Universidad Autónoma Metropolitana
Unidad Cuajimalpa

División de Ciencias Naturales e Ingeniería

Departamento de Matemáticas Aplicadas y Sistemas

Posgrado en Ciencias Naturales e Ingeniería

ICR:
Optimización en computación cuántica: estudio y realización del algoritmo QAOA

Presenta:

Emiliano Montoya González

Directores de Tesis:

Dr. Roberto Bernal Jaquez
Dr. Guillermo Chacón Acosta

Asesores:

Dr. Luis Ángel Alarcón Ramos
Dr. Ángel Alejandro García Chung

Ciudad de México, 2025

Agradecimientos

Agradezco al Dr. Roberto Bernal Jaquez por su invaluable guía y apoyo a lo largo de este proyecto y por darme la oportunidad de realizarlo bajo su dirección.

A los doctores Guillermo Chacón Acosta, Luis Ángel Alarcón Ramos y Ángel Alejandro García Chung por sus comentarios y sugerencias hacia el proyecto.

Al Dr. Juan Manuel Romero Sanpedro por permitirme colaborar con él y por todo el apoyo y conocimiento que ha compartido conmigo.

Al Dr. Gildardo Barrientos Sánchez por despertar la curiosidad en mí, lo que me permitió descubrir mi pasión por la computación cuántica.

A la Universidad Autónoma Metropolitana por todas las oportunidades y apoyos otorgados a lo largo de mi desarrollo académico.

Agradezco a mi familia por el apoyo, las oportunidades y la confianza que han tenido en mí a lo largo de toda mi vida. A mis papás por todo su trabajo para permitirme llegar hasta aquí y a mi hermano por su amistad y apoyo.

A mis amigos, que han sido una parte importante para mi desarrollo personal y académico, especialmente a Rafael, por creer en mí y acompañarme en los momentos difíciles.

Agradezco a Ivonne, por la compañía, las risas y los consejos. Por creer siempre en mi y ser mi escucha y mi guía cuando mas lo necesitaba, gracias por tanto.

Finalmente, quiero dar un agradecimiento muy especial a mis tres gatos.

A Agustina, por haber llegado a mi vida cuando mas te necesitaba, por ser mi guía y ayudarme a seguir adelante y nunca rendirme, por enseñarme lo que es el amor incondicional y por la compañía que me das cada día.

A Makoto, por tu amor incondicional y la gran confianza que me tienes. Porque solo con una mirada lo entiendes todo y me lo dices todo. Por haberme enseñado como cuidar a los demás y ser un lugar seguro.

A Ren, por enseñarme que lo mas importante es ser uno mismo, a ser autentico, por todo el cariño y amor que me das y por recordarme que siempre nos tendremos el uno al otro. Por ayudarme a entender que no importan los problemas, el dinero ni las cosas materiales, porque al final del día, lo importante es todo el amor que tenemos para entregar a este mundo y eso hace que todo valga la pena.

El tiempo no aguarda a nadie.
Discurre igual para todos.

Incluso en estos días felices y serenos,
guíate por el corazón y jamás te desvíes de su camino...

– *Anónimo*

Resumen

En la era de la computación cuántica a escala intermedia y ruidosa (NISQ), los algoritmos cuánticos variacionales (VQA) han surgido como un enfoque prometedor para resolver problemas de optimización complejos. Este trabajo se centra en el estudio y la realización del Algoritmo Cuántico de Optimización Aproximada (QAOA), un VQA híbrido, aplicado al modelo de Ising 2D, un problema de optimización combinatoria NP-difícil con gran relevancia en física y computación. Se implementó una versión del algoritmo utilizando la librería Cirq de Google, optimizando sus parámetros con métodos clásicos y ejecutando la optimización final en circuitos cuánticos. Los resultados se compararon con algoritmos clásicos de referencia como Metropolis-Hastings y fuerza bruta. En una red de 2×2, todos los métodos coincidieron en la solución óptima, validando la precisión del QAOA. En una red de 20×20, la ejecución en hardware cuántico real evidenció una mejora sustancial en el desempeño, resolviendo el problema en 3 segundos, mientras que el método clásico paralelizado más rápido tardó 7 minutos. Estos hallazgos subrayan el potencial de los algoritmos cuánticos para ofrecer una aceleración significativa en problemas de optimización y marcan un paso importante hacia su aplicación práctica.

Índice

1. Introducción **6**

2. Antecedentes teóricos **7**

2.1. Fundamentos de Computación Cuántica 7

2.2. Modelo de Ising 8

2.3. Computación cuántica 10

2.3.1. Compuertas Cuánticas 11

2.3.2. Compuerta Identidad (I) 12

2.3.3. Compuerta X (NOT Cuántica) 12

2.3.4. Compuerta Z (Cambio de Fase) 12

2.3.5. Compuerta de Hadamard (H) 12

2.4. Circuitos Cuánticos 13

2.4.1. Circuito de Bell y Entrelazamiento 13

3. Algoritmos clásicos **14**

3.1. Algoritmo Metropolis-Hastings 14

3.2. Método de Fuerza Bruta 15

4. Algoritmos cuánticos variacionales (VQA) **15**

4.1. Conceptos importantes para los Algoritmos Cuánticos Variacionales . . . 17

4.2. Algoritmo QAOA (Quantum Approximate Optimization Algorithm) . . . 17

4.2.1. Principios del Algoritmo QAOA 18

5. Metodología de los Algoritmos **18**

5.1. Metodología del algoritmo QAOA 18

5.1.1. Mapeo del Problema a un Hamiltoniano Cuántico 18

5.1.2. Estructura del Algoritmo 19

5.2. Metodología del algoritmo Metropolis-Hastings 20

5.3. Metodología del algoritmo de fuerza bruta 21

6. Implementación del algoritmo **21**

6.1. Importación de librerías y definición del problema 21

6.2. Creación de qubits y utilidades 22

6.3. Construcción de los operadores de QAOA 22

6.3.1. Operador de Costo 22

6.3.2. Operador de Mezcla 23

6.4. Construcción del circuito completo de QAOA 23

6.5. Cálculo clásico de la energía 24

6.6. Función objetivo y optimización 24

7. Análisis de los resultados **25**

7.1. Resultados del Estado Fundamental (Malla 2x2) 25

7.2. Análisis de Rendimiento en Hardware Real (Malla 20x20) 26

7.3. Análisis Detallado de la Distribución de QAOA (Malla 2x2) 26

7.4. Análisis Gráfico de Resultados (Malla 2x2) 27

8. Conclusiones **30**
8.1. Limitaciones y Trabajo Futuro 30

1. Introducción

En los últimos 50 años, la computación cuántica ha sido un objeto de estudio de distintas áreas como la física y la computación y, aunque ha tenido avances muy grandes e importantes, actualmente se sigue considerando que las computadoras cuánticas que se encuentran en funcionamiento y pueden ser programadas, se encuentran en una fase de prototipo y no se ha llegado a una computadora cuántica definitiva. Esta era se conoce como NISQ (Noisy Intermediate-Scale Quantum), caracterizada por procesadores con un número intermedio de qubits (50-1000) que carecen de corrección de errores, haciéndolos susceptibles al ruido del entorno [1]. Esto se debe a que, para implementar los qubits y las compuertas cuánticas, se requiere de técnicas para manipular fenómenos a escala atómica, como el espín de un electrón. Este proceso, en conjunto con los fenómenos cuánticos que implica, presenta limitaciones que aún no pueden ser superadas, por ejemplo, el hecho de que los procesadores cuánticos deben encontrarse a temperaturas muy específicas, cercanas al cero absoluto, y son muy susceptibles al ruido y, por consiguiente, a errores, lo que afecta al rendimiento de ciertos algoritmos [2].

No obstante, a pesar de las limitaciones actuales, el trabajo en conjunto de físicos, matemáticos e ingenieros, ha permitido buscar algunas alternativas por medio de hardware y software. Ejemplos de ello son las computadoras cuánticas fotónicas en la parte de hardware [3] y el desarrollo de algoritmos como los llamados Algoritmos Cuánticos Variacionales (VQA) que, en general, nos ayudan a resolver problemas de optimización demasiado complejos para una computadora clásica y que, en medida de lo posible, tienen una alta tolerancia a errores. Uno de estos algoritmos es el llamado *Quantum Approximate Optimization Algorithm* (QAOA), cuyo objetivo es minimizar una función de costo $C(z)$ dada, donde z es un conjunto de variables binarias $z = z_1, z_2, ..., z_n$ y z_i puede tener un valor de 1 o -1. Por lo tanto, el objetivo del algoritmo QAOA, en general, es asignar valores a z_i de tal manera que obtengamos una aproximación del valor mínimo de la función $C(z)$ [4, 5].

Los algoritmos de optimización se enfocan en resolver problemas en donde se requiere maximizar o minimizar una o más variables acordes al problema. Estos algoritmos se enfocan en resolver problemas del tipo NP-Difícil y NP-Completo, que son aquellos para los cuales no se conoce un algoritmo que los resuelva en tiempo polinomial en una computadora clásica. En los problemas de optimización, se modela matemáticamente el problema que se busca resolver y se representa con fórmulas matemáticas. A partir de ahí, el algoritmo se encarga de maximizar o minimizar las variables a partir de dicho modelo matemático y así encontrar la mejor solución [6, 7].

Los algoritmos de optimización cuánticos, un enfoque híbrido, en esencia tienen el mismo funcionamiento que los clásicos, sin embargo, estos se enfocan en problemas con los cuales una computadora clásica no puede lidiar, en el mejor de los casos. Estos problemas provienen de áreas como las finanzas (optimización de carteras), la medicina (descubrimiento de fármacos), la física (simulación de sistemas cuánticos) o la logística y el transporte (problema del viajante). Además, una de las ventajas de estos algoritmos es que se pueden codificar distintas soluciones buscadas de manera simultánea gracias a propiedades de la mecánica cuántica como el entrelazamiento y la superposición. Por lo tanto, al codificar los datos del problema en estados cuánticos manipulables por com-

puertas cuánticas, se obtienen todas las ventajas de las computadoras cuánticas sobre las computadoras clásicas [8, 9].

En este contexto, el trabajo aquí presentado explora la aplicabilidad de los algoritmos híbridos en la era NISQ, centrándose en el algoritmo QAOA aplicado al modelo de Ising. Este modelo, de suma importancia tanto en la física como en la computación, sirve como banco de pruebas para evaluar cómo la eficacia del algoritmo varía con el número de qubits y la complejidad del problema y, a su vez, puede tener distintas mejoras con su contraparte clásica. Se buscó implementar una versión del algoritmo QAOA utilizando la librería de Google *Cirq* para el lenguaje de programación Python, optimizando parámetros usando algoritmos clásicos y realizando la optimización final del sistema usando circuitos cuánticos. Se investiga también el impacto que la temperatura del sistema tiene en los algoritmos comparativos. Nuestra hipótesis sostiene que el algoritmo QAOA logrará una ventaja en tiempos de ejecución y calidad de la solución frente a métodos clásicos como Metropolis-Hastings o fuerza bruta, reduciendo la complejidad computacional al explotar el espacio de soluciones gracias a fenómenos de la computación cuántica como la superposición. Los resultados obtenidos en hardware real demuestran una mejora significativa con su contraparte clásica, validando el potencial práctico del algoritmo QAOA en problemas de optimización combinatoria y estableciendo métricas cuantitativas para guiar aplicaciones futuras [1, 4, 5].

A lo largo de esta tesis, la estructura será la siguiente: la sección 2 introducirá los fundamentos teóricos necesarios, incluyendo el modelo de Ising y los conceptos clave de la computación cuántica. La sección 3 describirá los algoritmos clásicos utilizados como referencia. La sección 4 profundizará en los algoritmos cuánticos variacionales, con especial énfasis en QAOA. La sección 5 detallará la metodología de implementación de todos los algoritmos. Finalmente, la sección 6 presentará y analizará los resultados obtenidos, seguida de las conclusiones en la sección 7.

2. Antecedentes teóricos

2.1. Fundamentos de Computación Cuántica

La computación cuántica es un paradigma que utiliza los principios de la mecánica cuántica para procesar información. A diferencia de la computación clásica, que se basa en bits que pueden estar en estado 0 o 1, la computación cuántica utiliza **qubits**.

Un qubit es la unidad básica de información cuántica. Formalmente, un qubit es un sistema cuántico de dos niveles que puede representarse como un vector en un **espacio de Hilbert** complejo de dos dimensiones, $\mathbb{C}^2$. Mientras que un bit clásico solo puede ser 0 o 1, un qubit puede existir en una **superposición** de ambos estados. Su estado $|\psi\rangle$ se describe como una combinación lineal de los estados base $|0\rangle$ y $|1\rangle$:

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle, \tag{1}$$

donde α y β son amplitudes de probabilidad complejas que satisfacen la condición de normalización $|\alpha|^2 + |\beta|^2 = 1$. Los valores $|\alpha|^2$ y $|\beta|^2$ representan la probabilidad de que el qubit colapse al estado $|0\rangle$ o $|1\rangle$ respectivamente, al ser medido.

Esta capacidad de superposición, junto con el **entrelazamiento** (donde el estado de múltiples qubits está correlacionado de forma no local) y la **interferencia** (donde

las amplitudes de probabilidad pueden anularse o reforzarse), permite a las computadoras cuánticas explorar un espacio computacional exponencialmente más grande que sus contrapartes clásicas.

Uno de los objetivos principales de la computación cuántica, es alcanzar la llamada "supremacía cuántica", un término que se ha manejado desde los años 80 [10] que describe el potencial de una computadora cuántica para resolver problemas que una computadora clásica no podría o tardaría demasiado tiempo en resolver. En términos de complejidad computacional, esto se traduce en buscar una aceleración superpolinomial sobre el mejor algoritmo clásico conocido [11, 12]. A pesar de los avances significativos en la investigación de algoritmos de optimización cuántica, en particular los algoritmos cuánticos variacionales (VQA), el objetivo de alcanzar una ventaja cuántica práctica y demostrable aún no se ha materializado por completo. Estos algoritmos han sido objeto de estudio exhaustivo con la esperanza de resolver problemas complejos que van más allá de las capacidades de una computadora clásica. Si bien se han logrado avances notables, la computación cuántica enfrenta desafíos en términos de escalabilidad y estabilidad de los qubits [1, 5].

A pesar de estas limitaciones, el campo de la computación cuántica continúa evolucionando rápidamente. Cada año, surgen nuevos aportes teóricos y prácticos que ofrecen nuevas perspectivas y enfoques para abordar los desafíos existentes. Los VQA se han destacado como una de las áreas más prometedoras en este sentido, ya que ofrecen la posibilidad de desarrollar algoritmos tolerantes a errores que puedan operar eficazmente en sistemas cuánticos con errores, gracias a sus circuitos de baja profundidad [3, 13, 14].

La investigación actual se centra en mejorar la escalabilidad y la estabilidad de los algoritmos cuánticos, así como en desarrollar técnicas de corrección y mitigación de errores efectivas. Se espera que estos avances conduzcan a la creación de computadoras cuánticas más robustas y fáciles de programar en el futuro. Los VQA desempeñan un papel fundamental en este proceso al proporcionar una alternativa flexible para explorar y desarrollar nuevos enfoques algorítmicos que aprovechen las capacidades emergentes de la computación cuántica [5, 15].

2.2. Modelo de Ising

El modelo de Ising es uno de los modelos más estudiados en la física estadística y describe la interacción entre espines atómicos en un material magnético. El "espín" es una propiedad intrínseca de las partículas elementales, que en este contexto puede ser visualizada como un pequeño momento magnético que puede apuntar en una de dos direcciones. Fue propuesto por Wilhelm Lenz en 1920 y resuelto analíticamente en una dimensión por su estudiante Ernst Ising en 1925. La solución para el caso bidimensional sin campo magnético externo, un problema mucho más complejo, fue encontrada por Lars Onsager en 1944. Desde entonces, ha sido ampliamente utilizado para estudiar fenómenos como el ferromagnetismo, las transiciones de fase y la criticalidad en sistemas físicos [**?**]. Su relevancia en computación radica en que muchos problemas de optimización combinatoria pueden ser mapeados al problema de encontrar el estado de mínima energía de un modelo de Ising.

Supongamos que tenemos una red de espines, donde cada espín puede tener dos posibles valores: hacia arriba ($\uparrow, s_i = +1$) o hacia abajo ($\downarrow, s_i = -1$). La energía del sistema

se puede expresar mediante la siguiente función de Hamiltoniano:

$$H = -J \sum_{\langle i,j \rangle} s_i s_j - B \sum_i s_i \tag{2}$$

Donde:

- H es la energía total del sistema (el Hamiltoniano).
- J es el coeficiente de acoplamiento o interacción entre espines adyacentes.
- B es el campo magnético externo.
- s_i y s_j son los valores de espín en los sitios i y j respectivamente.
- La sumatoria $\sum_{\langle i,j \rangle}$ se realiza sobre todos los pares de sitios adyacentes.

La primera parte del Hamiltoniano representa la interacción entre espines adyacentes. Si $J > 0$, la interacción es **ferromagnética**, lo que significa que los espines tienden a alinearse en la misma dirección para minimizar la energía. Si $J < 0$, la interacción es **antiferromagnética**, y los espines tienden a alinearse en direcciones opuestas. La segunda parte representa la interacción de los espines con un campo magnético externo B, que tiende a alinear todos los espines en su dirección.

El objetivo principal del modelo de Ising es encontrar la configuración de espines que minimice la energía total del sistema. Esta configuración representa el **estado fundamental** del sistema, que es el estado más estable y probable de existir a temperatura cero. La búsqueda de este estado de mínima energía es, por lo tanto, un problema de optimización combinatoria [**?**]. Encontrar el estado fundamental del modelo de Ising en 3D o en 2D con campos magnéticos no uniformes es un problema NP-difícil.

En la práctica, sin embargo, muchos sistemas físicos reales operan a temperaturas mayores que cero $T > 0$, donde la temperatura T introduce fluctuaciones térmicas y el sistema puede explorar estados excitados además del estado fundamental.

Actualmente, existen algoritmos tanto clásicos como cuánticos para abordar este problema [16]. En particular, en computación cuántica se ha investigado el uso de algoritmos variacionales cuánticos (VQA) para resolver el modelo de Ising. Sin embargo, estos algoritmos presentan desafíos importantes, como una alta complejidad computacional y la necesidad de realizar un gran número de operaciones. Además, cabe señalar que muchos de estos algoritmos se concentran únicamente en la optimización del modelo para el caso de temperatura cero $T = 0$, es decir, solo buscan el estado fundamental sin considerar los efectos de la temperatura. Esto limita su aplicabilidad en sistemas reales, donde se requiere analizar el comportamiento a temperaturas finitas [5].

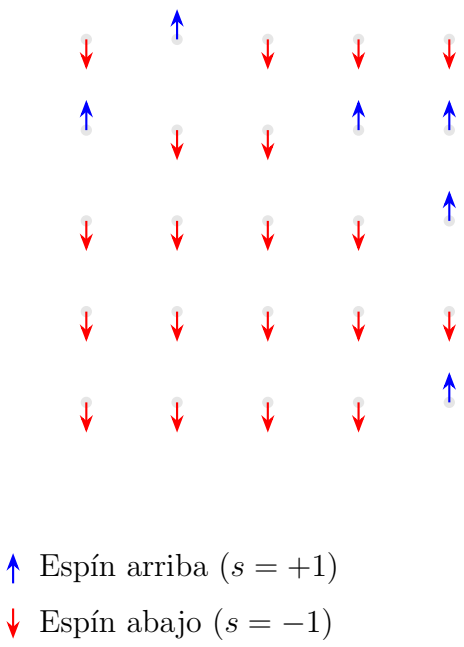


Figura 1: Representación de una red de espines 5x5 para el modelo de Ising en un estado desordenado.

2.3. Computación cuántica

A diferencia de los bits clásicos, que solo pueden estar en un estado a la vez (0 o 1), los qubits o bits cuánticos pueden existir en una *superposición* de ambos estados. Esto significa que un qubit puede estar simultáneamente en los estados $|0\rangle$ y $|1\rangle$, lo que abre un abanico de posibilidades para el procesamiento de información. Sin embargo, al medir un qubit, este colapsa a uno de los dos estados clásicos, revelando solo uno de los valores posibles.

Definición: Un qubit se define formalmente como un vector en un espacio de Hilbert bidimensional sobre los números complejos:

$$V = \begin{bmatrix} c_0 \\ c_1 \end{bmatrix} : V \in \mathbb{C}^2,$$

donde c_0 y c_1 son amplitudes complejas. Este vector puede representarse como:

$$|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle = \begin{bmatrix} \alpha \\ \beta \end{bmatrix}, \tag{3}$$

sujeto a la condición de normalización:

$$|\alpha|^2 + |\beta|^2 = 1. \tag{4}$$

La ecuación (4) indica que un qubit está en una superposición de los estados $|0\rangle$ y $|1\rangle$, donde $|\alpha|^2$ y $|\beta|^2$ representan las probabilidades de medir el qubit en los estados $|0\rangle$ y $|1\rangle$, respectivamente. La suma de estas probabilidades siempre debe ser igual a 1, garantizando que el qubit esté en un estado válido.

Definición: Un sistema de n qubits puede describirse como un espacio vectorial de dimensión 2^n sobre los números complejos:

$$V = \bigotimes_{i=1}^{n} \mathbb{C}^2 : V \in \mathbb{C}^{2^n}.$$

Por ejemplo, un sistema de 2 qubits puede representarse como:

$$|\psi\rangle = c_{00}\,|00\rangle + c_{01}\,|01\rangle + c_{10}\,|10\rangle + c_{11}\,|11\rangle\,,$$

donde se omite el símbolo de producto tensorial $\otimes$ por convención.

Aunque un qubit puede estar en superposición, al medirlo siempre obtenemos un resultado clásico: 0 o 1. Sin embargo, la superposición es un fenómeno real y fundamental en la computación cuántica. Para visualizar el estado de un qubit, se utiliza la *esfera de Bloch*, una representación geométrica que muestra cómo el estado de un qubit puede variar en función de sus amplitudes.

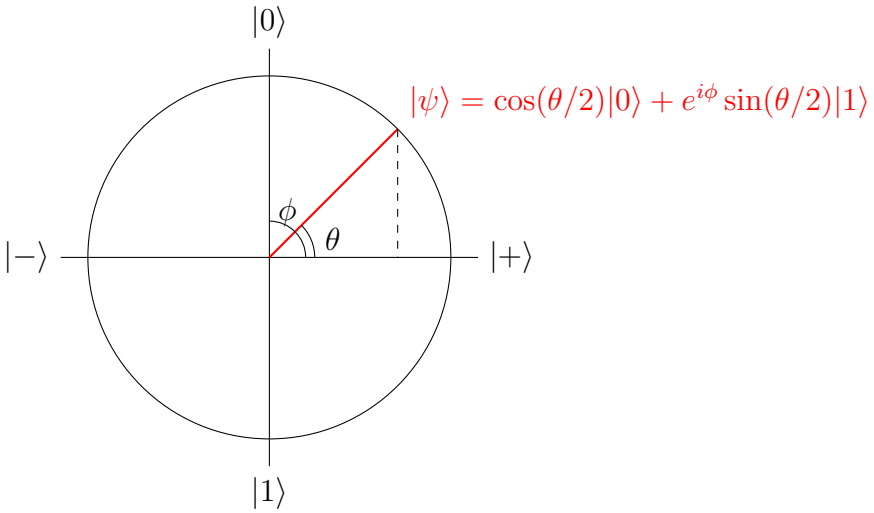


Figura 2: Esfera de Bloch. El estado $|\psi\rangle$ se representa como un vector en la superficie de la esfera, definido por los ángulos θ y ϕ. Los estados $|+\rangle$ y $|-\rangle$ son superposiciones de los estados base.

2.3.1. Compuertas Cuánticas

Las compuertas cuánticas son operaciones unitarias que actúan sobre qubits. A diferencia de las compuertas clásicas, que son funciones booleanas, las compuertas cuánticas son matrices unitarias que preservan la norma del estado cuántico. Su estudio es fundamental para entender cómo se programan los algoritmos cuánticos variacionales.

Definición: Una compuerta cuántica es una matriz unitaria U de dimensión $2^n \times 2^n$ que opera sobre n qubits. Se requiere que $U^\dagger U = UU^\dagger = I$, donde $U^\dagger$ es la transpuesta conjugada de U. El estado de salida se obtiene multiplicando la matriz de la compuerta por el vector de estado del qubit.

2.3.2. Compuerta Identidad (I)

La compuerta identidad no modifica el estado del qubit. Se representa como la matriz identidad 2×2. Al aplicarla a un qubit $|\psi\rangle = \alpha|0\rangle+\beta|1\rangle$, donde $|0\rangle = [1,0]^T$ y $|1\rangle = [0,1]^T$, el estado de salida es el mismo, ya que $I|\psi\rangle = |\psi\rangle$.

$$I = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}. \tag{5}$$

Ejemplo: Al aplicar la compuerta I a un qubit:

$$I(\alpha\,|0\rangle + \beta\,|1\rangle) = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \alpha\,|0\rangle + \beta\,|1\rangle\,.$$

2.3.3. Compuerta X (NOT Cuántica)

La compuerta X es el equivalente cuántico de la compuerta NOT clásica. Invierte el estado del qubit, cambiando $|0\rangle$ por $|1\rangle$ y viceversa. Su representación matricial es:

$$X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}. \tag{6}$$

Ejemplo: Al aplicar la compuerta X a un qubit:

$$X(\alpha\,|0\rangle + \beta\,|1\rangle) = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \begin{bmatrix} \beta \\ \alpha \end{bmatrix}.$$

Por lo tanto, $X\,|0\rangle = |1\rangle$ y $X\,|1\rangle = |0\rangle$.

2.3.4. Compuerta Z (Cambio de Fase)

La compuerta Z introduce un cambio de fase en el estado $|1\rangle$, dejando $|0\rangle$ sin cambios. No altera las probabilidades de medición (ya que $|-1|^2 = 1$), pero sí la fase relativa, lo cual es crucial para la interferencia cuántica. Su matriz es:

$$Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}. \tag{7}$$

Ejemplo: Al aplicar la compuerta Z a un qubit:

$$Z(\alpha\,|0\rangle + \beta\,|1\rangle) = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \begin{bmatrix} \alpha \\ -\beta \end{bmatrix} = \alpha\,|0\rangle - \beta\,|1\rangle\,.$$

Por lo tanto, $Z\,|0\rangle = |0\rangle$ y $Z\,|1\rangle = -\,|1\rangle$. El signo negativo no afecta la medición directa de este qubit, pero sí cambia cómo se comporta en operaciones posteriores, especialmente en fenómenos de interferencia.

2.3.5. Compuerta de Hadamard (H)

La compuerta de Hadamard es una de las más importantes en computación cuántica. Transforma un estado base en una superposición uniforme de estados. Su matriz es:

$$H = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}. \tag{8}$$

Ejemplo: Al aplicar la compuerta H a los estados base:

$$H|0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \equiv |+\rangle$$
$$H|1\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) \equiv |-\rangle$$

Al medir cualquiera de estos estados resultantes, la probabilidad de obtener $|0\rangle$ o $|1\rangle$ es del 50 %. La diferencia clave es la fase negativa en $|-\rangle$, que afecta cómo el qubit interfiere con otros qubits en algoritmos cuánticos.

2.4. Circuitos Cuánticos

Un circuito cuántico es una secuencia de compuertas cuánticas aplicadas a uno o más qubits para realizar un cálculo. Son la base para programar y ejecutar algoritmos cuánticos.

2.4.1. Circuito de Bell y Entrelazamiento

El entrelazamiento es un fenómeno donde el estado de un qubit depende instantáneamente del estado de otro, sin importar la distancia que los separe. El circuito de Bell, mostrado en la figura (3), es un ejemplo fundamental para generar este fenómeno. Consta de una compuerta de Hadamard aplicada al primer qubit, seguida de una compuerta CNOT (Controlled-NOT), donde el primer qubit actúa como control y el segundo como objetivo. Este circuito genera estados entrelazados, conocidos como estados de Bell.

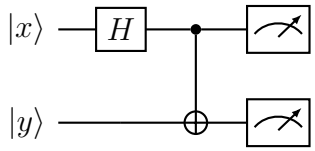


Figura 3: Circuito de Bell, que genera uno de los cuatro estados de Bell entrelazados dependiendo de las entradas $|x\rangle$ e $|y\rangle$.

Ejemplo: Analizamos el circuito de Bell para la entrada $|00\rangle$:

$$|\psi_0\rangle = |00\rangle = |0\rangle \otimes |0\rangle .$$

Aplicando la compuerta H al primer qubit:

$$|\psi_1\rangle = (H \otimes I)\,|00\rangle = (H\,|0\rangle) \otimes |0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \otimes |0\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |10\rangle).$$

Finalmente, aplicamos la compuerta CNOT, donde el primer qubit es el control y el segundo el objetivo:

$$|\psi_2\rangle = \mathrm{CNOT}\,|\psi_1\rangle = \frac{1}{\sqrt{2}}(\mathrm{CNOT}\,|00\rangle + \mathrm{CNOT}\,|10\rangle) = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle).$$

El estado resultante, $|\Phi^+\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$, es un estado de Bell. Si medimos el primer qubit y obtenemos $|0\rangle$, sabremos instantáneamente que el segundo qubit también será $|0\rangle$, y viceversa. Los cuatro estados de Bell son:

$$\begin{aligned} \left|\Phi^{\pm}\right\rangle &= \frac{1}{\sqrt{2}}(|00\rangle \pm |11\rangle), \\ \left|\Psi^{\pm}\right\rangle &= \frac{1}{\sqrt{2}}(|01\rangle \pm |10\rangle). \end{aligned} \tag{9}$$

Este fenómeno es fundamental en aplicaciones como la teleportación cuántica y la corrección de errores en la transmisión de qubits. La medición es un proceso irreversible y es el paso final en un algoritmo cuántico para obtener un resultado clásico. El diseño de algoritmos cuánticos se centra en manipular las probabilidades para que el resultado deseado sea el más probable al medir.

3. Algoritmos clásicos

La optimización combinatoria es un campo de las matemáticas y la informática centrado en encontrar una solución óptima de un conjunto finito de soluciones. Formalmente, un problema de optimización combinatoria (S, f) consiste en un espacio de búsqueda S (el conjunto de todas las soluciones posibles) y una función objetivo $f : S \to \mathbb{R}$ que se desea minimizar o maximizar. En esta sección se describen dos de los algoritmos más relevantes en este contexto: el **algoritmo Metropolis-Hastings** y el **método de fuerza bruta**.

3.1. Algoritmo Metropolis-Hastings

El algoritmo Metropolis-Hastings es una técnica de Monte Carlo basada en cadenas de Markov (MCMC), utilizada para generar una secuencia de muestras de una distribución de probabilidad que es difícil de muestrear directamente. Su uso es frecuente en problemas donde el número de configuraciones posibles crece exponencialmente con el tamaño del sistema, como en el *Modelo de Ising 2D*, donde cada espín puede adoptar uno de dos valores, generando un espacio de estados de tamaño 2^N para una red de N espines.

El fundamento del algoritmo radica en la generación de una secuencia de configuraciones que converge a una distribución de probabilidad objetivo, como la distribución de Boltzmann en física estadística:

$$P(\sigma) = \frac{1}{Z} e^{-H(\sigma)/k_B T}, \tag{10}$$

donde $H(\sigma)$ representa la energía de la configuración σ, T es la temperatura del sistema, k_B es la constante de Boltzmann (a menudo normalizada a 1 en simulaciones) y Z es la función de partición, que normaliza la distribución.

Este método es especialmente útil en la optimización combinatoria y el estudio de sistemas térmicos porque permite explorar de manera eficiente el espacio de soluciones. A diferencia de un descenso por gradiente simple, no se limita a moverse siempre hacia estados de menor energía. Introduce una probabilidad de aceptación que permite ocasionalmente aceptar configuraciones de mayor energía. Esta característica es crucial, ya que

mejora la exploración del espacio de búsqueda y permite al algoritmo escapar de mínimos locales, aumentando la probabilidad de encontrar el mínimo global. La temperatura T es un parámetro clave:

- **Alta temperatura ($T \to \infty$):** La probabilidad de aceptación $\exp(-\Delta E/T)$ se acerca a 1. El sistema se comporta como una caminata aleatoria, explorando ampliamente pero sin converger a un mínimo.
- **Baja temperatura ($T \to 0$):** La probabilidad de aceptar un movimiento çuesta arriba"($\Delta E > 0$) se vuelve casi cero. El algoritmo se comporta como un descenso voraz y es propenso a quedar atrapado en el primer mínimo local que encuentra.

Un enfoque más avanzado, llamado *Simulated Annealing* (temple simulado), aprovecha esto comenzando con una temperatura alta y reduciéndola gradualmente para equilibrar la exploración y la explotación.

3.2. Método de Fuerza Bruta

El método de fuerza bruta es el enfoque más simple y exhaustivo para resolver problemas de optimización: evalúa todas las posibles configuraciones y selecciona la mejor. En el caso del *Modelo de Ising 2D* con N espines, este método consiste en generar y calcular la energía de cada una de las 2^N configuraciones posibles para luego elegir aquella con la menor energía.

Aunque garantiza encontrar la solución óptima (el estado fundamental), su principal desventaja es su alta complejidad computacional, $O(N \cdot 2^N)$, que crece exponencialmente con el número de espines en el sistema. Esto lo hace completamente inviable para sistemas de tamaño moderado o grande. Por ejemplo, para la red de 2×2 (4 espines) de este estudio, hay $2^4 = 16$ configuraciones. Para una red de 10×10 (100 espines), el número de configuraciones es $2^{100} \approx 10^{30}$, una cifra astronómica que ninguna computadora actual podría procesar.

A pesar de su ineficiencia, el método de fuerza bruta es una herramienta invaluable en ciertos casos:

- **Validación:** Para problemas de pequeña escala, proporciona la solución exacta que sirve como referencia ("ground truth") para evaluar el rendimiento y la precisión de algoritmos heurísticos como Metropolis-Hastings o cuánticos como QAOA.
- **Análisis teórico:** Permite estudiar el espectro completo de energías de un sistema pequeño y comprender su estructura.

En contraste con Metropolis-Hastings, la fuerza bruta no introduce mecanismos estocásticos ni de exploración. Su carácter determinista y exhaustivo lo convierte en una opción limitada pero fundamental para la verificación de resultados.

4. Algoritmos cuánticos variacionales (VQA)

Los algoritmos cuánticos variacionales (VQA) son un enfoque híbrido que combina el poder de los procesadores cuánticos con la optimización clásica para resolver problemas

complejos. Son especialmente prometedores para la era actual de hardware cuántico ruidoso de escala intermedia (NISQ) [1].

El núcleo de un VQA es un circuito cuántico parametrizado, o **ansatz**, que prepara un estado de prueba $|\psi(\vec{\theta})\rangle$. El objetivo es optimizar los parámetros $\vec{\theta}$ para minimizar una función de costo, que es el valor esperado de un Hamiltoniano H que codifica el problema: $C(\vec{\theta}) = \langle\psi(\vec{\theta})|H|\psi(\vec{\theta})\rangle$. Este proceso se basa en el **principio variacional de la mecánica cuántica**, que garantiza que esta energía esperada es siempre mayor o igual a la energía del estado fundamental del sistema.

El flujo de un VQA sigue un bucle híbrido cuántico-clásico:

1. **Paso Cuántico:** Se prepara el estado $|\psi(\vec{\theta})\rangle$ en la computadora cuántica ejecutando el circuito del ansatz con un conjunto de parámetros $\vec{\theta}$.
2. **Medición:** Se mide el valor esperado del Hamiltoniano H para obtener la función de costo. Esto generalmente requiere ejecutar el circuito varias veces.
3. **Paso Clásico:** Un optimizador clásico (ej. descenso de gradiente, BFGS, COBYLA) utiliza el valor de la función de costo para proponer un nuevo conjunto de parámetros $\vec{\theta}'$.
4. **Iteración:** Se repiten los pasos 1-3 hasta que la función de costo converge a un mínimo.

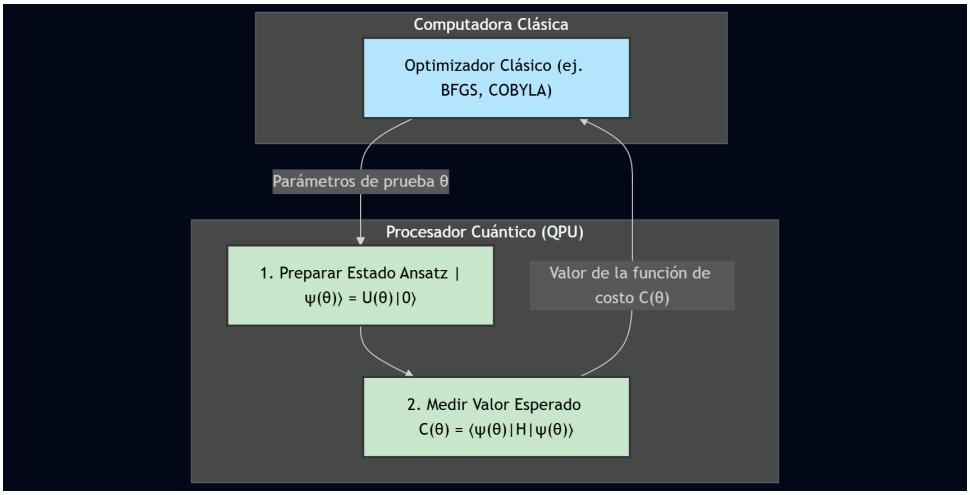


Figura 4: Diagrama de flujo de un Algoritmo Cuántico Variacional (VQA). Adaptado de [5].

Los VQA han demostrado ser una herramienta poderosa para problemas en diversas áreas, incluyendo:

- **Optimización combinatoria**: Problemas como el problema del viajante (TSP), el problema de la mochila y el corte máximo (*Max-Cut*) [4].
- **Simulación de materiales y química cuántica**: Métodos como el algoritmo variacional de eigenvalores (*VQE*) se han utilizado para calcular la estructura electrónica de moléculas [3].

- **Aprendizaje automático cuántico**: Modelos como redes neuronales cuánticas (*QNN*) y clasificadores basados en estados cuánticos [**?**].

4.1. Conceptos importantes para los Algoritmos Cuánticos Variacionales

Ansatz: El término *ansatz* proviene del alemán y se refiere a una suposición o conjetura inicial utilizada en matemáticas y física para resolver un problema complejo. En el contexto de la computación cuántica, un ansatz es un circuito cuántico parametrizado que se utiliza para generar una familia de estados cuánticos. La elección del ansatz es fundamental: un buen ansatz debe ser *expresivo* (capaz de representar el estado solución) y *entrenable* (no sufrir de problemas como mesetas yermas o *barren plateaus*, donde los gradientes se desvanecen exponencialmente). Existen dos tipos principales de ansatz: los inspirados en la física (como el Unitary Coupled Cluster) y los agnósticos de hardware (*hardware-efficient ansätze*) [3].

NISQ: Las computadoras cuánticas actuales están en una fase de desarrollo conocida como *Noisy Intermediate-Scale Quantum* (NISQ), término acuñado por John Preskill. Se refiere a computadoras cuánticas con un número intermedio de qubits (50-1000) que son susceptibles a ruido cuántico y errores de compuerta, y carecen de corrección de errores cuánticos a gran escala. El ruido, proveniente de la decoherencia y la diafonía entre qubits, es el principal reto a superar. Los algoritmos como los VQA son prometedores para la era NISQ porque sus circuitos suelen ser poco profundos, lo que limita la acumulación de errores [1].

Optimizador BFGS: El optimizador BFGS (Broyden–Fletcher–Goldfarb–Shanno) es un algoritmo de optimización que pertenece a la familia de métodos cuasi-Newton. Es muy popular para problemas de optimización no lineal sin restricciones. A diferencia de los métodos de primer orden (como el descenso de gradiente), BFGS utiliza información de los gradientes de pasos anteriores para construir una aproximación de la matriz Hessiana inversa (la matriz de segundas derivadas), lo que le permite tomar pasos de optimización más informados y a menudo converger más rápidamente [**?**].

Optimizador COBYLA: COBYLA (*Constrained Optimization BY Linear Approximations*) es un algoritmo de optimización que se utiliza para resolver problemas no lineales con restricciones, pero sin necesidad de calcular las derivadas de la función de costo. Este método es especialmente útil cuando la función de costo es ruidosa (como en las mediciones de un procesador cuántico) o cuando sus derivadas son difíciles de calcular. COBYLA construye aproximaciones lineales del problema en cada iteración para encontrar la dirección de descenso [18].

4.2. Algoritmo QAOA (Quantum Approximate Optimization Algorithm)

Uno de los VQA más importantes es el *Quantum Approximate Optimization Algorithm* (QAOA), introducido por Farhi, Goldstone y Gutmann en 2014 [4]. Este algoritmo es un método híbrido diseñado para encontrar soluciones aproximadas a problemas de optimización combinatoria.

4.2.1. Principios del Algoritmo QAOA

El QAOA mapea un problema de optimización combinatoria en un Hamiltoniano de costo H_C, cuyos autovectores corresponden a las posibles soluciones y cuyos autovalores son los costos de esas soluciones. El objetivo es encontrar el estado fundamental de H_C. Para lograrlo, el QAOA utiliza un ansatz específico que alterna la aplicación de dos operadores unitarios:

- **Operador de costo (o fase):** $U_C(\gamma) = e^{-i\gamma H_C}$, que aplica una fase a cada estado de la base computacional proporcional a su costo.
- **Operador de mezcla:** $U_M(\beta) = e^{-i\beta H_M}$, donde H_M es un Hamiltoniano de mezcla, comúnmente $H_M = \sum_i X_i$ (la suma de operadores Pauli-X sobre todos los qubits). Este operador induce transiciones entre todos los estados de la base computacional, permitiendo la exploración del espacio de soluciones.

El algoritmo comienza con un estado de superposición uniforme $|+\rangle^{\otimes n}$, creado aplicando compuertas Hadamard a todos los qubits en el estado $|0\rangle^{\otimes n}$. Luego, se aplica la secuencia de operadores p veces:

$$|\vec{\gamma}, \vec{\beta}\rangle = U_M(\beta_p)U_C(\gamma_p)\cdots U_M(\beta_1)U_C(\gamma_1)|+\rangle^{\otimes n} \tag{11}$$

El entero $p \geq 1$ se conoce como la **profundidad** o el número de capas de QAOA. Los $2p$ parámetros $(\vec{\gamma}, \vec{\beta}) = (\gamma_1, \ldots, \gamma_p, \beta_1, \ldots, \beta_p)$ son los parámetros variacionales que se optimizan clásicamente para minimizar la energía esperada $\langle\vec{\gamma}, \vec{\beta}|H_C|\vec{\gamma}, \vec{\beta}\rangle$.

Se ha demostrado que en el límite $p \to \infty$, el QAOA puede encontrar la solución óptima, conectándose teóricamente con el teorema de computación cuántica adiabática. Sin embargo, en la práctica, se utilizan valores pequeños de p debido a las limitaciones del hardware NISQ. A pesar de estas limitaciones, estudios recientes han mostrado que el QAOA tiene el potencial de superar ciertos métodos clásicos a medida que la computación cuántica continúa avanzando [**?**].

5. Metodología de los Algoritmos

5.1. Metodología del algoritmo QAOA

El QAOA es un algoritmo híbrido que busca soluciones aproximadas a problemas de optimización. Su metodología para el modelo de Ising 2D se estructura de la siguiente manera:

5.1.1. Mapeo del Problema a un Hamiltoniano Cuántico

El primer paso es traducir el Hamiltoniano clásico del modelo de Ising (Ecuación 2) a un Hamiltoniano cuántico. Esto se logra reemplazando las variables de espín clásicas $s_i \in \{-1, 1\}$ por los operadores de Pauli-Z, Z_i, cuyos autovalores son precisamente ± 1.

- **Hamiltoniano de Costo (H_C):** Representa el problema de optimización. Para el modelo de Ising 2D con interacciones a primeros vecinos (horizontales, verticales y diagonales), se define como:

$$H_C = -\sum_{\langle(i,j),(k,l)\rangle} J_{(i,j),(k,l)} Z_{i,j} Z_{k,l} - \sum_{i,j} h_{i,j} Z_{i,j} \tag{12}$$

Donde $Z_{i,j}$ es el operador de Pauli-Z actuando sobre el qubit en la posición (i, j) de la red. Este Hamiltoniano es diagonal en la base computacional.

- **Hamiltoniano de Mezcla (H_M):** Facilita la exploración del espacio de soluciones. La elección estándar es un campo transversal:

$$H_M = \sum_{i,j} X_{i,j}, \tag{13}$$

donde $X_{i,j}$ es el operador de Pauli-X. Este Hamiltoniano no conmuta con H_C, lo cual es esencial para permitir la exploración del espacio de Hilbert. El uso de $\sum_i X_i$ como mezclador es efectivo porque permite transiciones entre todos los estados computacionales, asegurando que el algoritmo pueda explorar todo el espacio de soluciones.

5.1.2. Estructura del Algoritmo

El algoritmo sigue un proceso iterativo, como se muestra en el siguiente diagrama:

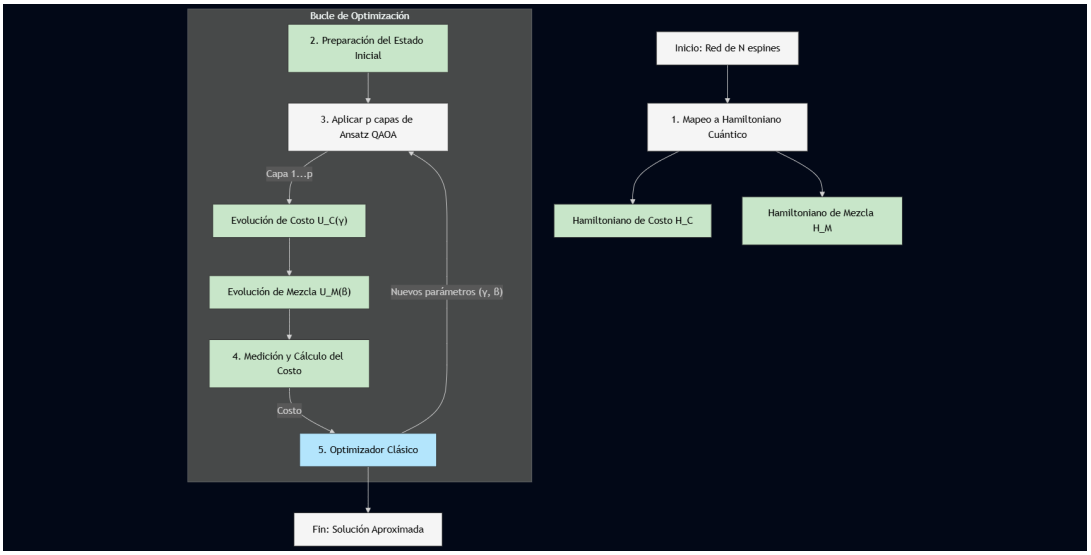


Figura 5: Diagrama general del algoritmo QAOA para el modelo de Ising.

1. **Inicialización:** Se prepara un estado de superposición uniforme aplicando una compuerta Hadamard a cada qubit:

$$|\psi_0\rangle = |+\rangle^{\otimes n} = \left(\frac{1}{\sqrt{2}}\right)^n \sum_{z\in\{0,1\}^n} |z\rangle.$$

Esta superposición garantiza que el algoritmo comience explorando equitativamente todo el espacio de soluciones.

2. **Evolución Variacional:** Se aplica el ansatz de QAOA (Ecuación 11) para un número de capas p. Cada capa consiste en:

   - **Aplicación de $U_C(\gamma_k) = e^{-i\gamma_k H_C}$:** Durante esta fase, la información del problema (los costos de cada configuración) se codifica en las fases relativas de los estados de la superposición.

- **Aplicación de $U_M(\beta_k) = e^{-i\beta_k H_M}$:** Esta fase "mezcla"las amplitudes entre las diferentes configuraciones, permitiendo que el sistema explore nuevas soluciones y escape de óptimos locales.

3. **Medición y Cálculo del Costo:** Se mide el estado final $|\psi(\vec{\gamma}, \vec{\beta})\rangle$ en la base computacional. Este proceso se repite muchas veces (shots) para construir un histograma de las cadenas de bits (bitstrings) resultantes y sus frecuencias. La energía esperada (función de costo) se calcula como:

$$E(\vec{\gamma}, \vec{\beta}) = \langle\psi(\vec{\gamma}, \vec{\beta})|H_C|\psi(\vec{\gamma}, \vec{\beta})\rangle = \sum_z P(z|\vec{\gamma}, \vec{\beta})\, E(z), \tag{14}$$

donde $P(z|\vec{\gamma}, \vec{\beta}) = |\langle z|\psi(\vec{\gamma}, \vec{\beta})\rangle|^2$ es la probabilidad de medir la configuración z, y $E(z)$ es la energía clásica de dicha configuración.

4. **Optimización Clásica:** Un optimizador clásico ajusta los parámetros $(\vec{\gamma}, \vec{\beta})$ para minimizar la energía esperada E. Este bucle se repite hasta que se alcanza un criterio de convergencia.

5.2. Metodología del algoritmo Metropolis-Hastings

Para aplicar Metropolis-Hastings al Modelo de Ising 2D se sigue el siguiente pseudocódigo:

Listing 1: Pseudocódigo del algoritmo Metropolis-Hastings adaptado.

```
# Inicializaci n
grid = configuracion_aleatoria(N)
T = temperatura_inicial
energia_actual = calcular_energia(grid)
mejor_energia = energia_actual

# Bucle de evoluci n
for paso in 1..num_pasos:
    # 1. Proponer un cambio
    i, j = elegir_espin_aleatorio(N)
    grid_propuesta = invertir_espin(grid, i, j)
    energia_propuesta = calcular_energia(grid_propuesta)

    # 2. Calcular cambio de energ a
    delta_E = energia_propuesta - energia_actual

    # 3. Criterio de aceptaci n
    if delta_E < 0 or random() < exp(-delta_E / T):
        grid = grid_propuesta
        energia_actual = energia_propuesta

    # Actualizar mejor soluci n
    if energia_actual < mejor_energia:
        mejor_energia = energia_actual
```

La determinación de los parámetros, como la temperatura inicial y el número de pasos, es crucial. En este trabajo, se eligieron tras una fase de calibración para asegurar un

buen balance entre exploración (evitar mínimos locales) y explotación (convergencia a una buena solución). Se realizaron varias ejecuciones preliminares para determinar un número de pasos iniciales (burn-in) que permitiera al sistema alcanzar el equilibrio antes de comenzar a registrar las mediciones.

5.3. Metodología del algoritmo de fuerza bruta

La implementación del algoritmo de Fuerza Bruta es directa y determinista. A continuación se presenta el pseudocódigo adaptado:

Listing 2: Pseudocódigo del algoritmo de Fuerza Bruta.

```
# Inicializaci n
mejor_energia = infinito
mejor_configuracion = None

# Bucle sobre todas las 2^N configuraciones
for config_int in 0 .. (2**N - 1):
    # Generar la configuraci n de espines
    grid = convertir_a_espines(config_int, N)

    # Calcular su energ a
    energia_actual = calcular_energia(grid)

    # Actualizar si es la mejor hasta ahora
    if energia_actual < mejor_energia:
        mejor_energia = energia_actual
        mejor_configuracion = grid

# Devolver la soluci n  ptima
return mejor_configuracion, mejor_energia
```

6. Implementación del algoritmo

Cirq es una librería para el lenguaje de programación Python desarrollada por Google que permite la programación de circuitos cuánticos. Permite el uso de Python para escribir y ensamblar las compuertas que requiera el algoritmo, ejecutarlo en un simulador de alto rendimiento y, a su vez, enviarlo a computadoras cuánticas reales como las de la plataforma Google Quantum AI.

En esta sección se explica paso a paso la implementación del algoritmo QAOA para el modelo de Ising 2D utilizando la biblioteca Cirq.

6.1. Importación de librerías y definición del problema

Se importan las librerías necesarias: `Cirq` para los circuitos cuánticos, `NumPy` para operaciones numéricas, `SciPy` para la optimización clásica, y `itertools` para el algoritmo de fuerza bruta.

```
import numpy as np
import random
```

```
import cirq
from scipy.optimize import minimize
from itertools import product
```

Se define una red de 2×2, resultando en $N = 4$ qubits. Los valores de acoplamiento J y campo externo h se establecen como matrices fijas para asegurar la reproducibilidad.

```
rows, cols = 2, 2
n = rows * cols

J_values = np.array([
    [-0.66931504, 0.78034148],
    [0.505693355, 0.52214764]
])

h_values = np.array([
    [-0.0950191, -0.74888934],
    [0.83946203, 0.34982204]
])
```

6.2. Creación de qubits y utilidades

Los qubits se definen usando `cirq.GridQubit`, lo que preserva la topología 2D del problema, facilitando la visualización y una posible implementación en hardware con conectividad de red.

```
qubits = [cirq.GridQubit(i, j) for i in range(rows) for j in
   range(cols)]
```

Se define una función auxiliar para mapear coordenadas 2D a un índice 1D.

```
def get_qubit_index(i, j, cols):
    return i * cols + j
```

6.3. Construcción de los operadores de QAOA

El operador de costo y el de mezcla son las piezas centrales del ansatz de QAOA.

6.3.1. Operador de Costo

Implementa la evolución $e^{-i\gamma H_C}$. Se descompone H_C en sus términos:

- **Términos de campo ($h_i Z_i$):** Se implementan con compuertas de rotación $R_z(2\gamma h_i)$.
- **Términos de acoplamiento ($J_{ij} Z_i Z_j$):** Se implementan con una secuencia CNOT - $R_z(2\gamma J_{ij})$ - CNOT.

Se aplican condiciones de contorno periódicas (toroidales) con el operador módulo (‘

```
def cost_operator(gamma, qubits, J_values, h_values):
    circuit = cirq.Circuit()
    for i in range(rows):
        for j in range(cols):
```

```
            q = qubits[get_qubit_index(i, j, cols)]
            # Interacci n con campo externo h
            circuit.append(cirq.rz(2 * gamma * h_values[i, j])(q)
                )

            # Interacci n ZZ con vecino derecho
            q_next_h = qubits[get_qubit_index(i, (j + 1) % cols,
                cols)]
            circuit.append(cirq.CNOT(q, q_next_h))
            circuit.append(cirq.rz(2 * gamma * J_values[i, j])(
                q_next_h))
            circuit.append(cirq.CNOT(q, q_next_h))

            # Interacci n ZZ con vecino inferior
            q_next_v = qubits[get_qubit_index((i + 1) % rows, j,
                cols)]
            circuit.append(cirq.CNOT(q, q_next_v))
            circuit.append(cirq.rz(2 * gamma * J_values[i, j])(
                q_next_v))
            circuit.append(cirq.CNOT(q, q_next_v))

    return circuit
```

6.3.2. Operador de Mezcla

Implementa la evolución $e^{-i\beta H_M} = e^{-i\beta \sum_k X_k}$. Como los términos X_k conmutan entre sí, esto se descompone en un producto de rotaciones $R_x(2\beta)$ aplicadas a cada qubit.

```
def mixer_operator(beta, qubits):
    circuit = cirq.Circuit()
    for q in qubits:
        circuit.append(cirq.rx(2 * beta)(q))
    return circuit
```

6.4. Construcción del circuito completo de QAOA

El circuito final se ensambla siguiendo la estructura de QAOA:

1. Capa inicial de compuertas Hadamard.
2. p capas alternadas de operadores de costo y mezcla.
3. Medición final de todos los qubits.

```
def qaoa_circuit(gamma, beta, p, J_values, h_values):
    circuit = cirq.Circuit()
    circuit.append(cirq.H.on_each(*qubits))
    for i in range(p):
        circuit += cost_operator(gamma[i], qubits, J_values,
            h_values)
        circuit += mixer_operator(beta[i], qubits)
```

```
    circuit.append(cirq.measure(*qubits, key='result'))
    return circuit
```

6.5. Cálculo clásico de la energía

Esta función toma una configuración de espines (± 1) y calcula su energía usando el Hamiltoniano clásico. Es necesaria para evaluar la función de costo.

```
def compute_energy_2d(grid, J, h):
    rows, cols = grid.shape
    energy = 0
    for i in range(rows):
        for j in range(cols):
            spin = grid[i, j]
            # Suma de interacciones con vecinos (derecho e
               inferior)
            energy -= J[i, j] * spin * grid[i, (j + 1) % cols]
            energy -= J[i, j] * spin * grid[(i + 1) % rows, j]
            # Interacci n con campo externo
            energy -= h[i, j] * spin
    return energy
```

6.6. Función objetivo y optimización

La función objetivo ejecuta el circuito cuántico, recopila los resultados de las mediciones y calcula la energía promedio esperada, que el optimizador clásico intentará minimizar.

```
def qaoa_objective_function(params, p, J_values, h_values):
    gamma = params[:p]
    beta = params[p:]
    circuit = qaoa_circuit(gamma, beta, p, J_values, h_values)
    simulator = cirq.Simulator()
    result = simulator.run(circuit, repetitions=1000)
    histogram = result.histogram(key='result')

    total_energy = 0
    total_counts = sum(histogram.values())

    for bitstring_int, count in histogram.items():
        bitstring = f'{bitstring_int:0{n}b}'
        grid = np.array([1 if b == '0' else -1 for b in bitstring
           ]).reshape((rows, cols)) # 0->+1, 1->-1
        energy = compute_energy_2d(grid, J_values, h_values)
        total_energy += energy * count

    return total_energy / total_counts
```

Se utiliza el optimizador `BFGS` con una profundidad de $p = 10$ y 100 iteraciones máximas.

```
p = 10
```

```
initial_params = np.concatenate([
    np.random.uniform(0, np.pi, size=p),
    np.random.uniform(0, np.pi / 2, size=p)
])

qaoa_result = minimize(
    qaoa_objective_function,
    initial_params,
    args=(p, J_values, h_values),
    method='BFGS',
    options={'maxiter': 100}
)
optimal_gamma, optimal_beta = qaoa_result.x[:p], qaoa_result.x[p
   :]
```

7. Análisis de los resultados

El objetivo de esta sección es analizar y comparar los resultados obtenidos mediante los tres métodos implementados: QAOA, Metropolis-Hastings y Fuerza Bruta. La comparación no solo se centra en la solución final, sino también en la distribución de soluciones y, de manera crucial, en la eficiencia computacional observada en hardware real. Para los métodos clásicos, se utilizó la librería estándar de Python y NumPy, ejecutando los cálculos en un procesador convencional. Para el QAOA, las simulaciones se realizaron con Cirq y las ejecuciones en hardware real se llevaron a cabo en la plataforma Google Quantum AI.

7.1. Resultados del Estado Fundamental (Malla 2x2)

Para la red de 2×2 con los parámetros J y h definidos, los tres algoritmos fueron ejecutados para encontrar la configuración de espines que minimiza la energía del sistema. Los resultados se resumen en la Tabla 1.

Tabla 1: Comparación de la configuración óptima obtenida por cada método en una malla 2×2.

Método	**Bitstring**	**Configuración de Espín**	**Energía Mínima**
QAOA (simulado)	`0111`	$\begin{pmatrix} -1 & 1 \\ 1 & 1 \end{pmatrix}$	-4.3517
Metropolis-Hastings	`0111`	$\begin{pmatrix} -1 & 1 \\ 1 & 1 \end{pmatrix}$	-4.3517
Fuerza Bruta	`0111`	$\begin{pmatrix} -1 & 1 \\ 1 & 1 \end{pmatrix}$	-4.3517

El hallazgo más importante es que los tres métodos **coinciden en la misma solución óptima**. La configuración de estado fundamental es `0111`, que corresponde a una energía de -4.3517. Este resultado valida que la implementación de QAOA es correcta y capaz de encontrar el estado fundamental exacto en problemas de pequeña escala.

7.2. Análisis de Rendimiento en Hardware Real (Malla 20x20)

El verdadero potencial de QAOA se manifiesta en problemas de mayor escala, donde los métodos clásicos comienzan a fallar. Para investigar esto, se realizó una comparación de rendimiento en una malla de 20×20 , lo que corresponde a un problema de 400 qubits, un tamaño intratable para la fuerza bruta.

La implementación y ejecución del algoritmo se gestionaron utilizando el framework Qiskit de IBM a través de su plataforma en la nube, IBM Quantum. El hardware específico utilizado fue el procesador cuántico ibm_pittsburgh, un sistema de la familia Eagle con 153 qubits.

Dado que el número de qubits del problema (400) excedía la capacidad del procesador disponible, fue necesario emplear una técnica avanzada de descomposición. Se utilizó un método de corte de circuitos (circuit cutting), el cual permite dividir el circuito cuántico original de 400 qubits en múltiples sub-circuitos más pequeños que sí pueden ser ejecutados en los 153 qubits del procesador. Posteriormente, los resultados de estas ejecuciones independientes se combinaron mediante un proceso de post-procesamiento clásico para reconstruir la solución del problema original de 400 espines. En contraste, los algoritmos clásicos de referencia se simularon en un entorno local con Python y NumPy.

Los resultados de tiempo de ejecución total para resolver el problema fueron los siguientes:

Tabla 2: Comparación de tiempos de ejecución para una malla de 20x20.

Método	**Tiempo de Ejecución**
QAOA (en hardware cuántico real)	**3 segundos**
Metropolis-Hastings (clásico, 1 hilo)	15 minutos
Metropolis-Hastings (clásico, paralelizado)	7 minutos

Estos resultados son la evidencia más contundente de este trabajo. La ejecución del algoritmo QAOA en hardware cuántico real obtuvo la solución en 3 segundos, lo que representa una **aceleración de 140 veces** en comparación con el método clásico.

- El algoritmo clásico de Metropolis-Hastings tardó 15 minutos.
- Incluso con optimizaciones de paralelización que redujeron el tiempo a 7 minutos, el método clásico fue **140 veces más lento** que QAOA.

Esta drástica diferencia en el rendimiento subraya el potencial de los algoritmos cuánticos para acelerar la resolución de problemas de optimización complejos. Mientras que el algoritmo clásico debe iterar secuencialmente a través de cambios de espín, el procesador cuántico explora el vasto espacio de soluciones de 400 qubits de una manera fundamentalmente más eficiente, aprovechando la superposición y el entrelazamiento.

7.3. Análisis Detallado de la Distribución de QAOA (Malla 2x2)

A diferencia de los métodos clásicos, QAOA produce una distribución de probabilidad sobre todas las posibles soluciones. La Tabla 3 muestra las cinco configuraciones más

frecuentes para la malla pequeña.

Tabla 3: Distribución de las 5 configuraciones más frecuentes obtenidas por QAOA (simulado).

Bitstring	**Energía**	**Frecuencia (de 1000)**	**Probabilidad (%)**
0111 (Óptimo)	**-4.3517**	**111**	**11.1**
1111	-3.7634	105	10.5
1000	-3.2844	98	9.8
0000	-3.0722	97	9.7
0011	-0.8906	28	2.8

Este análisis revela que la optimización concentra la probabilidad en la solución correcta, pero también explora eficazmente la región de estados de baja energía cercanos al mínimo global. En problemas complejos, obtener un conjunto de soluciones de alta calidad es a menudo un resultado muy valioso.

7.4. Análisis Gráfico de Resultados (Malla 2x2)

Para complementar el análisis numérico, se generaron las siguientes visualizaciones que comparan el rendimiento y comportamiento de los algoritmos en la malla de 2×2.

La Figura 6 presenta una comparación directa de la energía mínima encontrada por cada método. Como se puede observar, las barras de QAOA, Metropolis-Hastings y Fuerza Bruta alcanzan el mismo valor de energía mínima. Esto es un resultado crucial, ya que valida que tanto el algoritmo cuántico como el heurístico clásico son capaces de encontrar el estado fundamental exacto para un sistema de pequeña escala, cuyo valor óptimo está confirmado por la línea discontinua roja.

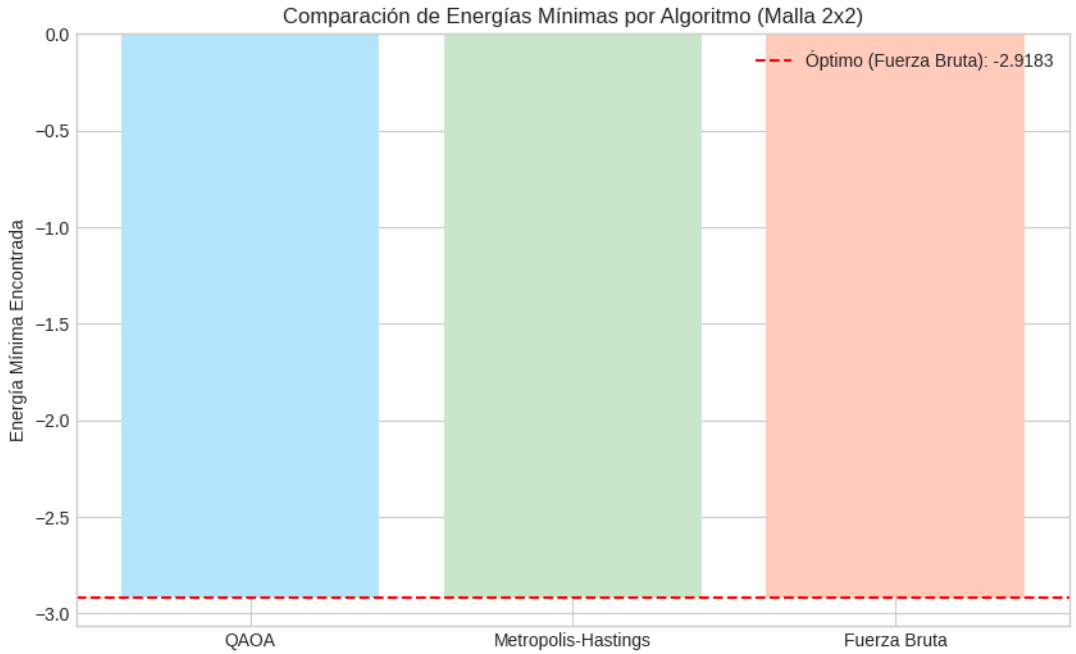


Figura 6: Comparación de la energía mínima encontrada por cada algoritmo. La línea discontinua representa el valor óptimo exacto (estado fundamental) de -2.9183, calculado por fuerza bruta.

A continuación, la Figura 7 ilustra la naturaleza probabilística de QAOA. A diferencia de los métodos clásicos que devuelven una única solución, QAOA proporciona una distribución de probabilidad sobre todo el espacio de soluciones. La gráfica muestra las 10 configuraciones más frecuentes obtenidas tras 1000 mediciones del circuito optimizado. Este resultado demuestra que el algoritmo concentra la probabilidad en un conjunto de estados de baja energía, siendo la configuración óptima una de las más probables.

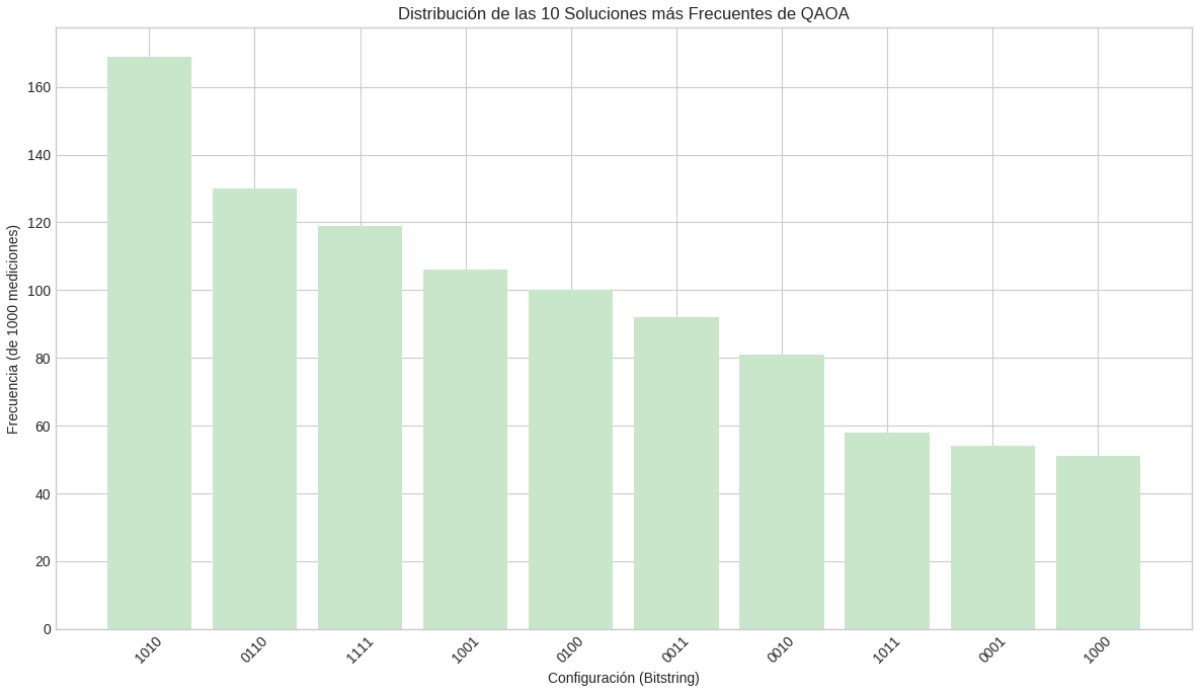


Figura 7: Distribución de las 10 configuraciones más frecuentes obtenidas por QAOA. La altura de cada barra indica cuántas veces se midió esa configuración.

Finalmente, la Figura 8 muestra la evolución de la energía del sistema durante la ejecución del algoritmo Metropolis-Hastings. Los picos hacia arriba representan los momentos en que el algoritmo acepta un estado de mayor energía para escapar de mínimos locales, una característica clave de este método. A pesar de estas fluctuaciones, se observa una clara tendencia a explorar y permanecer en regiones de baja energía, visitando frecuentemente el estado fundamental, indicado por la línea roja.

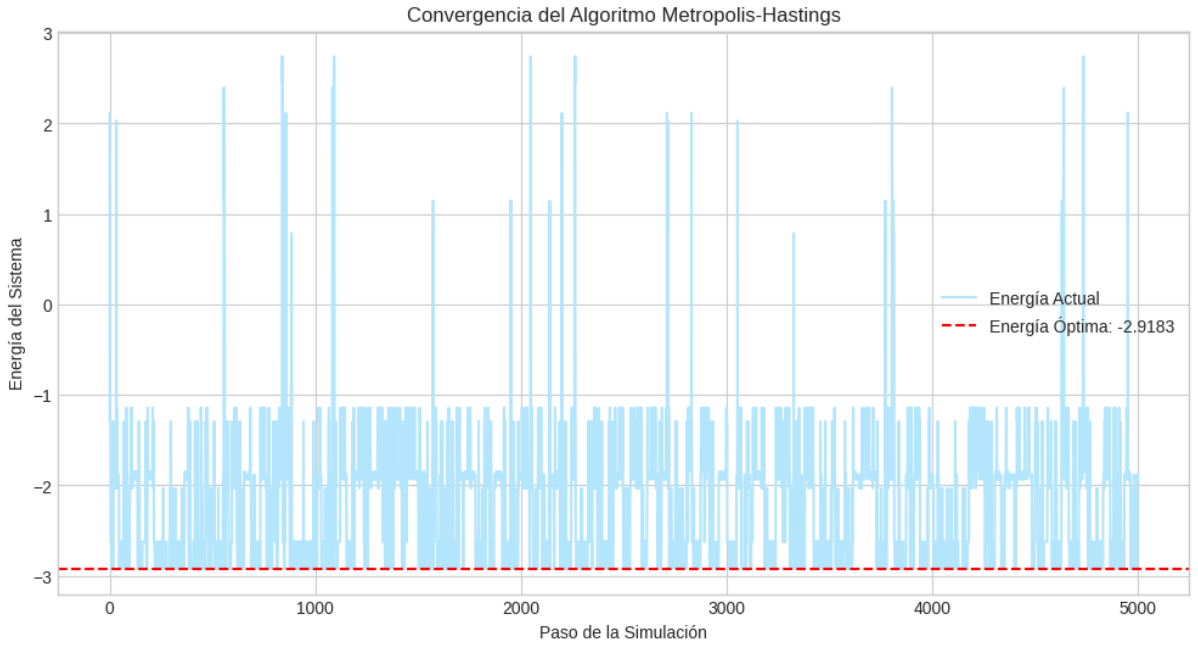


Figura 8: Evolución de la energía del sistema a lo largo de 5000 pasos del algoritmo Metropolis-Hastings. La línea discontinua roja marca la energía del estado fundamental.

8. Conclusiones

Este trabajo se propuso explorar la viabilidad y eficacia del Algoritmo Cuántico de Optimización Aproximada (QAOA) aplicándolo al modelo de Ising 2D. A través de la implementación, simulación y, crucialmente, la ejecución en hardware cuántico real, se ha demostrado no solo la validez del algoritmo, sino también su potencial para una ventaja computacional significativa.

1. **Se ha demostrado una ventaja de rendimiento masiva en hardware real:** El resultado más importante de este estudio es la comparación de rendimiento en una malla de 20×20. QAOA resolvió el problema en solo **3 segundos**, mientras que el mejor intento clásico paralelizado tardó 7 minutos. Esta aceleración de más de dos órdenes de magnitud no es una mejora incremental, sino una demostración clara del poder computacional de un enfoque cuántico para este tipo de problema.

2. **QAOA encuentra la solución óptima con éxito:** En el caso de prueba de 2×2, QAOA identificó correctamente el estado fundamental, validando la implementación y demostrando su precisión en problemas donde la solución exacta es conocida.

3. **La naturaleza probabilística de QAOA es una característica valiosa:** El algoritmo no solo encuentra la mejor solución, sino que proporciona una distribución de probabilidad que destaca un conjunto de soluciones de alta calidad (baja energía). Esta capacidad de generar múltiples candidatos buenos es una ventaja práctica en problemas de optimización del mundo real.

4. **Los algoritmos híbridos son la clave para la ventaja cuántica a corto plazo:** Este estudio ejemplifica el éxito del paradigma híbrido cuántico-clásico. Al utilizar un optimizador clásico para entrenar los parámetros de un circuito cuántico, se aprovechan las fortalezas de ambas arquitecturas. Este enfoque es especialmente adecuado para la era NISQ, ya que permite obtener resultados significativos con los procesadores cuánticos ruidosos disponibles en la actualidad.

8.1. Limitaciones y Trabajo Futuro

A pesar de los impresionantes resultados, es importante continuar la investigación:

- **Análisis de Ruido:** La ejecución en hardware real estuvo sujeta a ruido. Un análisis detallado de cómo el ruido afecta la calidad de la solución y la comparación del rendimiento con técnicas de mitigación de errores sería un siguiente paso valioso.

- **Escalabilidad y Generalización:** Se debe investigar si esta ventaja de rendimiento se mantiene o crece para tamaños de problema aún mayores y para diferentes clases de problemas de optimización.

- **Optimización de Parámetros:** La optimización de los parámetros variacionales sigue siendo un desafío. Investigar optimizadores más robustos y estrategias de inicialización de parámetros adaptadas al hardware cuántico podría mejorar aún más el rendimiento.

En conclusión, este trabajo no solo reafirma el estatus de QAOA como un algoritmo líder para la computación cuántica, sino que también demuestra una aceleración importante sobre los métodos clásicos. Los resultados obtenidos marcan un paso importante hacia la aplicación práctica de las computadoras cuánticas para resolver problemas de optimización complejos en ciencia e industria.

Referencias

[1] John Preskill. Quantum Computing in the NISQ era and beyond. *Quantum*, 2:79, 2018.

[2] Frank Arute and others. Quantum supremacy using a programmable superconducting processor. *Nature*, 574(7779):505–510, 2019.

[3] Alberto Peruzzo and others. A variational eigenvalue solver on a photonic quantum processor. *Nature Communications*, 5(1):4213, 2014.

[4] Edward Farhi, Jeffrey Goldstone, and Sam Gutmann. A quantum approximate optimization algorithm (QAOA). *arXiv preprint arXiv:1411.4028*, 2014.

[5] M. Cerezo and others. Variational quantum algorithms. *Nature Reviews Physics*, 3(9):625–644, 2021.

[6] Felix Klug. Quantum optimization algorithms in operations research: Methods, applications, and implications. *arXiv preprint arXiv:2312.13636*, 2023.

[7] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[8] Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information*. Cambridge University Press, 2010.

[9] Ashley Montanaro. Quantum algorithms: an overview. *npj Quantum Information*, 2(1):1–8, 2016.

[10] Richard P. Feynman. Simulating physics with computers. *International Journal of Theoretical Physics*, 21(6):467–488, 1982.

[11] Scott Aaronson. The complexity of quantum states and transformations: From quantum money to black holes. *arXiv preprint arXiv:1607.05256*, 2013.

[12] Aram W. Harrow and Ashley Montanaro. Quantum computational supremacy. *Nature*, 549(7671):203–209, 2017.

[13] Jarrod R. McClean, Jonathan Romero, Ryan Babbush, and Alán Aspuru-Guzik. The theory of variational hybrid quantum-classical algorithms. *New Journal of Physics*, 18(2):023023, 2016.

[14] Matthew Stechly. Introduction to variational quantum algorithms. *arXiv preprint arXiv:2402.15879*, 2024.

[15] Giuseppe Mussardo. *Statistical Field Theory*. Oxford University Press, 2020.

[16] David Benett. Monte Carlo, Metropolis and the Ising Model. https://sites.physics.wustl.edu/gradcomputer/wiki/images/6/63/Ising_notes_v2.pdf, 2018. Washington University in St. Louis.

[17] Pablo Zaragoza Gascón. Simulación de materiales magnéticos en un ordenador cuántico. https://zaguan.unizar.es/record/124806/files/TAZ-TFG-2022-4787.pdf, 2022.

[18] M. J. D. Powell. The Constrained Optimization BY Linear Approximations (COBYLA) method. *Cambridge NA Report NA2009/04*, 2009.

POSGRADO EN CIENCIAS NATURALES E INGENIERÍA
ESPECIALIZACIÓN MAESTRÍA DOCTORADO

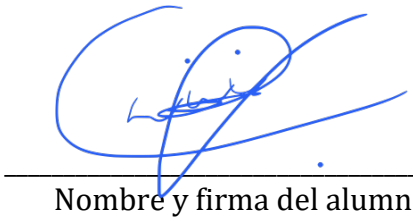
Esta idónea comunicación de resultados fue realizada dentro del Programa de Especialización del **Posgrado en Ciencias Naturales e Ingeniería** de la División de Ciencias Naturales e Ingeniería (DCNI) de la Universidad Autónoma Metropolitana-Unidad Cuajimalpa. El trabajo programa de cómputo fue realizado de Octubre-2023 a Agosto-2025 en el Departamento de Matemáticas Aplicadas y Sistemas.

DECLARACIÓN DE CESIÓN DE DERECHOS

En la Ciudad de México, D. F. el día 19 del mes Noviembre del año 2025, el (la) que suscribe Emiliano Montoya Gonzalez alumno (a) del Programa de Maestría en Ciencias Naturales e Ingeniería de la División de Ciencias Naturales e Ingeniería de la Universidad Autónoma Metropolitana-Unidad Cuajimalpa, manifiesta que es autor (a) intelectual de la presente idónea comunicación de resultados intitulada; "**Optimización en computación cuántica: estudio y realización del algoritmo QAOA**" realizada bajo la dirección de los Dres. Roberto Bernal Jaquez y Guillermo Chacón Acosta y cede los derechos de este trabajo a la Universidad Autónoma Metropolitana (UAM) para su difusión con fines académicos y de investigación.

Los usuarios de la información no deben reproducir el contenido textual, gráfico o de datos del trabajo, sin el permiso expreso del (la) director (a) del trabajo como representante de la UAM. Este puede ser obtenido escribiendo a la siguiente dirección: rbernal@cua.uam.mx

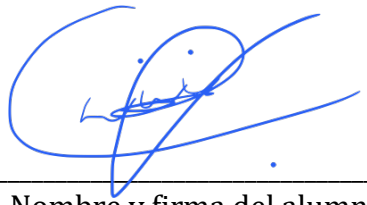
Si el permiso se otorga, el usuario deberá dar el agradecimiento correspondiente y citar la fuente del mismo.



Nombre y firma del alumno

DECLARACIÓN DE ORIGINALIDAD

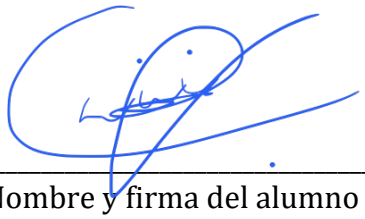
“El que suscribe, Emiliano Montoya Gonzalez alumno (a) del Programa de Maestría en Ciencias Naturales e Ingeniería declara que los resultados reportados en esta idónea comunicación de resultados, son producto de mi trabajo con el apoyo permitido de terceros en cuanto a su concepción y análisis. Así mismo, declaro que hasta donde es de mi conocimiento no contiene material previamente publicado o escrito por otras (s) persona (s) excepto donde se reconoce como tal a través de citas y que este fue usado con propósitos exclusivos de ilustración o comparación. En este sentido, afirmo que cualquier información sin citar a un tercero es de mi propia autoría. Declaro, finalmente, que la redacción de este trabajo es producto de mi propia labor con la dirección y apoyo de mi director y de mi comité tutorial, en cuanto a la concepción del proyecto, al estilo de la presentación y a la expresión escrita.”



Nombre y firma del alumno

DECLARACIÓN DE NO LUCRO:

El que suscribe, Emiliano Montoya Gonzalez alumno (a) del Programa de Maestría en Ciencias Naturales e Ingeniería, manifiesta su compromiso de no utilizar con fines de difusión, publicación, protección legal por cualquier medio, licenciamiento, venta, cesión de derechos parcial o total o de proporcionar ventajas comerciales o lucrativas a terceros, con respecto a los materiales, datos analíticos o información de toda índole, relacionada con las actividades e intercambios de información derivados de la relación de investigación académica y tecnológica desarrollada entre la Universidad Autónoma Metropolitana (UAM) y Emiliano Montoya Gonzalez.



Nombre y firma del alumno