

The Quantum World

A Rapid Introduction: Day 3

Reuben R. W. Wang ¹

¹*Engineering Product Development*
Singapore University of Technology and Design

IAP, 2019

Recap

Last lesson, we went over the following concepts:

Recap

Last lesson, we went over the following concepts:

- **Commutators:** Defined commutators between operators and their properties.

Recap

Last lesson, we went over the following concepts:

- **Commutators:** Defined commutators between operators and their properties.
- **Inner Products and Expectation:** Defined inner products in infinite dimensional Hilbert spaces and expectation values.

Last lesson, we went over the following concepts:

- **Commutators:** Defined commutators between operators and their properties.
- **Inner Products and Expectation:** Defined inner products in infinite dimensional Hilbert spaces and expectation values.
- **Hermitian Operators and the Spectral Theorem:** Hermitian observables will always have real eigenvalues and orthonormal eigenvectors.

Recap

Last lesson, we went over the following concepts:

- **Commutators:** Defined commutators between operators and their properties.
- **Inner Products and Expectation:** Defined inner products in infinite dimensional Hilbert spaces and expectation values.
- **Hermitian Operators and the Spectral Theorem:** Hermitian observables will always have real eigenvalues and orthonormal eigenvectors.
- **Measurement:** Measuring an observable collapses the state into an eigenstate producing the associated eigenvalue.

Last lesson, we went over the following concepts:

- **Commutators:** Defined commutators between operators and their properties.
- **Inner Products and Expectation:** Defined inner products in infinite dimensional Hilbert spaces and expectation values.
- **Hermitian Operators and the Spectral Theorem:** Hermitian observables will always have real eigenvalues and orthonormal eigenvectors.
- **Measurement:** Measuring an observable collapses the state into an eigenstate producing the associated eigenvalue.
- **Stationary States and 1D potentials:** Separable solutions allow us for energy eigenstate solutions in 1D potentials.

Dirac's Bras and Kets: Matrix Mechanics

- To define an inner product, we require a definition of **conjugate transposition**. Conjugate transpose of a ket is called the '**bra**':

$$\langle\psi| = (|\psi\rangle^*)^T = |\psi\rangle^\dagger$$

Dirac's Bras and Kets: Matrix Mechanics

- To define an inner product, we require a definition of **conjugate transposition**. Conjugate transpose of a ket is called the '**bra**':

$$\langle\psi| = (|\psi\rangle^*)^T = |\psi\rangle^\dagger$$

- The symbol we use for conjugate transposition ($\dagger$) is the '**dagger**'.

Dirac's Bras and Kets: Matrix Mechanics

- To define an inner product, we require a definition of **conjugate transposition**. Conjugate transpose of a ket is called the '**bra**':

$$\langle\psi| = (|\psi\rangle^*)^T = |\psi\rangle^\dagger$$

- The symbol we use for conjugate transposition ($\dagger$) is the '**dagger**'.
- **Inner products** are done by closing the '*bra-ket*':

$$\langle\psi, \phi\rangle = \langle\psi|\phi\rangle$$

.

Dirac's Bras and Kets: Matrix Mechanics

- To define an inner product, we require a definition of **conjugate transposition**. Conjugate transpose of a ket is called the '**bra**':

$$\langle\psi| = (|\psi\rangle^*)^T = |\psi\rangle^\dagger$$

- The symbol we use for conjugate transposition ($\dagger$) is the '**dagger**'.
- **Inner products** are done by closing the '*bra-ket*':

$$\langle\psi, \phi\rangle = \langle\psi|\phi\rangle$$

- **Expectations** are performed in the same way with the observable wedged in-between:

$$\langle\hat{Q}\rangle_\psi = \langle\psi|\hat{Q}|\psi\rangle$$

Dirac's Bras and Kets: Matrix Representation

- Operators that act on these kets/states/wave functions will adopt a **matrix representation**.

Dirac's Bras and Kets: Matrix Representation

- Operators that act on these kets/states/wave functions will adopt a **matrix representation**.
- The entries of these matrices for a given operator $\hat{Q}$ are:

$$[\hat{Q}]_{ij} = \langle \psi_i | \hat{Q} | \psi_j \rangle$$

Dirac's Bras and Kets: Matrix Representation

- Operators that act on these kets/states/wave functions will adopt a **matrix representation**.
- The entries of these matrices for a given operator $\hat{Q}$ are:

$$[\hat{Q}]_{ij} = \langle \psi_i | \hat{Q} | \psi_j \rangle$$

- It is imperative to **specify which basis** we are working with in order to construct the matrix representation of an operator.

Dirac's Bras and Kets: Matrix Representation

- Operators that act on these kets/states/wave functions will adopt a **matrix representation**.
- The entries of these matrices for a given operator $\hat{Q}$ are:

$$[\hat{Q}]_{ij} = \langle \psi_i | \hat{Q} | \psi_j \rangle$$

- It is imperative to **specify which basis** we are working with in order to construct the matrix representation of an operator.
- To be explicit, given some basis $\mathcal{B} = \{|\psi_1\rangle, |\psi_2\rangle, \dots, |\psi_n\rangle\}$, the matrix representation of $\hat{Q}$ will be:

$$\hat{Q} = \begin{bmatrix} \langle \psi_1 | \hat{Q} | \psi_1 \rangle & \langle \psi_1 | \hat{Q} | \psi_2 \rangle & \dots & \langle \psi_1 | \hat{Q} | \psi_n \rangle \\ \langle \psi_2 | \hat{Q} | \psi_1 \rangle & \langle \psi_2 | \hat{Q} | \psi_2 \rangle & \dots & \langle \psi_2 | \hat{Q} | \psi_n \rangle \\ \vdots & & \ddots & \vdots \\ \langle \psi_n | \hat{Q} | \psi_1 \rangle & \langle \psi_n | \hat{Q} | \psi_2 \rangle & \dots & \langle \psi_n | \hat{Q} | \psi_n \rangle \end{bmatrix}$$

Overview

1 TLDR; Classical Computers

2 Quantum Computers

TLDR; Classical Computers: Classical Machines

- There are many models of classical computation.

TLDR; Classical Computers: Classical Machines

- There are many models of classical computation.
- Some of these are *universal* while others are not.

TLDR; Classical Computers: Classical Machines

- There are many models of classical computation.
- Some of these are *universal* while others are not.
- This idea of *universal computation* was proposed by Alan Turing in 1936: any 'reasonable' computation can be solved by a *universal machine*.

TLDR; Classical Computers: Classical Machines

- There are many models of classical computation.
- Some of these are *universal* while others are not.
- This idea of *universal computation* was proposed by Alan Turing in 1936: any 'reasonable' computation can be solved by a *universal machine*.
- The **circuit model** is what we will be looking more closely at because it will pave the way for the circuit model of quantum computation.

TLDR; Classical Computers: Classical Machines

- There are many models of classical computation.
- Some of these are *universal* while others are not.
- This idea of *universal computation* was proposed by Alan Turing in 1936: any ‘reasonable’ computation can be solved by a *universal machine*.
- The **circuit model** is what we will be looking more closely at because it will pave the way for the circuit model of quantum computation.
- The circuit model is a generalization of the common *boolean circuitry* with universal gates (e.g. {AND, NOT}).

TLDR; Classical Computers: Byte-Sized Complexity Theory

- To appreciate the advantages of quantum computing, it would be useful to have an understanding of basic complexity theory.

TLDR; Classical Computers: Byte-Sized Complexity Theory

- To appreciate the advantages of quantum computing, it would be useful to have an understanding of basic complexity theory.
- For computational problems, if the number of operations is a polynomial function of the size of the input n , we call these class of problems *polynomial hard* ($\in P$).

TLDR; Classical Computers: Byte-Sized Complexity Theory

- To appreciate the advantages of quantum computing, it would be useful to have an understanding of basic complexity theory.
- For computational problems, if the number of operations is a polynomial function of the size of the input n , we call these class of problems *polynomial hard* ($\in P$).
- These problems are contained in the bigger class of *NP*-hard problems (nondeterministic polynomial time).

TLDR; Classical Computers: Byte-Sized Complexity Theory

- To appreciate the advantages of quantum computing, it would be useful to have an understanding of basic complexity theory.
- For computational problems, if the number of operations is a polynomial function of the size of the input n , we call these class of problems *polynomial hard* ($\in P$).
- These problems are contained in the bigger class of *NP*-hard problems (nondeterministic polynomial time).
- Finally, there is a special class of problems which are known as *NP*-complete. Solving *NP*-complete problems allows us to solve **all** *NP* problems.

TLDR; Classical Computers: Byte-Sized Complexity Theory

A set visualization of the aforementioned classes is given below.

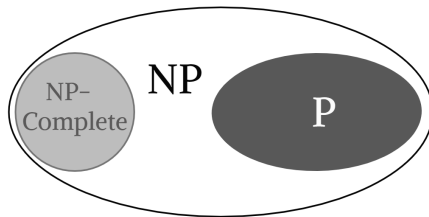


Figure: Complexity Classes

Overview

1 TLDR; Classical Computers

2 Quantum Computers

Quantum Computers: Today

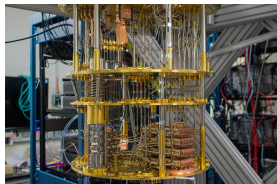


Figure: Bristlecone
(Google)

Quantum Computers: Today

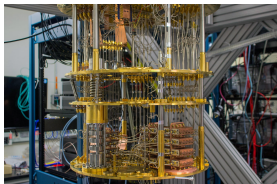


Figure: Bristlecone
(Google)

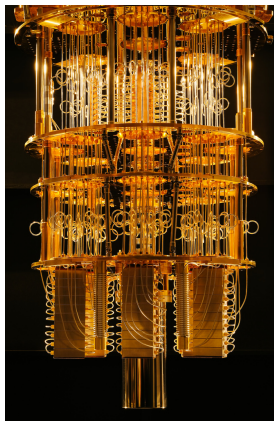


Figure: IBM's Quantum
Computer

Quantum Computers: Today

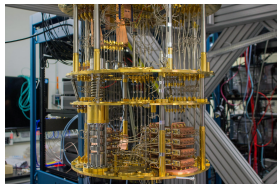


Figure: Bristlecone
(Google)

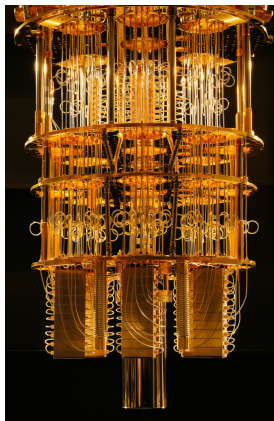


Figure: IBM's Quantum
Computer



Figure: D-Wave's
Quantum
Computer

Quantum Computers: Today



Figure: IBM's Commercial Quantum Computer

Quantum Computers: Today

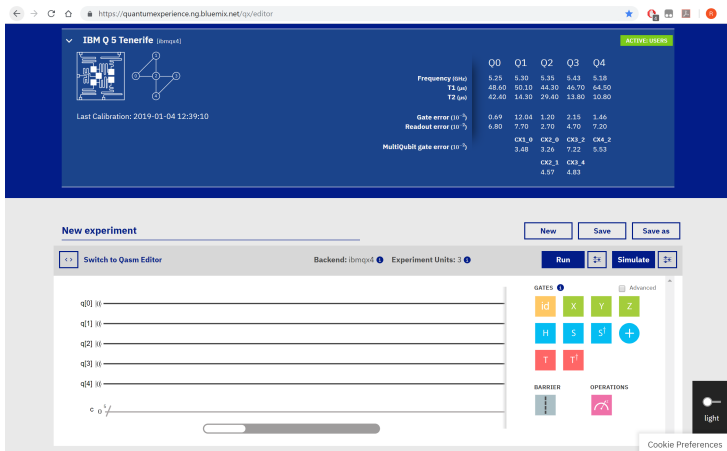
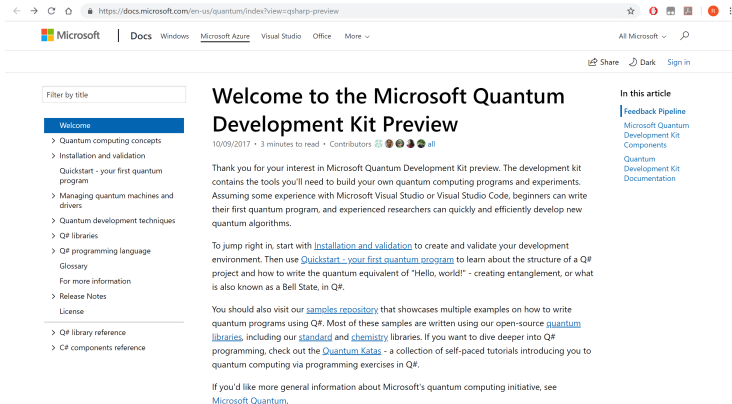


Figure: IBM QuantumExperience

Quantum Computers: Today



The screenshot shows a web browser displaying the Microsoft Quantum Development Kit Preview page. The browser's address bar shows the URL <https://docs.microsoft.com/en-us/quantum/index?view=qsharp-preview>. The Microsoft Docs navigation bar is visible at the top, with links for Windows, Microsoft Azure, Visual Studio, Office, and More. The main content area features a large heading "Welcome to the Microsoft Quantum Development Kit Preview" and a subheading "10/09/2017 • 3 minutes to read • Contributors". The page includes a sidebar with a "Filter by title" search box and a list of links under the "Welcome" section, such as "Quantum computing concepts", "Installation and validation", and "Managing quantum machines and drivers". The main text area contains a paragraph thanking the user for their interest and providing information about the development kit, followed by a section titled "To jump right in, start with" which links to "Installation and validation" and "Quickstart - your first quantum program". A right sidebar titled "In this article" contains links for "Feedback Pipeline", "Microsoft Quantum Development Kit Components", and "Quantum Development Kit Documentation".

← → ↺ 🏠 <https://docs.microsoft.com/en-us/quantum/index?view=qsharp-preview> ☆ 🔔 📄 🌙 ⌵

Microsoft | Docs Windows Microsoft Azure Visual Studio Office More ▾ All Microsoft 🔍

🔗 Share 🌙 Dark 📄 Sign in

Filter by title

Welcome to the Microsoft Quantum Development Kit Preview

10/09/2017 • 3 minutes to read • Contributors 🧑🏫 🧑🏻 🧑🏼 🧑🏽 🧑🏾 all

Thank you for your interest in Microsoft Quantum Development Kit preview. The development kit contains the tools you'll need to build your own quantum computing programs and experiments. Assuming some experience with Microsoft Visual Studio or Visual Studio Code, beginners can write their first quantum program, and experienced researchers can quickly and efficiently develop new quantum algorithms.

To jump right in, start with [Installation and validation](#) to create and validate your development environment. Then use [Quickstart - your first quantum program](#) to learn about the structure of a Q# project and how to write the quantum equivalent of "Hello, world!" - creating entanglement, or what is also known as a Bell State, in Q#.

You should also visit our [samples repository](#) that showcases multiple examples on how to write quantum programs using Q#. Most of these samples are written using our open-source [quantum libraries](#), including our [standard](#) and [chemistry](#) libraries. If you want to dive deeper into Q# programming, check out the [Quantum Katas](#) - a collection of self-paced tutorials introducing you to quantum computing via programming exercises in Q#.

If you'd like more general information about Microsoft's quantum computing initiative, see [Microsoft Quantum](#).

In this article

- [Feedback Pipeline](#)
- [Microsoft Quantum Development Kit Components](#)
- [Quantum Development Kit Documentation](#)

Filter by title

- Welcome
- > Quantum computing concepts
- > Installation and validation
 - Quickstart - your first quantum program
- > Managing quantum machines and drivers
- > Quantum development techniques
- > Q# libraries
- > Q# programming language
 - Glossary
 - For more information
- > Release Notes
 - License
- > Q# library reference
- > C# components reference

Figure: Microsoft Q# Development Kit

Quantum Computers: The Qubit

- The fundamental constituent of a quantum computer is a quantum bit (**qubit**).

Quantum Computers: The Qubit

- The fundamental constituent of a quantum computer is a quantum bit (**qubit**).
- In theory, any 2-level quantum system can be realized as an implementation of a qubit (e.g. Mach-Zehnder interferometer $\{|u\rangle, |d\rangle\}$).

Quantum Computers: The Qubit

- The fundamental constituent of a quantum computer is a quantum bit (**qubit**).
- In theory, any 2-level quantum system can be realized as an implementation of a qubit (e.g. Mach-Zehnder interferometer $\{|u\rangle, |d\rangle\}$).
- A commonly adopted 2-level system is the *spin* of spin- $\frac{1}{2}$ particles. (think of quantum spin as the 2-level quantized version of the classical spin for a charged sphere).

Quantum Computers: The Qubit

- The fundamental constituent of a quantum computer is a quantum bit (**qubit**).
- In theory, any 2-level quantum system can be realized as an implementation of a qubit (e.g. Mach-Zehnder interferometer $\{|u\rangle, |d\rangle\}$).
- A commonly adopted 2-level system is the *spin* of spin- $\frac{1}{2}$ particles. (think of quantum spin as the 2-level quantized version of the classical spin for a charged sphere).
- This quantization allows us to represent the *orthogonal* states of our spin qubit as follows:

$$|0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \quad |1\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

Quantum Computers: The Qubit

- We have 2 vector spaces in which one is 'embedded' in the other. These vector spaces are:

Quantum Computers: The Qubit

- We have 2 vector spaces in which one is 'embedded' in the other. These vector spaces are:
 - The **2D** vector space of **spin states** (quantum state space).

Quantum Computers: The Qubit

- We have 2 vector spaces in which one is 'embedded' in the other. These vector spaces are:
 - The **2D** vector space of **spin states** (quantum state space).
 - The **3D** vector space of **rotations** (real space).

Quantum Computers: The Qubit

- We have 2 vector spaces in which one is 'embedded' in the other. These vector spaces are:
 - The **2D** vector space of **spin states** (quantum state space).
 - The **3D** vector space of **rotations** (real space).
- This means that in any arbitrary direction in real space, there lives a 2D complex vector space.

Quantum Computers: The Qubit

- We have 2 vector spaces in which one is 'embedded' in the other. These vector spaces are:
 - The **2D** vector space of **spin states** (quantum state space).
 - The **3D** vector space of **rotations** (real space).
- This means that in any arbitrary direction in real space, there lives a 2D complex vector space.
- A method of visualizing is known as the *Bloch sphere* geometric representation. This will not be covered.

Quantum Computers: The Qubit

- We have 2 vector spaces in which one is 'embedded' in the other. These vector spaces are:
 - The **2D** vector space of **spin states** (quantum state space).
 - The **3D** vector space of **rotations** (real space).
- This means that in any arbitrary direction in real space, there lives a 2D complex vector space.
- A method of visualizing is known as the *Bloch sphere* geometric representation. This will not be covered.
- I give an alternative means of visualization that may be useful and I find rather intuitive.

Quantum Computers: The Qubit

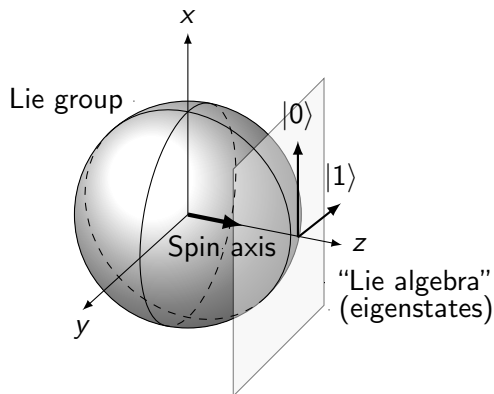


Figure: Visualization of Spin Rotations

Quantum Computers: Quantum Gates

- As in classical computers, we now want to be able to perform operations on our qubits.

Quantum Computers: Quantum Gates

- As in classical computers, we now want to be able to perform operations on our qubits.
- These can be done in the form of quantum gates (similar to logic gates in the classical circuit model).

Quantum Computers: Quantum Gates

- As in classical computers, we now want to be able to perform operations on our qubits.
- These can be done in the form of quantum gates (similar to logic gates in the classical circuit model).
- Physical operations to a closed system in quantum mechanics must be unitary, so we need to look for unitary operators which can act as our quantum gates.

Quantum Computers: Quantum Gates

- As in classical computers, we now want to be able to perform operations on our qubits.
- These can be done in the form of quantum gates (similar to logic gates in the classical circuit model).
- Physical operations to a closed system in quantum mechanics must be unitary, so we need to look for unitary operators which can act as our quantum gates.
- When adopting spin as qubits, we are given some nice tools to work with related to spin observables (Pauli and identity matrices).

Quantum Computers: Quantum Gates

- As in classical computers, we now want to be able to perform operations on our qubits.
- These can be done in the form of quantum gates (similar to logic gates in the classical circuit model).
- Physical operations to a closed system in quantum mechanics must be unitary, so we need to look for unitary operators which can act as our quantum gates.
- When adopting spin as qubits, we are given some nice tools to work with related to spin observables (Pauli and identity matrices).

$$\mathbb{I} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad \sigma_x = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \quad \sigma_y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}, \quad \sigma_z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$$

Quantum Computers: Quantum Gates

- The eigenstates of the Pauli matrices (in the z-basis) are:

$$\hat{\sigma}_x : |+\rangle = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \quad |-\rangle = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ -1 \end{bmatrix}$$

$$\hat{\sigma}_y : |\uparrow\rangle = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ i \end{bmatrix}, \quad |\downarrow\rangle = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ -i \end{bmatrix}$$

$$\hat{\sigma}_z : |0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \quad |1\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

Quantum Computers: Quantum Gates

- The eigenstates of the Pauli matrices (in the z-basis) are:

$$\hat{\sigma}_x : |+\rangle = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \quad |-\rangle = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ -1 \end{bmatrix}$$

$$\hat{\sigma}_y : |\uparrow\rangle = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ i \end{bmatrix}, \quad |\downarrow\rangle = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ -i \end{bmatrix}$$

$$\hat{\sigma}_z : |0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \quad |1\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

- The Pauli matrices follow the following commutation relation called the **spin algebra**:

$$[\hat{\sigma}_i, \hat{\sigma}_j] = 2i\epsilon_{ijk}\hat{\sigma}_k$$

Quantum Computers: Quantum Gates

- The eigenstates of the Pauli matrices (in the z-basis) are:

$$\hat{\sigma}_x : |+\rangle = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \quad |-\rangle = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ -1 \end{bmatrix}$$

$$\hat{\sigma}_y : |\uparrow\rangle = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ i \end{bmatrix}, \quad |\downarrow\rangle = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ -i \end{bmatrix}$$

$$\hat{\sigma}_z : |0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \quad |1\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

- The Pauli matrices follow the following commutation relation called the **spin algebra**:

$$[\hat{\sigma}_i, \hat{\sigma}_j] = 2i\epsilon_{ijk}\hat{\sigma}_k$$

- Pauli matrices are all **unitary**, **Hermitian** and **involutory**.

Quantum Computers: Quantum Gates

Check the commutation relations of the Pauli matrices. What are their anti-commutation relations?

(*Hint: The anti-commutator is defined as $\{\hat{A}, \hat{B}\} = \hat{A}\hat{B} + \hat{B}\hat{A}$.)*

Quantum Computers: Quantum Gates

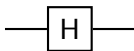


Figure: Hadamard Gate

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

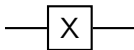


Figure: Pauli-X/NOT Gate

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

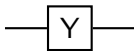


Figure: Pauli-Y Gate

$$\begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}$$

Quantum Computers: Quantum Gates



Figure: Pauli-Z Gate

$$\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$$



Figure: Phase Gate

$$\begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix}$$



Figure: $\pi/8$ Gate

$$\begin{bmatrix} 1 & 0 \\ 0 & e^{i\pi/4} \end{bmatrix}$$

Quantum Computers: Quantum Gates

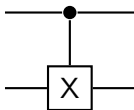


Figure: CNOT Gate

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

Break

Quantum Computers: More Qubits

- Having a computer (even a quantum one) with one bit (qubit) isn't very useful.

Quantum Computers: More Qubits

- Having a computer (even a quantum one) with one bit (qubit) isn't very useful.
- In quantum computers, the addition of more qubits scales the computational capacity exponential (unlike in classical computers).

Quantum Computers: More Qubits

- Having a computer (even a quantum one) with one bit (qubit) isn't very useful.
- In quantum computers, the addition of more qubits scales the computational capacity exponential (unlike in classical computers).
- For multi-particle (many-body) quantum systems, we require the use of **tensor products** (presented as **Kronecker products**).

Quantum Computers: More Qubits

- Having a computer (even a quantum one) with one bit (qubit) isn't very useful.
- In quantum computers, the addition of more qubits scales the computational capacity exponential (unlike in classical computers).
- For multi-particle (many-body) quantum systems, we require the use of **tensor products** (presented as **Kronecker products**).
- This 'expands' the Hilbert space of our quantum system $\rightarrow$ given the Hilbert spaces of each sub-quantum system being $\{\mathcal{H}_1, \dots, \mathcal{H}_n\}$, then the Hilbert space of the total system is given by $\mathcal{H}_1 \otimes \dots \otimes \mathcal{H}_n$, with $\dim(\mathcal{H}_1 \otimes \dots \otimes \mathcal{H}_n) = \dim(\mathcal{H}_1) \dots \dim(\mathcal{H}_n)$.

Definition

Given 2 vectors:

$$|\psi\rangle = \begin{bmatrix} \vdots \\ \psi_j \\ \psi_{j+1} \\ \vdots \end{bmatrix}, \quad |\phi\rangle = \begin{bmatrix} \vdots \\ \phi_j \\ \phi_{j+1} \\ \vdots \end{bmatrix}$$

with $|\psi\rangle \in V$ and $|\phi\rangle \in W$, the **Kronecker product** of $|\psi\rangle$ and $|\phi\rangle$ is given by:

$$|\psi\rangle \otimes |\phi\rangle = |\psi\rangle |\phi\rangle = |\psi\phi\rangle = \begin{bmatrix} \vdots \\ \psi_j |\phi\rangle \\ \psi_{j+1} |\phi\rangle \\ \vdots \end{bmatrix}$$

Definition

Given 2 matrices:

$$\hat{A} = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & & \ddots & \vdots \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{bmatrix}, \quad \hat{B} = \begin{bmatrix} b_{11} & b_{12} & \dots & b_{1m} \\ b_{21} & b_{22} & \dots & b_{2m} \\ \vdots & & \ddots & \vdots \\ b_{m1} & b_{m2} & \dots & b_{mm} \end{bmatrix}$$

the **Kronecker product** of $\hat{A}$ and $\hat{B}$ is defined as:

$$\hat{A} \otimes \hat{B} = \begin{bmatrix} a_{11}\hat{B} & a_{12}\hat{B} & \dots & a_{1n}\hat{B} \\ a_{21}\hat{B} & a_{22}\hat{B} & \dots & a_{2n}\hat{B} \\ \vdots & & \ddots & \vdots \\ a_{n1}\hat{B} & a_{n2}\hat{B} & \dots & a_{nn}\hat{B} \end{bmatrix}$$

The action of these operators on states in the larger space is:

$$(\hat{A}_1 \otimes \dots \otimes \hat{A}_n)(|\psi\rangle_1 \otimes \dots \otimes |\psi\rangle_n) = \hat{A}_1 |\psi\rangle_1 \otimes \dots \otimes \hat{A}_n |\psi\rangle_n$$

Quantum Computers: Quantum Circuits

We now introduce the formalism of quantum circuit diagrams. Quantum circuits are drawn in the following steps:

Quantum Computers: Quantum Circuits

We now introduce the formalism of quantum circuit diagrams. Quantum circuits are drawn in the following steps:

- Specify the qubit inputs.

Quantum Computers: Quantum Circuits

We now introduce the formalism of quantum circuit diagrams. Quantum circuits are drawn in the following steps:

- Specify the qubit inputs.
- Connect quantum wires from inputs to outputs.

Quantum Computers: Quantum Circuits

We now introduce the formalism of quantum circuit diagrams. Quantum circuits are drawn in the following steps:

- Specify the qubit inputs.
- Connect quantum wires from inputs to outputs.
- Specify intermediate gates to take inputs to outputs.

Quantum Computers: Quantum Circuits

We now introduce the formalism of quantum circuit diagrams. Quantum circuits are drawn in the following steps:

- Specify the qubit inputs.
- Connect quantum wires from inputs to outputs.
- Specify intermediate gates to take inputs to outputs.
- Include any classical post-processing required.

Quantum Computers: Quantum Circuits

We now introduce the formalism of quantum circuit diagrams. Quantum circuits are drawn in the following steps:

- Specify the qubit inputs.
- Connect quantum wires from inputs to outputs.
- Specify intermediate gates to take inputs to outputs.
- Include any classical post-processing required.

Example:

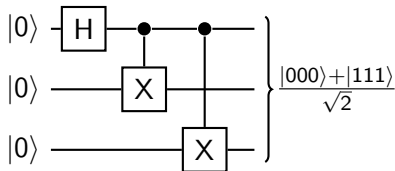


Figure: Quantum Circuit for GHZ State

Quantum Computers: Entanglement

- Consider the state:

$$|\Phi^+\rangle = \frac{|0\rangle_A \otimes |0\rangle_B + |1\rangle_A \otimes |1\rangle_B}{\sqrt{2}} = \frac{|00\rangle_{AB} + |11\rangle_{AB}}{\sqrt{2}}$$

Quantum Computers: Entanglement

- Consider the state:

$$|\Phi^+\rangle = \frac{|0\rangle_A \otimes |0\rangle_B + |1\rangle_A \otimes |1\rangle_B}{\sqrt{2}} = \frac{|00\rangle_{AB} + |11\rangle_{AB}}{\sqrt{2}}$$

- This state is unique because if you measure and determined the state in A , the state in B is automatically known!

Quantum Computers: Entanglement

- Consider the state:

$$|\Phi^+\rangle = \frac{|0\rangle_A \otimes |0\rangle_B + |1\rangle_A \otimes |1\rangle_B}{\sqrt{2}} = \frac{|00\rangle_{AB} + |11\rangle_{AB}}{\sqrt{2}}$$

- This state is unique because if you measure and determined the state in A , the state in B is automatically known!
- There seems to be this binding correlation between particles A and B no matter the distance between them. This phenomena is what physicist term **quantum entanglement**.

Quantum Computers: Teleportation

- We can now look at the first quantum protocol, **quantum teleportation**.

Quantum Computers: Teleportation

- We can now look at the first quantum protocol, **quantum teleportation**.
- It has use because of the **no-cloning theorem**:

Theorem

*The no-cloning theorem states that it is impossible to create an identical copy of some arbitrary **unknown** quantum state $|\psi\rangle$.*

Quantum Computers: Teleportation

- We can now look at the first quantum protocol, **quantum teleportation**.
- It has use because of the **no-cloning theorem**:

Theorem

*The no-cloning theorem states that it is impossible to create an identical copy of some arbitrary **unknown** quantum state $|\psi\rangle$.*

- In this protocol the goal is transmitting quantum information between 2 individuals (Alice and Bob) who agree on a prior scheme.

Quantum Computers: Teleportation

- We can now look at the first quantum protocol, **quantum teleportation**.
- It has use because of the **no-cloning theorem**:

Theorem

*The no-cloning theorem states that it is impossible to create an identical copy of some arbitrary **unknown** quantum state $|\psi\rangle$.*

- In this protocol the goal is transmitting quantum information between 2 individuals (Alice and Bob) who agree on a prior scheme.
- This will be done by the sending classical bits and utilizing quantum entangle.

The Protocol:

The Protocol:

- Alice has a single qubit quantum state which she wants to send over to Bob. Alice and Bob also share a 2-qubit entangled state, which means Alice has 2 qubits and Bob, 1.

The Protocol:

- Alice has a single qubit quantum state which she wants to send over to Bob. Alice and Bob also share a 2-qubit entangled state, which means Alice has 2 qubits and Bob, 1.
- Alice then performs a measurement in the **Bell basis** on her 2 qubits which grants her 2 classical bits.

The Protocol:

- Alice has a single qubit quantum state which she wants to send over to Bob. Alice and Bob also share a 2-qubit entangled state, which means Alice has 2 qubits and Bob, 1.
- Alice then performs a measurement in the **Bell basis** on her 2 qubits which grants her 2 classical bits.
- Alice then sends her 2 classical bits over a classical channel to Bob.

The Protocol:

- Alice has a single qubit quantum state which she wants to send over to Bob. Alice and Bob also share a 2-qubit entangled state, which means Alice has 2 qubits and Bob, 1.
- Alice then performs a measurement in the **Bell basis** on her 2 qubits which grants her 2 classical bits.
- Alice then sends her 2 classical bits over a classical channel to Bob.
- Bob receives the 2 classical bits, and performs the necessary unitary on his qubit according to a previously agreed upon rule, obtaining Alice's original qubit. The state has been effectively teleported.

Quantum Computers: Teleportation

To do a measurement in the Bell basis, we run our outputs through a circuit that allows us to measure in the computational basis:

A CNOT gate from the 1st to the 2nd qubit, then a Hadamard to the 1st qubit.

$$|\Phi^+\rangle = \frac{|00\rangle + |11\rangle}{\sqrt{2}} \xrightarrow{\text{CNOT}} \frac{|00\rangle + |10\rangle}{\sqrt{2}} = |+\rangle |0\rangle \xrightarrow{H \otimes I} |00\rangle$$

$$|\Psi^+\rangle = \frac{|01\rangle + |10\rangle}{\sqrt{2}} \xrightarrow{\text{CNOT}} \frac{|01\rangle + |11\rangle}{\sqrt{2}} = |+\rangle |1\rangle \xrightarrow{H \otimes I} |01\rangle$$

$$|\Phi^-\rangle = \frac{|00\rangle - |11\rangle}{\sqrt{2}} \xrightarrow{\text{CNOT}} \frac{|00\rangle - |10\rangle}{\sqrt{2}} = |-\rangle |0\rangle \xrightarrow{H \otimes I} |10\rangle$$

$$|\Psi^-\rangle = \frac{|01\rangle - |10\rangle}{\sqrt{2}} \xrightarrow{\text{CNOT}} \frac{|01\rangle - |11\rangle}{\sqrt{2}} = |-\rangle |1\rangle \xrightarrow{H \otimes I} |11\rangle$$

Quantum Computers: Teleportation

After sending the classical bits, we apply the respective **recovery operators**:

Quantum Computers: Teleportation

After sending the classical bits, we apply the respective **recovery operators**:

$$00 \rightarrow \mathbb{I}$$

$$01 \rightarrow \sigma_x$$

$$10 \rightarrow \sigma_z$$

$$11 \rightarrow \sigma_x \sigma_z$$

Quantum Computers: Teleportation

After sending the classical bits, we apply the respective **recovery operators**:

$$00 \rightarrow \mathbb{I}$$

$$01 \rightarrow \sigma_x$$

$$10 \rightarrow \sigma_z$$

$$11 \rightarrow \sigma_x \sigma_z$$

The Q-teleportation protocol as a circuit is given below:

Quantum Computers: Teleportation

After sending the classical bits, we apply the respective **recovery operators**:

$$00 \rightarrow \mathbb{I}$$

$$01 \rightarrow \sigma_x$$

$$10 \rightarrow \sigma_z$$

$$11 \rightarrow \sigma_x \sigma_z$$

The Q-teleportation protocol as a circuit is given below:

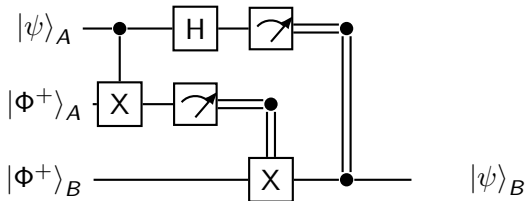


Figure: Quantum Teleportation Circuit

Thank you!

<https://tinyurl.com/TQWday3>