Intro to Quantum Computing

Michael Williams de la Bastida

UCL

August 22, 2023

Aims

- I can’t make you an expert in a day (sorry!)
- Be able to build and analyse quantum circuits
- Understand how quantum computers work
- Understand how quantum computing fits in
- Feel confident to learn more on your own

Outline

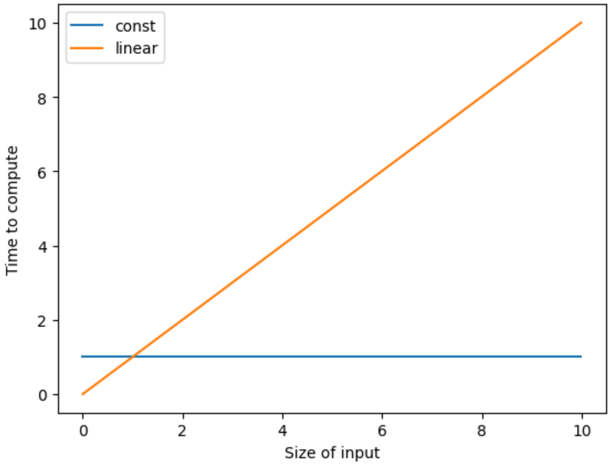
Menti

Before we start, join the mentimeter interactive session.

Why bother?

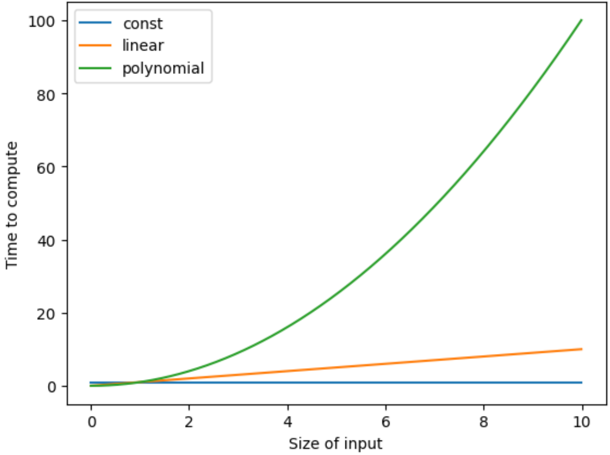
Complexity

- Check odd or even
- Add two numbers



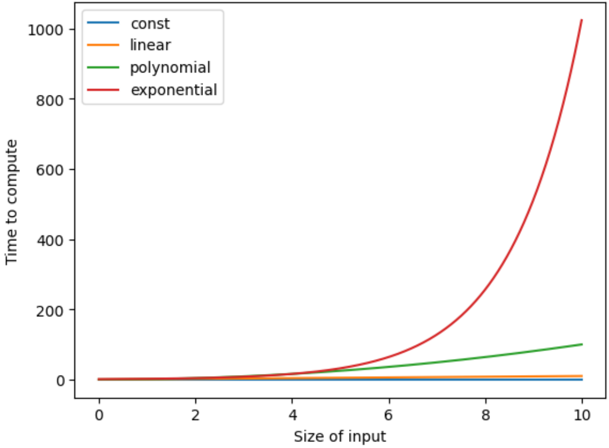
Complexity

- Check odd or even
- Add two numbers
- Multiply two numbers



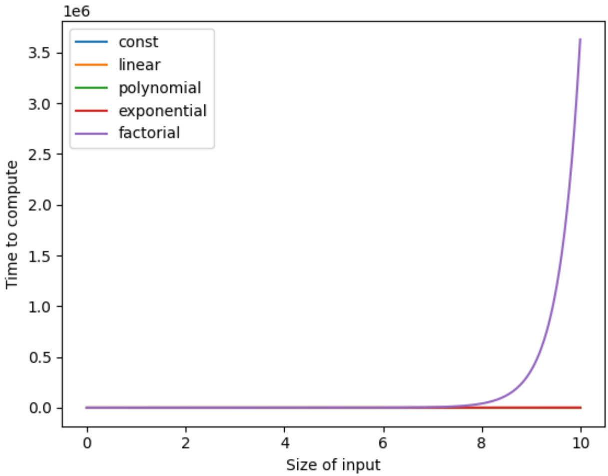
Complexity

- Check odd or even
- Add two numbers
- Multiply two numbers
- Factorise a number



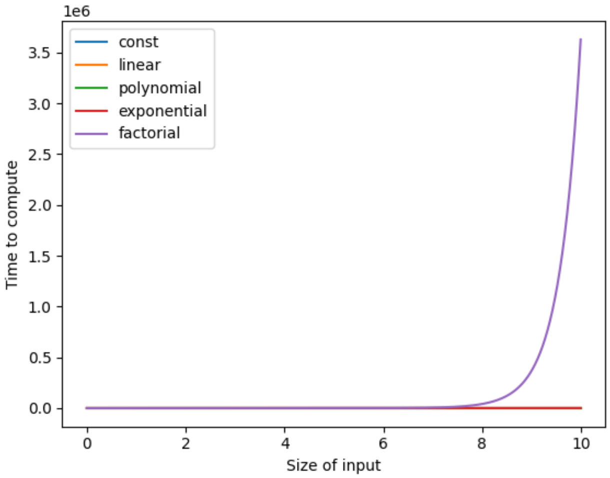
Complexity

- Check odd or even
- Add two numbers
- Multiply two numbers
- Factorise a number
- Find the quickest route for a delivery truck



Complexity

- Find the quickest route for a delivery truck
- Find all possible energy states of a molecule



Feynman

Simulating Physics with Computers

Richard P. Feynman

Department of Physics, California Institute of Technology, Pasadena, California 91107

Received May 7, 1981

1. INTRODUCTION

On the program it says this is a keynote speech—and I don't know what a keynote speech is. I do not intend in any way to suggest what should be in this meeting as a keynote of the subjects or anything like that. I have my own things to say and to talk about and there's no implication that anybody needs to talk about the same thing or anything like it. So what I want to talk about is what Mike Dertouzos suggested that nobody would talk about. I want to talk about the problem of simulating physics with computers and I mean that in a specific way which I am going to explain. The reason for doing this is something that I learned about from Ed Fredkin, and my entire interest in the subject has been inspired by him. It has to do with learning something about the possibilities of computers, and also something about possibilities in physics. If we suppose that we know all

Uses for Quantum Computers

- Design Pharmaceuticals

Uses for Quantum Computers

- Design Pharmaceuticals
- Save energy

Uses for Quantum Computers

- Design Pharmaceuticals
- Save energy
- Encryption / Decryption

Uses for Quantum Computers

- Design Pharmaceuticals
- Save energy
- Encryption / Decryption
- Machine Learning

Uses for Quantum Computers

- Design Pharmaceuticals
- Save energy
- Encryption / Decryption
- Machine Learning
- More we don’t know yet!

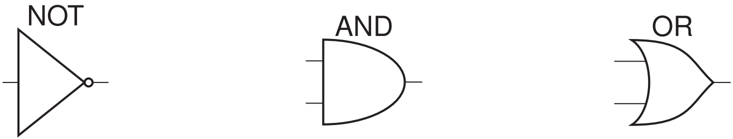
Computational Logic

What does a computer do?

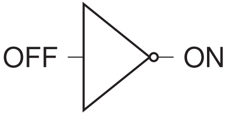
We're going to approach quantum computing from the computing side, and add in quantum mechanics as we go.

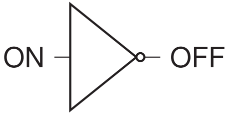
1. Arithmetic / Logic
2. Read / Write Memory
3. Control Flow

Logic Gates



NOT Gate

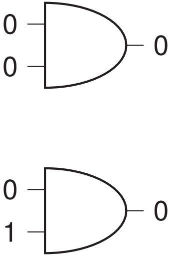




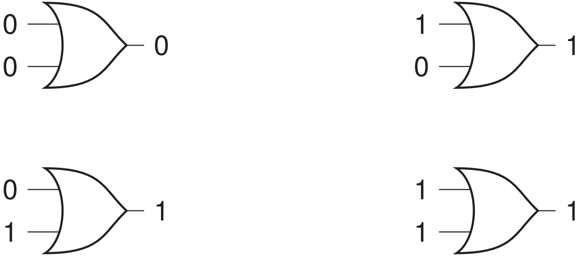
NOT Gate

NOT Gate

AND Gate

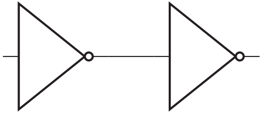


OR Gate



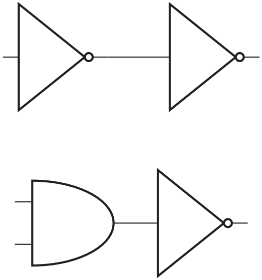
Combining gates

We can combine gates into circuits.



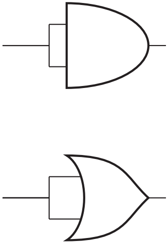
Combining gates

We can combine gates into circuits.



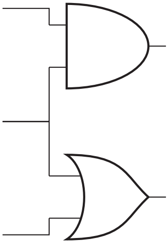
FANOUT

We can split the wire in two.

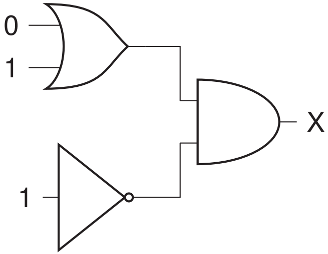


FANOUT

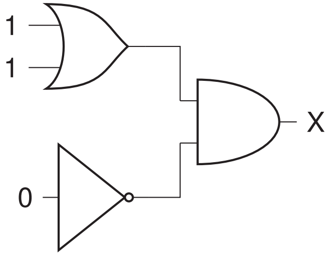
This allows us to share inputs between gates.



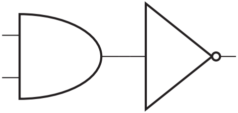
First Circuit

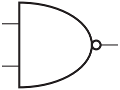


First Circuit



Universality of NAND





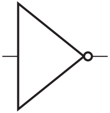
Theorem

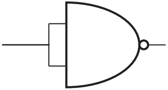
All logical operations can be completed with the NAND gate, they are said to be universal.

Try to put these together to make something which behaves the same way as NOT, AND and OR gate.[1]

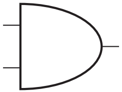
[1] (Hint: You can split lines or cross them over!)

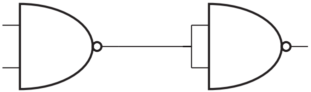
Universality of NAND



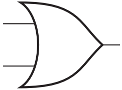


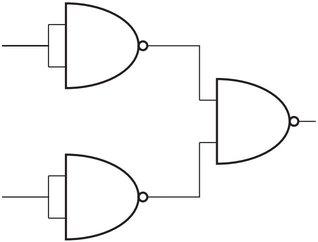
Universality of NAND





Universality of NAND





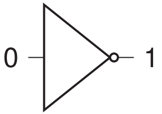
States

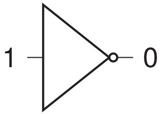
For binary systems we can define two states of each input.

OFF = $|0\rangle$ ON = $|1\rangle$

Operators

Rather than draw the gate we can use its name, like we would for a function.

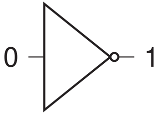


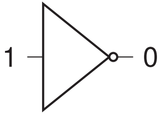


$$NOT\,|0\rangle = |1\rangle \qquad NOT\,|1\rangle = |0\rangle$$

Operators

Rather than draw the gate we can use its name, like we would for a function.





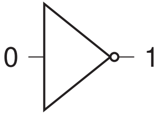
$NOT\,|0\rangle = |1\rangle \qquad NOT\,|1\rangle = |0\rangle$

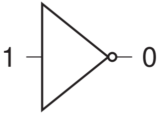
We can go one step further and use a symbol.

$\hat{N}\,|0\rangle = |1\rangle \qquad \hat{N}\,|1\rangle = |0\rangle$

Operators

Rather than draw the gate we can use its name, like we would for a function.





$$NOT\,|0\rangle = |1\rangle \qquad NOT\,|1\rangle = |0\rangle$$

We can go one step further and use a symbol.

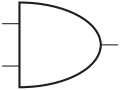
$$\hat{N}\,|0\rangle = |1\rangle \qquad \hat{N}\,|1\rangle = |0\rangle$$

Gates are represented by symbols with hats, called *operators*.

Multi-bit Operators

We can apply multi-bit operators to multi-bit states.

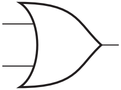
$$\hat{A}|00\rangle = |0\rangle$$
$$\hat{A}|01\rangle = |0\rangle$$
$$\hat{A}|10\rangle = |0\rangle$$
$$\hat{A}|11\rangle = |1\rangle$$



Multi-bit Operators

We can apply multi-bit operators to multi-bit states.

$$\hat{O}|00\rangle = |0\rangle$$
$$\hat{O}|01\rangle = |1\rangle$$
$$\hat{O}|10\rangle = |1\rangle$$
$$\hat{O}|11\rangle = |1\rangle$$



Rules for Operators

Operators act to their right.

$$\hat{N}\,|0\rangle = |1\rangle$$

Rules for Operators

Operators act to their right.

$$\hat{N}|0\rangle = |1\rangle$$

$$\hat{N}\hat{A}|11\rangle = \hat{N}\left(\hat{A}|11\rangle\right) = \hat{N}|1\rangle = |0\rangle$$

Rules for Operators

Operators only act on states with the correct size of input

$$\hat{A}\,|0\rangle = 0$$

$$\hat{N}\,|11\rangle = 0$$

Note that this is 0 and not a state at all!

Rules for Operators

Operators can be applied multiple times.

$$\hat{N}\hat{N}|1\rangle = \hat{N}|0\rangle = |1\rangle$$

Identity Operator

We define the Identity Operator $\hat{\mathbb{I}}$ which does nothing to change the state.

$$\hat{\mathbb{I}}\,|0\rangle = |0\rangle \qquad \hat{\mathbb{I}}\,|1\rangle = |1\rangle$$

This is equivalent to a wire in our circuit.

0 ———————— 0

Identity Operator

This allows us write about operators independent of states.

$$\hat{N}\hat{N}|0\rangle = \hat{N}|1\rangle = \hat{\mathbb{I}}|0\rangle$$

$$\hat{N}\hat{N}|1\rangle = \hat{N}|0\rangle = \hat{\mathbb{I}}|1\rangle$$

$$\hat{N}\hat{N} = \hat{\mathbb{I}}$$

States

For binary systems we can define two states of each input.

OFF = $|0\rangle$ ON = $|1\rangle$

States

For binary systems we can define two states of each input.

OFF = $|0\rangle$ ON = $|1\rangle$

The states have a property called the **inner product**.

States

For binary systems we can define two states of each input.

OFF = $|0\rangle$ ON = $|1\rangle$

The states have a property called the **inner product**.

$$\langle 0|0\rangle = \langle 1|1\rangle = 1$$

$$\langle 0|1\rangle = \langle 1|0\rangle = 0$$

States

For binary systems we can define two states of each input.

OFF = $|0\rangle$ ON = $|1\rangle$

The states have a property called the **inner product**.

$$\langle 0|0\rangle = \langle 1|1\rangle = 1$$

$$\langle 0|1\rangle = \langle 1|0\rangle = 0$$

This is essentially the question 'are these the same state?'

Multi-bit States

To combine single bit states into larger sizes we use a *tensor product*.[2]

$$|x\rangle \otimes |y\rangle = |xy\rangle$$

$$\text{e.g. } |0\rangle \otimes |1\rangle = |01\rangle$$

[2] We don't want to multiply the values, just list them in order!

Multi-bit Inner Product

We can work out the inner product using the single bit definitions.

$$\langle ab|cd\rangle$$

$$= (\langle a| \otimes \langle b|)(|c\rangle \otimes |d\rangle)$$

$$= \langle a|c\rangle \otimes \langle b|d\rangle$$

Multi-bit Inner Product

We can work out the inner product using the single bit definitions.

$$\langle ab|cd\rangle$$

$$= (\langle a| \otimes \langle b|)(|c\rangle \otimes |d\rangle)$$

$$= \langle a|c\rangle \otimes \langle b|d\rangle$$

Similar states give 1:

$$\langle 00|00\rangle = 1$$

Different states give 0:

$$\langle 00|11\rangle = 0$$

$$\langle 01|11\rangle = 0$$

$$\langle 10|11\rangle = 0$$

Rules for Operators

However, we can use the tensor product to make larger operators.

$$\hat{\mathbb{I}} \otimes \hat{N} |11\rangle = (\hat{\mathbb{I}} \otimes \hat{N})(|1\rangle \otimes |1\rangle)$$

$$= \hat{\mathbb{I}} |1\rangle \otimes \hat{N} |1\rangle$$

$$= |1\rangle \otimes |0\rangle$$

$$= |10\rangle$$

Similarly $(\hat{N} \otimes \hat{\mathbb{I}}) |11\rangle = |01\rangle$ and $(\hat{N} \otimes \hat{N}) |11\rangle = |00\rangle$

Outer Product

So far we've learned the **tensor product**

$$|0\rangle \otimes |1\rangle = |01\rangle$$

Outer Product

So far we've learned the **tensor product**

$$|0\rangle \otimes |1\rangle = |01\rangle$$

and the **inner product**

$$\langle 0|1\rangle = 0$$

Outer Product

So far we've learned the **tensor product**

$$|0\rangle \otimes |1\rangle = |01\rangle$$

and the **inner product**

$$\langle 0|1\rangle = 0$$

Can you guess what an **outer product** looks like?

Outer Product

So far we've learned the **tensor product**

$$|0\rangle \otimes |1\rangle = |01\rangle$$

and the **inner product**

$$\langle 0|1\rangle = 0$$

Can you guess what an **outer product** looks like?

$$|0\rangle \langle 1|$$

Outer Product

The outer product can be used to change a state. We start with $|1\rangle$.

$$|0\rangle \langle 1| \; |1\rangle$$

Outer Product

The outer product can be used to change a state. We start with $|1\rangle$.

$$|0\rangle\langle 1|\ |1\rangle$$

$$= |0\rangle \langle 1|1\rangle$$

Outer Product

The outer product can be used to change a state. We start with $|1\rangle$.

$$|0\rangle \langle 1| \; |1\rangle$$

$$= |0\rangle \langle 1|1\rangle$$

$$= |0\rangle \times 1$$

Outer Product

The outer product can be used to change a state. We start with $|1\rangle$.

$$|0\rangle \langle 1| \; |1\rangle$$

$$= |0\rangle \langle 1|1\rangle$$

$$= |0\rangle \times 1$$

The state has been changed from $|1\rangle$ to $|0\rangle$!

$$|OUT\rangle \langle IN|$$

Operators from Outer Products

We already saw that $|0\rangle\langle 1|$ changed 1 to 0.

$|1\rangle\langle 0|$ does the opposite, it changes 0 to 1.

Operators from Outer Products

We already saw that $|0\rangle\langle 1|$ changed 1 to 0.

$|1\rangle\langle 0|$ does the opposite, it changes 0 to 1.

These are the conditions for the NOT gate and $\hat{N}$ operator.

$$\hat{N} = |0\rangle\langle 1| + |1\rangle\langle 0|$$

Operators from Outer Products

Lets see how he $\hat{N}$ operator works.

$$\hat{N} = |0\rangle \langle 1| + |1\rangle \langle 0|$$

$$\hat{N} |0\rangle = (|0\rangle \langle 1| + |1\rangle \langle 0|) |0\rangle$$

Operators from Outer Products

Lets see how he $\hat{N}$ operator works.

$$\hat{N} = |0\rangle \langle 1| + |1\rangle \langle 0|$$

$$\hat{N} |0\rangle = (|0\rangle \langle 1| + |1\rangle \langle 0|) |0\rangle$$

$$\hat{N} |0\rangle = |0\rangle \langle 1|0\rangle + |1\rangle \langle 0|0\rangle$$

Operators from Outer Products

Lets see how he $\hat{N}$ operator works.

$$\hat{N} = |0\rangle \langle 1| + |1\rangle \langle 0|$$

$$\hat{N}|0\rangle = (|0\rangle \langle 1| + |1\rangle \langle 0|)|0\rangle$$

$$\hat{N}|0\rangle = |0\rangle \langle 1|0\rangle + |1\rangle \langle 0|0\rangle$$

$$\hat{N}|0\rangle = |0\rangle \times 0 + |1\rangle \times 1$$

Operators from Outer Products

Lets see how he $\hat{N}$ operator works.

$$\hat{N} = |0\rangle\langle 1| + |1\rangle\langle 0|$$

$$\hat{N}|0\rangle = (|0\rangle\langle 1| + |1\rangle\langle 0|)|0\rangle$$

$$\hat{N}|0\rangle = |0\rangle\langle 1|0\rangle + |1\rangle\langle 0|0\rangle$$

$$\hat{N}|0\rangle = |0\rangle \times 0 + |1\rangle \times 1$$

$$\hat{N}|0\rangle = |1\rangle$$

The AND Operator

Lets build up the $\hat{A}$ operator.

Input	Output	Outer Product
$\|00\rangle$	$\|0\rangle$	$\|0\rangle\langle 00\|$

The AND Operator

Lets build up the $\hat{A}$ operator.

Input	Output	Outer Product
$\|00\rangle$	$\|0\rangle$	$\|0\rangle \langle 00\|$
$\|01\rangle$	$\|0\rangle$	$\|0\rangle \langle 01\|$

The AND Operator

Lets build up the $\hat{A}$ operator.

Input	Output	Outer Product
$\|00\rangle$	$\|0\rangle$	$\|0\rangle\langle 00\|$
$\|01\rangle$	$\|0\rangle$	$\|0\rangle\langle 01\|$
$\|10\rangle$	$\|0\rangle$	$\|0\rangle\langle 10\|$

The AND Operator

Lets build up the $\hat{A}$ operator.

Input	Output	Outer Product
$\|00\rangle$	$\|0\rangle$	$\|0\rangle \langle 00\|$
$\|01\rangle$	$\|0\rangle$	$\|0\rangle \langle 01\|$
$\|10\rangle$	$\|0\rangle$	$\|0\rangle \langle 10\|$
$\|11\rangle$	$\|1\rangle$	$\|1\rangle \langle 11\|$

The AND Operator

Lets build up the $\hat{A}$ operator.

Input	Output	Outer Product
$\|00\rangle$	$\|0\rangle$	$\|0\rangle\langle 00\|$
$\|01\rangle$	$\|0\rangle$	$\|0\rangle\langle 01\|$
$\|10\rangle$	$\|0\rangle$	$\|0\rangle\langle 10\|$
$\|11\rangle$	$\|1\rangle$	$\|1\rangle\langle 11\|$

$$\hat{A} = |0\rangle\langle 00| + |0\rangle\langle 01| + |0\rangle\langle 10| + |1\rangle\langle 11|$$

The AND Operator

Lets build up the $\hat{A}$ operator.

Input	Output	Outer Product
$\|00\rangle$	$\|0\rangle$	$\|0\rangle\langle 00\|$
$\|01\rangle$	$\|0\rangle$	$\|0\rangle\langle 01\|$
$\|10\rangle$	$\|0\rangle$	$\|0\rangle\langle 10\|$
$\|11\rangle$	$\|1\rangle$	$\|1\rangle\langle 11\|$

$$\hat{A} = |0\rangle\langle 00| + |0\rangle\langle 01| + |0\rangle\langle 10| + |1\rangle\langle 11|$$

Notice that the states on the right have two bits but the states on the left only have one!

The OR Operator

Lets build up the $\hat{O}$ operator.

Input	Output	Outer Product
$\|00\rangle$	$\|0\rangle$	$\|0\rangle\langle 00\|$
$\|01\rangle$	$\|1\rangle$	$\|1\rangle\langle 01\|$
$\|10\rangle$	$\|1\rangle$	$\|1\rangle\langle 10\|$
$\|11\rangle$	$\|1\rangle$	$\|1\rangle\langle 11\|$

$$\hat{O} = |0\rangle\langle 00| + |1\rangle\langle 01| + |1\rangle\langle 10| + |1\rangle\langle 11|$$

Quantum States

Postulates of QM

Postulates of QM

Postulate 1

Any isolated physical system is completely described by a state vector.

A two state quantum system is called a **Qubit**.

$$|\Psi\rangle = \alpha |0\rangle + \beta |1\rangle$$

Where $\alpha, \beta \in \mathbb{C}$ (they are complex numbers) [3]

[3] where $\chi = a + ib$ is a complex number, $\chi^* = a - ib$.

Postulates of QM

Postulate 1

Any isolated physical system is completely described by a state vector.

A two state quantum system is called a **Qubit**.

$$|\Psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle$$

Where α, $\beta \in \mathbb{C}$ (they are complex numbers) [3]

$$\langle\Psi| = \langle 0|\,\alpha^* + \langle 1|\,\beta^*$$

[3] where $\chi = a + ib$ is a complex number, $\chi^* = a - ib$.

Postulates of QM

Postulate 1

Any isolated physical system is completely described by a state vector.

A two state quantum system is called a **Qubit**.

$$|\Psi\rangle = \alpha|0\rangle + \beta|1\rangle$$

Where $\alpha, \beta \in \mathbb{C}$ (they are complex numbers) [3]

$$\langle\Psi| = \langle 0|\alpha^* + \langle 1|\beta^*$$

$$\langle\Psi|\Psi\rangle = |\alpha|^2\langle 0|0\rangle + \alpha^*\beta\langle 0|1\rangle + \alpha\beta^*\langle 1|0\rangle + |\beta|^2\langle 1|1\rangle$$

[3] where $\chi = a + ib$ is a complex number, $\chi^* = a - ib$.

Postulates of QM

Postulate 1

Any isolated physical system is completely described by a state vector.

A two state quantum system is called a **Qubit**.

$$|\Psi\rangle = \alpha |0\rangle + \beta |1\rangle$$

Where α, $\beta \in \mathbb{C}$ (they are complex numbers) [3]

$$\langle\Psi| = \langle 0| \alpha^* + \langle 1| \beta^*$$

$$\langle\Psi|\Psi\rangle = |\alpha|^2 \langle 0|0\rangle + \alpha^* \beta \langle 0|1\rangle + \alpha\beta^* \langle 1|0\rangle + |\beta|^2 \langle 1|1\rangle$$

$$1 = |\alpha|^2 + |\beta|^2$$

[3] where $\chi = a + ib$ is a complex number, $\chi^* = a - ib$.

Inner Product of Different States

Because we have amplitudes for each state $|0\rangle$ and $|1\rangle$ the inner products of two quantum states have values between 0 and 1.

$$|\Psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle$$
$$|\Phi\rangle = \gamma\,|0\rangle + \delta\,|1\rangle$$

$$\langle\Psi|\Phi\rangle = \alpha^*\gamma\,\langle 0|0\rangle + \beta^*\delta\,\langle 1|1\rangle$$

$$\langle\Psi|\Phi\rangle = \alpha^*\gamma + \beta^*\delta$$

$$0 \leqslant \langle\Psi|\Phi\rangle \leqslant 1$$

Postulates of QM

Postulate 1

Any isolated physical system is completely described by a state vector.

This is true not just for qubits, but for any size of quantum system.

$$|\Psi\rangle = c_0 |\psi_0\rangle + c_1 |\psi_1\rangle + \cdots + c_n |\psi_n\rangle$$

Where $\{c_0, c_1, \ldots, c_n\} \in \mathbb{C}$

$$|c_0|^2 + |c_1|^2 + \cdots + |c_n|^2 = 1$$

Postulates of QM

Postulates of QM

Postulate 2

Evolution of a closed system is by *unitary transformation*.

Unitary here means that the inner product $\langle\Psi|\Psi\rangle$ is unchanged.

$$\hat{U}\,|\Psi\rangle = |\Phi\rangle$$

$$\langle\Phi|\Phi\rangle = 1$$

Postulates of QM

1. State Vector
2. Time Evolution
3. **Measurements**
4. Composite Systems

Postulates of QM

Postulate 3

Measurements of quantum systems project the system onto one of its states.

For a single qubit state $|\Psi\rangle = \alpha |0\rangle + \beta |1\rangle$, a measurement of the qubit will transform it into the state $|0\rangle$ with probability $|\alpha|^2$ or $|1\rangle$ with probability $|\beta|^2$.

Postulates of QM

Postulates of QM

Postulate 4

The state vector of a composite system is the tensor product of the component systems state vectors.

$$|\Psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle$$

$$|\Phi\rangle = \gamma\,|0\rangle + \delta\,|1\rangle$$

$$|\Psi\rangle \otimes |\Phi\rangle = \alpha\gamma\,|00\rangle + \alpha\delta\,|01\rangle + \beta\gamma\,|10\rangle + \beta\delta\,|11\rangle$$

Quantum Entanglement

Definition

To objects are entangled when their joint state $|\Psi\rangle$ cannot be expressed as the product of two states $|\psi\rangle \otimes |\psi\rangle$.

$$|\Psi\rangle \neq |\psi\rangle \otimes |\psi\rangle$$

Quantum Entanglement

Definition

To objects are entangled when their joint state $|\Psi\rangle$ cannot be expressed as the product of two states $|\psi\rangle \otimes |\psi\rangle$.

$$|\Psi\rangle \neq |\psi\rangle \otimes |\psi\rangle$$

Exercise

Show that the state $|\Psi\rangle = \frac{1}{\sqrt{2}}|00\rangle + \frac{1}{\sqrt{2}}|11\rangle$ is entangled by proving there are no values of of α_0, α_1, β_0, β_1 such that:

$$|\Psi\rangle = (\alpha_0 |0\rangle + \beta_0 |1\rangle)(\alpha_1 |0\rangle + \beta_1 |1\rangle)$$

Number of Parameters

In the example where we combine two qubits

$$|\Psi\rangle \otimes |\Phi\rangle = \alpha\gamma\,|00\rangle + \alpha\delta\,|01\rangle + \beta\gamma\,|10\rangle + \beta\delta\,|11\rangle$$

We need 4 terms to describe the state. If we take the tensor product with another qubit:

Number of Parameters

In the example where we combine two qubits

$$|\Psi\rangle \otimes |\Phi\rangle = \alpha\gamma\,|00\rangle + \alpha\delta\,|01\rangle + \beta\gamma\,|10\rangle + \beta\delta\,|11\rangle$$

We need 4 terms to describe the state. If we take the tensor product with another qubit:

$$|\chi\rangle = \nu\,|0\rangle + \mu\,|1\rangle$$

Number of Parameters

In the example where we combine two qubits

$$|\Psi\rangle \otimes |\Phi\rangle = \alpha\gamma\,|00\rangle + \alpha\delta\,|01\rangle + \beta\gamma\,|10\rangle + \beta\delta\,|11\rangle$$

We need 4 terms to describe the state. If we take the tensor product with another qubit:

$$|\chi\rangle = \nu\,|0\rangle + \mu\,|1\rangle$$

$$|\Psi\rangle \otimes |\Phi\rangle \otimes |\chi\rangle = \nu\alpha\gamma\,|000\rangle + \nu\alpha\delta\,|001\rangle + \nu\beta\gamma\,|010\rangle + \nu\beta\delta\,|011\rangle + \mu\alpha\gamma\,|100\rangle + \mu\alpha\delta\,|101\rangle + \mu\beta\gamma\,|110\rangle + \mu\beta\delta\,|111\rangle$$

Number of Parameters

In the example where we combine two qubits

$$|\Psi\rangle \otimes |\Phi\rangle = \alpha\gamma\,|00\rangle + \alpha\delta\,|01\rangle + \beta\gamma\,|10\rangle + \beta\delta\,|11\rangle$$

We need 4 terms to describe the state. If we take the tensor product with another qubit:

$$|\chi\rangle = \nu\,|0\rangle + \mu\,|1\rangle$$

$$|\Psi\rangle \otimes |\Phi\rangle \otimes |\chi\rangle = \nu\alpha\gamma\,|000\rangle + \nu\alpha\delta\,|001\rangle + \nu\beta\gamma\,|010\rangle + \nu\beta\delta\,|011\rangle + \mu\alpha\gamma\,|100\rangle + \mu\alpha\delta\,|101\rangle + \mu\beta\gamma\,|110\rangle + \mu\beta\delta\,|111\rangle$$

Now there are 8 terms!

Generally, the number of terms is 2^N, where N is the number of qubits.

Too many states

The state/operator notation works great for logical circuits. Inputs and Output are states, and gates are operators which transform the states.

It also works really well for representing quantum states.

However, it gets a little hard to work with for larger states.

State Vectors

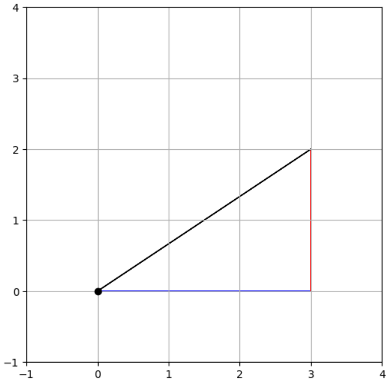
We can handle complex states more easily if we switch to expressing states using vector notation.

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

Vectors

Vectors are often used to point to a location in space.

For instance, here's a representation of the vector $\begin{pmatrix} 3 \\ 2 \end{pmatrix}$



Bloch Sphere

Vectors can be thought of as arrows with a certain length and direction.
Quantum states can be represented by a point on the the **Bloch Sphere**.

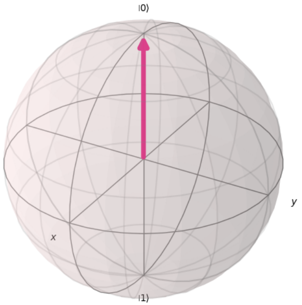


Figure: $|\Psi\rangle = |0\rangle$

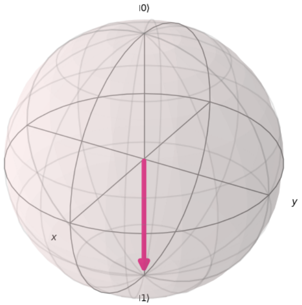


Figure: $|\Psi\rangle = |1\rangle$

We'll explore the the Bloch sphere in the practical parts of the day.

Rules for Vectors

With two vectors $|x\rangle = \begin{pmatrix} a_0 \\ b_0 \end{pmatrix}$ and $|y\rangle = \begin{pmatrix} a_1 \\ b_1 \end{pmatrix}$, we define:

Addition

$$|x\rangle + |y\rangle = \begin{pmatrix} a_0 + a_1 \\ b_0 + b_1 \end{pmatrix}$$

Scalar Multiplication

$$n * |x\rangle = \begin{pmatrix} n * a_0 \\ n * b_0 \end{pmatrix}$$

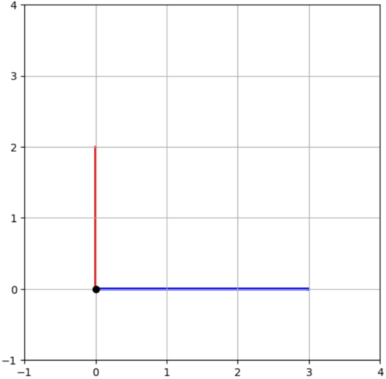
Inner Product[4]

$$\langle x|y\rangle = a_0^* a_1 + b_0^* b_1$$

[4] if $z = x + iy$ then $z^* = x - iy$

Vector Addition

The example we saw already illustrated vector addition.

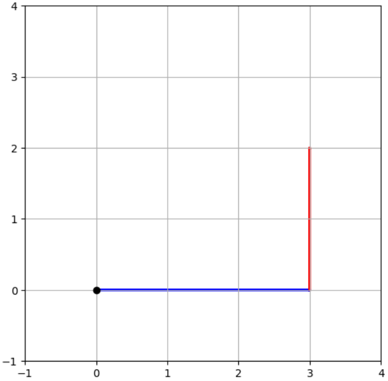


This shows the addition:

$$\begin{pmatrix} 3 \\ 0 \end{pmatrix} + \begin{pmatrix} 0 \\ 2 \end{pmatrix}$$

Vector Addition

The example we saw already illustrated vector addition.



This shows the addition:

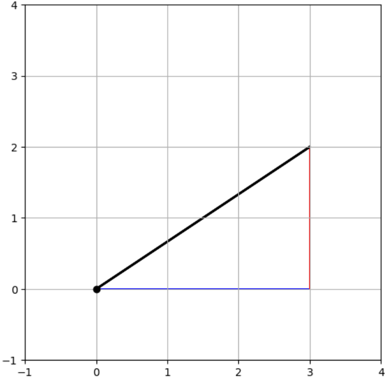
$$\begin{pmatrix} 3 \\ 0 \end{pmatrix} + \begin{pmatrix} 0 \\ 2 \end{pmatrix}$$

Vector Addition

The example we saw already illustrated vector addition.

This shows the addition:

$$\begin{pmatrix} 3 \\ 0 \end{pmatrix} + \begin{pmatrix} 0 \\ 2 \end{pmatrix}$$



Rules for Vectors

With two vectors $|x\rangle = \begin{pmatrix} a_0 \\ b_0 \end{pmatrix}$ and $|y\rangle = \begin{pmatrix} a_1 \\ b_1 \end{pmatrix}$, we define:

Addition

$$|x\rangle + |y\rangle = \begin{pmatrix} a_0 + a_1 \\ b_0 + b_1 \end{pmatrix}$$

Scalar Multiplication

$$n * |x\rangle = \begin{pmatrix} n * a_0 \\ n * b_0 \end{pmatrix}$$

Inner Product[4]

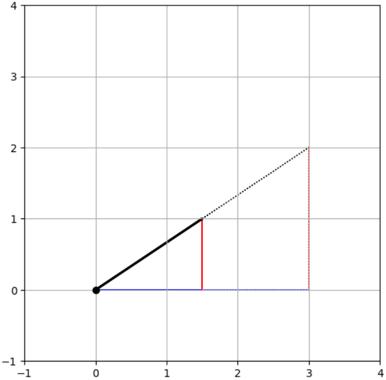
$$\langle x|y\rangle = a_0^* a_1 + b_0^* b_1$$

[4] if $z = x + iy$ then $z^* = x - iy$

Vector Scaling

We can scale this vector too.

$$\frac{1}{2} * \begin{pmatrix} 3 \\ 2 \end{pmatrix} = \begin{pmatrix} 1.5 \\ 1 \end{pmatrix}$$



Rules for Vectors

With two vectors $|x\rangle = \begin{pmatrix} a_0 \\ b_0 \end{pmatrix}$ and $|y\rangle = \begin{pmatrix} a_1 \\ b_1 \end{pmatrix}$, we define:

Addition

$$|x\rangle + |y\rangle = \begin{pmatrix} a_0 + a_1 \\ b_0 + b_1 \end{pmatrix}$$

Scalar Multiplication

$$n * |x\rangle = \begin{pmatrix} n * a_0 \\ n * b_0 \end{pmatrix}$$

Inner Product[4]

$$\langle x|y\rangle = a_0^* a_1 + b_0^* b_1$$

[4] if $z = x + iy$ then $z^* = x - iy$

State Vectors

We can handle complex states more easily if we switch to expressing states using vector notation.

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

$$\langle 0| = \begin{pmatrix} 1 & 0 \end{pmatrix} \qquad \langle 1| = \begin{pmatrix} 0 & 1 \end{pmatrix}$$

Vector Product

The inner product we've been using looks very neat with vectors.

$$\langle 0|0\rangle = \begin{pmatrix} 1 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = 1 * 1 + 0 * 0 = 1$$

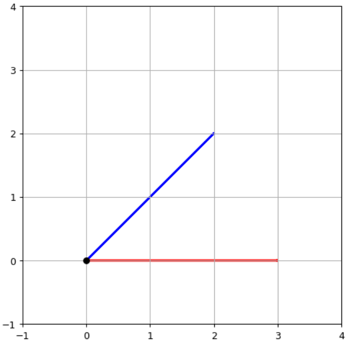
Generally,

$$\langle x|y\rangle = \begin{pmatrix} a & b \end{pmatrix} \begin{pmatrix} c \\ d \end{pmatrix} = ac + bd$$

Inner Product

We can think of the inner product graphically too.

$$\langle x|y\rangle = \begin{pmatrix} 3 & 0 \end{pmatrix} \begin{pmatrix} 2 \\ 2 \end{pmatrix}$$

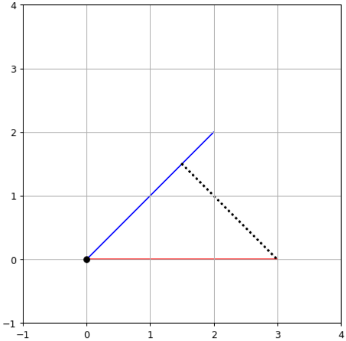


Inner Product

We can think of the inner product graphically too.

$$\langle x|y\rangle = \begin{pmatrix} 3 & 0 \end{pmatrix} \begin{pmatrix} 2 \\ 2 \end{pmatrix}$$

First we project vector x onto y.
Then we find the length of the new projected vector $|z|$.



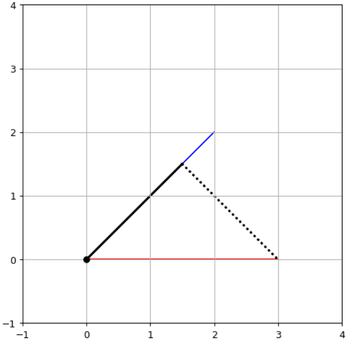
Inner Product

We can think of the inner product graphically too.

$$\langle x|y\rangle = \begin{pmatrix} 3 & 0 \end{pmatrix} \begin{pmatrix} 2 \\ 2 \end{pmatrix}$$

First we project vector x onto y.
Then we find the length of the new projected vector $|z|$.
Finally, we multiply this by the length of y, $|y|$.

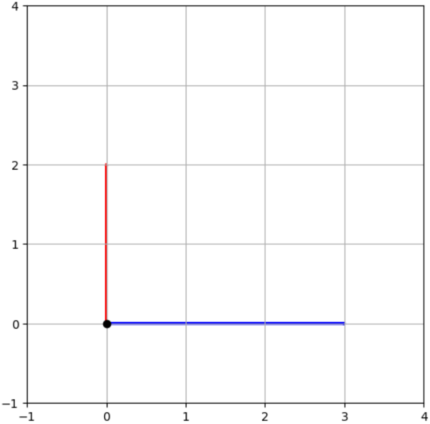
$$\langle x|y\rangle = |z||y|$$



Orthogonal States

By using this projection method, we see that two orthogonal (perpendicular) states have an inner product of 0.

$$\langle x|y\rangle = \begin{pmatrix} 3 & 0 \end{pmatrix} \begin{pmatrix} 0 \\ 2 \end{pmatrix} = 0$$



Computational Basis

Writing our two states $|0\rangle$ and $|1\rangle$ in this way allows us to make any 2 dimensional vector from a combination of them.

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

$$\begin{pmatrix} a \\ b \end{pmatrix} = a \begin{pmatrix} 1 \\ 0 \end{pmatrix} + b \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

We can this property **spanning** the 2d vectors.

Computational Basis

Additionally, they cannot be written as linear combinations of each other.

$$a \begin{pmatrix} 1 \\ 0 \end{pmatrix} \neq b \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

Computational Basis

Additionally, they cannot be written as linear combinations of each other.

$$a \begin{pmatrix} 1 \\ 0 \end{pmatrix} \neq b \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

They are therefore **linearly independent**.

Spanning and linear independence are the two criteria that make a set of vectors a basis.

Computational Basis

Additionally, they cannot be written as linear combinations of each other.

$$a\begin{pmatrix}1\\0\end{pmatrix} \neq b\begin{pmatrix}0\\1\end{pmatrix}$$

They are therefore **linearly independent**.

Spanning and linear independence are the two criteria that make a set of vectors a basis.

$\{|0\rangle, |1\rangle\}$ is called the **computational basis** because it relates most clearly to states of bits.

X Basis

s Spanning and linear independence seem pretty obvious for the computational basis, but we could make a basis from a different set of vectors.

$$|+\rangle = \frac{1}{\sqrt{2}}\begin{pmatrix}1\\1\end{pmatrix} \qquad |-\rangle = \frac{1}{\sqrt{2}}\begin{pmatrix}1\\-1\end{pmatrix}$$

X Basis

s Spanning and linear independence seem pretty obvious for the computational basis, but we could make a basis from a different set of vectors.

$$|+\rangle = \frac{1}{\sqrt{2}}\begin{pmatrix}1\\1\end{pmatrix} \qquad |-\rangle = \frac{1}{\sqrt{2}}\begin{pmatrix}1\\-1\end{pmatrix}$$

$$|+\rangle = \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle \qquad |-\rangle = \frac{1}{\sqrt{2}}|0\rangle - \frac{1}{\sqrt{2}}|1\rangle$$

X Basis

s Spanning and linear independence seem pretty obvious for the computational basis, but we could make a basis from a different set of vectors.

$$|+\rangle = \frac{1}{\sqrt{2}}\begin{pmatrix}1\\1\end{pmatrix} \qquad |-\rangle = \frac{1}{\sqrt{2}}\begin{pmatrix}1\\-1\end{pmatrix}$$

$$|+\rangle = \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle \qquad |-\rangle = \frac{1}{\sqrt{2}}|0\rangle - \frac{1}{\sqrt{2}}|1\rangle$$

The basis $\{|+\rangle, |-\rangle\}$ is called the X basis, for reasons we'll see soon.

X Basis

s Spanning and linear independence seem pretty obvious for the computational basis, but we could make a basis from a different set of vectors.

$$|+\rangle = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 1 \end{pmatrix} \qquad |-\rangle = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ -1 \end{pmatrix}$$

$$|+\rangle = \frac{1}{\sqrt{2}} |0\rangle + \frac{1}{\sqrt{2}} |1\rangle \qquad |-\rangle = \frac{1}{\sqrt{2}} |0\rangle - \frac{1}{\sqrt{2}} |1\rangle$$

The basis $\{|+\rangle, |-\rangle\}$ is called the X basis, for reasons we'll see soon.

Exercise

Show that the set $\{|+\rangle, |-\rangle\}$ is a basis for the 2d vectors.

Multi-bit States

To combine single bit states into larger sizes we use a *tensor product*.

$$|x\rangle \otimes |y\rangle = |xy\rangle$$

$$\begin{pmatrix} a_0 \\ b_0 \end{pmatrix} \otimes \begin{pmatrix} a_1 \\ b_1 \end{pmatrix} = \begin{pmatrix} a_0 \begin{pmatrix} a_1 \\ b_1 \end{pmatrix} \\ b_0 \begin{pmatrix} a_1 \\ b_1 \end{pmatrix} \end{pmatrix} = \begin{pmatrix} a_0 a_1 \\ a_0 b_1 \\ b_0 a_1 \\ b_0 b_1 \end{pmatrix}$$

Matrices

A **Matrix** can be thought of as a collection of vectors.

$$M = \begin{pmatrix} a_0 & b_0 & c_0 \\ a_1 & b_1 & c_1 \\ a_2 & b_2 & c_2 \end{pmatrix}$$

Rules for Matrices

Matrices behave quite similarly to vectors.

With two matrices $\hat{X} = \begin{pmatrix} a_0 & b_0 \\ c_0 & d_0 \end{pmatrix}$ and $\hat{Y} = \begin{pmatrix} a_1 & b_1 \\ c_1 & d_1 \end{pmatrix}$

Addition	**Scalar Multiplication**
$\hat{X} + \hat{Y} = \begin{pmatrix} a_0 + a_1 & b_0 + b_1 \\ c_0 + c_1 & d_0 + d_1 \end{pmatrix}$	$n * \hat{X} = \begin{pmatrix} n * a_0 & n * b_0 \\ n * c_0 & n * d_0 \end{pmatrix}$

Matrix Multiplication

When a matrix multiplies a vector, it transforms the vector to a new one.

$$\hat{X}\,|y\rangle = \begin{pmatrix} a_0 & b_0 \\ c_0 & d_0 \end{pmatrix} \begin{pmatrix} \alpha \\ \beta \end{pmatrix} = \begin{pmatrix} a_0\,\alpha + b_0\,\beta \\ c_0\,\alpha + d_0\,\beta \end{pmatrix}$$

This just gives us another vector.

Exercise

Find the vector v:

$$v = \begin{pmatrix} 2 & 7 \\ -2 & 1 \end{pmatrix} \begin{pmatrix} -4 \\ 3 \end{pmatrix}$$

Matrix Multiplication

Similarly matrices can be multiplied together

$$\hat{X}\hat{Y} = \begin{pmatrix} a_0 & b_0 \\ c_0 & d_0 \end{pmatrix} \begin{pmatrix} \alpha & \gamma \\ \beta & \delta \end{pmatrix} = \begin{pmatrix} a_0\alpha + b_0\beta & a_0\gamma + b_0\delta \\ c_0\alpha + d_0\beta & c_0\gamma + d_0\delta \end{pmatrix}$$

Matrix Multiplication

Similarly matrices can be multiplied together

$$\hat{X}\hat{Y} = \begin{pmatrix} a_0 & b_0 \\ c_0 & d_0 \end{pmatrix} \begin{pmatrix} \alpha & \gamma \\ \beta & \delta \end{pmatrix} = \begin{pmatrix} a_0\alpha + b_0\beta & a_0\gamma + b_0\delta \\ c_0\alpha + d_0\beta & c_0\gamma + d_0\delta \end{pmatrix}$$

Exercise

Find the matrix M:

$$M = \begin{pmatrix} 2 & 7 \\ -2 & 1 \end{pmatrix} \begin{pmatrix} -4 & 0 \\ 3 & 3 \end{pmatrix}$$

Matrix Form of Operators

$\hat{N}\left|0\right\rangle$

$\hat{N}\left|1\right\rangle$

Matrix Form of Operators

$$\hat{N}\left|0\right\rangle \qquad \hat{N}\left|1\right\rangle$$

$$\hat{N}\begin{pmatrix}1\\0\end{pmatrix} \qquad \hat{N}\begin{pmatrix}0\\1\end{pmatrix}$$

Matrix Form of Operators

$$\hat{N}\left|0\right\rangle \qquad \hat{N}\left|1\right\rangle$$

$$\hat{N}\begin{pmatrix}1\\0\end{pmatrix} \qquad \hat{N}\begin{pmatrix}0\\1\end{pmatrix}$$

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

Matrix Form of Operators

$$\hat{N}\left|0\right\rangle$$

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

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

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

$$\hat{N}\left|1\right\rangle$$

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

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

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

Matrix form of Operators

We can actually find out the matrix form of an operator from its outer product form.

$$\hat{N} = |1\rangle\langle 0| + |0\rangle\langle 1|$$

Matrix form of Operators

We can actually find out the matrix form of an operator from its outer product form.

$$\hat{N} = |1\rangle\langle 0| + |0\rangle\langle 1|$$

$$\hat{N} = 0\,|0\rangle\langle 0| + 1\,|1\rangle\langle 0| + 1\,|0\rangle\langle 1| + 0\,|1\rangle\langle 1|$$

Matrix form of Operators

We can actually find out the matrix form of an operator from its outer product form.

$$\hat{N} = |1\rangle\langle 0| + |0\rangle\langle 1|$$

$$\hat{N} = 0\,|0\rangle\langle 0| + 1\,|1\rangle\langle 0| + 1\,|0\rangle\langle 1| + 0\,|1\rangle\langle 1|$$

$$\hat{N} = \begin{matrix} 0\,|0\rangle\langle 0| & 1\,|0\rangle\langle 1| \\ 1\,|1\rangle\langle 0| & 0\,|1\rangle\langle 1| \end{matrix}$$

Matrix form of Operators

We can actually find out the matrix form of an operator from its outer product form.

$$\hat{N} = |1\rangle\langle 0| + |0\rangle\langle 1|$$

$$\hat{N} = 0\,|0\rangle\langle 0| + 1\,|1\rangle\langle 0| + 1\,|0\rangle\langle 1| + 0\,|1\rangle\langle 1|$$

$$\hat{N} = \begin{matrix} 0\,|0\rangle\langle 0| & 1\,|0\rangle\langle 1| \\ 1\,|1\rangle\langle 0| & 0\,|1\rangle\langle 1| \end{matrix}$$

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

Multi-bit Operators

The **only** time an AND gate returns *ON* is when we have both inputs on.

$$\hat{A}|11\rangle = |1\rangle$$

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

$$\hat{A}\begin{pmatrix}0\\0\\0\\1\end{pmatrix} = \begin{pmatrix}0\\1\end{pmatrix}$$

Multi-bit Operators

The $\hat{A}$ operator has the form:

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

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

Multi-bit Operators

and the **only** time an OR gate returns *OFF* is when we have both inputs off.

$$\hat{O}\,|00\rangle = |0\rangle$$

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

$$\hat{O}\begin{pmatrix}1\\0\\0\\0\end{pmatrix} = \begin{pmatrix}1\\0\end{pmatrix}$$

Multi-bit Operators

The $\hat{0}$ operator has the form:

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

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

Logic Summary

We now have three(!) equivalent ways of thinking about computational logic.

1. Circuits
2. States and Operators
3. Vectors and Matrices

These are the same tools we'll use to understand quantum algorithms later on.

Quantum Circuits

Components

- Quantum circuits flow from left to right.

Components

- Quantum circuits flow from left to right.
- We use a wire to represent each qubit.
- Wires cannot be split. (No FANOUT)

Components

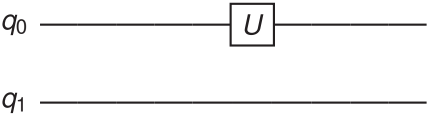
- Quantum circuits flow from left to right.
- We use a wire to represent each qubit.
- Wires cannot be split. (No FANOUT)
- Boxes represent Operators.

Components

- Quantum circuits flow from left to right.
- We use a wire to represent each qubit.
- Wires cannot be split. (No FANOUT)
- Boxes represent Operators.
- Each qubit starts in state $|0\rangle$ unless stated otherwise.

Components

- Quantum circuits flow from left to right.
- We use a wire to represent each qubit.
- Wires cannot be split. (No FANOUT)
- Boxes represent Operators.
- Each qubit starts in state $|0\rangle$ unless stated otherwise.



Pauli Matrices

$$\sigma_X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \qquad \sigma_Y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix} \qquad \sigma_Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

Pauli Matrices

Once we add in the identity $\mathbb{I} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$, we have a **basis** for the 2x2 matrices.

Exercise

Show that the set $\{\sigma_x, \sigma_y, \sigma_z, I\}$ is a basis for the 2x2 matrices.

Pauli Matrices

Prove that the Pauli matrices are self-inverse.

$$\sigma_X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \qquad \sigma_Y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix} \qquad \sigma_Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

$$\sigma_X \sigma_X = \sigma_Y \sigma_Y = \sigma_Z \sigma_Z = \mathbb{I}$$

Pauli Gates

$$\sigma_X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \qquad \sigma_Y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix} \qquad \sigma_Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

Hadamard Gate

One of the most common gates is the Hadamard gate, which maximally mixes the $|0\rangle$ and $|1\rangle$ states.

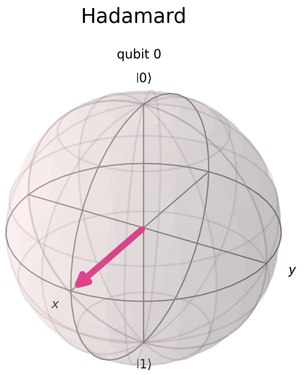
$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$$

so $\hat{H}\,|0\rangle = (|0\rangle + |1\rangle)/\sqrt{2}$ and $\hat{H}\,|1\rangle = (|0\rangle - |1\rangle)/\sqrt{2}$.

Which puts the state half way between $|0\rangle$ and $|1\rangle$.

Visualising Gates

Let's visualise the action of these one qubits gates using the Bloch Sphere.



We can see that the Hadamard gate has moved the state so that it now

Visualising Gates

We know that for a state $|\Psi\rangle = \alpha|0\rangle + \beta|1\rangle$ we expect:

$$\hat{X}|\Psi\rangle = \alpha|1\rangle + \beta|0\rangle$$

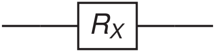
Lets visualise this on the bloch sphere:

Rotation Gates

We can turn these into rotations by an angle θ by mixing them with the identity.

$$R_X = \cos\frac{\theta}{2}\mathbb{I} - i\sin\frac{\theta}{2}\sigma_x$$

$$R_X = \begin{pmatrix} \cos\frac{\theta}{2} & -i\sin\frac{\theta}{2} \\ -i\sin\frac{\theta}{2} & \cos\frac{\theta}{2} \end{pmatrix}$$



Rotation Gates

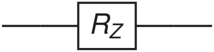
$$R_Y = \cos\frac{\theta}{2}\mathbb{I} - i\sin\frac{\theta}{2}\sigma_y$$

$$R_Y = \begin{pmatrix} \cos\frac{\theta}{2} & -\sin\frac{\theta}{2} \\ \sin\frac{\theta}{2} & \cos\frac{\theta}{2} \end{pmatrix}$$

Rotation Gates

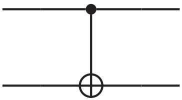
$$R_Z = \cos\frac{\theta}{2}\mathbb{I} - i\sin\frac{\theta}{2}\sigma_z$$

$$R_Z = \begin{pmatrix} \cos\frac{\theta}{2} - i\sin\frac{\theta}{2} & 0 \\ 0 & \cos\frac{\theta}{2} + i\sin\frac{\theta}{2} \end{pmatrix}$$



Two-qubit Gates

Controlled-NOT



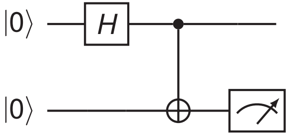
$$U_{CN} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

Swap Gate

We can swap the states of two qubits using three CNOT gates.

Measurement

The last basic component is the measurement symbol, which will terminate the line.



IMBQ

Now we're ready to run our first quantum circuits.
We're going to do this using a Jupyter notebook

Metrics

The size of circuits is measured with two different numbers.

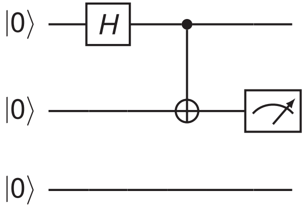
1. Depth - The maximum number of gates which have to be applied in sequence.
2. Qubit Count - The maximum number of qubits in use at any one time.

Depth

Depth is calculated as the maximum number of gates which must be applied in sequence.

Qubit Count

Qubit count is the maximum number of physical qubits that have to be in use at any one time.



Even though we have three qubits, the qubit count for this circuit is two.

Information & Communication

Information

What sort of things can we do with information?

Information

What sort of things can we do with information?

- Read
- Edit
- Delete
- Move
- Copy
- Encryption / decryption

Readout

When we measure a state we get each possible result with some probability.

Readout

When we measure a state we get each possible result with some probability.

We're interested in knowing these probabilities.

Readout

When we measure a state we get each possible result with some probability.

We're interested in knowing these probabilities.

So we have to make lots of measurements.

Readout

When we measure a state we get each possible result with some probability.

We're interested in knowing these probabilities.

So we have to make lots of measurements.

Which requires preparing the state each time!

Information

What sort of things can we do with information?

Information

What sort of things can we do with information?

- Read
- Edit
- Delete
- Move
- Copy
- Encryption / decryption

No Cloning Theorem

Theorem

There is no operation which can copy arbitrary quantum states.

We prove this by supposing it is possible.

$$|\Psi\rangle \otimes |s\rangle \xrightarrow{U} |\Psi\rangle \otimes |\Psi\rangle \qquad |\Phi\rangle \otimes |s\rangle \xrightarrow{U} |\Phi\rangle \otimes |\Phi\rangle$$

No Cloning Theorem

Theorem

There is no operation which can copy arbitrary quantum states.

We prove this by supposing it is possible.

$$|\Psi\rangle \otimes |s\rangle \xrightarrow{U} |\Psi\rangle \otimes |\Psi\rangle \qquad |\Phi\rangle \otimes |s\rangle \xrightarrow{U} |\Phi\rangle \otimes |\Phi\rangle$$

$$(\langle\Psi| \otimes \langle s|)(|\Phi\rangle \otimes |s\rangle) = (\langle\Psi| \otimes \langle\Psi|)(|\Phi\rangle \otimes |\Phi\rangle)$$

No Cloning Theorem

Theorem

There is no operation which can copy arbitrary quantum states.

We prove this by supposing it is possible.

$$|\Psi\rangle \otimes |s\rangle \xrightarrow{U} |\Psi\rangle \otimes |\Psi\rangle \qquad |\Phi\rangle \otimes |s\rangle \xrightarrow{U} |\Phi\rangle \otimes |\Phi\rangle$$

$$(\langle\Psi| \otimes \langle s|)(|\Phi\rangle \otimes |s\rangle) = (\langle\Psi| \otimes \langle\Psi|)(|\Phi\rangle \otimes |\Phi\rangle)$$

$$\langle\Psi|\Phi\rangle \otimes \langle s|s\rangle = \langle\Psi|\Phi\rangle \otimes \langle\Psi|\Phi\rangle$$

No Cloning Theorem

Theorem

There is no operation which can copy arbitrary quantum states.

We prove this by supposing it is possible.

$$|\Psi\rangle \otimes |s\rangle \xrightarrow{U} |\Psi\rangle \otimes |\Psi\rangle \qquad |\Phi\rangle \otimes |s\rangle \xrightarrow{U} |\Phi\rangle \otimes |\Phi\rangle$$

$$(\langle\Psi| \otimes \langle s|)(|\Phi\rangle \otimes |s\rangle) = (\langle\Psi| \otimes \langle\Psi|)(|\Phi\rangle \otimes |\Phi\rangle)$$

$$\langle\Psi|\Phi\rangle \otimes \langle s|s\rangle = \langle\Psi|\Phi\rangle \otimes \langle\Psi|\Phi\rangle$$

$$\langle\Psi|\Phi\rangle = (\langle\Psi|\Phi\rangle)^2$$

So either $|\Psi\rangle$ and $|\Phi\rangle$ are the same state or they're orthogonal!

Information

What sort of things can we do with information?

Information

What sort of things can we do with information?

- Read
- Edit
- Delete
- Move
- Copy
- Encryption / decryption

Private Key Distribution

Alice and Bob know they will soon have a message to send over the internet, but it's critical it stays private from their friend Eve.[5]

[5] They are planning surprise a party for Eve's cat, who is alive and well.

Private Key Distribution

Alice and Bob know they will soon have a message to send over the internet, but it's critical it stays private from their friend Eve.[5]

They could met up in person ahead of time, and agree on a secret **key**, a string of random bits.

$$k = 0011100010110010 1\ldots$$

[5] They are planning surprise a party for Eve's cat, who is alive and well.

Private Key Distribution

When the time comes, Alice prepares her message in binary, then adds the key to her own bits modulo 2.

$$\begin{array}{c} 01100010100111011 \\ + \\ \color{red}{00111000101100101} \\ \downarrow \\ 01011010001011110 \end{array}$$

---

[6] in this case addition and subtraction give the same result. Try it out.

Private Key Distribution

When the time comes, Alice prepares her message in binary, then adds the key to her own bits modulo 2.

$$\begin{array}{c} 0110001010011011 \\ + \\ \color{red}{0011100010110010} \\ \downarrow \\ 0101101000101110 \end{array}$$

Now her message is encrypted. Bob can decode it by doing the same operation.[6]

Even if Eve intercepted it the message she couldn't know what the original values were without having the key.

[6] in this case addition and subtraction give the same result. Try it out.

Quantum Key Distribution

This system works great provided that:

1. You know you'll need to send the message
2. You can meet beforehand
3. No one steals either copy of the key

It would be much better to create a new key at the time of each message.

However, if Alice and Bob tried to send each other messages containing the key before the message, Eve could listen in.

Quantum Key Distribution

There is a way that Alice and Bob can create a key for themselves by sending Qubits to each other.

1. Alice creates two strings of random bits, *a* and *b*.

Quantum Key Distribution

There is a way that Alice and Bob can create a key for themselves by sending Qubits to each other.

1. Alice creates two strings of random bits, *a* and *b*.
2. She encodes the bits of *a* using the bits of *b*.

	a=0	a=1
b=0	$\vert 0\rangle$	$\vert 1\rangle$
b=1	$\vert +\rangle$	$\vert -\rangle$

Quantum Key Distribution

There is a way that Alice and Bob can create a key for themselves by sending Qubits to each other.

1. Alice creates two strings of random bits, *a* and *b*.
2. She encodes the bits of *a* using the bits of *b*.

	a=0	a=1
b=0	$\|0\rangle$	$\|1\rangle$
b=1	$\|+\rangle$	$\|-\rangle$

3. She then sends these to Bob.

Quantum Key Distribution

There is a way that Alice and Bob can create a key for themselves by sending Qubits to each other.

1. Alice creates two strings of random bits, *a* and *b*.
2. She encodes the bits of *a* using the bits of *b*.

	a=0	a=1
b=0	$\|0\rangle$	$\|1\rangle$
b=1	$\|+\rangle$	$\|-\rangle$

3. She then sends these to Bob.
4. Alice keeps *b* private for now.

Quantum Key Distribution

There is a way that Alice and Bob can create a key for themselves by sending Qubits to each other.

1. Alice creates two strings of random bits, *a* and *b*.
2. She encodes the bits of *a* using the bits of *b*.

	a=0	a=1
b=0	$\|0\rangle$	$\|1\rangle$
b=1	$\|+\rangle$	$\|-\rangle$

3. She then sends these to Bob.
4. Alice keeps *b* private for now.
5. Bob picks at random which basis to measure Alice's encoded bits in, and records his choices as *c*.

Quantum Key Distribution

There is a way that Alice and Bob can create a key for themselves by sending Qubits to each other.

1. Alice creates two strings of random bits, *a* and *b*.
2. She encodes the bits of *a* using the bits of *b*.

	a=0	a=1
b=0	$\|0\rangle$	$\|1\rangle$
b=1	$\|+\rangle$	$\|-\rangle$

3. She then sends these to Bob.
4. Alice keeps *b* private for now.
5. Bob picks at random which basis to measure Alice's encoded bits in, and records his choices as *c*.
6. Alice publicly announces *b*.

Quantum Key Distribution

There is a way that Alice and Bob can create a key for themselves by sending Qubits to each other.

1. Alice creates two strings of random bits, *a* and *b*.
2. She encodes the bits of *a* using the bits of *b*.

	a=0	a=1
b=0	$\|0\rangle$	$\|1\rangle$
b=1	$\|+\rangle$	$\|-\rangle$

3. She then sends these to Bob.
4. Alice keeps *b* private for now.
5. Bob picks at random which basis to measure Alice's encoded bits in, and records his choices as *c*.
6. Alice publicly announces *b*.
7. Alice and Bob discard any bits for which *b* is different from *c*.

Quantum Key Distribution

Let's see an example.

$$a = 0110011$$
$$b = 1011001$$

	a=0	a=1
b=0	$\|0\rangle$	$\|1\rangle$
b=1	$\|+\rangle$	$\|-\rangle$

$$\downarrow$$

$$|+\rangle\,|1\rangle\,|-\rangle\,|+\rangle\,|0\rangle\,|1\rangle\,|-\rangle$$

Quantum Key Distribution

Let's see an example.

$$a = 0110011$$
$$b = 1011001$$

	a=0	a=1
b=0	$\|0\rangle$	$\|1\rangle$
b=1	$\|+\rangle$	$\|-\rangle$

$$\downarrow$$

$$|+\rangle\,|1\rangle\,|-\rangle\,|+\rangle\,|0\rangle\,|1\rangle\,|-\rangle$$

$$c = 1110100$$

Quantum Key Distribution

Let's see an example.

$$a = 0110011$$
$$b = 1011001$$

	a=0	a=1
b=0	$\|0\rangle$	$\|1\rangle$
b=1	$\|+\rangle$	$\|-\rangle$

$$\downarrow$$

$$|+\rangle\ |1\rangle\ |-\rangle\ |+\rangle\ |0\rangle\ |1\rangle\ |-\rangle$$

$$c = 1110100$$

$$\downarrow$$

$$|+\rangle\ |+\rangle\ |-\rangle\ |1\rangle\ |-\rangle\ |1\rangle\ |0\rangle$$

$$\downarrow$$

Quantum Key Distribution

Let's see an example.

$a = 0110011$
$b = 1011001$

	a=0	a=1
b=0	$\|0\rangle$	$\|1\rangle$
b=1	$\|+\rangle$	$\|-\rangle$

$\downarrow$

$|+\rangle\ |1\rangle\ |-\rangle\ |+\rangle\ |0\rangle\ |1\rangle\ |-\rangle$

$c = 1110100$

$\downarrow$

$|+\rangle\ |+\rangle\ |-\rangle\ |1\rangle\ |-\rangle\ |1\rangle\ |0\rangle$

$\downarrow$

0 0 1 1 0 1 1

Checking for Eavesdropping

The No Cloning Theorem prevents Eve from copying the qubits Alice sends to Bob.

Lets say she'd like to gain information about which state they share without disturbing the state.

$$|\psi\rangle |u\rangle \rightarrow |\psi\rangle |v\rangle$$
$$|\phi\rangle |u\rangle \rightarrow |\phi\rangle |v'\rangle$$

Checking for Eavesdropping

The No Cloning Theorem prevents Eve from copying the qubits Alice sends to Bob.

Lets say she'd like to gain information about which state they share without disturbing the state.

$$|\psi\rangle\,|u\rangle \rightarrow |\psi\rangle\,|v\rangle$$
$$|\phi\rangle\,|u\rangle \rightarrow |\phi\rangle\,|v'\rangle$$

If Eve can make $|v\rangle$ and $|v'\rangle$ different, then she can tell which state they had!

Checking for Eavesdropping

The No Cloning Theorem prevents Eve from copying the qubits Alice sends to Bob.

Lets say she'd like to gain information about which state they share without disturbing the state.

$$|\psi\rangle|u\rangle \rightarrow |\psi\rangle|v\rangle$$
$$|\phi\rangle|u\rangle \rightarrow |\phi\rangle|v'\rangle$$

If Eve can make $|v\rangle$ and $|v'\rangle$ different, then she can tell which state they had!

$$\langle\psi|\phi\rangle\langle v|v'\rangle = \langle\psi|\phi\rangle\langle u|u\rangle$$
$$\langle v|v'\rangle = \langle u|u\rangle = 1$$

Checking for Eavesdropping

The No Cloning Theorem prevents Eve from copying the qubits Alice sends to Bob.

Lets say she'd like to gain information about which state they share without disturbing the state.

$$|\psi\rangle\,|u\rangle \rightarrow |\psi\rangle\,|v\rangle$$
$$|\phi\rangle\,|u\rangle \rightarrow |\phi\rangle\,|v'\rangle$$

If Eve can make $|v\rangle$ and $|v'\rangle$ different, then she can tell which state they had!

$$\langle\psi|\phi\rangle\,\langle v|v'\rangle = \langle\psi|\phi\rangle\,\langle u|u\rangle$$
$$\langle v|v'\rangle = \langle u|u\rangle = 1$$

Because $\langle v|v'\rangle = 1$, they must be the same state. Eve can't learn about Alice and Bob's state without changing it.

Checking for Eavesdropping

Alice and Bob can do a final check to see if someone is interfering with their states.

Checking for Eavesdropping

Alice and Bob can do a final check to see if someone is interfering with their states.

They remove a portion of their key and share publicly to look for discrepancies.

Checking for Eavesdropping

Alice and Bob can do a final check to see if someone is interfering with their states.

They remove a portion of their key and share publicly to look for discrepancies.

If there are too many differences, they know that someone was intercepting the message.

Hardware

Required properties

The criteria for quantum information processing are:

- Well defined two-level system
- Ability to initialise the state
- Long qubit coherence times[7]
- Universal gate set
- Measurement

[7] Compared to the time it takes to implement a gate.

Desirable properties

These are pretty loose criteria but in reality some designs are better than others.

- Low noise
- Qubit connectivity
- Easy to scale up
- Reliable
- Cheap to build and use

Hardware

- Trapped Ions
- Superconducting Qubits

Trapped Ions

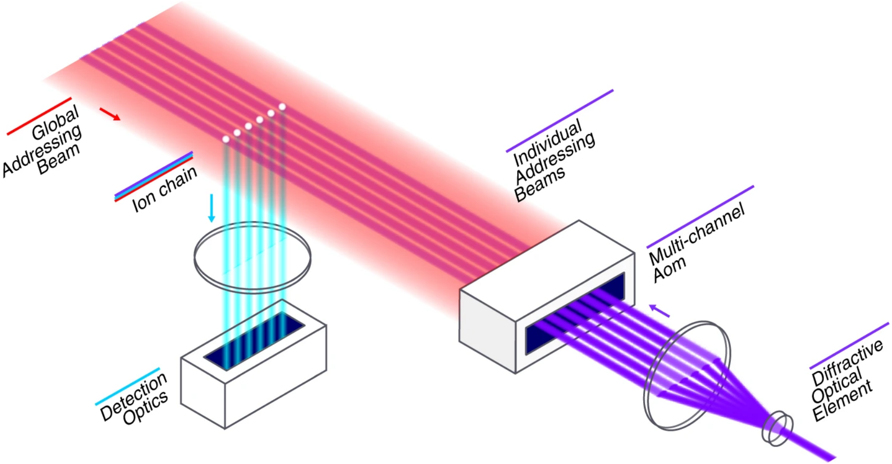


Figure: IONQ

Trapped Ions

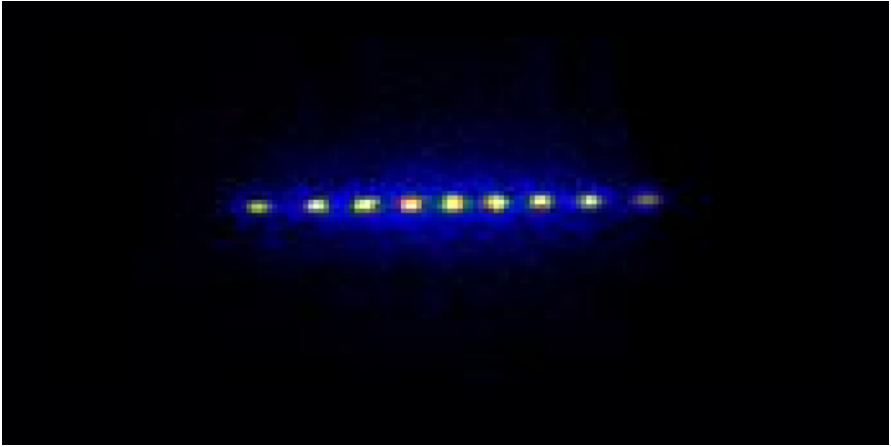
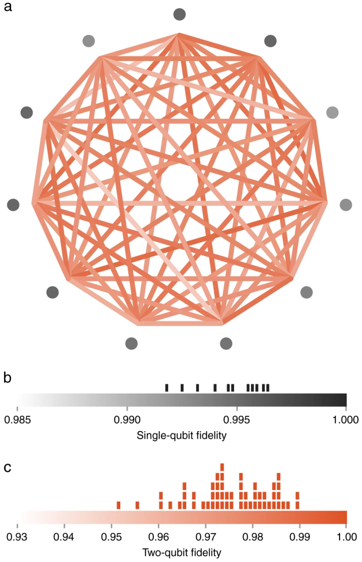


Figure: Trapped Ions - University of Oxford

Trapped Ions



Trapped Ions: Connectivity

Qubits are defined using two energy levels of the ion.
We can manipulate the energy of the Ion by using a laser with a resonant frequency.

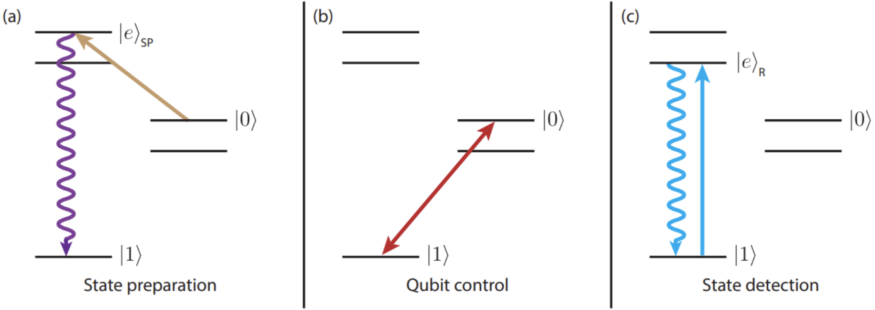


Figure: Bruzewicz et al 2019

Trapped Ions: Gates

The GPi and GPi2 gates are used for single qubit operations.

$$GPi = \begin{pmatrix} 0 & e^{-i\phi} \\ e^{-i\phi} & 0 \end{pmatrix} \qquad GPi2 = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & -ie^{-i\phi} \\ -ie^{-i\phi} & 1 \end{pmatrix}$$

The *Mølmer-Sørenson* gate is used for entangling qubits.

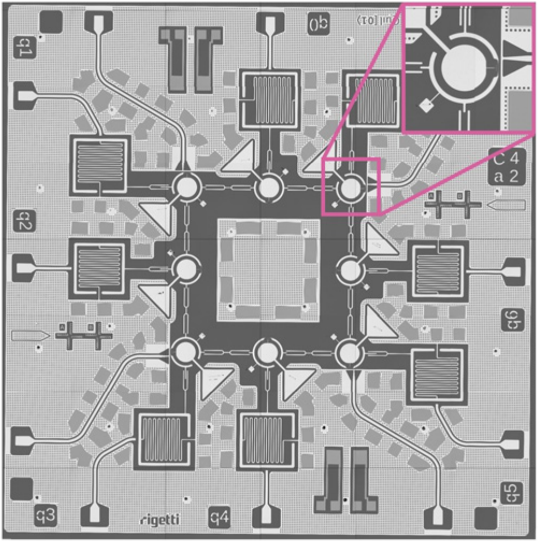
$$\frac{-i}{\sqrt{2}} \begin{pmatrix} i & 0 & 0 & e^{-i(\phi_0+\phi_1)} \\ 0 & i & e^{-i(\phi_0-\phi_1)} & 0 \\ 0 & e^{-i(\phi_0-\phi_1)} & i & 0 \\ e^{-i(\phi_0+\phi_1)} & 0 & 0 & i \end{pmatrix}$$

Hardware

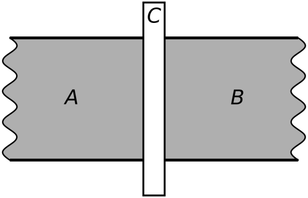
- Trapped Ions
- Superconducting Qubits

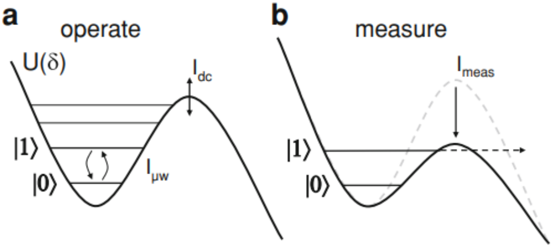
Superconducting Qubits

A



Superconducting Qubits





Superconducting Qubits: Gates

The R_x and R_y gates are used for single qubit operations.

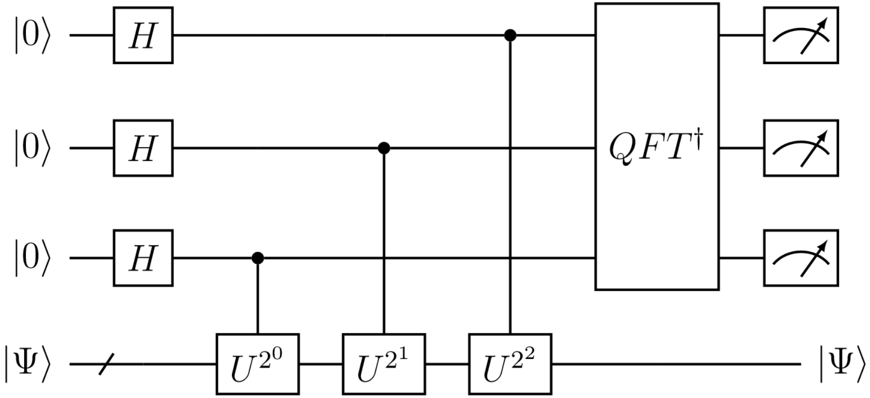
$$R_X = \begin{pmatrix} \cos\frac{\theta}{2} & -i\sin\frac{\theta}{2} \\ -i\sin\frac{\theta}{2} & \cos\frac{\theta}{2} \end{pmatrix} \qquad R_Y = \begin{pmatrix} \cos\frac{\theta}{2} & -\sin\frac{\theta}{2} \\ \sin\frac{\theta}{2} & \cos\frac{\theta}{2} \end{pmatrix}$$

The Controlled-Z CZ gate is used for entangling qubits.

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

QPE

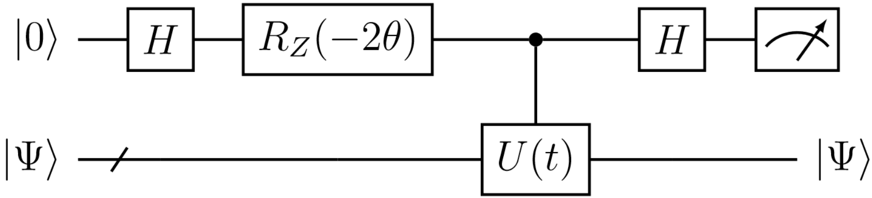
The Quantum Phase Estimator is a way of working out the energy of each state of a system.



Iterative QPE

We can reduce the number of qubits we need by only using one extra at a time.

However, the circuit depth increases.



NISQ

Current quantum computers are very limited in qubit count and depth.

We therefore need to design algorithms for the **N**oisy **I**ntermediate **S**mall **Q**uantum Computers.

Most algorithm research conducted today is focused on NISQ applications.

VQE

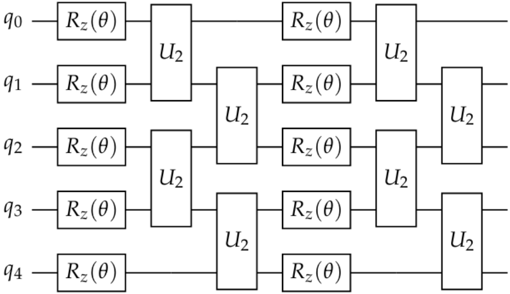
The Variational Quantum Eigensolver is a way to get the lowest energy state $|\psi_0\rangle$ of a system $|\Psi\rangle$.

1. Initial state input $|\Phi\rangle$

VQE

The Variational Quantum Eigensolver is a way to get the lowest energy state $|\psi_0\rangle$ of a system $|\Psi\rangle$.

1. Initial state input $|\Phi\rangle$
2. State evolves according to parameterised circuit $\hat{U}(\vec{\alpha})\,|\Phi\rangle = |\Psi(\vec{\alpha})\rangle$



VQE

The Variational Quantum Eigensolver is a way to get the lowest energy state $|\psi_0\rangle$ of a system $|\Psi\rangle$.

1. Initial state input $|\Phi\rangle$
2. State evolves according to parameterised circuit $\hat{U}(\vec{\alpha})\,|\Phi\rangle = |\Psi(\vec{\alpha})\rangle$
3. Measure the energy of the state $E = \langle\Psi(\alpha)|\,\hat{H}\,|\Psi(\alpha)\rangle$

Postulates of QM

Postulate 1

Any isolated physical system is completely described by a state vector.

This is true not just for qubits, but for any size of quantum system.

$$|\Psi\rangle = c_0 |\psi_0\rangle + c_1 |\psi_1\rangle + \cdots + c_n |\psi_n\rangle$$

Where $\{c_0, c_1, \ldots, c_n\} \in \mathbb{C}$

$$|c_0|^2 + |c_1|^2 + \cdots + |c_n|^2 = 1$$

Variational Principle

The variational principal says that we can find the ground state $|\Psi_0\rangle$ by finding a minimum in the energy of parameterised states.

$$E = \langle\Psi(\alpha)| \hat{H} |\Psi(\alpha)\rangle$$

$$E_0 \leqslant \langle\Psi(\alpha)| \hat{H} |\Psi(\alpha)\rangle$$

$|\Psi(\alpha)\rangle$ is the ground state $|\Psi_0\rangle$ when we find a minimum in E!

VQE

The Variational Quantum Eigensolver is a way to get the lowest energy state $|\psi_0\rangle$ of a system $|\Psi\rangle$.

1. Initial state input $|\Phi\rangle$
2. State evolves according to parameterised circuit $\hat{U}(\vec{\alpha})\,|\Phi\rangle = |\Psi(\vec{\alpha})\rangle$
3. Measure the energy of the state $E = \langle\Psi(\alpha)|\,\hat{H}\,|\Psi(\alpha)\rangle$
4. Update the parameters $\vec{\alpha}$

VQE

The Variational Quantum Eigensolver is a way to get the lowest energy state $|\psi_0\rangle$ of a system $|\Psi\rangle$.

1. Initial state input $|\Phi\rangle$
2. State evolves according to parameterised circuit $\hat{U}(\vec{\alpha})\,|\Phi\rangle = |\Psi(\vec{\alpha})\rangle$
3. Measure the energy of the state $E = \langle\Psi(\alpha)|\,\hat{H}\,|\Psi(\alpha)\rangle$
4. Update the parameters $\vec{\alpha}$
5. Repeat Steps 1-3 until we find the minimum energy.

VQE Pros & Cons

- Reduces circuit depth.
- Doesn't need any ancilla qubits
- Classical methods for initial state
- Finds real E_0
- Resistant to noise

VQE Pros & Cons

- Reduces circuit depth.
- Doesn't need any ancilla qubits
- Classical methods for initial state
- Finds real E_0
- Resistant to noise

- Need to run lots of circuits
- Might not be able to update parameters
- Global or local minimum?

Compilers

Why we need compilers

We can write circuits using lots of languages.

Why we need compilers

We can write circuits using lots of languages.

Real devices are limited in qubit count and connectivity.

Why we need compilers

We can write circuits using lots of languages.

Real devices are limited in qubit count and connectivity.

We want to run algorithms efficiently.

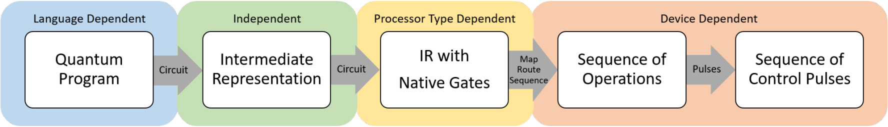
Why we need compilers

We can write circuits using lots of languages.

Real devices are limited in qubit count and connectivity.

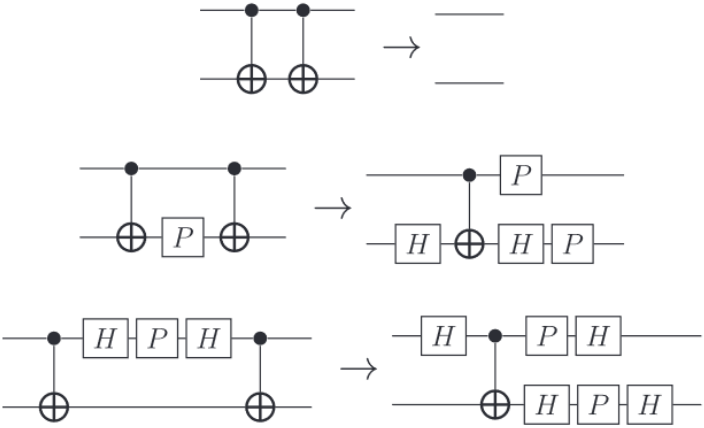
We want to run algorithms efficiently.

Compiler Flow



Redundant Gates

We might inadvertently write a circuit with gates that cancel out.
Compilers remove these using templates.



Intermediate Representation

Programming languages output circuits in a standard format called an **Intermediate Representation**.

Qiskit

Read only

Open in Quantum Lab

```
from qiskit import QuantumRegister, ClassicalRegister, QuantumCircuit
from numpy import pi

qreg_q = QuantumRegister(4, 'q')
creg_c = ClassicalRegister(4, 'c')
circuit = QuantumCircuit(qreg_q, creg_c)

circuit.h(qreg_q[0])
circuit.cx(qreg_q[0], qreg_q[1])
circuit.measure(qreg_q[1], creg_c[1])
```

Intermediate Representation

OpenQASM 2.0

```
OPENQASM 2.0;
include "qelib1.inc";

qreg q[2];
creg c[2];

h q[0];
cx q[0],q[1];
measure q[1] -> c[1];
```

Qiskit

Read only

Open in Quantum Lab

```python
from qiskit import QuantumRegister, ClassicalRegister, QuantumCircuit
from numpy import pi

qreg_q = QuantumRegister(4, 'q')
creg_c = ClassicalRegister(4, 'c')
circuit = QuantumCircuit(qreg_q, creg_c)

circuit.h(qreg_q[0])
circuit.cx(qreg_q[0], qreg_q[1])
circuit.measure(qreg_q[1], creg_c[1])
```

Native Gates

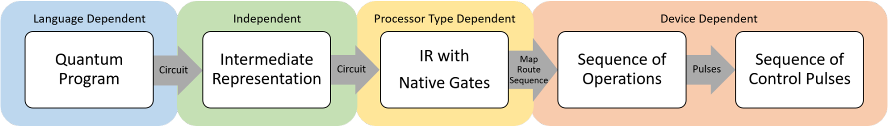
Trapped Ion (IONQ)

- GPi
- GPi2
- Molmer-Sorensen

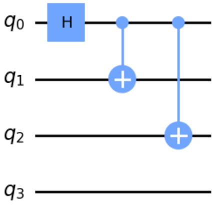
Super Conductor (Rigetti)

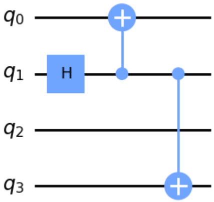
- Rx
- Ry
- CZ

Compiler Flow

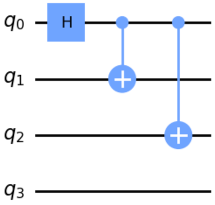


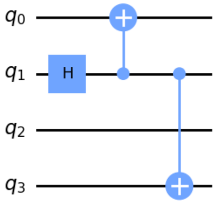
Equivalent Circuits

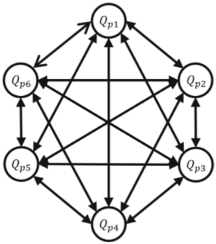




Equivalent Circuits







Connectivity

Figure: Qubit map for IBMQ Manilla

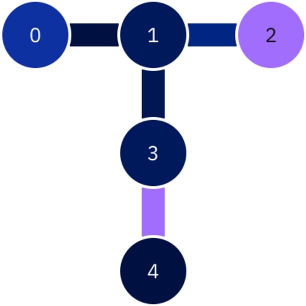


Figure: Qubit map for IBMQ Quito

Quality

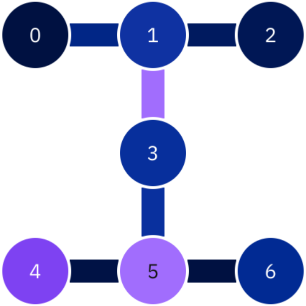


Figure: Qubit map for IBMQ Lagos

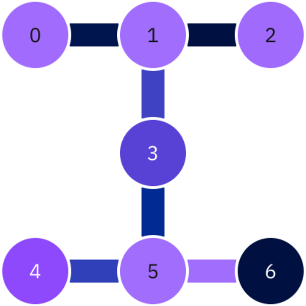
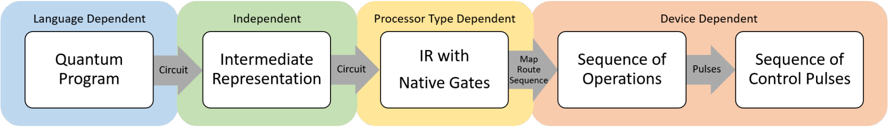


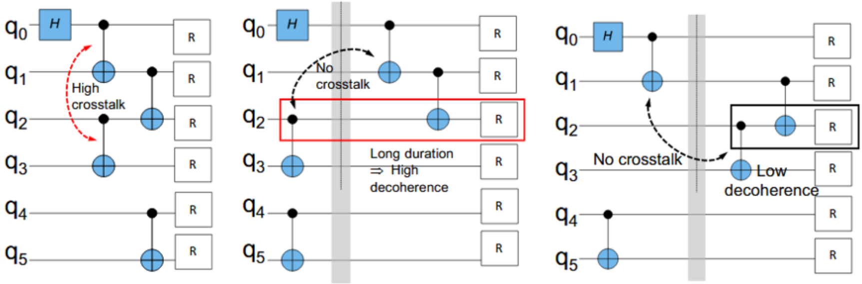
Figure: Qubit map for IBMQ Perth

Compiler Flow



Crosstalk

Crosstalk occurs when we apply two gates at once, causing the signals to interfere.



What we didn’t cover

- Adiabatic Quantum Computing
- Non-reversible computing
- Braiding
- Error Correction
- Algorithms

Further Reading

- IMBQ Textbook
- Feynman Lectures
- Neilsen & Chuang