Quantum Circuits and Algorithms

- Modular Arithmetic, XOR
- Reversible Computation revisited
- Quantum Gates revisited
- A taste of quantum algorithms: Deutsch algorithm
- Other algorithms, general overviews
- Measurements revisited

Sources:

John P. Hayes, Mike Frank Michele Mosca, Artur Ekert, Bulitko, Rezania. Dave Bacon, 156 Jorgensen, dabacon@cs.caltech.edu, Stephen Bartlett

Some review, some new.

- This lecture reviews some of the most important facts from the class,
- Possibly in a new light
- To help you understand not only the top quantum algorithms
- But also a philosophy and methodology of creating quantum algorithms.

Outline

- Review and new ideas useful for quantum algorithms
- Introduction to quantum algorithms
  - Define algorithms and computational complexity
  - Discuss factorization as an important algorithm for information security
- Quantum algorithms
  - What they contribute to computing and cryptography
  - Deutsch algorithm and Deutsch-Jozsa algorithm
  - Shor’s quantum algorithm for efficient factorization
  - Quantum search algorithms
  - Demonstrations of quantum algorithms
  - Ongoing quantum algorithms research

Review of quantum formalism, circuits and new ideas useful in quantum algorithms

Universal Quantum gates

- Ideally, we'd like a set of gates that allows us to generate all unitary operations on n qubits
- The controlled-NOT plus all 1-qubit gates is universal in this sense
- However, this set of gates in infinite, and therefore not "reasonable"
- We are happy with finite sets of gates that allow us to approximate any unitary operation on n qubits (more in Chapter 4 of *Nielsen and Chuang*)

Universal Q-Gates: History

- Deutsch '89:
  - Universal 3-qubit Toffoli-like gate.
- diVincenzo '95:
  - Adequate set of 2-qubit gates.
- Barenco '95:
  - Universal 2-qubit gate.
- Deutsch *et al.* '95
  - Almost all 2-qubit gates are universal.
- Barenco *et al.* '95
  - CNOT + set of 1-qubit gates is adequate.
- **Later development of discrete gate sets...**

Deutsch: Generalized 3-bit Toffoli gate:

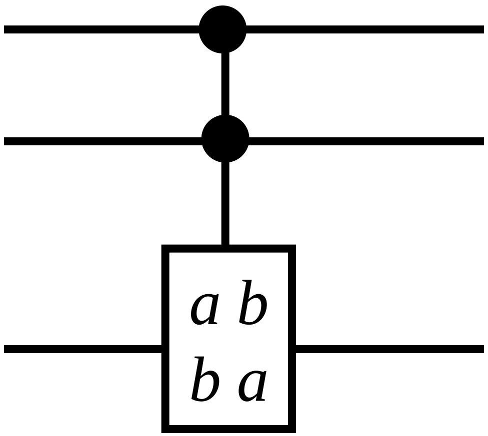
- The following gate is universal:

$$\begin{bmatrix} 1 & & & & & & & \\ & 1 & & & & & & \\ & & 1 & & & & & \\ & & & 1 & & & & \\ & & & & 1 & & & \\ & & & & & 1 & & \\ & & & & & & a & b \\ & & & & & & b & a \end{bmatrix}$$

$$a = ie^{i\pi\alpha/2}(1+e^{i\pi\alpha})/2$$

$$b = ie^{i\pi\alpha/2}(1-e^{i\pi\alpha})/2$$

(Where α is any irrational number.)

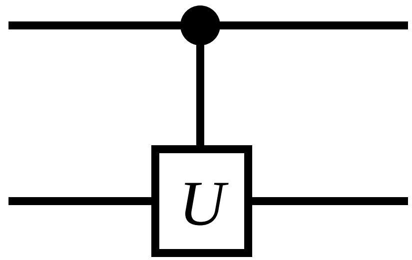


Barenco's 2-bit generalized CNOT gate

$$A(\phi,\alpha,\theta)=\begin{bmatrix}1 & & & \\ & 1 & & \\ & & e^{i\alpha}\cos\theta & -ie^{i(\alpha-\phi)}\sin\theta \\ & & -ie^{i(\alpha+\phi)}\sin\theta & e^{i\alpha}\cos\theta\end{bmatrix}$$

- where ϕ,α,θ,π are relatively irrational
- Also works, *e.g.*, for $\phi=\pi$, $\alpha=\pi/2$:

$$A(\pi,\tfrac{\pi}{2},\theta)=\begin{bmatrix}1 & & & \\ & 1 & & \\ & & i\cos\theta & -\sin\theta \\ & & -\sin\theta & i\cos\theta\end{bmatrix}$$



Barenco *et al.* '95 results

- **Universality** of CNOT + 1-qubit gates
  - 2-qubit Barenco gate already known universal
  - 4 **1-qubit gates** + **2 CNOTs** suffice to build it
- **Construction of generalized Toffoli gates**
  - 3-bit version via ***five*** 2-qubit gates
  - n-qubit version via $O(n^2)$ 2-qubit gates
  - No auxilliary qubits needed for the above
    - All operations done "in place" on input qubits.
  - n-bit version via $O(n)$ 2-qubit gates, given 1 work qubit

Modular arithmetic

- For any positive integer N, we say a is congruent to b modulo N (denoted $a \equiv b \bmod N$) if and only if N divides a-b
- E.g.

$$\ldots, -10, -5, 0, 5, 10, 15 \ldots \equiv 0 \bmod 5$$
$$\ldots -14, -9, -4, 1, 6, 11, 16 \ldots \equiv 1 \bmod 5$$
$$\ldots -13, -8, -3, 2, 7, 12, 17 \ldots \equiv 2 \bmod 5$$
$$\ldots -12, -7, -2, 3, 8, 13, 18 \ldots \equiv 3 \bmod 5$$
$$\ldots -11, -6, -1, 4, 9, 14, 19 \ldots \equiv 4 \bmod 5$$

Modular arithmetic

- For any positive integer N, and for any integer a, define $a \bmod N$ to be the unique integer, $\bar{a}$, between 0 and N-1 such that $a \equiv \bar{a} \bmod N$
- For positive integers, a, we can say that $\bar{a}$ is the remainder when we divide a by N.
- If N=2, then $a \bmod 2 = 0$ if a is even

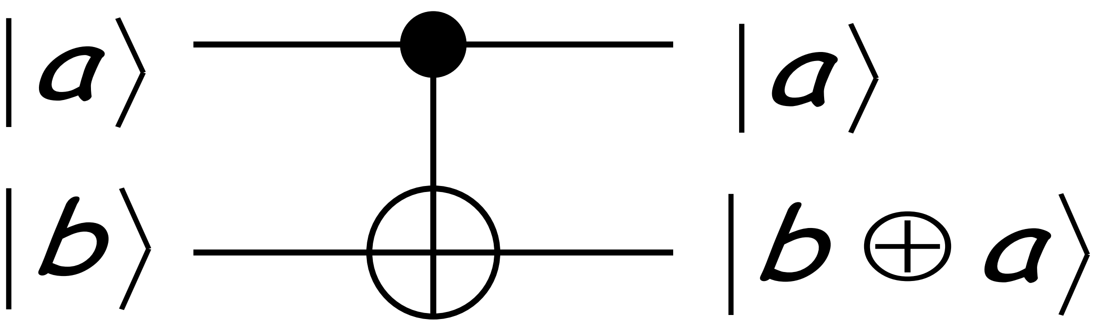
  $a \bmod 2 = 1$ if a is odd

Modulo versus XOR

- For $a, b \in \{0,1\}$

$$a \oplus b = (a+b) \bmod 2$$

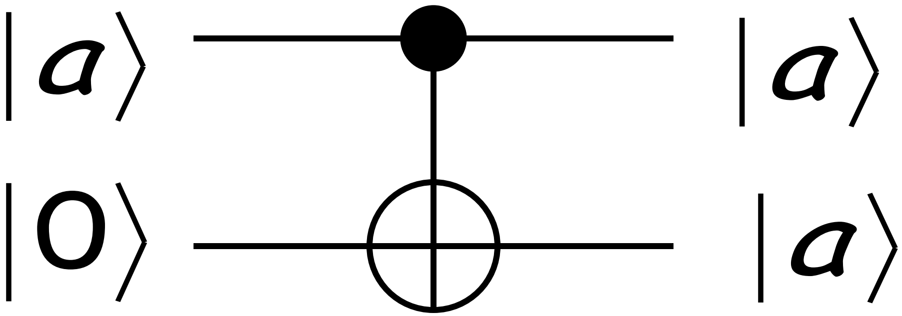
- The controlled-NOT also realizes the reversible XOR function



reminder

Controlled-NOT can be used to copy classical information

- If we initialize b=0, then the C-NOT can be used to copy "classical" information



- We can use this operation in the copy part of reversible computation

Reversibly computing f(x)

- Suppose we know how to compute

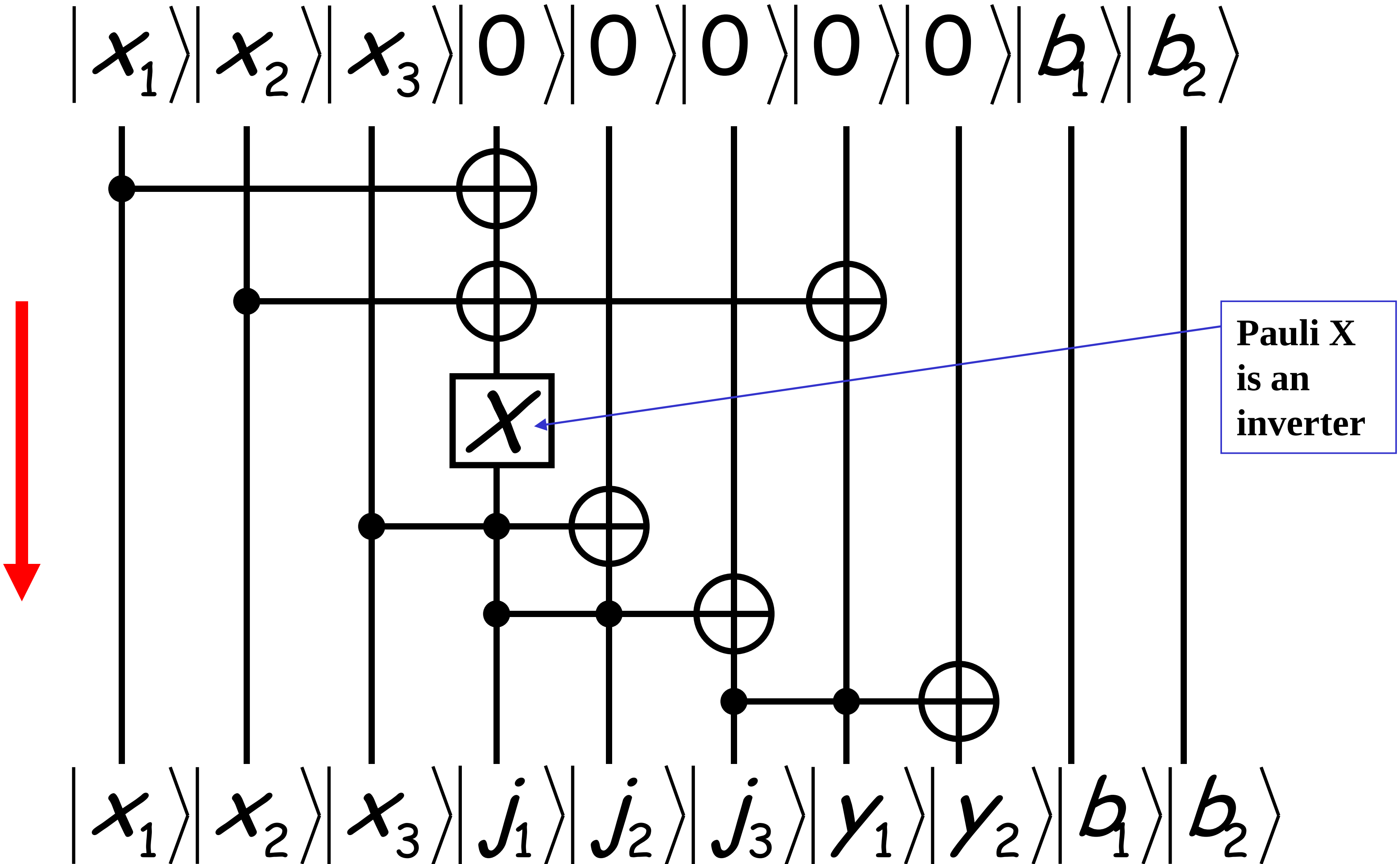
$$f:\{0,1\}^n \rightarrow \{0,1\}^m$$

- We can realize the following reversible implementation of f

$$|x\rangle|b\rangle \rightarrow |x\rangle|b \oplus f(x)\rangle$$

Reversibly computing $f(x)=y_1y_2$

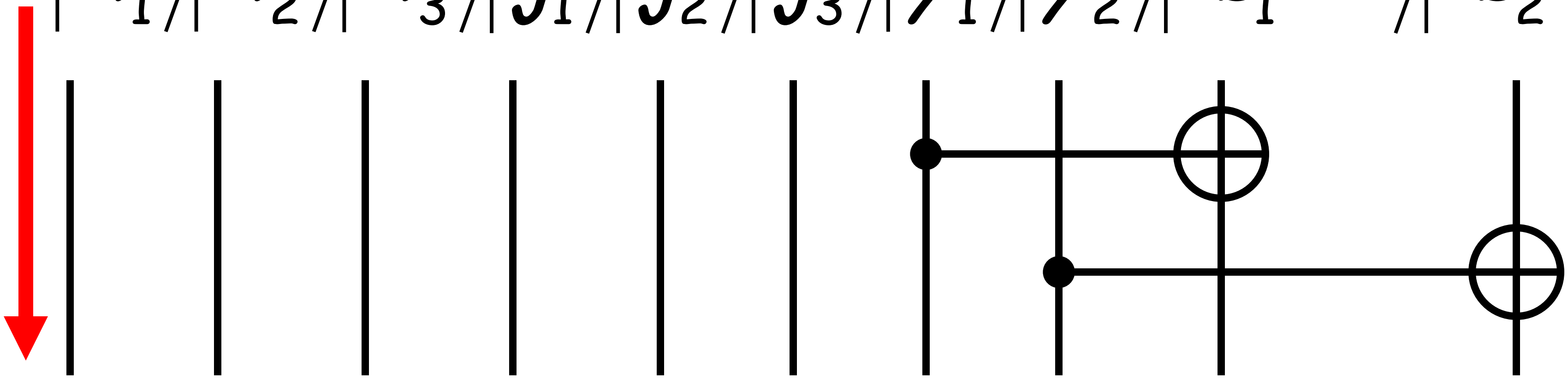
Step 1: Compute f(x)



Reversibly computing $f(x)=y_1y_2$

Step 2: Add answer to output register

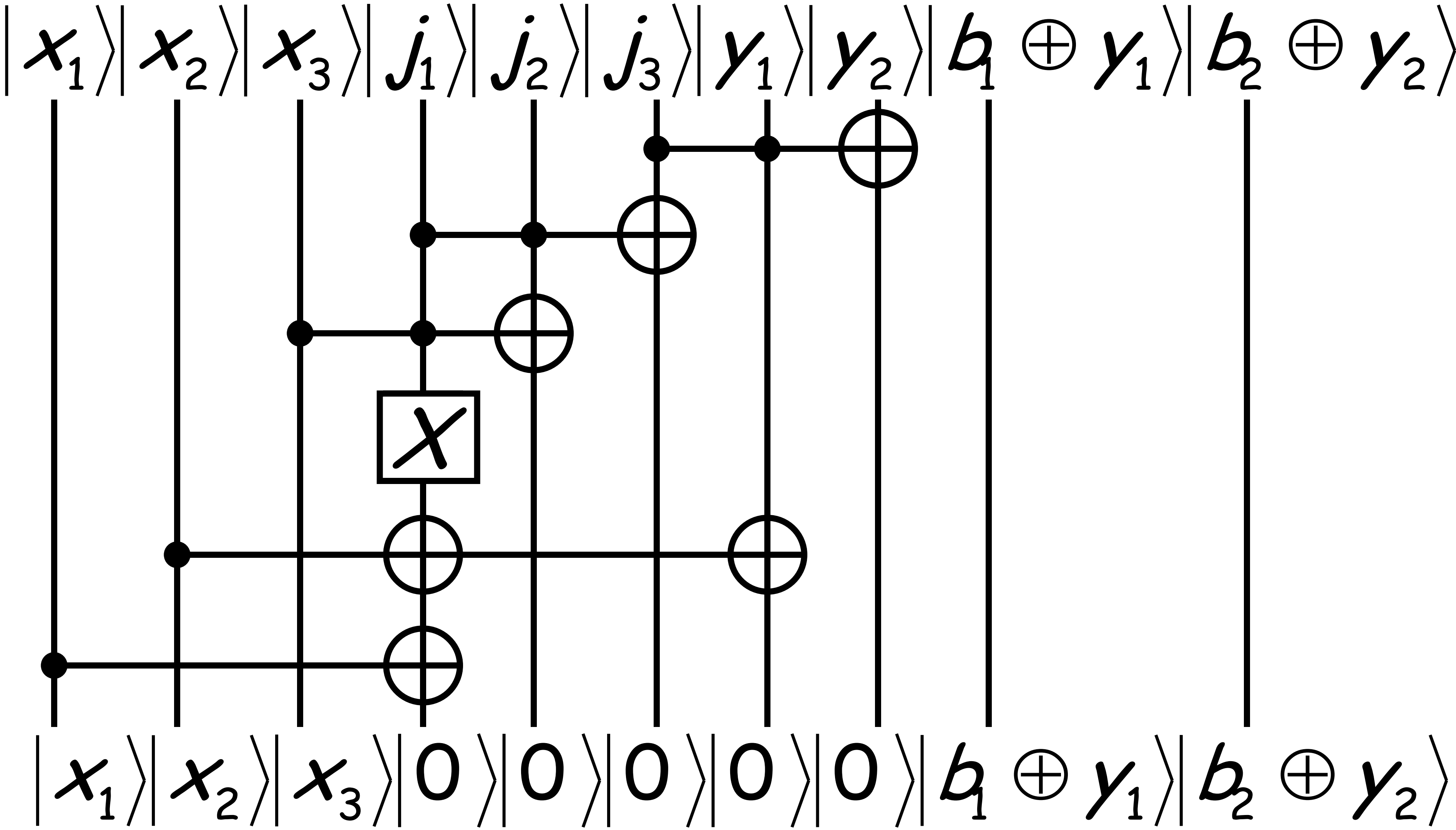
$$|x_1\rangle|x_2\rangle|x_3\rangle|j_1\rangle|j_2\rangle|j_3\rangle|y_1\rangle|y_2\rangle|b_1\rangle|b_2\rangle$$



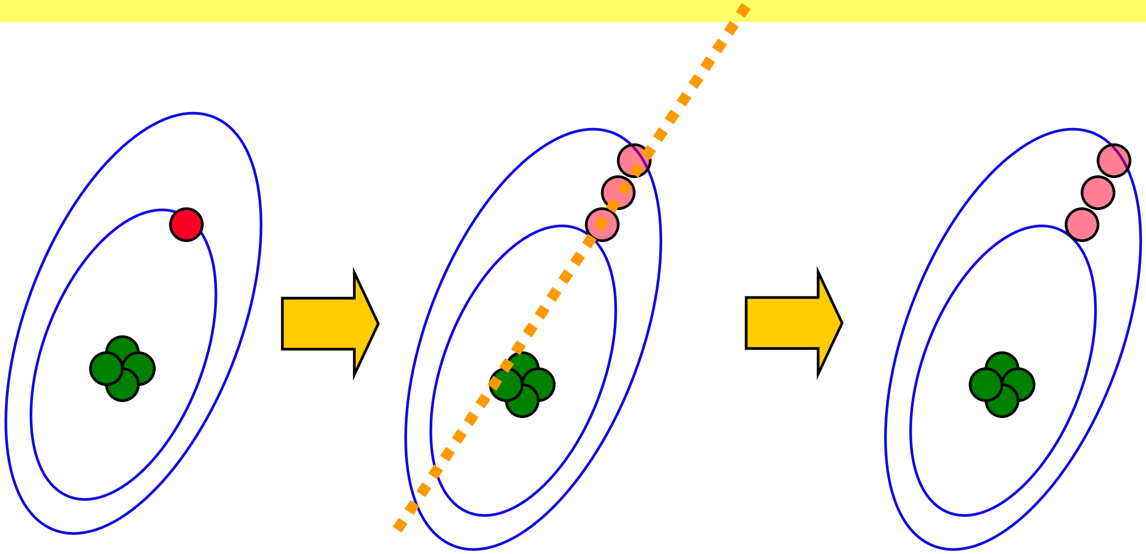
$$|x_1\rangle|x_2\rangle|x_3\rangle|j_1\rangle|j_2\rangle|j_3\rangle|y_1\rangle|y_2\rangle|b_1\oplus y_1\rangle|b_2\oplus y_2\rangle$$

Reversibly computing $f(x)=y_1y_2$

Step 3: Uncompute f(x)



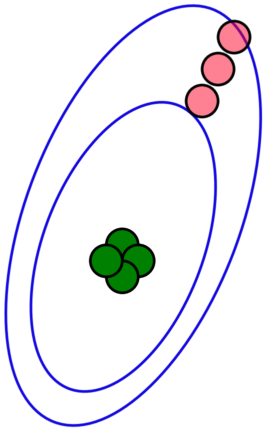
A quantum gate



$$|0\rangle \;\text{---}\; \boxed{\sqrt{\text{NOT}}} \;\text{---}\; \frac{i}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle$$

$$|1\rangle \;\text{---}\; \boxed{\sqrt{\text{NOT}}} \;\text{---}\; \frac{1}{\sqrt{2}}|0\rangle + \frac{i}{\sqrt{2}}|1\rangle$$

???



What is $\frac{i}{\sqrt{2}}\left|0\right\rangle + \frac{1}{\sqrt{2}}\left|1\right\rangle$ supposed to mean?

One thing we know about it

If we measure $\alpha_0 |0\rangle + \alpha_1 |1\rangle$

we get $|0\rangle$ with probability $|\alpha_0|^2$

and $|1\rangle$ with probability $|\alpha_1|^2$

Please recall the notation!

$|0\rangle$ corresponds to $\begin{pmatrix} 1 \\ 0 \end{pmatrix}$

$|1\rangle$ corresponds to $\begin{pmatrix} 0 \\ 1 \end{pmatrix}$

$\alpha_0|0\rangle + \alpha_1|1\rangle$ corresponds to $\alpha_0\begin{pmatrix} 1 \\ 0 \end{pmatrix} + \alpha_1\begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} \alpha_0 \\ \alpha_1 \end{pmatrix}$

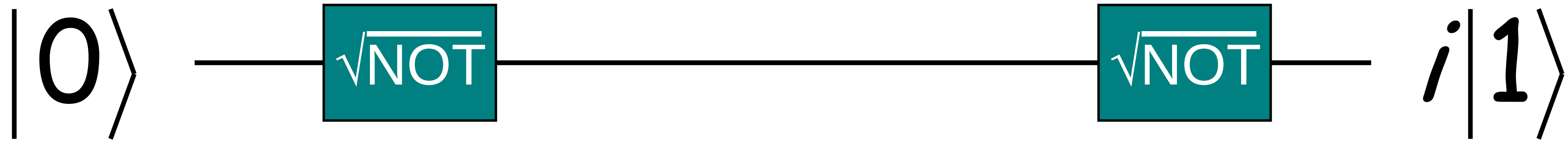
Two very important 1-qubit gates

corresponds to $\begin{pmatrix} \frac{i}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & \frac{i}{\sqrt{2}} \end{pmatrix}$

Another useful gate:
(Hadamard gate)

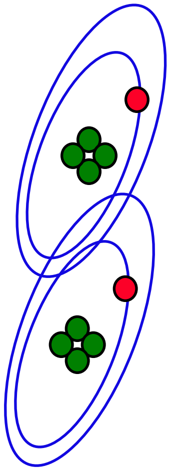
$$H = \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & \frac{-1}{\sqrt{2}} \end{pmatrix}$$

Unexpected result again!



$$\begin{pmatrix} 0 \\ i \end{pmatrix} = \begin{pmatrix} \frac{i}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & \frac{i}{\sqrt{2}} \end{pmatrix} \begin{pmatrix} \frac{i}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & \frac{i}{\sqrt{2}} \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix}$$

Tensor Product again!



$$= \begin{matrix}|0\rangle \\ |0\rangle\end{matrix} = |00\rangle = \begin{pmatrix}1\\0\\0\\0\end{pmatrix}$$

$$\begin{pmatrix}1\\0\\0\\0\end{pmatrix} = |00\rangle = |0\rangle|0\rangle = \begin{pmatrix}1\\0\end{pmatrix} \otimes \begin{pmatrix}1\\0\end{pmatrix} = |0\rangle \otimes |0\rangle$$

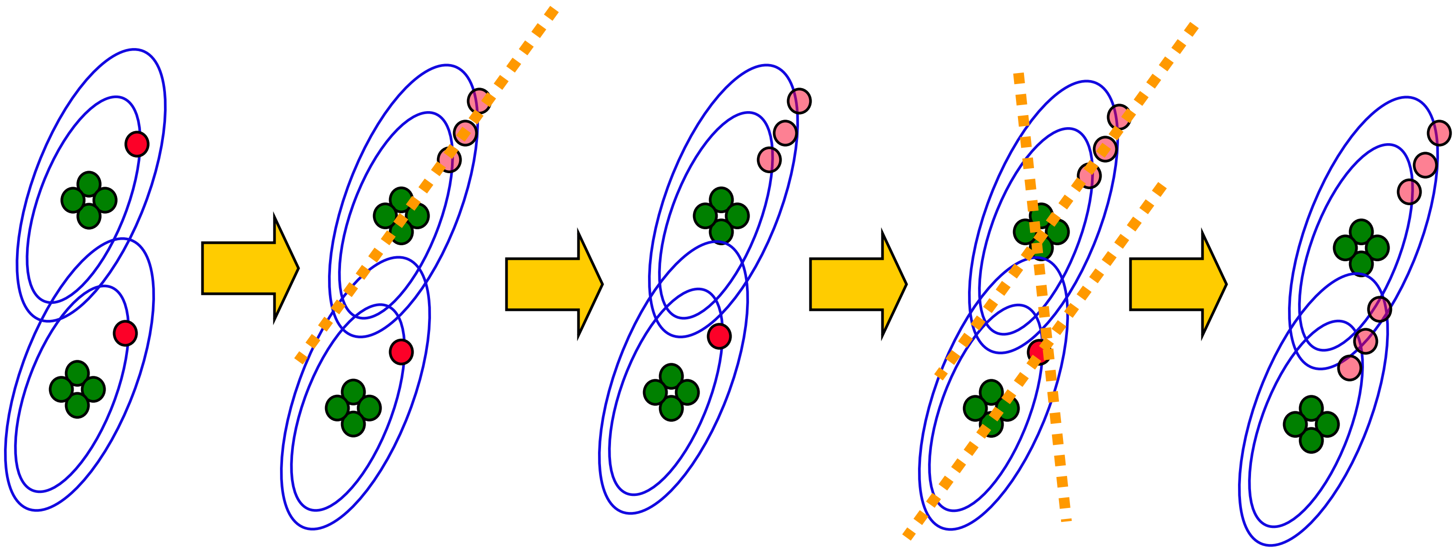
Local versus Global description of a 2-qubit state

$$\left(\frac{i}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle\right)\otimes|0\rangle$$

$$= \left(\frac{i}{\sqrt{2}}|0\rangle \otimes |0\rangle + \frac{1}{\sqrt{2}}|1\rangle \otimes |0\rangle\right)$$

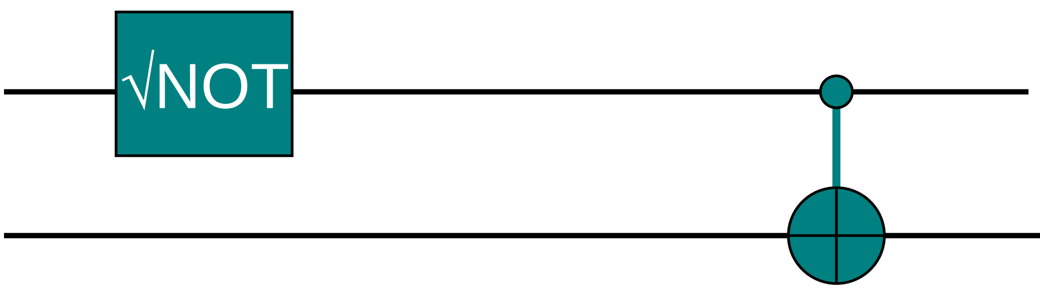
$$= \left(\frac{i}{\sqrt{2}}|00\rangle + \frac{1}{\sqrt{2}}|10\rangle\right)$$

A quantum computation: Entanglement



$|0\rangle$

$|0\rangle$



$$\frac{i}{\sqrt{2}}|0\rangle|0\rangle + \frac{1}{\sqrt{2}}|1\rangle|1\rangle$$

$$|00\rangle \xrightarrow{\sqrt{NOT}\otimes I} \frac{i}{\sqrt{2}}|00\rangle + \frac{1}{\sqrt{2}}|10\rangle \xrightarrow{c-NOT} \frac{i}{\sqrt{2}}|00\rangle + \frac{1}{\sqrt{2}}|11\rangle$$

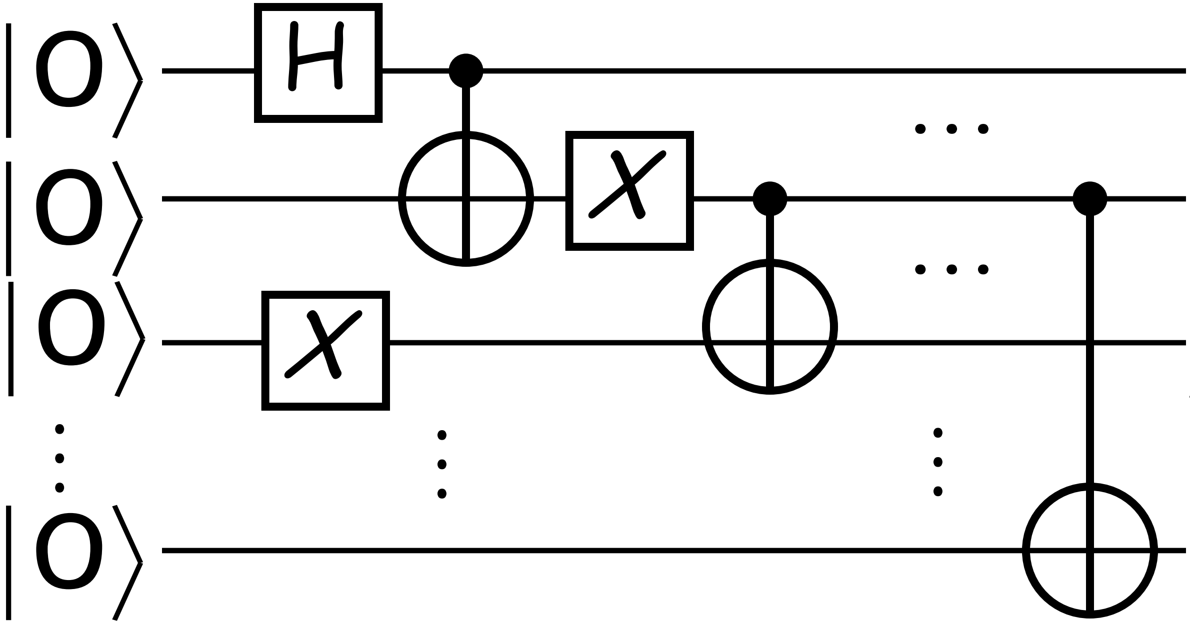
A quantum computation: Entanglement

$$|00\rangle \xrightarrow{\sqrt{NOT}\otimes I} \frac{i}{\sqrt{2}}|00\rangle + \frac{1}{\sqrt{2}}|10\rangle \xrightarrow{c-NOT} \frac{i}{\sqrt{2}}|00\rangle + \frac{1}{\sqrt{2}}|11\rangle$$

$$\begin{pmatrix}1\\0\\0\\0\end{pmatrix} \xrightarrow{\frac{1}{\sqrt{2}}\begin{bmatrix}i & 1\\1 & i\end{bmatrix}\otimes I} \frac{1}{\sqrt{2}}\begin{pmatrix}i\\0\\1\\0\end{pmatrix} \xrightarrow{\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&0&1\\0&0&1&0\end{bmatrix}} \frac{1}{\sqrt{2}}\begin{pmatrix}i\\0\\0\\1\end{pmatrix}$$

$$\frac{1}{\sqrt{2}}\begin{bmatrix}i & 1\\1 & i\end{bmatrix}\otimes I = \frac{1}{\sqrt{2}}\begin{bmatrix}i&0&1&0\\0&i&0&1\\1&0&i&0\\0&1&0&i\end{bmatrix}$$

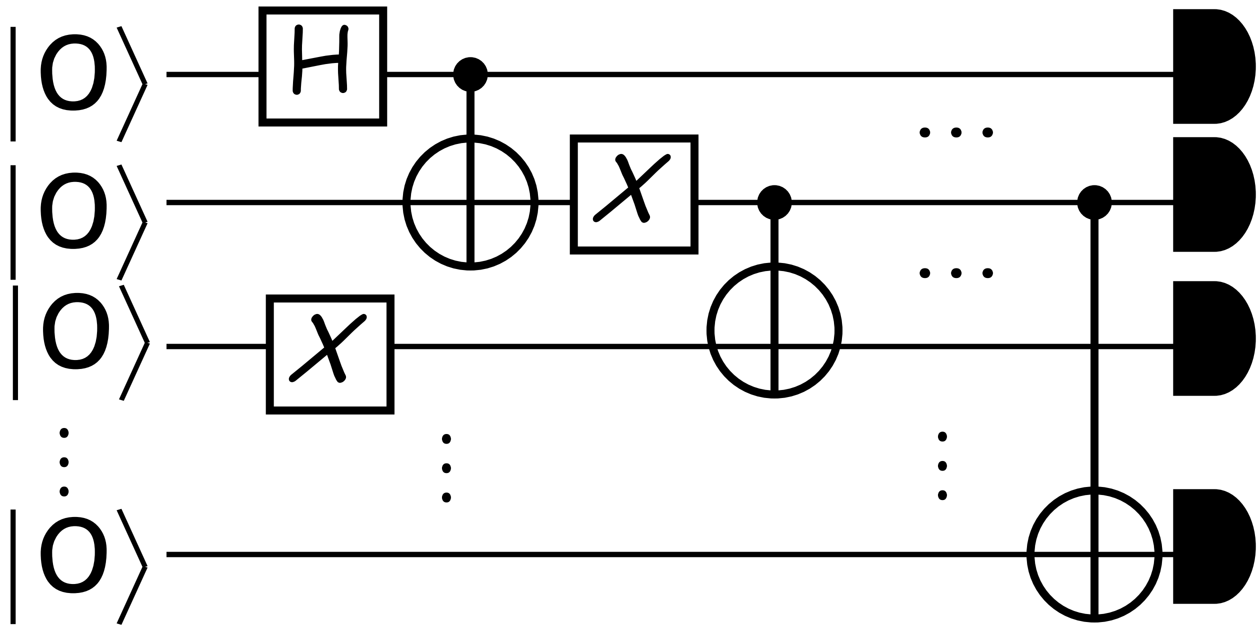
Quantum Circuit Model



$$\sum_{x \in \{0,1\}^n} \alpha_x |x\rangle$$

$$\sum_{x \in \{0,1\}^n} |\alpha_x|^2 = 1$$

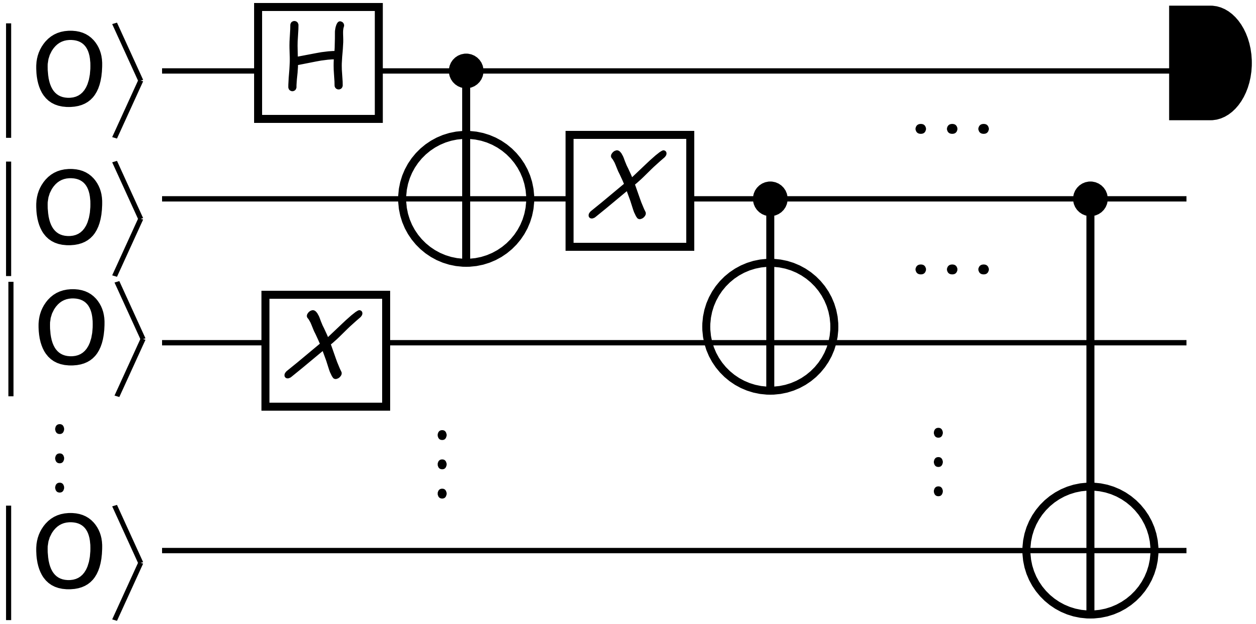
Measurement



Measuring all n qubits yields the result

$|x\rangle = |x_1\rangle|x_2\rangle\cdots|x_n\rangle$ with probability $|\alpha_x|^2$

Partial Measurement



Partial Measurement

Suppose we measure the first bit of $\sum_{x\in\{0,1\}^n} \alpha_x |x\rangle$ which can be rewritten as

$$\sum_{y\in\{0,1\}^{n-1}} \alpha_{0y}|0\rangle|y\rangle + \sum_{y\in\{0,1\}^{n-1}} \alpha_{1y}|1\rangle|y\rangle$$

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

remaining qubits

qubit 0

$$= a_0|0\rangle\left(\sum_{y\in\{0,1\}^{n-1}} \frac{\alpha_{0y}}{a_0}|y\rangle\right) + a_1|1\rangle\left(\sum_{y\in\{0,1\}^{n-1}} \frac{\alpha_{1y}}{a_1}|y\rangle\right)$$

$$a_0 = \sqrt{\sum_{y\in\{0,1\}^{n-1}} |\alpha_{0y}|^2} \qquad a_1 = \sqrt{\sum_{y\in\{0,1\}^{n-1}} |\alpha_{1y}|^2}$$

Partial Measurement

The probability of obtaining $|0\rangle$ is

$$|a_0|^2 = \sum_{y \in \{0,1\}^{n-1}} |\alpha_{0y}|^2$$

and in this case the remaining qubits will be left in the state

$$\sum_{y \in \{0,1\}^{n-1}} \frac{\alpha_{0y}}{a_0} |y\rangle$$

(reminiscent of Bayes' theorem)

Measurement: observer breaks a closed system

- Note that the act of measurement involves interacting the formerly closed system with an external system (the "observer" or "measuring apparatus").

- So the evolution of the system is no longer necessarily unitary.

Note that "global" phase doesn't matter

Measuring $\sum_{x\in\{0,1\}^n} \alpha_x |x\rangle$ gives $|x\rangle$ with probability $|\alpha_x|^2$

Measuring $\sum_{x\in\{0,1\}^n} e^{i\varphi}\alpha_x |x\rangle$ gives $|x\rangle$ with probability $|e^{i\varphi}\alpha_x|^2 = |\alpha_x|^2$

Note that "global" phase doesn't matter

Can we apply some unitary operation that will make the phase measurable? No!

$$U\left(\sum_{x\in\{0,1\}^n} e^{i\varphi}\beta_x|x\rangle\right) = e^{i\varphi}U\left(\sum_{x\in\{0,1\}^n}\beta_x|x\rangle\right)$$

Another tensor product fact

$$\left(\alpha\begin{bmatrix} a \\ b \end{bmatrix}\right)\otimes\begin{bmatrix} c \\ d \end{bmatrix}=\begin{bmatrix} \alpha ac \\ \alpha ad \\ \alpha bc \\ \alpha bd \end{bmatrix}=\begin{bmatrix} a \\ b \end{bmatrix}\otimes\left(\alpha\begin{bmatrix} c \\ d \end{bmatrix}\right)$$

$$=\alpha\left(\begin{bmatrix} a \\ b \end{bmatrix}\otimes\begin{bmatrix} c \\ d \end{bmatrix}\right)$$

Another tensor product fact

So

$$|x\rangle\left(\alpha|y\rangle\right) = \left(\alpha|x\rangle\right)|y\rangle = \alpha|x\rangle|y\rangle$$

….please remember….

Now we have a base of facts to discuss the most interesting aspect of quantum computing - quantum algorithms that are different than for normal (Turing machine-like, circuit-like) computing.

Basic Ideas of Quantum Algorithms

Quantum Algorithms give 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

$$O\left(2^{n^{1/3}\log(n)^{2/3}}\right).$$

Example: discrete Fourier transform

- Problem: for a given vector (x_j), j= 1,..., N, what is the discrete Fourier transform (DFT) vector

$$y_j = \sum_{k=1}^{N} \exp(2\pi i(j-1)(k-1)/N) x_k$$

- Algorithm:
  – a detailed step- by- step method to calculate the DFT (y_j) for any instance (x_j)
- With such an algorithm, one could:
  – write a DFT program to run on a computer
  – build a custom chip that calculates the DFT
  – train a team of children to execute the algorithm (remember the Andleman DNA algorithm and children with Lego?)

Computational complexity of DFT

- For the DFT, *N* could be the dimension of the vector

$$y_j = \sum_{k=1}^{N} \exp(2\pi i(j-1)(k-1)/N) x_k$$

- To calculate each y_j, must sum *N* terms
- This sum must be performed for *N* different y_j
- Computational complexity of DFT: requires N^2 steps
- DFTs are important --> a lot of work in optical computing (1950s, 1960s) to do fast DFTs
- 1965: Tukey and Cooley invent the Fast Fourier Transform (FFT), requires *N* log*N* steps
- FFT much faster --> optical computing almost dies overnight

Example: Factoring

- Factoring: given a number, what are its prime factors?
- Considered a "hard" problem in general, especially for numbers that are products of 2 large primes

Example: 4633 = 41 x 113

RSA-129

114381625757888867669235779976146612010218 296721242362562561842935706935245733897830597123563958705058989075147599290026879543541 = 3490529510847650949147849619903898133417764638493387843990820577 x 32769132993266709549961988190834461413177642967992942539798288533

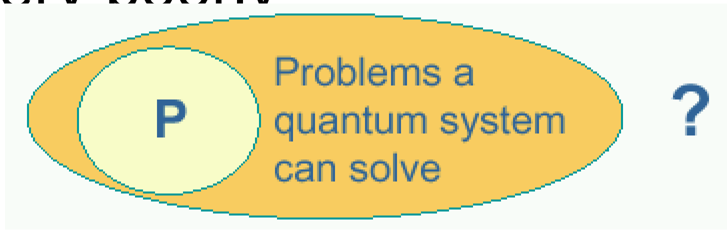
- Best factoring algorithm requires resources that grow exponentially in the size of the number (RSA-129 took 17 years)
- Example: factor a 300-digit number
  - Best algorithm: takes 10^{24} steps
  - On computer at THz speed: 150,000 years
- Difficulty of factoring is the basis of security for the RSA encryption scheme used, e.g., on the internet
- Information security of interest to private and public sectors

Quantum algorithms

Richard Feynman

David Deutsch

- Feynman (1982): there may be quantum systems that cannot be simulated efficiently on a "classical" computer
- Deutsch (1985): proposed that machines using quantum processes might be able to perform computations that "classical" computers can only perform very poorly



Concept of *quantum computer* emerged as a universal device to execute such quantum algorithms

Factoring with quantum systems

- ***Shor (1995):*** quantum factoring algorithm

Best classical algorithm: 10^{24} steps	Shor's quantum algorithm: 10^{10} steps
On classical THz computer: 150,000 years	On quantum THz computer: <1 second

- To implement Shor's algorithm, one could:
  - run it as a program on a "universal quantum computer"
  - design a custom quantum chip with hard- wired algorithm
  - find a quantum system that does it naturally! (?)

<u>Reminder to appreciate</u> : exponential savings is very good! Factor a 5,000 digit number:

–Classical computer (1ns/instruction, ~today's best algorithm)

- over 5 trillion years (the universe is ~ 10–16 billion years old).

–Quantum computer (1ns/instruction, ~Shor's algorithm)

- just over 2 minutes

….the power of quantum computing…...

Implications of Factoring and other quantum algorithms

- Information security and e-commerce are based on the use of **NP** problems that are not in **P**
  - must be “hard” (not in **P**) so that security is unbreakable
  - requires knowledge/ assumptions about the algorithmic and computational power of your adversaries
- Quantum algorithms (e. g., Shor’s factoring algorithm) require us to reassess the security of such systems
- Lessons to be learned:
  - algorithms and complexity classes can change!
  - information security is based on assumptions of what is hard and what is possible --> better be convinced of their validity

Shor's algorithm

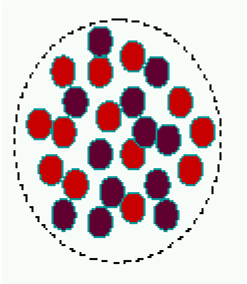
- **Hybrid algorithm to factor numbers**
- **Quantum component finds period r of sequence $a_1, a_2, \ldots, a_i, \ldots,$ given an oracle function that maps i to a_i**
- **Skeleton of the algorithm:**
  - create a superposition of all oracle inputs and call the oracle
  - apply a quantum Fourier transform 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**

Shor Type Algorithms

1985 Deutsch's algorithm	demonstrates task quantum computer can perform in one shot that classically takes two shots.
1992 Deutsch-Jozsa algorithm	demonstrates an exponential separation between classical deterministic and quantum algorithms.
1993 Bernstein-Vazirani algorithm	demonstrates a superpolynomial separation between probabilistic and quantum algorithms.
1994 Simon's algorithm	demonstrates an exponential separation between probabilistic and quantum algorithms.
1994 Shor's algorithm	demonstrates that quantum computers can efficiently factor numbers.

Search problems

- **Problem 1 : Given an unsorted database of *N* items, how long will it take to find a particular item x?**
  - **Check items one at a time. Is it x?**
  - **Average number of checks: *N/ 2***
- **Problem 2 : Given an unsorted database of *N* items, each either red or black, how many are red?**
  - **Start a tally**
  - **Check items one at a time. Is it red?**
    - **If it is red, add one to the tally**
    - **If it is black, don't change the tally**
  - **Must check all items: requires *N* checks**

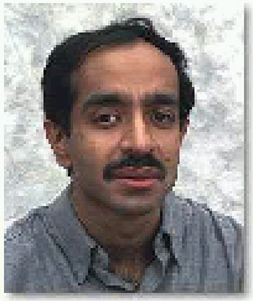


- **Not surprisingly, these are the best (classical) algorithms**
- **We can define quantum search algorithms that do better**

Oracles

- We need a "quantum way" to recognize a solution
- Define an *oracle* to be the unitary operator
- $O : |x> |q> \; --> \; |x> \; |q \oplus f(x) >$

  where $|q>$ is an ancilla qubit
- Could measure the ancilla qubit to determine if *x* is a solution
- Doesn't this "oracle" need to know the solution?
  - It just needs to recognize a solution when given one
  - Similar to NP problems
- One oracle call represents a fixed number of operations
- Address the complexity of a search algorithm in terms of the number of oracle calls --> separates scaling from fixed costs


Lov Grover

Quantum searching

- Grover (1996): quantum search algorithm
- For *M* solutions in a database containing *N* elements:

Classical search	Quantum search
N/M oracle calls	$(N/M)^{1/2}$ oracle calls

- Quantum search algorithm works by applying the oracle to superpositions of all elements,
  - it increases the amplitude of solutions (viewed as states)
- Quantum search requires that we know M/ *N* (at least approximately) prior to the algorithm, in order to perform the correct number of steps
- Failure to measure a solution --> run the algorithm again .

Quantum counting

- What if the number of solutions *M* is not known?
- Need *M* in order to determine the number of iterations of the Grover operator
- Classical algorithm requires *N* steps
- Quantum algorithm: Use phase estimation techniques
  - based on quantum Fourier transform (Shor)
  - requires $N^{1/2}$ oracle calls
- For a search with unknown number of solutions:
  - First perform quantum counting: $N^{1/2}$
  - With M, perform quantum search: $N^{1/2}$
  - Total search algorithm: still only $N^{1/2}$

Example of collaboration of two quantum algorithms

Can we do better than Grover quantum search?

- Quantum search algorithm provides a *quadratic speedup* over best classical algorithm

  Classical: N steps Quantum: $N^{1/2}$ steps

- Maybe there is a better quantum search algorithm
- Imagine one that requires **log *N*** steps:
  - Quantum search would be exponentially faster than any classical algorithm
  - Used for **NP** problems: could reduce them to **P** by searching all possible solutions
- **Unfortunately**, **NO**: Quantum search algorithm is "optimal"
- Any search- based method for **NP** problems is slow

How do quantum algorithms work?

- What makes a quantum algorithm potentially faster than any classical one?
  - **Quantum parallelism:** by using superpositions of quantum states, the computer is executing the algorithm on *all possible inputs at once*
  - **Dimension of quantum Hilbert space:** the "size" of the state space for the quantum system is *exponentially larger* than the corresponding classical system
  - **Entanglement capability:** different subsystems (qubits) in a quantum computer become *entangled*, exhibiting nonclassical correlations
- We don't really know what makes quantum systems more powerful than a classical computer
- Quantum algorithms are helping us understand the computational power of quantum *versus* **classical** systems

Quantum algorithms research

- **Require more quantum algorithms in order to:**
  - **solve problems more efficiently**
  - **understand the power of quantum computation**
  - **make valid/ realistic assumptions for *information security***
- **Problems for quantum algorithms research:**
  - **requires close collaboration between physicists and computer scientists**
  - **highly non- intuitive nature of quantum physics**
  - **even classical algorithms research is difficult**

Summary of quantum algorithms

- It may be possible to solve a problem on a quantum system much faster (i. e., using fewer steps) than on a classical computer
- Factorization and searching are examples of problems where quantum algorithms are known and are faster than any classical ones
- Implications for cryptography, information security
- Study of quantum algorithms and quantum computation is important in order to make assumptions about adversary's algorithmic and computational capabilities
- Leading to an understanding of the computational power of quantum versus classical systems

Deutsch's Problem

… everything started with small circuit of Deutsch…...

Deutsch's Problem

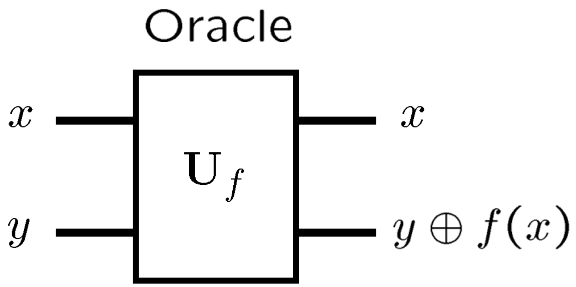
(Deutsch '85)

two qubits

$$\alpha_0|00\rangle + \alpha_1|01\rangle + \alpha_2|10\rangle + \alpha_3|11\rangle \qquad \begin{pmatrix}\alpha_0\\ \alpha_1\\ \alpha_2\\ \alpha_3\end{pmatrix} \begin{matrix}|00\rangle\\ |01\rangle\\ |10\rangle\\ |11\rangle\end{matrix}$$

David Deutsch

Delphi



$x, y, f(x) \in \{0,1\}$

Example $f(x) = x$:

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

Four possible functions $f(x)$:

$$\underbrace{f(x)=0 \quad f(x)=1}_{\text{Constant functions}} \quad \underbrace{f(x)=x \quad f(x)=\bar{x}}_{\text{Balanced functions}}$$

Deutsch's Problem

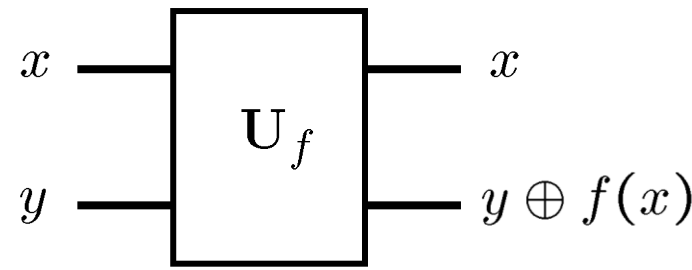
Determine whether f(x) is constant or balanced using <u>as few queries</u> to the oracle as possible.

Classical Deutsch

Four possible functions $f(x)$:

$$\underbrace{f(x)=0 \quad f(x)=1}_{\text{Constant functions}} \quad \underbrace{f(x)=x \quad f(x)=\bar{x}}_{\text{Balanced functions}}$$

Oracle

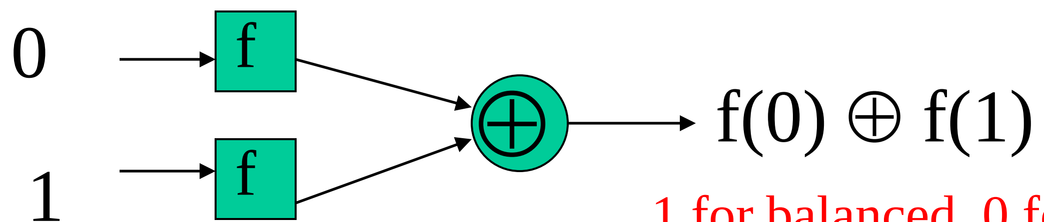


$$x, y, f(x) \in \{0, 1\}$$

Query input of x_0 and y_0 only gives information about $f(x_0)$.

Knowing $f(x_0)$ not enough to distinguish constant from balanced.

Classically we need to query the oracle **two times** to solve Deutsch's Problem



1 for balanced, 0 for constants

Quantum Deutsch:first explanation

1. Query oracle with $\frac{1}{2}(|00\rangle - |01\rangle + |10\rangle - |11\rangle)$
2. Producing the state $\frac{1}{2}\left(|0f(0)\rangle - |0\bar{f}(0)\rangle + |1f(1)\rangle - |1\bar{f}(1)\rangle\right)$

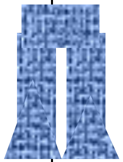
Substitute f

$f(x) = 0$	$f(x) = 1$	$f(x) = x$	$f(x) = \bar{x}$
$\frac{1}{2}(\|00\rangle - \|01\rangle + \|10\rangle - \|11\rangle)$	$\frac{1}{2}(\|01\rangle - \|00\rangle + \|11\rangle - \|10\rangle)$	$\frac{1}{2}(\|00\rangle - \|01\rangle + \|11\rangle - \|10\rangle)$	$\frac{1}{2}(\|01\rangle - \|00\rangle + \|10\rangle - \|11\rangle)$
$\frac{1}{2}\begin{pmatrix}1\\-1\\1\\-1\end{pmatrix}$	$\frac{1}{2}\begin{pmatrix}-1\\1\\-1\\1\end{pmatrix}$	$\frac{1}{2}\begin{pmatrix}1\\-1\\-1\\1\end{pmatrix}$	$\frac{1}{2}\begin{pmatrix}-1\\1\\1\\-1\end{pmatrix}$

Constant functions | Balanced functions

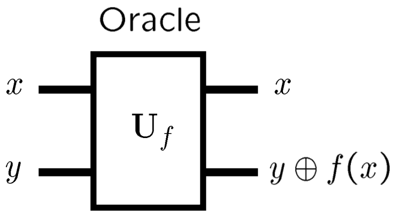
3. Apply the unitary transformation

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

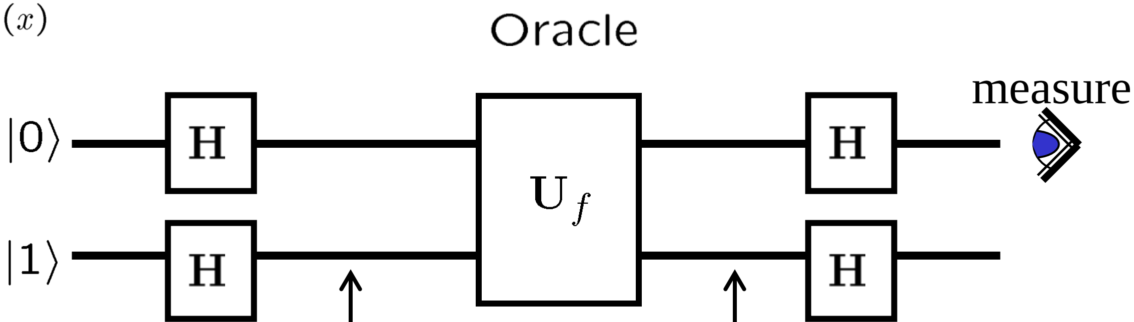


$\begin{pmatrix}0\\1\\0\\0\end{pmatrix}$	$\begin{pmatrix}0\\-1\\0\\0\end{pmatrix}$	$\begin{pmatrix}0\\0\\0\\1\end{pmatrix}$	$\begin{pmatrix}0\\0\\0\\-1\end{pmatrix}$
100 % |01>	100 % |01>	100 % |11>	100 % |11>

Deutsch Circuit



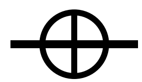
$x, y, f(x) \in \{0, 1\}$

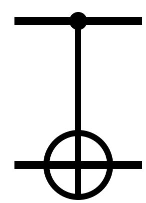


$$\frac{1}{2}(|00\rangle - |01\rangle + |10\rangle - |11\rangle)$$

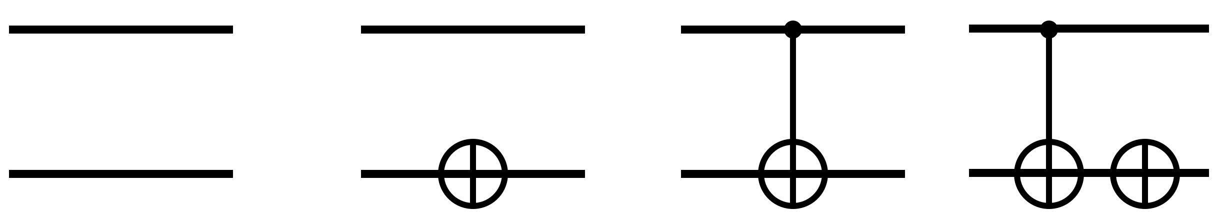
$$\frac{1}{2}\left(|0f(0)\rangle - |0\bar{f}(0)\rangle + |1f(1)\rangle - |1\bar{f}(1)\rangle\right)$$

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


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


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

$f(x) = 0$ $\quad$ $f(x) = 1$ $\quad$ $f(x) = x$ $\quad$ $f(x) = \bar{x}$

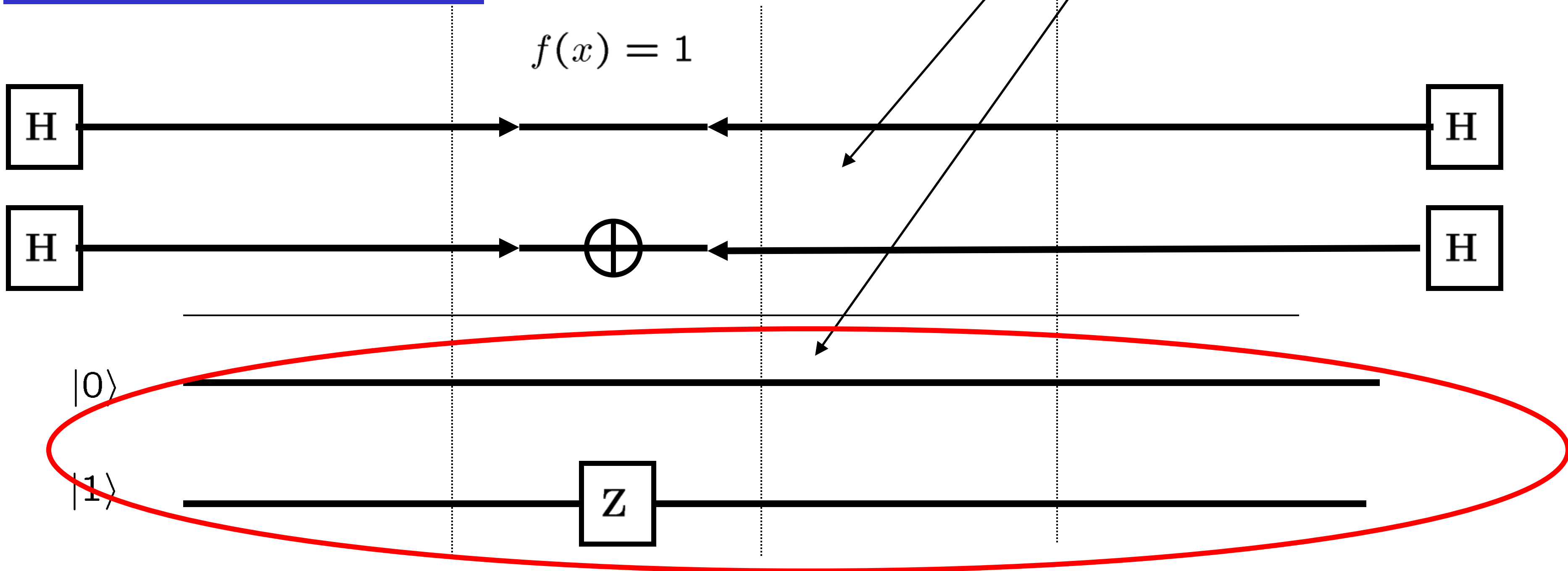


Quantum Deutsch: second explanation

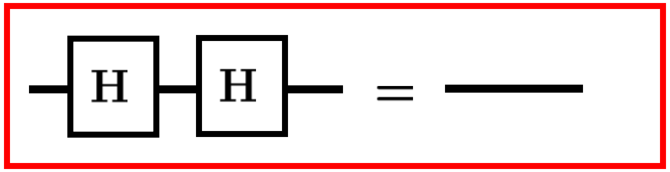
This kind of proof is often faster and more intuitive but it is better to check using matrices because you likely can make errors

$$\begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{pmatrix} = \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{pmatrix} \begin{pmatrix} \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

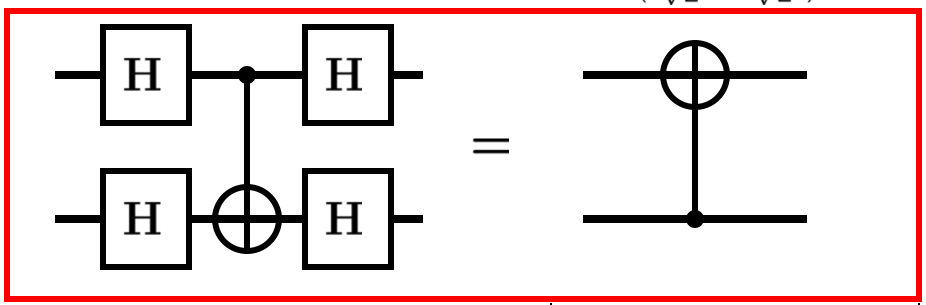
This circuit is replaced by this

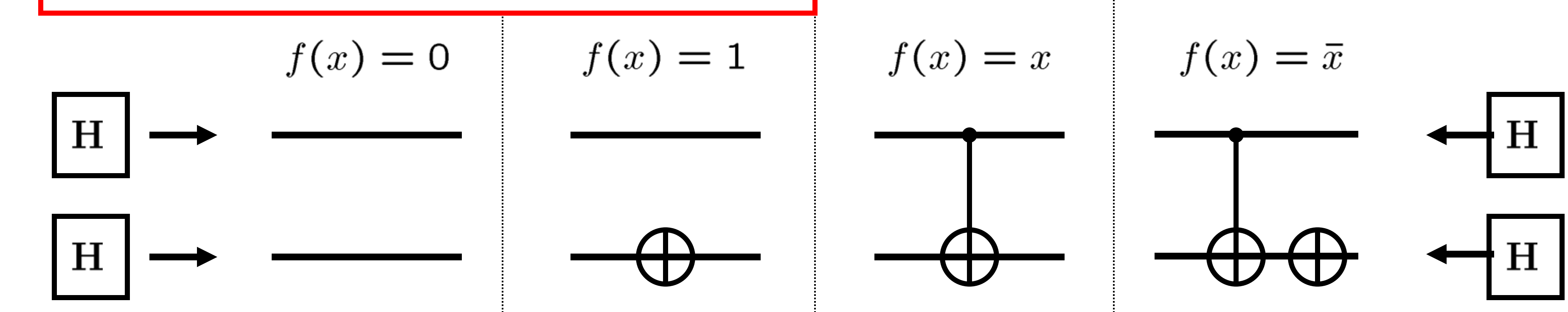


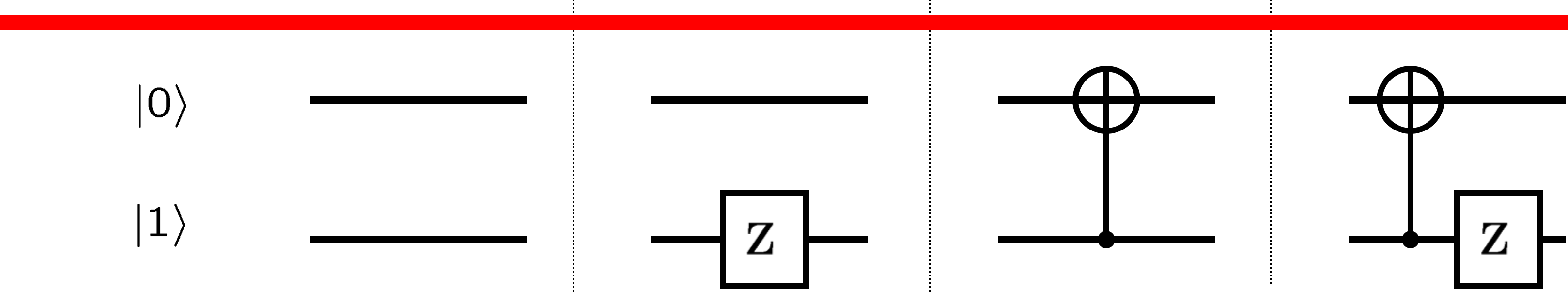
Quantum Deutsch: second explanation



$$\begin{pmatrix}\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}}\end{pmatrix}\begin{pmatrix}0 & 1 \\ 1 & 0\end{pmatrix}\begin{pmatrix}\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}}\end{pmatrix} = \begin{pmatrix}\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}}\end{pmatrix}\begin{pmatrix}\frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}}\end{pmatrix} = \begin{pmatrix}1 & 0 \\ 0 & -1\end{pmatrix}$$







This is obtained after connecting Hadamards and simplifying

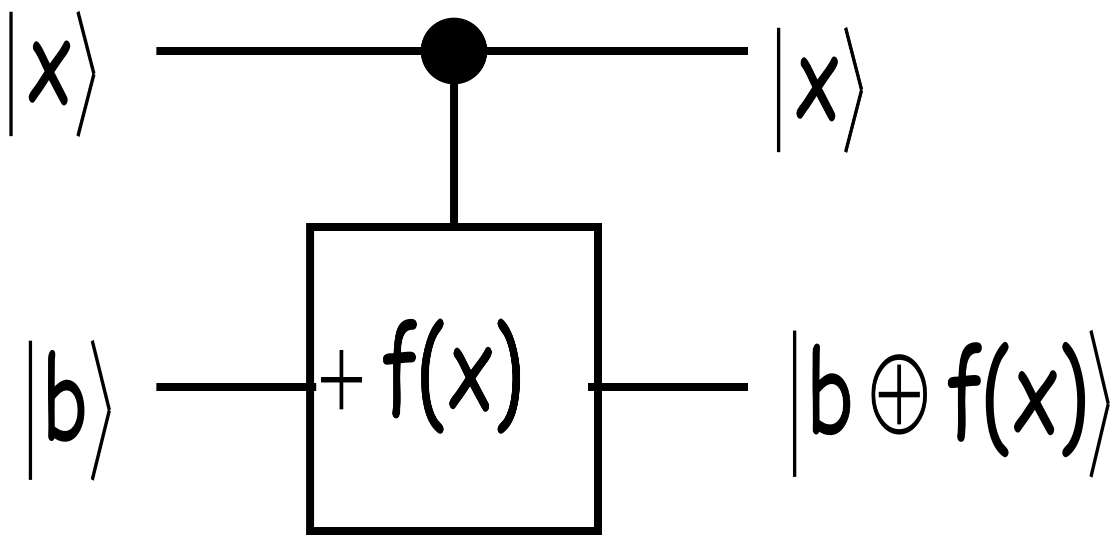
Generalize these ideas

- So, we can distinguish by measurement between first two circuits from bottom and second two circuits from bottom.
- This method is very general, we can build various oracles and check how they can be distinguished, by how many tests.
- In this case, we just need one test, but in a more general case we can have a decision tree for decision making.

Quantum Deutsch: third explanation

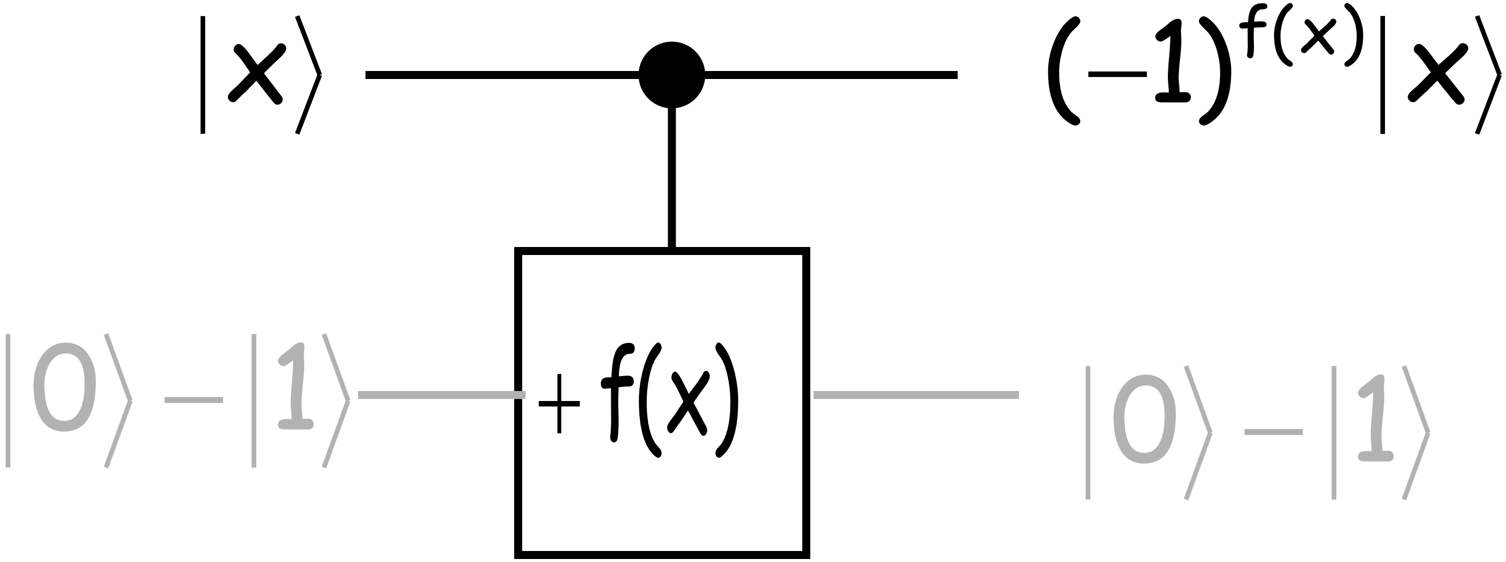
$$f : \{0,1\} \rightarrow \{0,1\}$$

Find $f(0) \oplus f(1)$ using only 1 evaluation of a reversible "black-box" circuit for f



Phase "kick-back" trick

The phase depends on function f(x)

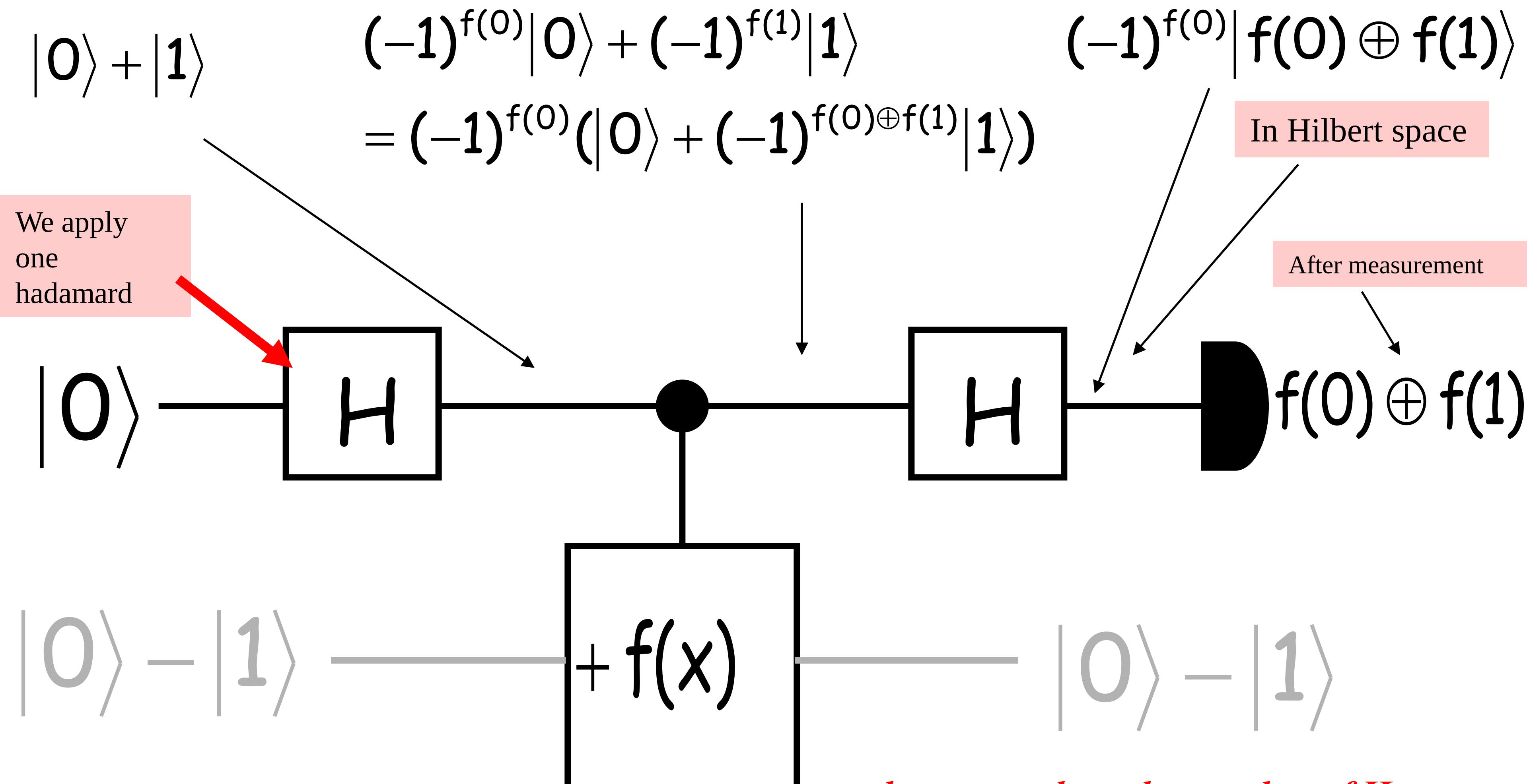


$$|x\rangle(|0\rangle - |1\rangle) \rightarrow |x\rangle(|f(x)\rangle - |f(x)\oplus 1\rangle)$$

$$= |x\rangle(-1)^{f(x)}(|0\rangle - |1\rangle)$$

$$= (-1)^{f(x)}|x\rangle(|0\rangle - |1\rangle)$$

A Deutsch quantum algorithm:
third explanation continued



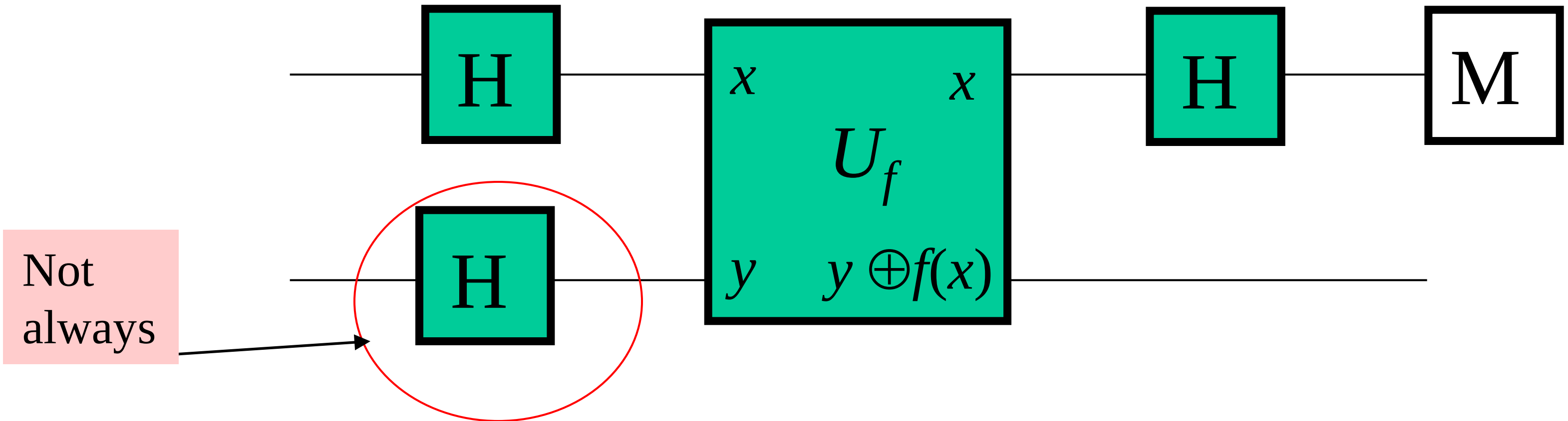
…here we reduce the number of H gates...

Deutsch Algorithm Philosophy

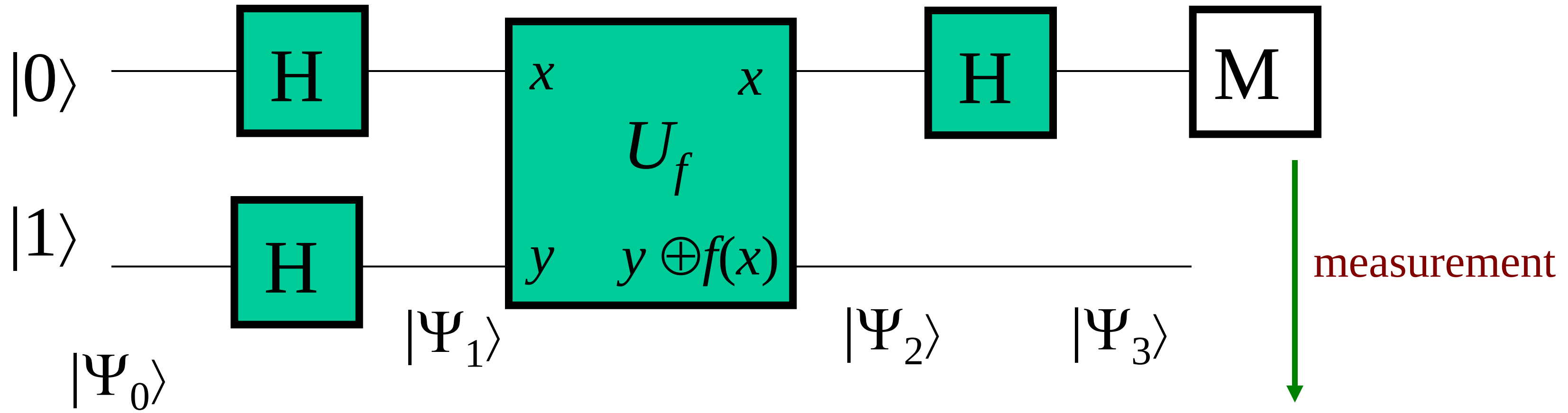
- Since we can prepare a superposition of all the inputs, we can learn a *global* property of f (i.e. a property that depends on ***all*** the values of f(x)) by only applying f once
- The global property is **encoded in the phase information**, which we learn via **interferometry**
- Classically, one application of f will only allow us to probe its value on one input

We use just one quantum evaluation by, in effect, computing *f*(0) and *f*(1) simultaneously

- **The Circuit:**

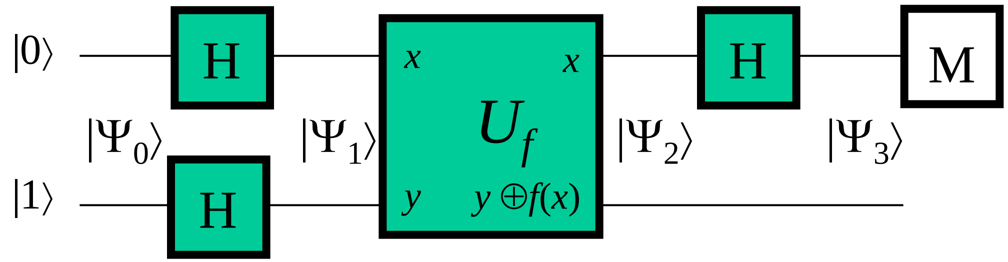


Deutsch's Algorithm



- Initialize with $|\Psi_0\rangle = |01\rangle$
- Create superposition of *x* states using the first Hadamard (H) gate. Set *y* control input using the second H gate
- Compute $f(x)$ using the special unitary circuit U_f
- Interfere the $|\Psi_2\rangle$ states using the third H gate
- Measure the *x* qubit

Deutsch's Algorithm with single qubit measurement



$$|\Psi_0\rangle = [|0\rangle][|1\rangle] \qquad |\Psi_1\rangle = \left[\frac{|0\rangle + |1\rangle}{\sqrt{2}}\right]\left[\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right]$$

$$|\Psi_2\rangle = \left[\frac{(-1)^{f(0)}|0\rangle + (-1)^{f(1)}|1\rangle}{\sqrt{2}}\right]\left[\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right] = \begin{cases} \pm\left[\dfrac{|0\rangle + |1\rangle}{\sqrt{2}}\right]\left[\dfrac{|0\rangle - |1\rangle}{\sqrt{2}}\right] & \text{if } f(0) = f(1) \\ \pm\left[\dfrac{|0\rangle - |1\rangle}{\sqrt{2}}\right]\left[\dfrac{|0\rangle - |1\rangle}{\sqrt{2}}\right] & \text{if } f(0) \neq f(1) \end{cases}$$

$$|\Psi_3\rangle = \begin{cases} \pm[|0\rangle]\left[\dfrac{|0\rangle - |1\rangle}{\sqrt{2}}\right] & \text{if } f(0) = f(1) \\ \pm[|1\rangle]\left[\dfrac{|0\rangle - |1\rangle}{\sqrt{2}}\right] & \text{if } f(0) \neq f(1) \end{cases}$$

Deutsch In Perspective

Quantum theory allows us to do in a single query what classically requires two queries.

What about problems where the computational complexity is exponentially more efficient?

Extended Deutsch's Problem

- Given black-box f:$\{0,1\}^n \rightarrow \{0,1\}$,
  - and a guarantee that f is either *constant* or *balanced* (1 on exactly ½ of inputs)
  - Which is it?
  - Minimize number of calls to f.
- Classical algorithm, worst-case:
  - Order 2^n time!
    - What if the first $2^{n\text{-}1}$ cases examined are all 0?
      - Function could be either constant or balanced.
    - Case number $2^{n\text{-}1}+1$: if 0, constant; if 1, balanced.
- **Quantum algorithm is exponentially faster!**
  - **(Deutsch & Jozsa, 1992.)**

Deutsch-Jozsa Problem

(1992)

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

function maps n bit strings to a single bit

f is promised to be either

(A) f is constant.

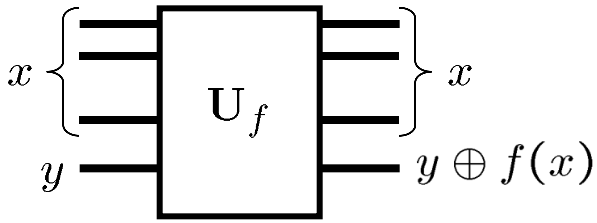
$f(x) = b \ \forall x$

(B) f is balanced.

$f(x) = 1$ if $x \in \mathcal{S}$ otherwise $f(x) = 0$

$\mathcal{S}$ has 2^{n-1} elements.

Oracle



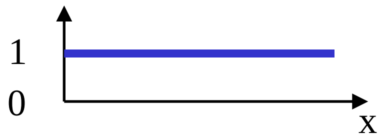
$x \in \{0,1\}^n$

$y, f(x) \in \{0,1\}$

Deutsch-Jozsa Problem

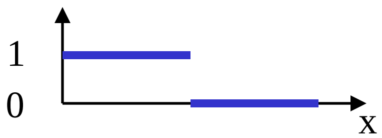
Determine whether f(x) is constant or balanced using as few queries to the oracle as possible.

Classical DJ



(A) f is constant.

$$f(x) = b \ \forall x$$



(B) f is balanced.

$$f(x) = 1 \text{ if } x \in \mathcal{S} \text{ otherwise } f(x) = 0$$

$\mathcal{S}$ has 2^{n-1} elements.

If we never want to be wrong:

Worst case we need $2^{n-1} + 1$ querries

If we allow a failure probability of ϵ, then

Randomly choose k different x_k to query.

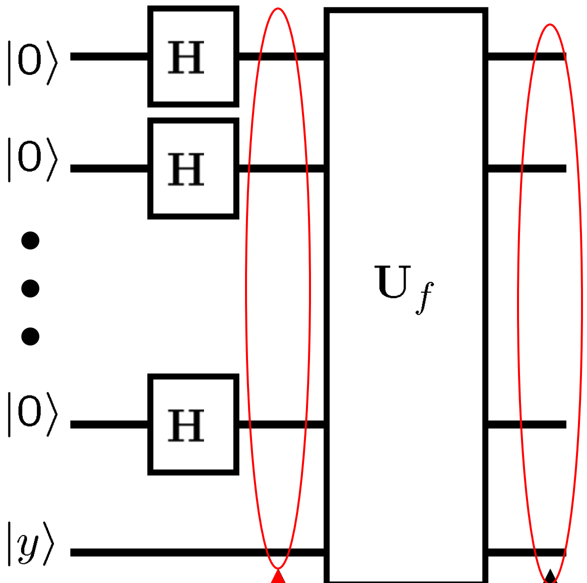
$$k = O\left(log\left(\frac{1}{\epsilon}\right)\right)$$

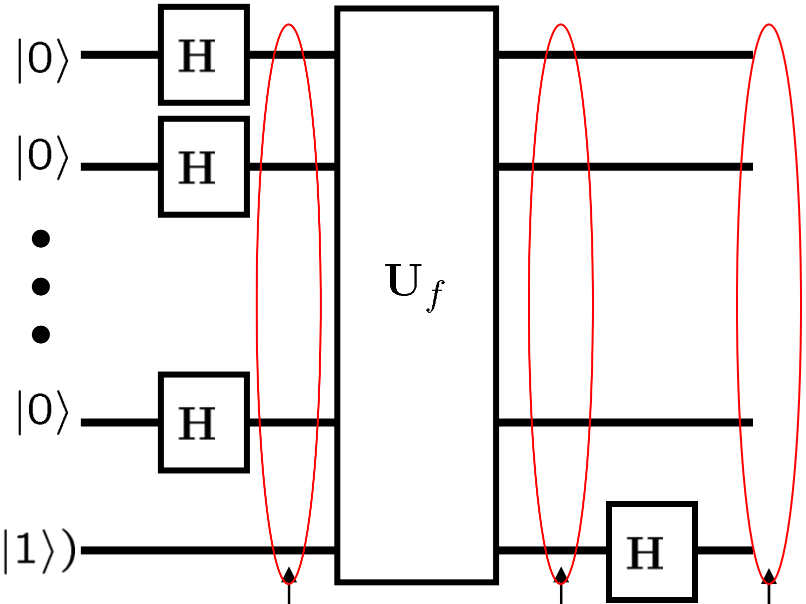
This is a probabilistic algorithm!

Determinstically hard, probabilistically easy.

Quantum DJ

Now we additionally apply Hadamard in output of the function





$$\frac{1}{\sqrt{2^n}}\sum_{x=0}^{2^n-1}|x\rangle|y\rangle$$

$$\frac{1}{\sqrt{2^n}}\sum_{x=0}^{2^n-1}|x\rangle|y\bigoplus f(x)\rangle$$

$$|x\rangle = |x_n x_{n-1}\dots x_1\rangle$$

$$\frac{1}{\sqrt{2^{n+1}}}\sum_{x=0}^{2^n-1}|x\rangle\left(|0\rangle - |1\rangle\right)$$

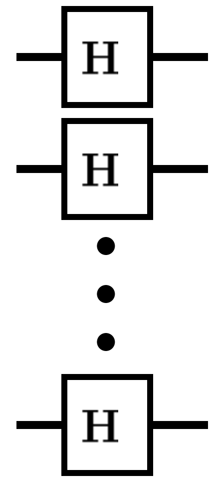
$$\frac{1}{\sqrt{2^{n+1}}}\sum_{x=0}^{2^n-1}|x\rangle\left(|f(x)\rangle - |\bar{f}(x)\rangle\right)$$

$$\frac{1}{\sqrt{2^n}}\sum_{x=0}^{2^n-1}(-1)^{f(x)}|x\rangle|1\rangle$$

Quantum DJ

Information about $f(x)$ is in the phase

$$\frac{1}{\sqrt{2^n}}\sum_{x=0}^{2^n-1}(-1)^{f(x)}|x\rangle|1\rangle$$



$= \mathbf{H}^{\otimes n}$

$$\mathbf{H}^{\otimes n}|x_n x_{n-1}\ldots x_1\rangle = \frac{1}{\sqrt{2}}(|0\rangle+(-1)^{x_n}|1\rangle)\frac{1}{\sqrt{2}}(|0\rangle+(-1)^{x_{n-1}}|1\rangle)\cdots\frac{1}{\sqrt{2}}(|0\rangle+(-1)^{x_1}|1\rangle)$$

$$= \frac{1}{\sqrt{2^n}}\sum_{y=0}^{2^n-1}(-1)^{x\cdot y}|y\rangle \qquad x\cdot y = x_n y_n + x_{n-1}y_{n-1} + \cdots + x_1 y_1 \bmod 2$$

All matrix elements of $\mathbf{H}^{\otimes n}$ are $\pm\frac{1}{\sqrt{2^n}}$

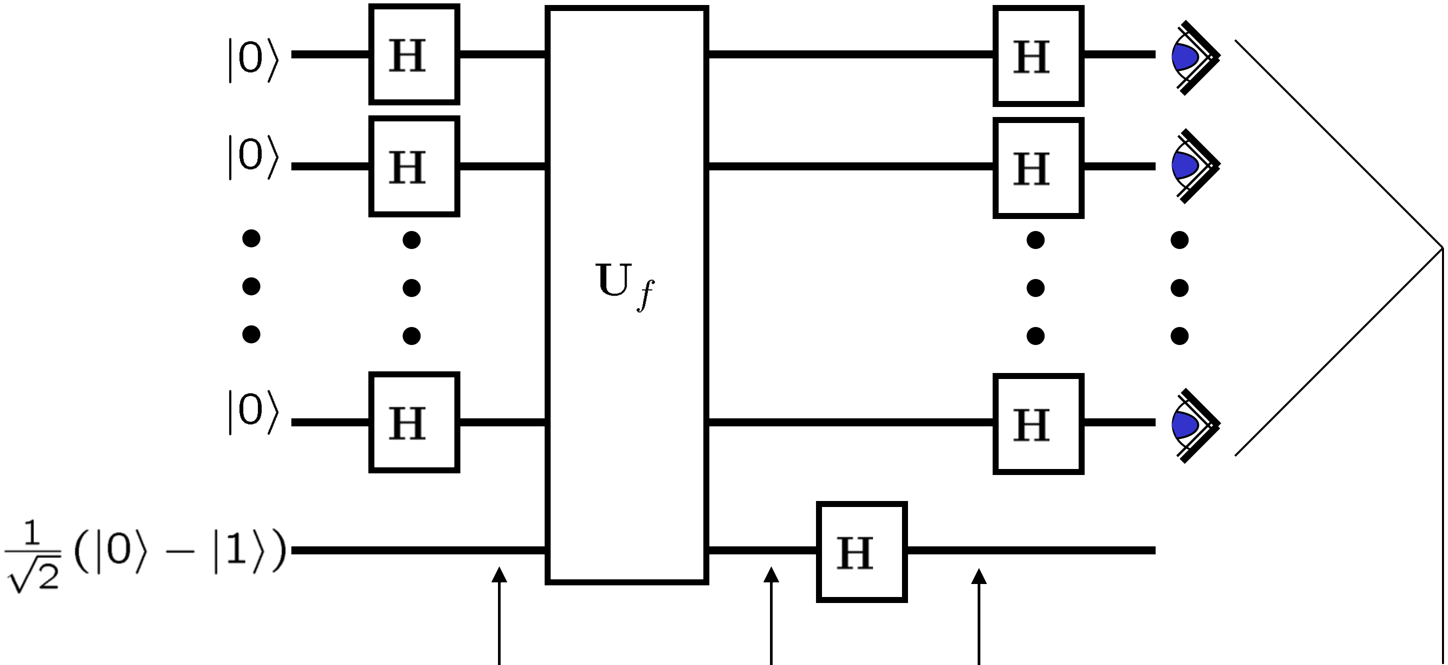
First row of $\mathbf{H}^{\otimes n}$ has all elements $+\frac{1}{\sqrt{2^n}}$

Other rows have equal number $\pm\frac{1}{\sqrt{2^n}}$ elements.

If $f(x)$ is constant, applying $\mathbf{H}^{\otimes n}$ to $\frac{1}{\sqrt{2^n}}\sum_{x=0}^{2^n-1}|x\rangle$ always gives $|0\rangle$.

If $f(x)$ is balanced, applying $\mathbf{H}^{\otimes n}$ to $\frac{1}{\sqrt{2^n}}\sum_{x=0}^{2^n-1}|x\rangle$ always gives a superposition without $|0\rangle$.

Full Quantum DJ



If all bits are 0, then f is constant.

If all bits are not 0, then f is balanced.

$$\frac{1}{\sqrt{2^{n+1}}}\sum_{x=0}^{2^n-1}|x\rangle\left(|0\rangle-|1\rangle\right)$$

$$\frac{1}{\sqrt{2^{n+1}}}\sum_{x=0}^{2^n-1}|x\rangle\left(|f(x)\rangle-|\bar{f}(x)\rangle\right)$$

$$\frac{1}{\sqrt{2^{n}}}\sum_{x=0}^{2^n-1}(-1)^{f(x)}|x\rangle|1\rangle$$

Solves DJ with a SINGLE query vs $2^{n-1}+1$ classical deterministic!!!!!!!!!

Deutsch-Josza Algorithm (contd)

- This algorithm distinguishes constant from balanced functions ***in one evaluation*** of f, versus $2^{n-1} + 1$ evaluations for classical deterministic algorithms
- Balanced functions have many interesting and some useful properties
  - K. Chakrabarty and **J.P. Hayes**, "Balanced Boolean functions," *IEE Proc: Digital Techniques*, vol. 145, pp 52 - 62, Jan. 1998.