Introduction to Quantum Computing

How I Learned to Stop Worrying and Love the Bomb

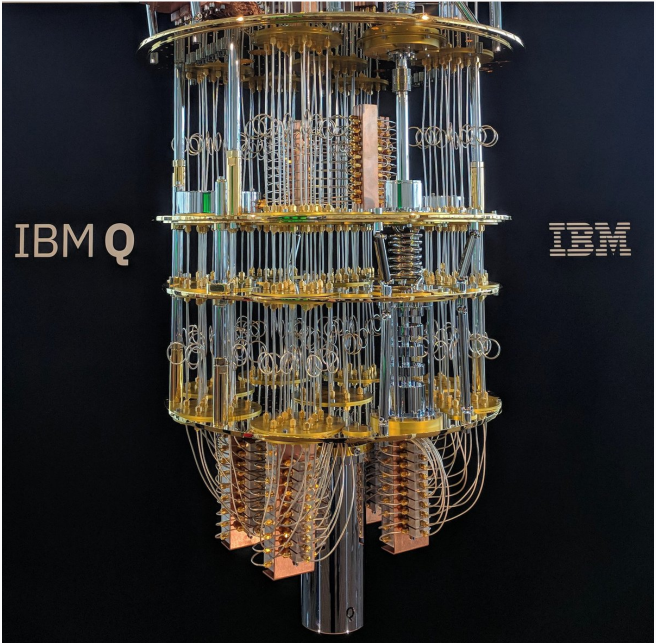
Dr. Omri Har-Shemesh

13. February 2019

Schmiede.ONE GmbH & Co. KG

Introduction

Ce n'est pas un lustre (This is not a Chandelier)



Big big picture

- New computing paradigm

Big big picture

- New computing paradigm
- Uses the rules of quantum mechanics

Big big picture

- New computing paradigm
- Uses the rules of quantum mechanics
- Might be able to solve some problems exponentially faster

Big big picture

- New computing paradigm
- Uses the rules of quantum mechanics
- Might be able to solve some problems exponentially faster
- Definitely could simulate quantum systems better

Big big picture

- New computing paradigm
- Uses the rules of quantum mechanics
- Might be able to solve some problems exponentially faster
- Definitely could simulate quantum systems better
- Realizable in large scale?

Big big picture

- New computing paradigm
- Uses the rules of quantum mechanics
- Might be able to solve some problems exponentially faster
- Definitely could simulate quantum systems better
- Realizable in large scale?
- ¯_(ツ)_/¯

Disclaimer

- Quantum Mechanics is technical

Disclaimer

- Quantum Mechanics is technical
- Quantum Mechanics is weird

Disclaimer

- Quantum Mechanics is technical
- Quantum Mechanics is weird
- Quantum Computing is technical

Disclaimer

- Quantum Mechanics is technical
- Quantum Mechanics is weird
- Quantum Computing is technical
- Quantum Algorithms are hard

Examples?

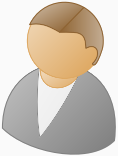
Breaking the RSA algorithm

Example: the RSA algorithm

Alice

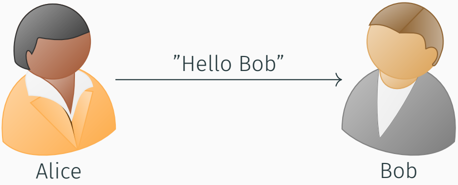
Example: the RSA algorithm

Alice

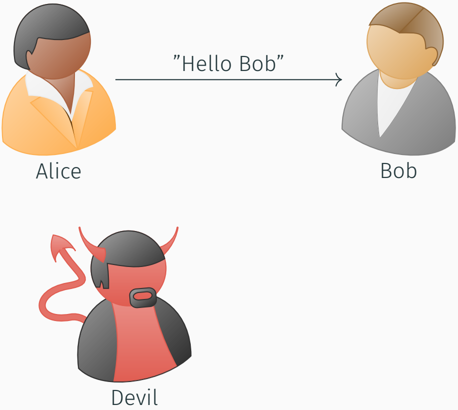


Bob

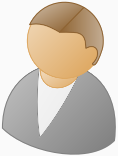
Example: the RSA algorithm



Example: the RSA algorithm



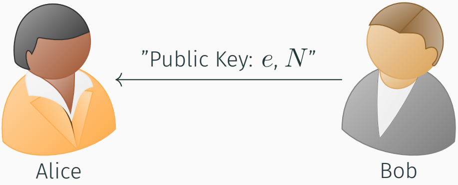
Alice



Bob

Generate Public/Private keys

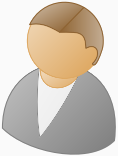
$$e, d, N = pq$$



Generate Public/Private keys

$$e, d, N = pq$$

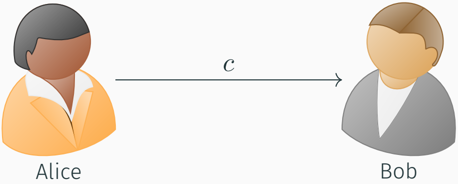
Alice



Bob

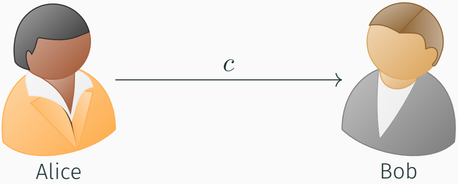
$$m = \text{encode}(\text{"Hello World"})$$
$$c = m^e \pmod{N}$$

Example: the RSA algorithm



$$m = \text{encode}(\text{"Hello World"})$$
$$c = m^e \pmod{N}$$

Example: the RSA algorithm



$$m = \text{encode}(\text{"Hello World"})$$
$$c = m^e \pmod{N}$$

$$m = c^d \pmod{N}$$
$$\text{decode}(m)$$

Example: the RSA algorithm - outline

It's easy to find $e, d, N \in \mathbb{N}$, such that:

Example: the RSA algorithm - outline

It's easy to find $e, d, N \in \mathbb{N}$, such that:

$$\boxed{(m^e)^d \equiv m \pmod{N}}$$

It's easy to find $e, d, N \in \mathbb{N}$, such that:

$$\boxed{(m^e)^d \equiv m \pmod N}$$

- $N = pq$ with p, q large prime numbers.

Example: the RSA algorithm - outline

It's easy to find $e, d, N \in \mathbb{N}$, such that:

$$\boxed{(m^e)^d \equiv m \pmod{N}}$$

- $N = pq$ with p, q large prime numbers.
- e, N - Public Key

Example: the RSA algorithm - outline

It's easy to find $e, d, N \in \mathbb{N}$, such that:

$$\boxed{(m^e)^d \equiv m \pmod{N}}$$

- $N = pq$ with p, q large prime numbers.
- e, N - Public Key
- d, N - Private Key

Example: the RSA algorithm - outline

It's easy to find $e, d, N \in \mathbb{N}$, such that:

$$\boxed{(m^e)^d \equiv m \pmod{N}}$$

- $N = pq$ with p, q large prime numbers.
- e, N - Public Key
- d, N - Private Key

Example: the RSA algorithm - outline

It's easy to find $e, d, N \in \mathbb{N}$, such that:

$$\boxed{(m^e)^d \equiv m \pmod{N}}$$

- $N = pq$ with p, q large prime numbers.
- e, N - Public Key
- d, N - Private Key

But it's hard to find p, q, such that $pq = N$.

Example: the RSA algorithm - Factorizing is Hard

Factorizing on a Classical Computer

Bits	Time	Notes
128	less than 2 seconds	
192	16 seconds	
256	35 minutes	
260	1 hour	
512	73 days	in 2009

Example: the RSA algorithm - Factorizing is Hard

Factorizing on a Classical Computer

Bits	Time	Notes
128	less than 2 seconds	
192	16 seconds	
256	35 minutes	
260	1 hour	
512	73 days	in 2009
768	1500 CPU years	Largest known (2010), took 2 years on hundreds of computers

Example: the RSA algorithm - Factorizing is Hard

Factorizing on a Classical Computer

Bits	Time	Notes
128	less than 2 seconds	
192	16 seconds	
256	35 minutes	
260	1 hour	
512	73 days	in 2009
768	1500 CPU years	Largest known (2010), took 2 years on hundreds of computers

Most RSA implementations use between 1024 and 4096 bits.

Would require ~ 4000 qubits to break 2048-bit RSA

Movie

The Technical Part

"Shut up and calculate"

David Mermin

Overview

- Representing computation with linear algebra
- Qubits, superposition and quantum logic gates
- Simplest problem where a quantum computer outperforms a classical one
- Bonus: Quantum entanglement

Representing classical bits as vectors

One bit with value 0, also written as $|0\rangle$ (Dirac vector notation)

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

One bit with value 1, also written as $|1\rangle$

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

Review: matrix multiplication

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} ax + by \\ cx + dy \end{pmatrix}$$

$$\begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} ax + by + cz \\ dx + ey + fz \\ gx + hy + iz \end{pmatrix}$$

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{pmatrix} w & x \\ y & z \end{pmatrix} = \begin{pmatrix} aw + by & ax + bz \\ cw + dy & cx + dz \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} a \\ b \\ c \\ d \end{pmatrix} = \begin{pmatrix} a \\ b \\ c \\ d \end{pmatrix}$$

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

What are the operations we can perform on one classical bit?

Identity	$f(x) = x$	$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$	$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$
Negation	$f(x) = \neg x$	$\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$	$\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$
Constant-0	$f(x) = 0$	$\begin{pmatrix} 1 & 1 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$	$\begin{pmatrix} 1 & 1 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$
Constant-1	$f(x) = 1$	$\begin{pmatrix} 0 & 0 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$	$\begin{pmatrix} 0 & 0 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$

Reversible computing

- Given the operation and the input, you can always infer the output.
  - For $Ax = b$, given b and A, you can uniquely find x.

Reversible computing

- Given the operation and the input, you can always infer the output.
  - For $Ax = b$, given b and A, you can uniquely find x.
- Permutations are reversible; erasing and overwriting are not
  - Identity and negation are reversible.
  - Constant-0 and Constant-1 are not reversible.

Reversible computing

- Given the operation and the input, you can always infer the output.
  - For $Ax = b$, given b and A, you can uniquely find x.
- Permutations are reversible; erasing and overwriting are not
  - Identity and negation are reversible.
  - Constant-0 and Constant-1 are not reversible.
- Quantum computers use only reversible operations.
  - In fact, all quantum operations are their own inverse.

Review: tensor product of vectors

$$\begin{pmatrix} x_0 \\ x_1 \end{pmatrix} \otimes \begin{pmatrix} y_0 \\ y_1 \end{pmatrix} = \begin{pmatrix} x_0 \begin{pmatrix} y_0 \\ y_1 \end{pmatrix} \\ x_1 \begin{pmatrix} y_0 \\ y_1 \end{pmatrix} \end{pmatrix} = \begin{pmatrix} x_0 y_0 \\ x_0 y_1 \\ x_1 y_0 \\ x_1 y_1 \end{pmatrix}$$

$$\begin{pmatrix} 1 \\ 2 \end{pmatrix} \otimes \begin{pmatrix} 3 \\ 4 \end{pmatrix} = \begin{pmatrix} 3 \\ 4 \\ 6 \\ 8 \end{pmatrix}$$

$$\begin{pmatrix} x_0 \\ x_1 \end{pmatrix} \otimes \begin{pmatrix} y_0 \\ y_1 \end{pmatrix} \otimes \begin{pmatrix} z_0 \\ z_1 \end{pmatrix} = \begin{pmatrix} x_0 y_0 z_0 \\ x_0 y_0 z_1 \\ x_0 y_1 z_0 \\ x_0 y_1 z_1 \\ x_1 y_0 z_0 \\ x_1 y_0 z_1 \\ x_1 y_1 z_0 \\ x_1 y_1 z_1 \end{pmatrix}$$

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

Representing multiple cbits

$$|00\rangle = \begin{pmatrix}1\\0\end{pmatrix} \otimes \begin{pmatrix}1\\0\end{pmatrix} = \begin{pmatrix}1\\0\\0\\0\end{pmatrix} \qquad |01\rangle = \begin{pmatrix}1\\0\end{pmatrix} \otimes \begin{pmatrix}0\\1\end{pmatrix} = \begin{pmatrix}0\\1\\0\\0\end{pmatrix}$$

$$|10\rangle = \begin{pmatrix}0\\1\end{pmatrix} \otimes \begin{pmatrix}1\\0\end{pmatrix} = \begin{pmatrix}0\\0\\1\\0\end{pmatrix} \qquad |11\rangle = \begin{pmatrix}0\\1\end{pmatrix} \otimes \begin{pmatrix}0\\1\end{pmatrix} = \begin{pmatrix}0\\0\\0\\1\end{pmatrix}$$

$$|100\rangle = \begin{pmatrix}0\\1\end{pmatrix} \otimes \begin{pmatrix}1\\0\end{pmatrix} \otimes \begin{pmatrix}1\\0\end{pmatrix} = \begin{pmatrix}0\\0\\0\\0\\1\\0\\0\\0\end{pmatrix}$$

- The tensor representation is called the product state.
- It can be factored back into the individual state representation.
- The product state of n bits is a vector of size 2^n.

Operations on multiple cbits: CNOT

- Takes two bits, one control bit and one target bit.
- If the control bit is set, flip the target bit, otherwise leave it.

Operations on multiple cbits: CNOT

- Takes two bits, one control bit and one target bit.
- If the control bit is set, flip the target bit, otherwise leave it.
- If most significant bit is control, and least-significant is target, then:

$$
\begin{array}{ll}
|00\rangle & |00\rangle \\
|01\rangle & |01\rangle \\
|10\rangle & |10\rangle \\
|11\rangle & |11\rangle
\end{array}
$$

Operations on multiple cbits: CNOT

- Takes two bits, one control bit and one target bit.
- If the control bit is set, flip the target bit, otherwise leave it.
- If most significant bit is control, and least-significant is target, then:

$$
\begin{array}{ccc}
|00\rangle & \longrightarrow & |00\rangle \\
|01\rangle & & |01\rangle \\
|10\rangle & & |10\rangle \\
|11\rangle & & |11\rangle
\end{array}
$$

Operations on multiple cbits: CNOT

- Takes two bits, one control bit and one target bit.
- If the control bit is set, flip the target bit, otherwise leave it.
- If most significant bit is control, and least-significant is target, then:

$$
\begin{aligned}
|00\rangle &\longrightarrow |00\rangle \\
|01\rangle &\longrightarrow |01\rangle \\
|10\rangle &\qquad\quad\;\; |10\rangle \\
|11\rangle &\qquad\quad\;\; |11\rangle
\end{aligned}
$$

- Takes two bits, one control bit and one target bit.
- If the control bit is set, flip the target bit, otherwise leave it.
- If most significant bit is control, and least-significant is target, then:

$$\begin{tikzcd} |00\rangle \arrow[r] & |00\rangle \\ |01\rangle \arrow[r] & |01\rangle \\ |10\rangle \arrow[dr] & |10\rangle \\ |11\rangle & |11\rangle \end{tikzcd}$$

Operations on multiple cbits: CNOT

- Takes two bits, one control bit and one target bit.
- If the control bit is set, flip the target bit, otherwise leave it.
- If most significant bit is control, and least-significant is target, then:

$$\begin{tikzcd}
|00\rangle \arrow[r] & |00\rangle \\
|01\rangle \arrow[r] & |01\rangle \\
|10\rangle \arrow[dr] & |10\rangle \\
|11\rangle \arrow[ur] & |11\rangle
\end{tikzcd}$$

Operations on multiple cbits: CNOT

- Takes two bits, one control bit and one target bit.
- If the control bit is set, flip the target bit, otherwise leave it.
- If most significant bit is control, and least-significant is target, then:

$$
\begin{aligned}
|00\rangle &\longrightarrow |00\rangle \\
|01\rangle &\longrightarrow |01\rangle \\
|10\rangle &\longrightarrow |11\rangle \\
|11\rangle &\longrightarrow |10\rangle
\end{aligned}
$$

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

Operations on multiple cbits: CNOT

$$C|10\rangle = C\left(\begin{pmatrix}0\\1\end{pmatrix}\otimes\begin{pmatrix}1\\0\end{pmatrix}\right) = \begin{pmatrix}1&0&0&0\\0&1&0&0\\0&0&0&1\\0&0&1&0\end{pmatrix}\begin{pmatrix}0\\0\\1\\0\end{pmatrix} = \begin{pmatrix}0\\0\\0\\1\end{pmatrix} = \begin{pmatrix}0\\1\end{pmatrix}\otimes\begin{pmatrix}0\\1\end{pmatrix} = |11\rangle$$

$$C|11\rangle = C\left(\begin{pmatrix}0\\1\end{pmatrix}\otimes\begin{pmatrix}0\\1\end{pmatrix}\right) = \begin{pmatrix}1&0&0&0\\0&1&0&0\\0&0&0&1\\0&0&1&0\end{pmatrix}\begin{pmatrix}0\\0\\0\\1\end{pmatrix} = \begin{pmatrix}0\\0\\1\\0\end{pmatrix} = \begin{pmatrix}0\\1\end{pmatrix}\otimes\begin{pmatrix}1\\0\end{pmatrix} = |10\rangle$$

Operations on multiple cbits: CNOT

$$C|00\rangle = C\left(\begin{pmatrix}1\\0\end{pmatrix}\otimes\begin{pmatrix}1\\0\end{pmatrix}\right) = \begin{pmatrix}1&0&0&0\\0&1&0&0\\0&0&0&1\\0&0&1&0\end{pmatrix}\begin{pmatrix}1\\0\\0\\0\end{pmatrix} = \begin{pmatrix}1\\0\\0\\0\end{pmatrix} = \begin{pmatrix}1\\0\end{pmatrix}\otimes\begin{pmatrix}1\\0\end{pmatrix} = |00\rangle$$

$$C|01\rangle = C\left(\begin{pmatrix}1\\0\end{pmatrix}\otimes\begin{pmatrix}0\\1\end{pmatrix}\right) = \begin{pmatrix}1&0&0&0\\0&1&0&0\\0&0&0&1\\0&0&1&0\end{pmatrix}\begin{pmatrix}0\\1\\0\\0\end{pmatrix} = \begin{pmatrix}0\\1\\0\\0\end{pmatrix} = \begin{pmatrix}1\\0\end{pmatrix}\otimes\begin{pmatrix}0\\1\end{pmatrix} = |01\rangle$$

Qubits and superposition

- Cbits are a special case of Qubits!
- A qubit is represented by $\binom{a}{b}$ where a and b are complex numbers such that $||a||^2 + ||b||^2 = 1$.
  - The cbit vectors $\binom{1}{0}$ and $\binom{0}{1}$ fit this definition.
- Example qubit values:

$$\begin{pmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{pmatrix} \quad \begin{pmatrix} \frac{1}{2} \\ \frac{\sqrt{3}}{2} \end{pmatrix} \quad \begin{pmatrix} -1 \\ 0 \end{pmatrix} \quad \begin{pmatrix} \frac{1}{\sqrt{2}} \\ \frac{-1}{\sqrt{2}} \end{pmatrix}$$

What does that mean?

What does that mean?

Superposition

The qubit is in a state of both $|0\rangle$ and $|1\rangle$. We can write this as:

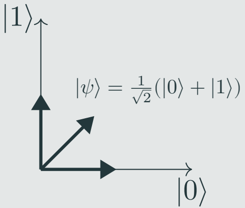
$$\begin{pmatrix} a \\ b \end{pmatrix} = a \begin{pmatrix} 1 \\ 0 \end{pmatrix} + b \begin{pmatrix} 0 \\ 1 \end{pmatrix} = a\,|0\rangle + b\,|1\rangle\,.$$

What does that mean?

Superposition

The qubit is in a state of both $|0\rangle$ and $|1\rangle$. We can write this as:

$$\begin{pmatrix} a \\ b \end{pmatrix} = a \begin{pmatrix} 1 \\ 0 \end{pmatrix} + b \begin{pmatrix} 0 \\ 1 \end{pmatrix} = a\,|0\rangle + b\,|1\rangle\,.$$



What does that mean?

Superposition

The qubit is in a state of both $|0\rangle$ and $|1\rangle$. We can write this as:

$$\begin{pmatrix} a \\ b \end{pmatrix} = a \begin{pmatrix} 1 \\ 0 \end{pmatrix} + b \begin{pmatrix} 0 \\ 1 \end{pmatrix} = a\,|0\rangle + b\,|1\rangle\,.$$

Amplitudes

a and b are called amplitudes. $||a||^2$ is the probability of the qubit being 0 when measured; $||b||^2$ is the probability of measuring 1.

What does that mean?

Superposition

The qubit is in a state of both $|0\rangle$ and $|1\rangle$. We can write this as:

$$\begin{pmatrix} a \\ b \end{pmatrix} = a \begin{pmatrix} 1 \\ 0 \end{pmatrix} + b \begin{pmatrix} 0 \\ 1 \end{pmatrix} = a\,|0\rangle + b\,|1\rangle\,.$$

Amplitudes

a and b are called amplitudes. $||a||^2$ is the probability of the qubit being 0 when measured; $||b||^2$ is the probability of measuring 1.

Measurement

The measurement of the qubit collapses its state. It will be in the state $|0\rangle$ if we measured 0 and $|1\rangle$ if we measured $1^\dagger$.

For example

The qubit $\begin{pmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{pmatrix}$ has a $\left|\left|\frac{1}{\sqrt{2}}\right|\right|^2 = \frac{1}{2}$ chance of collapsing to $|0\rangle$ or $|1\rangle$.

The qubit $\begin{pmatrix} 1 \\ 0 \end{pmatrix}$ has 100% chance of collapsing to $|0\rangle$, and $\begin{pmatrix} 0 \\ 1 \end{pmatrix}$ has a 100% chance of collapsing to $|1\rangle$.

Qubits and Superposition

- Multiple qubits are represented by the tensor product:
  $\begin{pmatrix} a \\ b \end{pmatrix} \otimes \begin{pmatrix} c \\ d \end{pmatrix} = \begin{pmatrix} ab \\ ad \\ bc \\ bd \end{pmatrix}$ with $||ac||^2 + ||ad||^2 + ||bc||^2 + ||bd||^2 = 1$.

- Multiple qubits are represented by the tensor product:
$$\begin{pmatrix} a \\ b \end{pmatrix} \otimes \begin{pmatrix} c \\ d \end{pmatrix} = \begin{pmatrix} ab \\ ad \\ bc \\ bd \end{pmatrix} \text{ with } ||ac||^2 + ||ad||^2 + ||bc||^2 + ||bd||^2 = 1.$$
- For example:

$$\begin{pmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{pmatrix} \otimes \begin{pmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{pmatrix} = \begin{pmatrix} \frac{1}{2} \\ \frac{1}{2} \\ \frac{1}{2} \\ \frac{1}{2} \end{pmatrix}; \quad \left\|\frac{1}{2}\right\|^2 = \frac{1}{4}; \quad \frac{1}{4} + \frac{1}{4} + \frac{1}{4} + \frac{1}{4} = 1$$

- Multiple qubits are represented by the tensor product:

$$\begin{pmatrix} a \\ b \end{pmatrix} \otimes \begin{pmatrix} c \\ d \end{pmatrix} = \begin{pmatrix} ab \\ ad \\ bc \\ bd \end{pmatrix} \text{ with } ||ac||^2 + ||ad||^2 + ||bc||^2 + ||bd||^2 = 1.$$

- For example:

$$\begin{pmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{pmatrix} \otimes \begin{pmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{pmatrix} = \begin{pmatrix} \frac{1}{2} \\ \frac{1}{2} \\ \frac{1}{2} \\ \frac{1}{2} \end{pmatrix}; \quad \left\|\frac{1}{2}\right\|^2 = \frac{1}{4}; \quad \frac{1}{4} + \frac{1}{4} + \frac{1}{4} + \frac{1}{4} = 1$$

$\rightarrow$ There's an equal chance (25%) of measuring $|00\rangle$, $|01\rangle$, $|10\rangle$, $|11\rangle$.

Operations on qubits

- We operate on qubits in the same way as on cbits: with matrices.
- All the operations we saw so far (bit flip, CNOT, etc...) work on qubits as well.
- In reality, the matrix operations model some device that manipulates the real qubits without measurement.
- Some gates only make sense in the quantum context...

- The Hadaramd gate puts a $|0\rangle$ or $|1\rangle$ bit into exact superposition: $H|0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$ and $H|1\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$.

The Hadamard gate

- The Hadaramd gate puts a $|0\rangle$ or $|1\rangle$ bit into exact superposition: $H|0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$ and $H|1\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$.

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

The Hadamard gate

- The Hadaramd gate puts a $|0\rangle$ or $|1\rangle$ bit into exact superposition: $H|0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$ and $H|1\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$.

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

- Note that $H^2 = HH = \mathbf{1}$ so $H^2|0\rangle = |0\rangle$ and $H^2|1\rangle = |1\rangle$.

The Hadamard gate

- The Hadaramd gate puts a $|0\rangle$ or $|1\rangle$ bit into exact superposition: $H|0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$ and $H|1\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$.

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

- Note that $H^2 = HH = \mathbf{1}$ so $H^2|0\rangle = |0\rangle$ and $H^2|1\rangle = |1\rangle$.
- This allows us to get out of superposition without measurement! So we can structure computations deterministically.

The Deutsch Oracle

The Deutsch oracle

- The Black Box is a deterministic function of one bit.

The Deutsch oracle

- The Black Box is a deterministic function of one bit.
- You can give it whatever input you want, and observe the output.

The Deutsch oracle

- The Black Box is a deterministic function of one bit.
- You can give it whatever input you want, and observe the output.
- How many queries do you need to determine the function on classical computer?

- The Black Box is a deterministic function of one bit.
- You can give it whatever input you want, and observe the output.
- How many queries do you need to determine the function on classical computer?
- And on a quantum computer?

- What if we want to determine whether the function is constant or variable?

- What if we want to determine whether the function is constant or variable?
- How many queries on a classical computer?

- What if we want to determine whether the function is constant or variable?
- How many queries on a classical computer?
- And on a quantum computer?

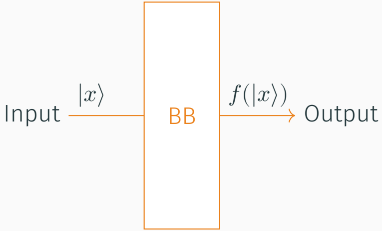
Problem: the `constant-0` and `constant-1` functions are non-reversible.

Before we can continue, we have to write them in a reversible way:

Problem: the `constant-0` and `constant-1` functions are non-reversible.

Before we can continue, we have to write them in a reversible way:

Before:

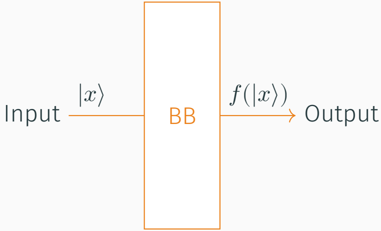


The Deutsch oracle

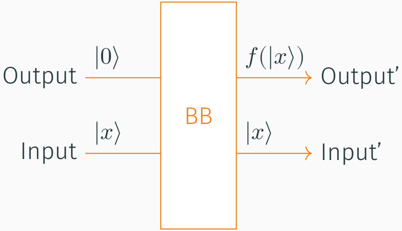
Problem: the `constant-0` and `constant-1` functions are non-reversible.

Before we can continue, we have to write them in a reversible way:

Before:

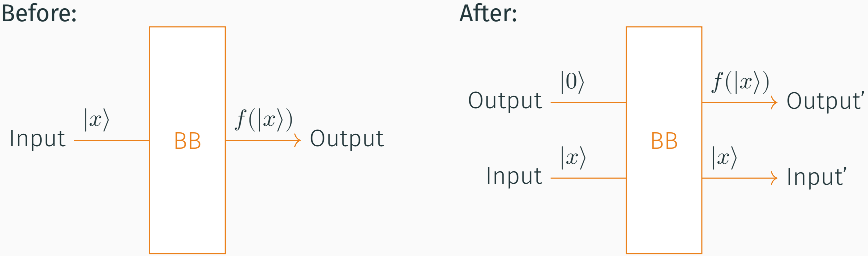


After:



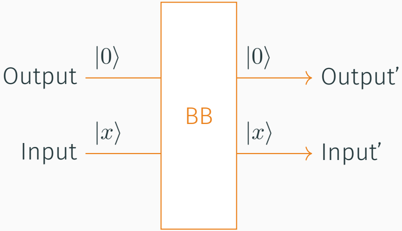
Problem: the `constant-0` and `constant-1` functions are non-reversible.

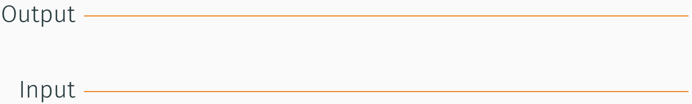
Before we can continue, we have to write them in a reversible way:



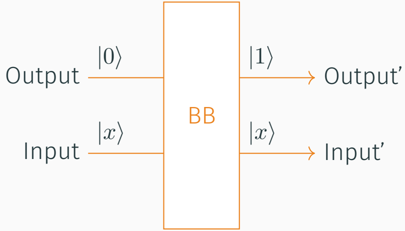
The black box leaves the input qubit unchanged, writing the output of the function to the output qubit.

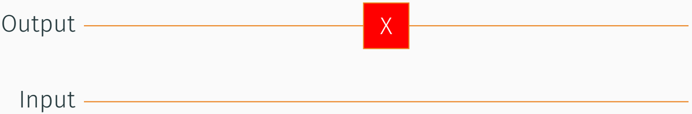
The Deutsch oracle: `constant-0`



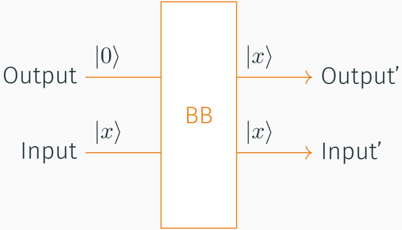


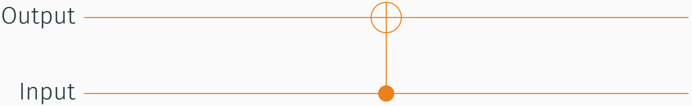
The Deutsch oracle: `constant-1`

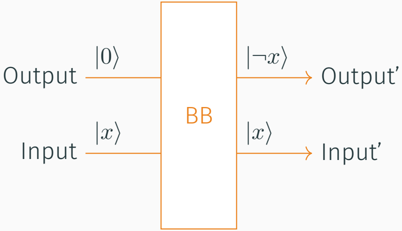


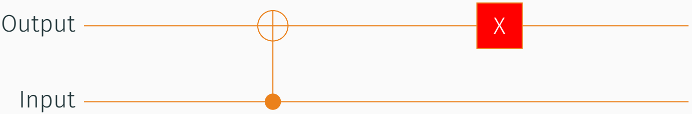


The Deutsch oracle: identity





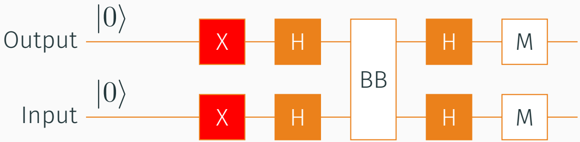

Output
$|0\rangle$
Input
$|x\rangle$
BB
$|\neg x\rangle$
Output'
$|x\rangle$
Input'


Output
Input
X

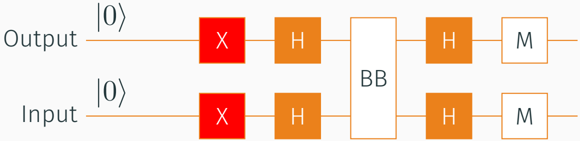
The Deutsch oracle: solution

- So how do we solve the problem in one operation?

- So how do we solve the problem in one operation?

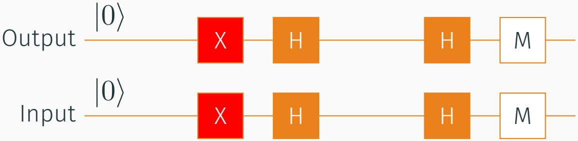


- So how do we solve the problem in one operation?



- If the black-box function is constant, we will measure $|11\rangle$.
- If the black-box function is variable, we will measure $|01\rangle$.

The Deutsch oracle: check for `constant-0`

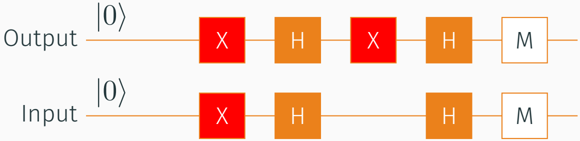


$$\text{Output'} = \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{pmatrix} \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix}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

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

Output is $|11\rangle$.

The Deutsch oracle: check for `constant-1`

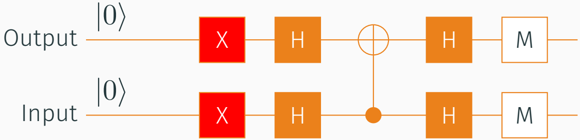


$$\text{Output'} = \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{pmatrix} \begin{pmatrix}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

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

Output is $|11\rangle$.

The Deutsch oracle: check for the identity

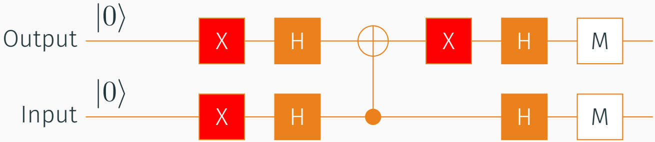


$$C\left(\begin{pmatrix}\frac{1}{\sqrt{2}}\\-\frac{1}{\sqrt{2}}\end{pmatrix}\otimes\begin{pmatrix}\frac{1}{\sqrt{2}}\\-\frac{1}{\sqrt{2}}\end{pmatrix}\right)=C\begin{pmatrix}\frac{1}{2}\\-\frac{1}{2}\\-\frac{1}{2}\\\frac{1}{2}\end{pmatrix}=\frac{1}{2}\begin{pmatrix}1&0&0&0\\0&1&0&0\\0&0&0&1\\0&0&1&0\end{pmatrix}\begin{pmatrix}1\\-1\\-1\\1\end{pmatrix}=\frac{1}{2}\begin{pmatrix}1\\-1\\1\\-1\end{pmatrix}$$

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

Output (after applying Hadamard gate): $|01\rangle$.

The Deutsch oracle: check for negation



Left as an exercise :)

Output: $|01\rangle$.

- We managed to determine whether the function is constant in one query!
- We magnified the difference between categories (CNOT gate) and neutralized the difference within the categories (NOT gate).
- There is a generalization of this to n-bit black boxes (Deutsch-Josza problem).
- It was an inspiration for Shor's algorithm!

Bonus Topic - Quantum Entanglement

Quantum entanglement

- If the product state of two qubits cannot be factored, they are said to be entangled

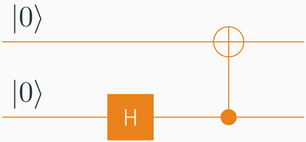
$$\begin{pmatrix} \frac{1}{\sqrt{2}} \\ 0 \\ 0 \\ \frac{1}{\sqrt{2}} \end{pmatrix} = \begin{pmatrix} a \\ b \end{pmatrix} \otimes \begin{pmatrix} c \\ d \end{pmatrix}$$

$$ac = \frac{1}{\sqrt{2}}$$
$$ad = 0$$
$$bc = 0$$
$$bd = \frac{1}{\sqrt{2}}$$

- The system of equations has no solution, so we cannot factor the quantum state.
- This state has a 50% chance of collapsing to $|00\rangle$ and 50% chance of collapsing to $|11\rangle$.

Quantum entanglement

To reach an entangled state is quite simple:



$$CH_1\left(\begin{pmatrix}1\\0\end{pmatrix}\otimes\begin{pmatrix}1\\0\end{pmatrix}\right) = C\left(\begin{pmatrix}\frac{1}{\sqrt{2}}\\\frac{1}{\sqrt{2}}\end{pmatrix}\otimes\begin{pmatrix}1\\0\end{pmatrix}\right) = \begin{pmatrix}1 & 0 & 0 & 0\\0 & 1 & 0 & 0\\0 & 0 & 0 & 1\\0 & 0 & 1 & 0\end{pmatrix}\begin{pmatrix}\frac{1}{\sqrt{2}}\\0\frac{1}{\sqrt{2}}\\0\end{pmatrix} = \begin{pmatrix}\frac{1}{\sqrt{2}}\\0\\0\\\frac{1}{\sqrt{2}}\end{pmatrix}$$

Quantum entanglement - consequences

- The qubits are forced to be equal (both are either 0 or 1)!
- If we measure one of them, the state of the other will collapse to be equal to the one we measured.
- This can happen even if they are very very far away from each other!
- The value is not predetermined.
- We cannot use this to transmit information, however.