

Project abstracts

PWF: Physics REBoot Venezuela

Bootcamp #3 : Quantum Information Science & Technology

1. El algoritmo de Shor (dificultad media)

Mentor: A definir

Todo número puede ser descompuesto en un producto de números primos. Esta descomposición es uno de los problemas más difíciles en la historia de las matemáticas y ciencias de la computación. Los mejores algoritmos desarrollados hoy en día requieren de un número muy grande de pasos que crece exponencialmente. Esto implica que a partir de un cierto valor del número a descomponer, el número de pasos requeridos es tan grande que ni las mejores supercomputadoras pueden ser usadas para llevar a cabo la factorización en un tiempo relativamente finito. Gran parte de los protocolos de criptografía se basan en esta técnica de encriptación usando números primos que representan una gran dificultad de descryptación.

En 1994, Peter Shor desarrolló un algoritmo cuántico para factorizar un entero más eficientemente que cualquier algoritmo clásico conocido. Esto representaba entonces una de las primeras señales de una ventaja cuántica y el comienzo del desarrollo e investigación en algoritmos cuánticos. Aún así, en el algoritmo de Shor se requieren de muchos qubits antes de poder romper los protocolos de seguridad y factorizar números primos muy grandes.

El algoritmo ha sido demostrado con dispositivos NISQ, y el récord hasta este día es la factorización del número 21 en 2021 [2].

El desafío consiste en entender el [algoritmo de Shor](#) y explicar cómo funciona. Para ello se puede crear un circuito cuántico que factorice el número 15.

Objetivos:

- Demostrar un algoritmo que sea más eficiente que uno clásico e.g., algoritmos de Deutsch-Josza, de Simon o de Bernstein-Vazirani.
- Explicar con diagramas como funciona el algoritmo de Shor
- Escribir un algoritmo para factorizar el número 15
- Demostrar su funcionamiento en un procesador cuántico real

Software recomendado: Qiskit

Referencias y enlaces:

[1] P. Shor, Proceedings 35th annual symposium on foundations of computer science 1994:


<https://ieeexplore.ieee.org/document/365700>

[2] Skosana & Tame 2021 SciRep 2021: <https://www.nature.com/articles/s41598-021-95973-w>

[3] Demostración para el número 15, Vandersypen et al, Nature 2001: 10.1038/414883a

[4] Nielsen and Chuang, [Quantum Computation and Quantum Information](#), Cambridge University Press

[5] Qiskit textbook: <https://qiskit.org/textbook/preface.html>

[6] Minute Physics:  How Quantum Computers Break Encryption | Shor's Algorithm Explained

2. Cubitización (dificultad media)

Mentor: M.Gómez Vilorio

Los procesadores cuánticos consisten de muchos sistemas de dos niveles (qubits) que pueden ser utilizados directamente para modelar la interacción de algunos problemas físicos. Por ejemplo, las propiedades ferromagnéticas de un imán se pueden modelar como una red de espines en interacción, donde cada espín representa un pequeño momento magnético.

Aun así, no todos los problemas en física cuántica se tratan con sistemas de dos niveles. La energía fundamental de una molécula depende del espín pero también de los niveles electrónicos ($N > 2$) de los electrones. Nos hace falta traducir un problema de partículas en un problema de sistemas a dos niveles. Este proceso se conoce como cubitización (del inglés *qubitization*). Para ello, se han desarrollado diferentes tipos de transformaciones, la más conocida se conoce como la [transformación de Jordan-Wigner](#) que traduce un problema de electrones (fermiones) en un problema de qubits. Así, un investigador es capaz de escribir un Hamiltoniano electrónico, transformarlo y simularlo directamente en una computadora cuántica.

Para este desafío, estudiaremos de qué se trata la transformación de Jordan-Wigner. Aprenderás que es la [segunda cuantización](#) y las propiedades de los fermiones. Aplicando lo que aprendas podrás expresar el Hamiltoniano de un sistema simple en términos de qubits para poder analizarlo en un procesador cuántico.

Objetivos:

- Entender las propiedades de los fermiones ¿en qué se diferencian de los bosones?
- Entender la segunda cuantización para los fermiones, aprender las identidades básicas del álgebra de anticonmutadores
- Entender la transformación de Wigner-Jordan y las propiedades de las cadenas de Pauli
- Demostrar su funcionamiento traducir un problema simple a dos electrones, si posible demostrar su funcionamiento en un procesador cuántico

QDK recomendado: Qiskit, Qutip, Azure Quantum

Referencias y enlaces:

- a) Fermiones y bosones: [Fermions and bosons](#)
- b) Segunda cuantización: [Second Quantization - Azure Quantum | Microsoft Docs](#) y [Segunda cuantificación Estados cuánticos de muchos cuerpos Operadores de creación y aniquilación](#)
- c) Jordan Wigner: [Jordan-Wigner Representation - Azure Quantum | Microsoft Docs](#) y [Simple examples of second quantization](#)

3. Diseño de un simulador cuántico (cualquier nivel)

Mentor: A definir

Richard Feynman propuso simular la mecánica cuántica usando sistemas cuánticos. Para desarrollar esta tecnología es necesario comprender cómo se comportan diferentes aspectos de la física existente de la interacción de los distintos componentes de la materia y cómo interactúan con otros como la luz. Esto puede permitir simular sistemas cuánticos de muchos cuerpos usando sistemas análogos que tengan comportamientos similares pero que son más accesibles de controlar. Para poder construir estos simuladores es necesario entender varias las facetas claves del experimento.

La serie de capas de complejidad de una computadora cuántica se conoce en inglés como el *quantum stack* (el pilón cuántico). El quantum stack va desde los algoritmos cuánticos al nivel de abstracción más alto, pasando por los programas que permiten traducir las instrucciones en pulsos que permiten ejecutar el experimento, hasta la capa más baja que es la física/experimental de la arquitectura.

En este desafío proponemos estudiar las capas más bajas del *stack*. Las preguntas a responder son: ¿Qué puede ser simulado con los sistemas actuales? Y ¿Cómo se llevan a cabo? ¿Cuáles son las limitaciones y cómo sobrepasarlas? ¿Qué tipos de simuladores se pueden construir?

Objetivos:

- Presenta el funcionamiento de las diferentes arquitecturas cuánticas y simuladores cuánticos.
- Escoge una arquitectura y presenta en detalles su funcionamiento y sus limitaciones
- Piensa y explora un problema de química, ingeniería, finanzas, u otro que pueda ser simulado en la arquitectura que escogiste ¿Cómo llevarías a cabo la simulación?
- Diseña la serie de instrucciones que serían necesarias para controlar la arquitectura escogida

Referencias y enlaces:

[1] Nielsen and Chuang, [Quantum Computation and Quantum Information](#), Cambridge University Press

[2] T. Wong, Introduction to Classical and Quantum Computing, Rooted Groove

4. Algoritmo variacional para la molécula de hidrógeno (dificultad media)

Mentor: A definir

El sueño de Richard Feynman era el de simular la física cuántica usando verdaderos sistemas cuánticos. Este sueño se está llevando a cabo, pero por los momentos estamos en lo que se conoce como la era NISQ o era cuántica ruidosa y de escala intermedia. Ruidosa porque los procesadores cuánticos tienen muchos errores relacionados con ruido externo y decoherencia cuántica. De escala intermedia porque para poder llevar a cabo algoritmos que demuestran la ventaja cuántica (como el algoritmo de Shor) se necesitan decenas de miles de qubits, y por los momentos los mejores procesadores tienen apenas unas centenas de qubits.

En 2014, Alberto Peruzzo y sus colaboradores presentaron un algoritmo híbrido bajo el nombre de VQE (*Variational quantum eigensolver*, solucionador variacional de autoestados) [a]. La idea consiste en utilizar métodos variacionales (suponer una solución y optimizarla) para conseguir las energías fundamentales de un sistema usando haciendo colaborar un procesador clásico con uno cuántico. De esta manera la parte cuántica del algoritmo puede correr sin problemas en un procesador cuántico ruidoso (NISQ) mientras que la optimización se hace usando optimizadores usuales.

Para este desafío queremos conseguir la energía fundamental de una molécula de hidrógeno (H_2) usando el algoritmo VQE. Para ello necesitarás conseguir la energía del sistema de manera clásica y luego crear un programa cuántico-clásica.

Objetivos:

- Utilizar una herramienta para conseguir la energía fundamental de la molécula de hidrógeno. Puedes usar PySCF [b], no es necesario explicar cómo funciona.
- Explicar cómo funciona el algoritmo VQE
- Implementar el algoritmo VQE para la molécula de hidrógeno y comparar con tu solución clásica. Mientras más pequeño sea el circuito mejor. ¿Puede aplicar el mismo tratamiento a otra molécula?
- Demostrar tu algoritmo en un verdadero procesador cuántico

SDK recomendado: Qiskit

Referencias y enlaces:

[a] A. Peruzzo et al, Nature Comm. (2014): 10.1038/ncomms5213 (2014)

[b] PySCF: <https://pyscf.org/>

[c] VQE qiskit: <https://qiskit.org/textbook/ch-applications/vqe-molecules.html>

5. Contribuye a un paquete de software cuántico (nivel básico)

Mentor: A definir

El mundo de la computación cuántica y del desarrollo tecnológico y científico se basa en la colaboración. Los SDK (software development kits) son recursos para modelar y escribir rutinas para trabajar algoritmos cuánticos, correr experimentos y modelar estos sistemas. Todos los recursos que hemos presentado en este bootcamp consisten de proyectos abiertos a la colaboración externa (*open source*).

La manera de contribuir a estos proyectos es el uso de repositorios accesibles al usuario. Páginas como [Github](#) y [Gitlab](#) permiten a los usuarios compartir códigos de programas, comentar estos códigos y colaborar mejorando las rutinas y detalles del código. La herramienta que lo permite se conoce como [git](#), un software que permite controlar una serie de archivos llevando un historial de versiones y sincronizarlos con un contenido en línea.

Este desafío consiste en aprender a usar git y aportar una pequeña contribución a un proyecto existente

Objetivos:

- Crear tu propio repositorio y demuestra el uso de git (puedes usar github o gitlab)
- Acceder a los repositorios de los proyectos presentados en el bootcamp y prepara una contribución que te gustaría llevar a cabo
- Hacer una contribución a uno de los repositorios (si es aceptaba mejor)
- Crear tu propio paquete de simulación de un sistema cuántico y compártelo al público. Debería poderse descargar y usar fácilmente.

Referencias:

- a) Guía de github: <https://docs.github.com/es/get-started>
- b) Guía de git: <https://m-monroyc22.medium.com/gu%C3%ADa-b%C3%A1sica-git-9f5c35f5acef>
- c) Qiskit: <https://github.com/Qiskit>
- d) Qutip: <https://github.com/qutip>
- e) QNE: <https://github.com/QuTech-Delft/qne-adk>
- f) AQIPT: <https://github.com/manuelmorgado/AQiPT>
- g) Quantum Open Source Foundation: <https://github.com/qosf/monthly-challenges>

6. Distribuir claves secretas usando cuántica (nivel básico)

Mentor: A definir

Uno de los problemas más fundamentales de la telecomunicación consiste en crear un protocolo para enviar una información secreta sin que sea leída por un tercero. Cualquier mensaje clásico puede ser interceptado, leído, copiado y modificado. Una manera de evitar algunos de estos problemas consiste en enviar un mensaje encriptado. Con un código secreto se hace más difícil leerlo y modificarlo (pero con suficiente tiempo se puede descryptar el mensaje).

La mecánica cuántica permite una solución a este problema. En 1984, Charles Bennet y Gilles Brassard desarrollaron un protocolo para enviar mensajes secretos utilizando canales cuánticos[a]. Los sistemas cuánticos son tan sensibles que cualquier interacción con un sistema macroscópico colapsa el estado del sistema. Así que interceptar el sistema modifica el resultado final. Más aún, el [teorema de no clonación](#) dice que ningún estado cuántico puede ser duplicado, esto le hace imposible copiar el mensaje para analizarlo posteriormente. De esta manera un protocolo cuántico es lo más seguro posible.

Este desafío consiste en desarrollar un protocolo para la distribución de claves secretas usando redes cuánticas.

Objetivos:

- El protocolo más simple para enviar información usando cuántica se conoce como [codificación superdensa](#). Crear un algoritmo que lleve a cabo este protocolo.
- Crear un programa que lleve a cabo el protocolo BB84 (Bennet-Brassard 1984).
- Crea un algoritmo para intercambio de entrelazamiento (entanglement swapping)
- Implementar protocolo en sistemas reales usando el Quantum Network Explorer [b]

Referencias y enlaces:

[a] Bennet, Brassard, Proceedings of the IEEE International Conference on Computers, Systems and Signal Processing, 1984:

<https://researcher.watson.ibm.com/researcher/files/us-bennetc/BB84highest.pdf>

[b] QNE: <https://www.quantum-network.com/>

[c] Bennet, Brassard, preprint 2020: <https://arxiv.org/abs/2003.06557>

7. Dinámica de un qubit único usando pulsos (dificultad media)

Mentor: M.Morgado

El formalismo de circuitos cuánticos y de compuertas unitarias se fundamenta en una implementación perfecta en arquitecturas cuánticas perfectas o que se encuentran en el régimen de tolerancia a fallos. Sin embargo, para lograr que los sistemas físicos cumplan tal régimen, requieren tener un nivel de controlabilidad total y alto nivel de coherencia.

Diferentes procesos no unitarios afectan la dinámica de los bit cuánticos. Mientras que la evolución temporal de la dinámica de qubits perfectos está definida por la ecuación de Schrödinger, en la ecuación de Lindblad representa un formalismo en el cual se incluyen procesos Markovianos disipativos. Estas ecuaciones tienen como requisito fundamental el Hamiltoniano que describe el sistema en todo tiempo. Sin embargo, para imitar los Hamiltonianos o efectos físicos sobre el sistema es necesario tomar en cuenta una función temporal que module el valor del Hamiltoniano en el tiempo dentro de la ecuación. Esto se conoce como el nivel de pulsos.

Este desafío consiste en realizar un estudio de la dinámica de los sistemas físicos que representan los qubits y ver cuales son los efectos más interesantes.

Objetivos:

- Definir el sistema físico y el qubit embebido en el sistema.
- Definir el Hamiltoniano del qubit.
- Modelar la dinámica del qubit para un Hamiltoniano independiente del tiempo.
- Modelar la dinámica del qubit con pulsos (Hamiltoniano dependiente del tiempo).
- ¿Se te ocurre cómo hacer compuertas?

Referencias y enlaces

[1] QuTiP

8. Divulgación cuántica (cualquier nivel)

Mentor: libre

En el auge de la informática cuántica es necesario divulgar los fundamentos, las maravillas y las dificultades del desarrollo de tecnologías cuánticas. Un buen divulgador no sólo puede traducir un problema complicado en una motivación comprensible para todo público sino que también enseña y desmitifica las ideas erróneas que uno puede tener sobre un tema científico.





Este desafío consiste en preparar una buena presentación sobre un tema abierto de tu interés. Tu presentación deberá mencionar datos históricos, proveer una motivación sobre el tema, usar imágenes que ilustren el tema y usar un lenguaje apto para todo público.

Objetivos:

- Preparar una presentación de divulgación sobre un fundamento de mecánica cuántica
- Preparar una presentación sobre por qué es interesante la información y la computación cuántica
- Explicar un tema avanzado (puedes usar temas de otros desafíos o preguntar a los expertos, tutores y organizadores).
- Crear una presentación interactiva utilizando un verdadero procesador cuántico

Referencias: tus conocimientos y tus contactos

Algunos videos que te pueden inspirar:

1.  [How Quantum Computers Break Encryption | Shor's Algorithm Explained](#) MinutePhysics
2.  [Todo lo que un Qubit puede Enseñarte sobre Física Cuántica](#) QuantumFracture
3.  [How Does a Quantum Computer Work?](#) Veritasium
4.  [Quantum Computers Explained – Limits of Human Technology](#) Kurzgesagt

9. Transpilador y compuertas universales (nivel básico)

Mentor: A definir

Un circuito cuántico consiste en un conjunto de compuertas cuánticas que se aplican en serie. Para llevar a cabo este concepto en procesadores reales hay que tener en cuenta dos aspectos: las compuertas nativas, la profundidad del circuito y la conectividad. Las compuertas nativas son aquellas que son naturales al sistema cuántico a utilizar. No todas las compuertas cuánticas pueden ser implementadas en una estructura dada y solo existe un número finito de compuertas nativas. Afortunadamente, toda compuerta puede ser descompuesta en una serie adecuada de compuertas. La profundidad del circuito se refiere al número de compuertas nativas a aplicar para llevar a cabo el circuito. Mientras menos profundo sea el circuito es más probable que se ejecute con éxito y evite los errores. La conectividad con la cercanía de los qubits, si dos qubits están muy separados compuertas a dos qubits no podrán aplicarse.

Si toda compuerta puede descomponerse o aproximarse con un conjunto de compuertas nativas dado, entonces se dice que este conjunto nativo es un conjunto de compuertas universales. Por ejemplo, dos compuertas de rotación y la compuerta X-controlada es un conjunto de compuertas universales.

Un transpilador es un programa que traduce cualquier circuito a un circuito de compuertas nativas, tomando en cuenta la conectividad. Esto siempre se puede hacer, pero transpilar de manera óptima para reducir la profundidad de un sistema es un problema abierto.

En este proyecto, deberás familiarizarte con el álgebra de compuertas y con el concepto de conjunto universal de compuertas cuánticas para crear tu propio transpilador.

Objetivos:

- Investigar cuáles son las compuertas nativas de diferentes arquitecturas de procesadores cuánticos. Demuestra cómo puedes descomponer una compuerta de 1 qubit usando esas compuertas nativas
- Investigar la compuerta Toffoli y tradúcela a varios conjuntos de compuertas universales. Mientras menos profundo mejor.
- Crear tu propio programa que traduzca un circuito en un conjunto de compuertas dado
- Un conjunto de compuertas universales a 1 qubit está dado por las compuertas Hadamard y T . Dada una compuerta de 1 qubit aleatoria, demuestra teórica o numéricamente que puedes aproximarla arbitrariamente con ese conjunto.

SDK recomendado: cualquiera

Referencias y enlaces:

[1] Qiskit textbook: <https://qiskit.org/textbook>

[2] How does Qiskit transpiler work:

<https://medium.com/qiskit/how-does-the-qiskit-transpiler-work-6710863beaac>

[3] Nielsen and Chuang, [Quantum Computation and Quantum Information](#), Cambridge University Press

[4] [Teorema de Solovay-Kitaev](#)

10. Mitigación de errores (nivel básico)

Mentor: A definir

El [código Morse](#) consiste en enviar una señal con un código binario de puntos y rayas. Es tan sencillo que puede llevarse a cabo con cualquier sistema físico con dos estados: dos corrientes diferentes, dos sonidos diferentes, ceros y unos en una computadora, etc. El código es simple pero si el sistema no es perfecto, pueden generarse errores en el mensaje. Un punto se puede convertir en un raya. Generalmente estos errores dependen de tantos factores que son impredecibles.

En el tratamiento de señales, la mitigación de errores consiste en conocer la frecuencia de estos errores para corregir la señal resultante. Si se conoce con qué probabilidad ocurre un cierto error, se puede crear un filtro y aplicarlo al mensaje final para obtener un resultado (aproximado) del mensaje original.

La mecánica cuántica es probabilística, los resultados de medir un qubit en superposición de 0 y 1 son también impredecibles. Así que todo algoritmo cuántico reposa en poder crear los estados cuánticos precisos para obtener probabilidades adecuadas. El problema es que los sistemas cuánticos son tan sensibles, que cualquier ruido o interacción indeseada, por más pequeña que sea puede desbalancear las probabilidades. Es interesante aplicar la mitigación de errores a estos sistemas para sobrepasar las limitaciones físicas del sistema.

En este proyecto deberás crear simular un sistema cuántico con ruido y demostrar la teoría de mitigación de errores para corregir los resultados del sistema.

Objetivos:

- Escoger un algoritmo cuántico interesante y modelar un sistema con ruido. Comparar los resultados de medición con y sin ruido.
- Demostrar la teoría de mitigación de errores. Crear un filtro que al aplicarlo a los resultados de tu sistema resulte en los resultados esperados.
- Aplicar tu rutina de mitigación de errores a un procesador cuántico real
- Demostrar las fidelidades de tu sistema real o modelo con ruido.

SDK recomendado: Qiskit, Qutip

Referencias y enlaces:

[1] Qiskit textbook: <https://qiskit.org/textbook/preface.html>

[2] Nielsen and Chuang, [Quantum Computation and Quantum Information](#), Cambridge University Press

11. Machine learning cuántico y algoritmos de optimización aproximada [Encoding and Classifier] (nivel avanzado)

Tutor: A. Maldonado

La computación tiene un área que en años recientes ha obtenido una gran popularidad como es el Machine Learning, este tiene como propósitos generar una función $h(x)$ a partir de un modelo como son las redes neuronales para aproximar a una función desconocida $f(x)$, Se tienen 3 bloques de estos como es el aprendizaje supervisado, el aprendizaje no supervisado y el aprendizaje por refuerzo, para este proyecto se usará el primero que se caracteriza por usar un conjunto de entrenamiento que tiene los valores x y $f(x)$ con el fin de hacer una clasificación o una predicción (regresión), se evaluará su desempeño a partir de otro conjunto donde solo se conocen los valores x y se aplicaran en nuestro modelo propuesto $h(x)$.

En este proyecto se recreará la clasificación usando un circuito variacional, con el cuál debemos identificar como pasar un valor clásico y que pueda interpretarlo una computadora cuántica, puede ser por su ángulo o la amplitud de nuestro vector de estado. Consideré un conjunto de entrenamiento y de prueba que tiene 5 variables, pueden tener valores flotantes como enteros, los primeros cuatro parámetros serán los que se codificarán y el último es el que nos dice la clase a la cuál pertenece 0,1. Diseñe la codificación y el preprocesamiento adecuado para poder implementar el circuito variacional que puede estar en el estado del arte de nuestro framework. Y mide un qubit que nos diga la clase a la cual nuestro circuito cuántico está proponiendo.

[Archivo del conjunto de entrenamiento](#)

[Archivo del conjunto de prueba](#)

Objetivos:

- Escoger la mejor manera para codificar datos clásicos en términos de qubits y un circuito variacional propuesto por el framework de Qiskit que logre realizar la clasificación binaria.
- Demostrar los conocimientos para definir tu propio circuito variacional e identificar el número de capas para tener un desempeño con respecto a la métrica accuracy de 100%.
- Demostrar con respecto a una equivalencia clásica usando redes neuronales para identificar con la curva ROC cuál es el mejor e identificar las limitantes del Machine learning cuántico usando una computadora cuántica real con máximo de 5 qubits.

Referencias y enlaces:

[1] Qiskit QML textbook:

https://qiskit.org/textbook-beta/course/machine-learning-course/?utm_source=Social&utm_medium=Twitter

[m_medium=Medium&utm_campaign=Announcement&utm_term=CTA&utm_content=qm
l-course](#)

12. Proyecto abierto (cualquier nivel)

Tutor: libre

El mundo de la computación cuántica necesita de ideas nuevas y creativas. Si hay algún tema que te interesa pero no está entre los desafíos propuestos puedes desarrollarlo tú mismo.

Objetivos:

- Definir un tema interesante inspirado en otros temas, contacta a expertos y organizadores para refinar la idea
- Explicar las dificultades de tu desafío, la teoría y la motivación detrás de este proyecto
- Llevar a cabo una demostración de tu proyecto
- Demostrar que es posible demostrar tu idea con las tecnologías actuales

Referencias: tus conocimientos y tus contactos