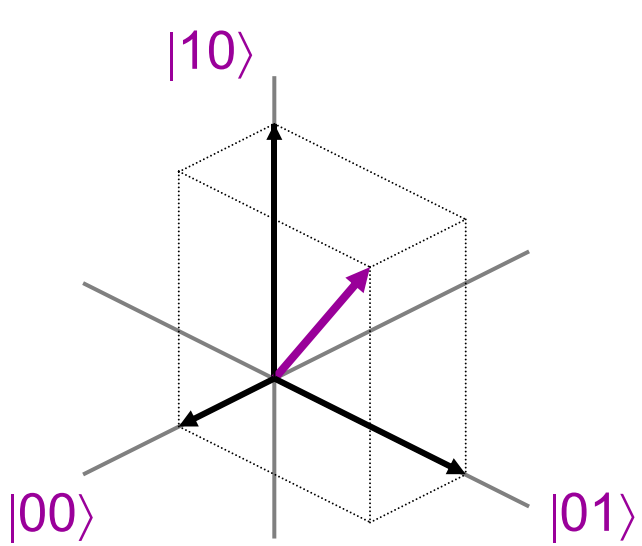
: : : : : :

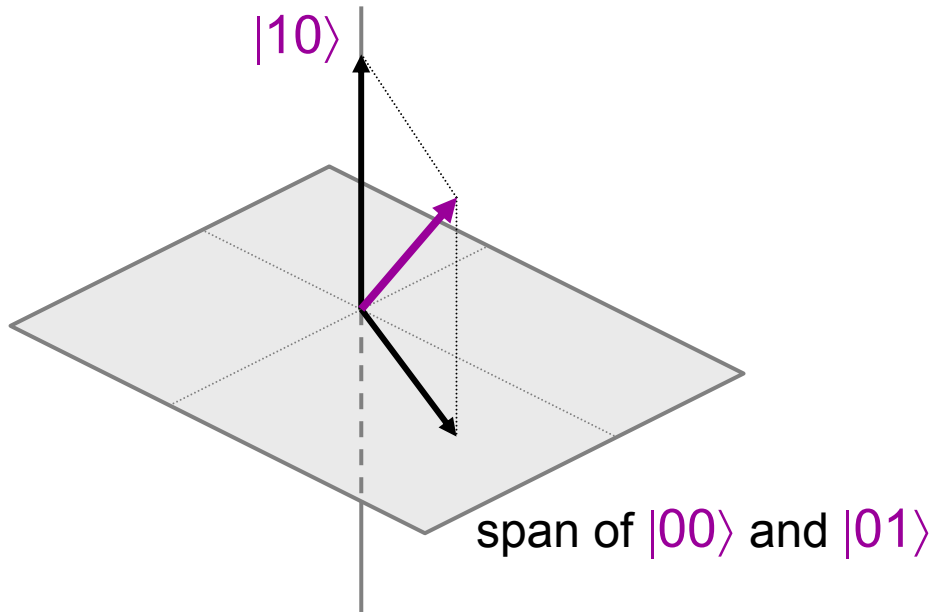
$U|1111\rangle$ = the $(2^n)^{\text{th}}$ column of U

} the columns of U are orthonormal

Recap (II)

- **von Neumann Measurement:** associated with a partition of the space into mutually orthogonal subspaces

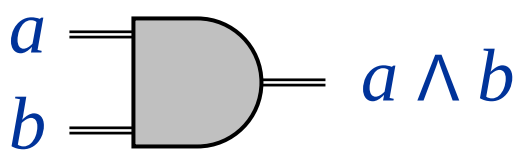
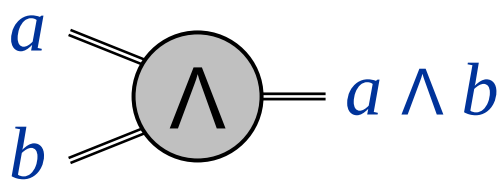
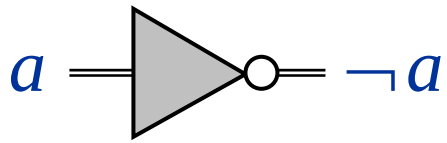
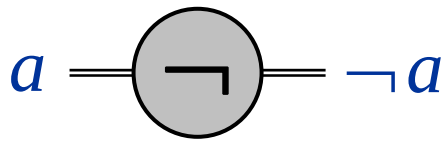




When the measurement is performed, the state collapses to each subspace with probability the square of the length of its projection on that subspace

- Recap: states, unitary operations, measurements
- Classical computations as circuits
- Simulating ***classical*** circuits with ***quantum*** circuits
- Simulating ***quantum*** circuits with ***classical*** circuits
- Simple quantum algorithms in the query scenario

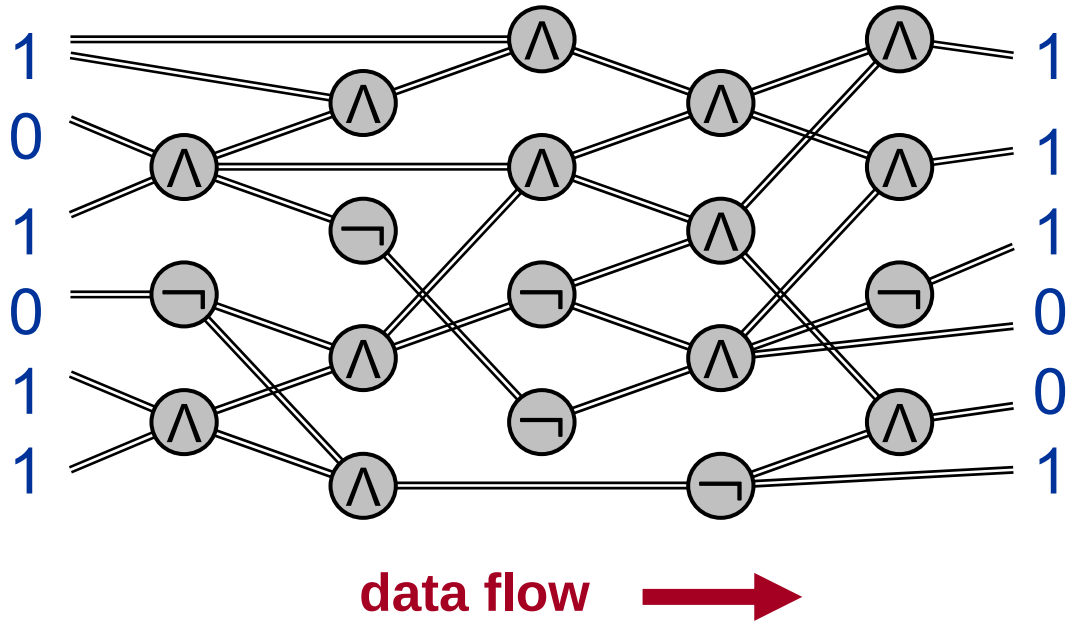
Classical (boolean logic) gates

	"old" notation	"new" notation
AND gate		
NOT gate		

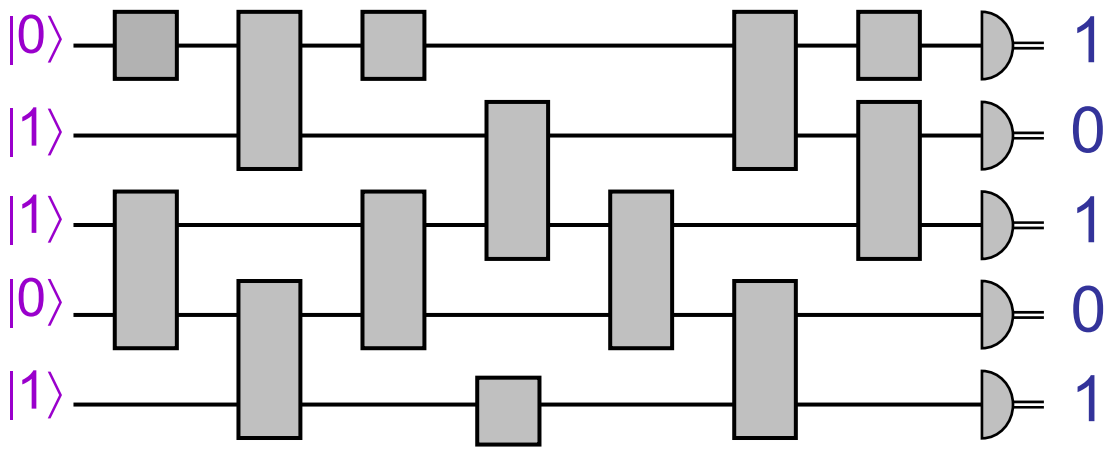
Note: an **OR** gate can be simulated by one **AND** gate and three **NOT** gates (since $a \vee b = \neg(\neg a \wedge \neg b)$)

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 $O(n \log n \, \mathrm{loglog} \, n)$
- Best currently-known ***quantum*** method: same

Factoring problem

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

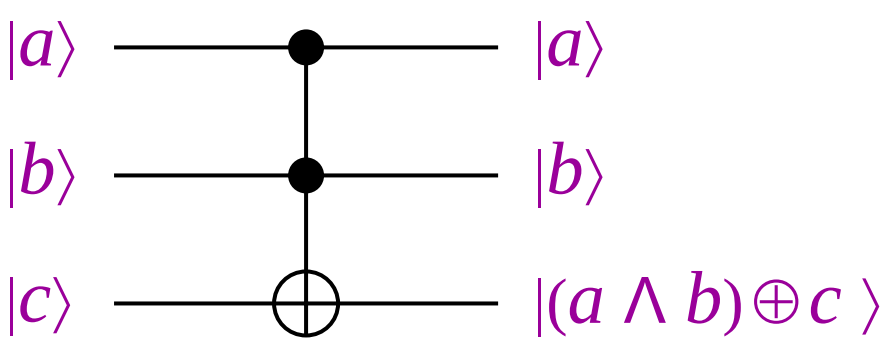
Output: their product (e.g. 101, 111)

- 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

- Recap: states, unitary operations, measurements
- Classical computations as circuits
- Simulating ***classical*** circuits with ***quantum*** circuits
- Simulating ***quantum*** circuits with ***classical*** circuits
- Simple quantum algorithms in the query scenario

Toffoli gate

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



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

Matrix representation:

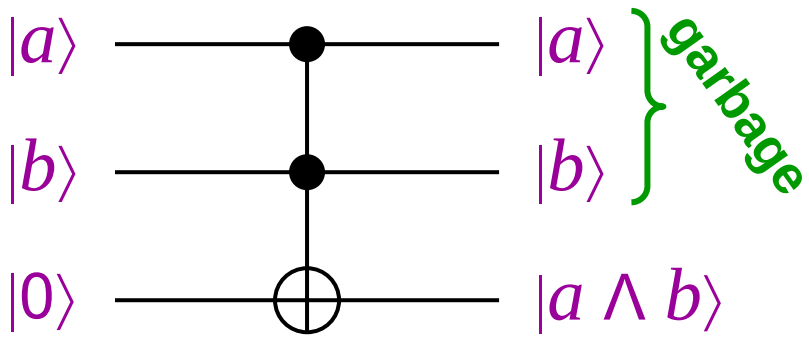
$$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}$$

Quantum simulation of classical

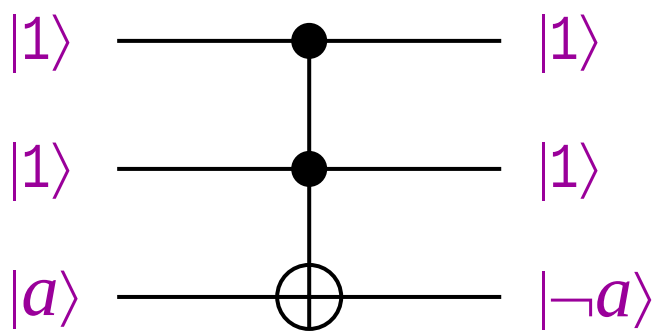
Theorem: a classical circuit of size s can be simulated by a quantum circuit of size $O(s)$

Idea: using Toffoli gates, one can simulate:

AND gates



NOT gates

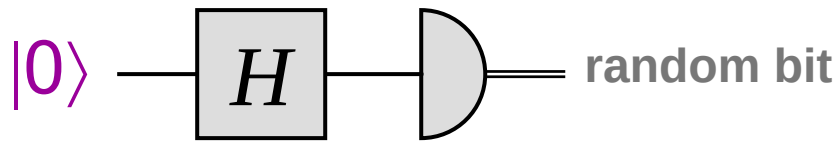


This garbage will have to be reckoned with later on …

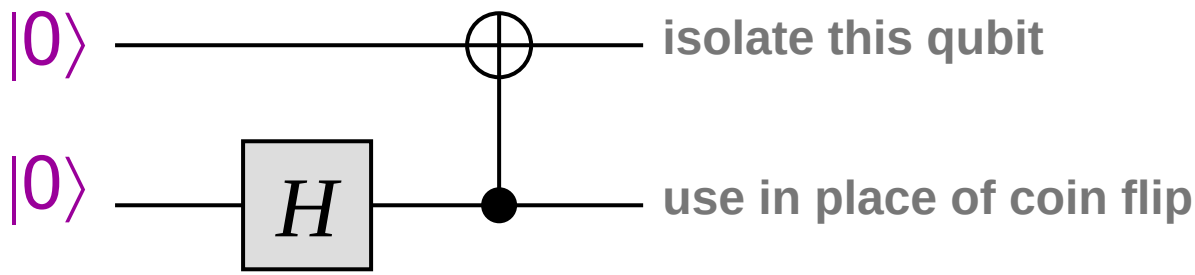
Simulating probabilistic algorithms

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:



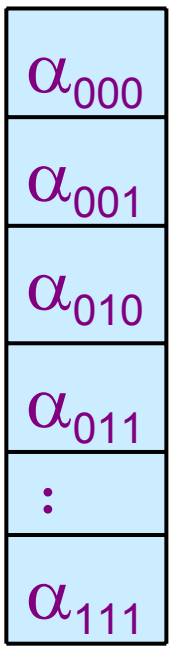
Exercise: prove that this works

- Recap: states, unitary operations, measurements
- Classical computations as circuits
- Simulating ***classical*** circuits with ***quantum*** circuits
- Simulating ***quantum*** circuits with ***classical*** circuits
- Simple quantum algorithms in the query scenario

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}



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

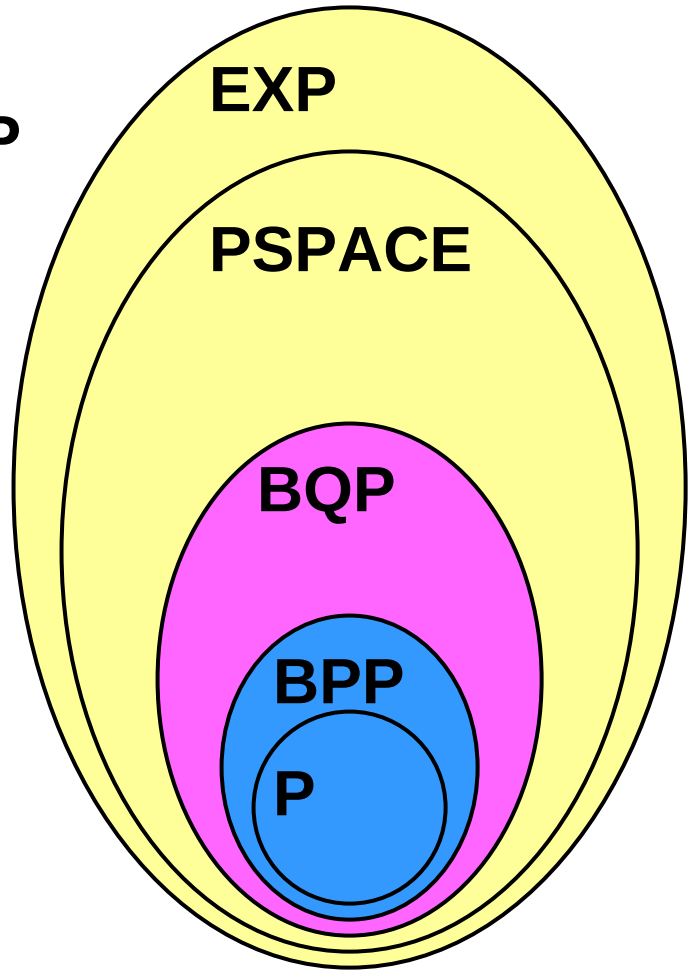
Exercise: show how to do the simulation using only a polynomial amount of ***space*** (memory)

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

$\mathbf{P \subseteq BPP \subseteq BQP \subseteq PSPACE \subseteq EXP}$



This picture will be fleshed out more later on

- Recap: states, unitary operations, measurements
- Classical computations as circuits
- Simulating ***classical*** circuits with ***quantum*** circuits
- Simulating ***quantum*** circuits with ***classical*** circuits
- Simple quantum algorithms in the query scenario

Query scenario

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

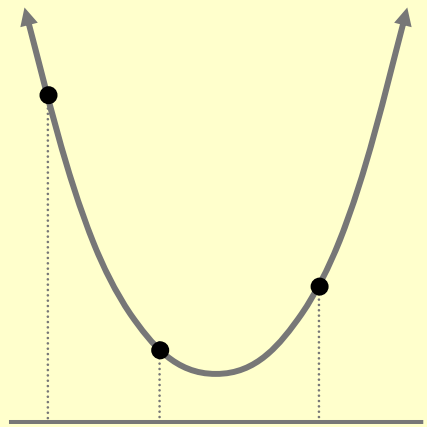
Goal: determine some information about f making as few queries to f (and other operations) as possible

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, … , c_d

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



Answer: $d+1$

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

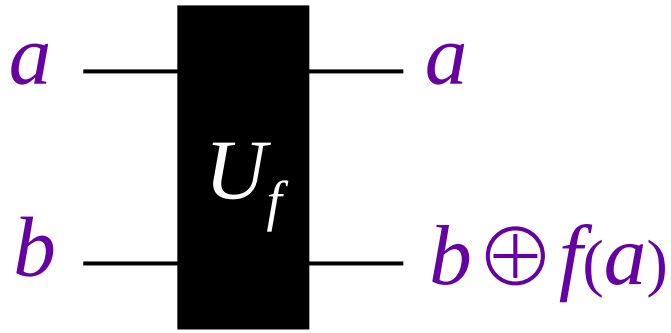
x	$f_4(x)$
0	1
1	0

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

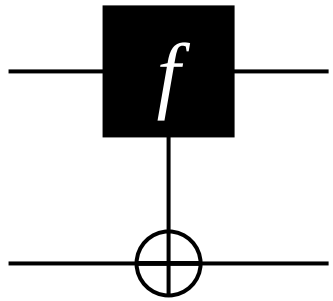
Any classical method requires ***two*** queries

What about a quantum method?

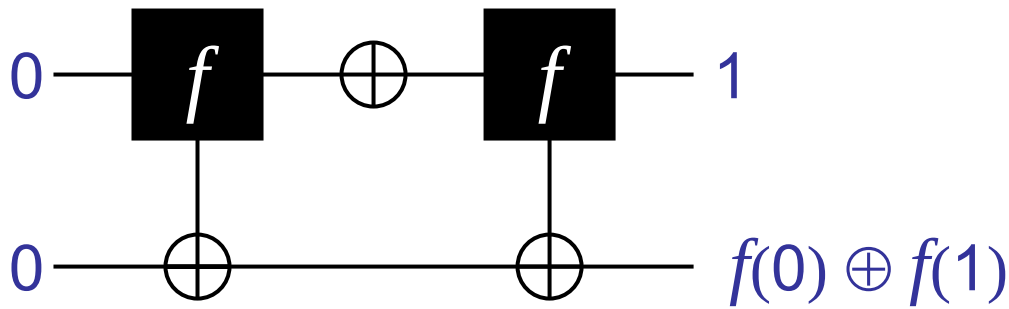
Reversible black box for f



alternate notation:

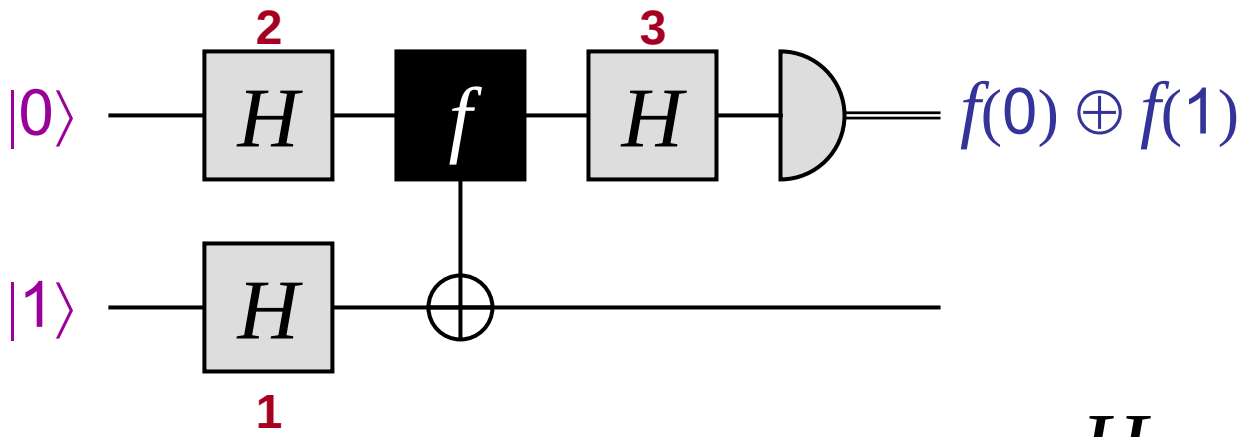


A classical algorithm:
(still requires 2 queries)



2 queries + **1** auxiliary operation

Quantum algorithm for Deutsch



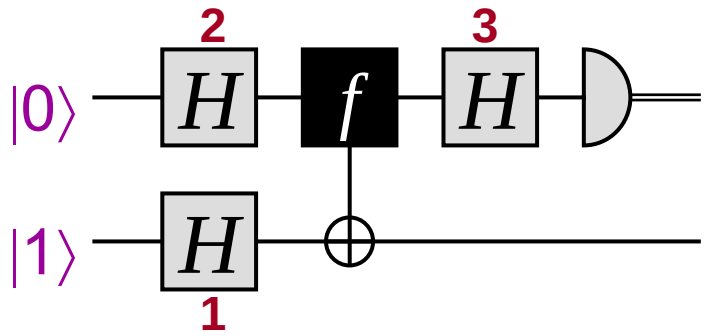
1 query + **4** auxiliary operations

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

How does this algorithm work?

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

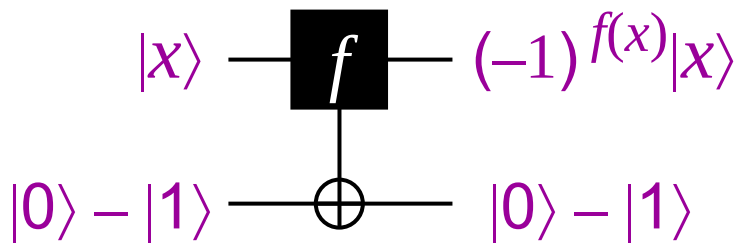
Quantum algorithm (1)



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

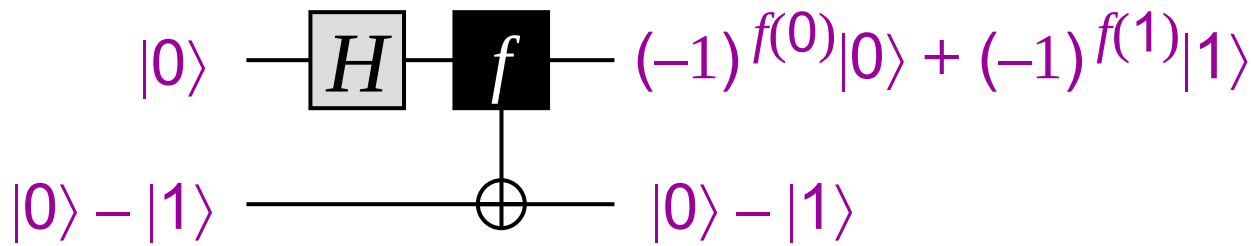
- **NOT** with eigenvalue -1
- I with eigenvalue $+1$

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

$\pm(|0\rangle + |1\rangle)$

x	$f_3(x)$
0	0
1	1

x	$f_4(x)$
0	1
1	0

$\pm(|0\rangle - |1\rangle)$

Quantum algorithm (3)

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

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

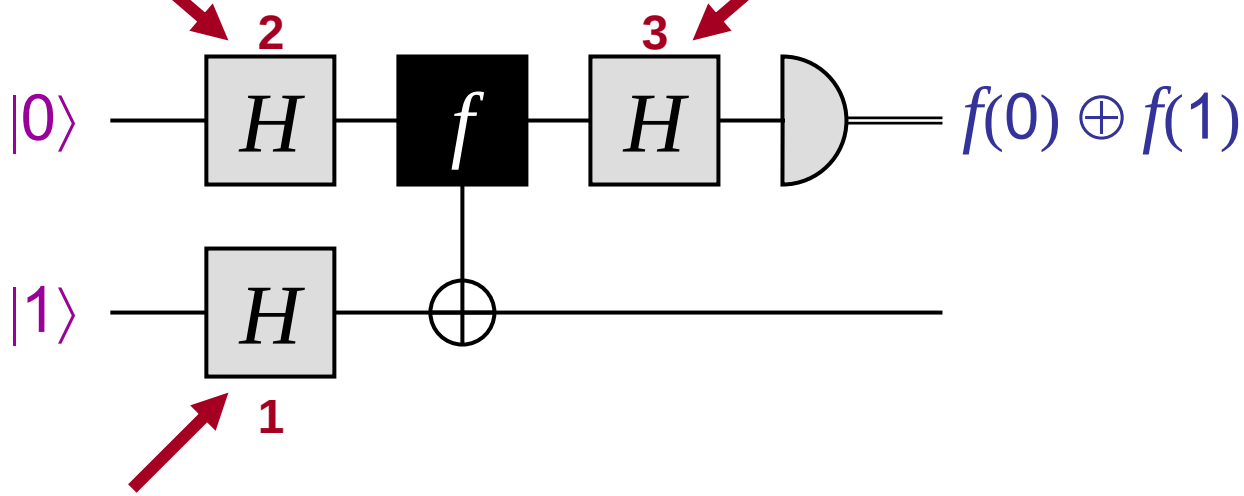
$$\pm(|0\rangle - |1\rangle) \xleftrightarrow{H} \pm|1\rangle$$

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: $|x\rangle \rightarrow (-1)^{f(x)}|x\rangle$


THE END