

UNIVERSIDAD POLITÉCNICA DE MADRID
ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA DE SISTEMAS
INFORMÁTICOS



Trabajo de Fin de Grado de Ingeniería de Computadores 2019/2020

TÍTULO:

COMPUTACIÓN CUÁNTICA: Introducción al paradigma cuántico universal, situación actual, herramientas de desarrollo, estudio e implementación del algoritmo Quantum Counting clásico, desarrollo de una versión simplificada del algoritmo y aplicaciones prácticas.

AUTOR:

Todor Krasimirov Ivanov

TUTOR:

Giannicola Scarpa

Resumen

La computación cuántica es un campo innovador en las ciencias de computación e información que actualmente se encuentra en vías de desarrollo con algoritmos e infraestructuras cada vez más elaborados. Este tipo de computación, en concreto la computación cuántica universal de puertas, posee un gran potencial de cálculo y, a lo largo de todo el proyecto, se detallan las características y propiedades que la hacen una computación alternativa y ventajosa frente a la computación clásica en problemas de cálculos masivos.

Además de incluir la información más relevante sobre el contexto histórico de la mecánica cuántica, por la cual la computación cuántica se ha podido desarrollar, se introducen los conceptos principales de esta para la construcción de algoritmos con el paradigma de programación cuántico universal, analizando, a su vez, las ventajas que supone la utilización de dicho paradigma.

También, se estudia el estado actual de la computación cuántica referente a tecnologías como los entornos de programación cuántica y las infraestructuras de máquinas cuánticas existentes de empresas relevantes como *IBM* o *Microsoft*.

Una vez introducidos los conceptos fundamentales sobre computación cuántica, se detalla la construcción del algoritmo cuántico *Quantum Counting* con el cual se cuenta el número de soluciones de una determinada función objetivo abstracta, adaptable a muchos problemas reales, demostrando una ventaja cuántica frente a los algoritmos clásicos más eficientes que realizan el mismo trabajo.

Análogamente, se investiga y se desarrolla un algoritmo cuántico muy reciente, más simple y eficiente que el *Quantum Counting* clásico, denominado *Simpler Quantum Counting*, el cual facilita la ejecución en ordenadores cuánticos reales por la reducción sustancial del número de operadores controlados.

Estos dos algoritmos, el clásico y el simplificado, son ejecutados en simuladores y máquinas cuánticas utilizando funciones prueba con el correspondiente análisis de resultados.

Finalmente, se enumeran una serie de aplicaciones prácticas del *Quantum Counting* útiles para la implementación de otros algoritmos cuánticos como el *Quantum Amplitude Estimation* o el algoritmo de Grover, además de otras utilidades en campos como la inteligencia artificial o problemas de explosión combinatoria, en los que la computación cuántica supone ventajas en cuanto a complejidad computacional.

Abstract

Quantum computing is a breakthrough field in information and computation science whose quantum software and hardware are in a constant development process improving increasingly its possibilities. This type of computation, specifically the universal quantum gates computation, holds a great potential of calculation and its features and properties are explained all along the project, demonstrating the advantages against classical computation when it comes to massive calculation problems.

Firstly, fundamental concepts of universal quantum computation are introduced focusing on algorithm construction using the quantum gate programming paradigm which advantages are analyzed. Also, an historic context of quantum mechanics is given to trace the evolution that brought quantum computing to reality.

Moreover, quantum computing state-of-art is studied researching recent technologies as quantum programming environments and physical quantum devices of outstanding corporations as *IBM* or *Microsoft*.

Once all fundamental concepts of quantum computing are explained, the construction of a quantum algorithm called *Quantum Counting* is detailed, whose purpose is to find the number of solutions of a given abstract function adaptable to real life problems. The quantum advantage of this algorithm is demonstrated by comparing it to the best classical algorithms with the same purpose.

Then, a simplified version of the *Quantum Counting* algorithm is studied and implemented, named as *Simpler Quantum Counting*. It is a brand new quantum algorithm which reduces the number of controlled operators in its related quantum circuit easing, consequently, its physical execution in quantum machines.

These two quantum algorithms, the classical version and the simplified one, are run in quantum simulators and physical quantum computers including their corresponding results and discussions.

At last, several useful applications of the *Quantum Counting* are listed and presented as future jobs. Some of them are associated with other quantum algorithms implementation as *Quantum Amplitude Estimation* or Grover's algorithm and the rest are aimed to artificial intelligence or combinatorial explosion problems where quantum computing is advantageous in terms of computational complexity.

ÍNDICE

1. Introducción	6
1.1. Definición del problema a resolver y objetivos	7
2. Contexto histórico de la mecánica cuántica	8
Experimento de Young	8
Hamiltoniano cuántico	11
Cuantización de Max Planck y Albert Einstein	11
Espacio de Hilbert	12
Modelo atómico de Ernest Rutherford	12
Modelo atómico de Niels Bohr	14
Números cuánticos	15
Hipótesis de De Broglie	15
Ecuación de Schrödinger	16
Principio de exclusión de Pauli	16
Principio de incertidumbre de Heisenberg	17
Ecuación de Dirac	17
Concepto de ordenador cuántico: Algoritmos de Shor y de Grover	18
3. Introducción a la computación cuántica	19
Tipos de computadores cuánticos	24
Qubits	26
Notación de Dirac	30
Operadores y puertas lógicas cuánticas	32
Circuitos cuánticos	36
Ventajas de la computación cuántica	40
Inconvenientes de la computación cuántica	41
4. Estado del arte	43
5. Herramientas para el desarrollo de programas cuánticos	51
Microsoft Quantum Development Kit	52
Qiskit	55
6. <i>Quantum Counting</i> : Algoritmo de búsqueda cuántico	58
Algoritmo de Grover	58
Transformada cuántica de Fourier	62
Quantum Phase Estimation	64
<i>Quantum Counting</i> : Búsqueda del número de soluciones	66
Complejidad computacional	68
Complejidad en memoria	69
Simulación del algoritmo en Q#	70
Simulación del algoritmo con Qiskit	73
Ejecución en el computador cuántico de <i>IBM</i>	73
7. Simpler <i>Quantum Counting</i>	75
Complejidad computacional y en memoria	80
Simulación del algoritmo en Q#	81
Simulación del algoritmo con Qiskit	82
Ejecución en el computador cuántico de <i>IBM</i>	84
Conclusiones	85
8. Aplicaciones prácticas	86
9. Líneas futuras	90

1. Introducción

A continuación, se detalla la estructura del desarrollo del proyecto de fin de grado titulado *Computación cuántica: Introducción al paradigma cuántico universal, situación actual, herramientas de desarrollo, estudio e implementación del algoritmo Quantum Counting clásico, desarrollo de una versión simplificada del algoritmo y aplicaciones prácticas*.

Primero, en el apartado 1.1 se resume el problema a resolver y los objetivos que se tratan a lo largo de todo el trabajo.

En la sección 2, se exponen los inicios de la mecánica cuántica y el transcurso de esta a través de la historia, mencionando los descubrimientos y avances más relevantes, terminando por el surgir del concepto de computación cuántica.

Después, en la sección 3, se introducen los conceptos y herramientas esenciales en la computación cuántica, hablando de las propiedades de la mecánica cuántica, de su utilidad para el tratamiento de información y de la representación de esta para la computación cuántica universal de puertas. También, se explican las ventajas e inconvenientes de este tipo de computación.

El apartado 4 se corresponde con una investigación sobre el estado del arte de la computación cuántica de puertas, donde se habla de la situación actual de los computadores cuánticos físicos y de los algoritmos que se pueden implementar. Adicionalmente, se habla de la ventaja cuántica alcanzada por *Google* y de su significado.

En la siguiente sección, la sección 5, se exponen algunas de las herramientas más conocidas disponibles para el desarrollo de programas cuánticos, como lenguajes de programación y kits de desarrollo software. Se estudian en profundidad las tecnologías que ofrece *Microsoft*, con su kit de desarrollo *Microsoft Quantum Development Kit*, su lenguaje de programación cuántico *Q#* y sus aspiraciones en los computadores cuánticos, e *IBM* con su entorno de trabajo *Qiskit*, las máquinas cuánticas de las que disponen y, también, sus futuros proyectos.

Posteriormente, en la sección 6, se detalla la construcción del algoritmo *Quantum Counting* clásico con las respectivas demostraciones de su funcionamiento, incluyendo explicaciones de todas las partes que lo componen: algoritmo de Grover, transformada cuántica de Fourier y la estimación de fase cuántica. Además, se realiza un análisis de la complejidad computacional y en memoria del algoritmo, junto con la correspondiente simulación del algoritmo con *Q#* y *Qiskit* y su ejecución en máquinas reales de *IBM*.

Al igual que en la sección 6, en la sección 7 se incluye exactamente lo mismo respecto de una versión simplificada del *Quantum Counting* que no utiliza la estimación de fase cuántica, demostrando dicho método, simulándolo con pruebas y ejecutándolas en computadores cuánticos reales. Este algoritmo simplificado se trata de una versión reciente y este trabajo recoge una de las primeras implementaciones con sus respectivas pruebas y experimentos. También, se discuten las complejidades del algoritmo y se da una comparativa con la versión clásica del algoritmo con las conclusiones correspondientes.

Finalmente, en los apartados 8 y 9, se investigan y exponen una serie de aplicaciones prácticas y trabajos futuros, tanto para el *Simpler Quantum Counting* como para otras áreas de interés en la computación cuántica universal tales como inteligencia artificial con *Quantum Machine Learning*.

1.1. Definición del problema a resolver y objetivos

Este proyecto aborda el tema de la computación cuántica universal o de puertas con desarrollos, explicaciones y análisis de cómo funciona este tipo de computación de forma general introduciendo conceptos primordiales de computación cuántica, herramientas de desarrollo de programas cuánticos y la situación actual de todo este campo de estudio, junto con una contextualización histórica de la mecánica cuántica.

En concreto, en el problema que se aborda es el de desarrollar e implementar el algoritmo de *Quantum Counting* clásico que hace referencia a un algoritmo de contabilización de soluciones dada una función objetivo mediante el paradigma cuántico universal.

También se investiga y se desarrolla una versión simplificada del algoritmo que recibe el nombre de *Simpler Quantum Counting*, cuyo objetivo es el mismo, pero utilizando otro método distinto al algoritmo clásico que mejora la complejidad en tiempo de ejecución y permite ejecutar el algoritmo en máquinas cuánticas reales, dadas las restricciones de estas.

El objetivo del proyecto es indagar en las tecnologías cuánticas estudiando las herramientas e infraestructuras actuales y desarrollar algoritmos útiles que supongan una ventaja frente a los que se pueden implementar con informática clásica, como es el *Simpler Quantum Counting*.

Adicionalmente, se tienen una serie de objetivos futuros como la aplicación directa de algoritmos cuánticos para resolver problemas intratables clásicamente o la aplicación de determinados algoritmos cuánticos para la creación de otros procesos necesarios en casi cualquier programa cuántico como es el *Quantum Amplitude Estimation*.

2. Contexto histórico de la mecánica cuántica

La computación cuántica tiene sus primeros orígenes potenciales en la historia de la mecánica cuántica con el descubrimiento de las propiedades y los comportamientos físicos que poseen los sistemas cuánticos o, en otras palabras, los sistemas cerrados de partículas tales como átomos, iones, fotones u otras partículas de escala atómica o subatómica que siguen las leyes de la mecánica cuántica.

En base a la información de [1, 2, 3, 4] se redacta la historia de la mecánica cuántica de forma cronológica,

Experimento de Young

[5, 6] En 1801, un científico inglés, Thomas Young, realizó un experimento que revolucionaría la física de aquel entonces, al que se denominó el experimento de la doble rendija de Young. Este famoso experimento indicaba que la luz se podía manifestar físicamente de dos formas: como onda o como corpúsculo. Esto mostró lo que hoy se conoce como la doble naturaleza de la luz.

La estructura del experimento estaba formada por un foco emisor de haces de luz, una lámina con dos rendijas y una pantalla oscura en la que se reflejaría la luz de forma visual. El experimento consistió en emitir haces de luz controladas desde el foco que posteriormente pasarían a través de las dos rendijas, reflejándose finalmente en la pantalla. Se puede ver la configuración física del experimento en la figura, aunque se ha añadido una lámina intermedia con una rendija para limitar el foco de luz.

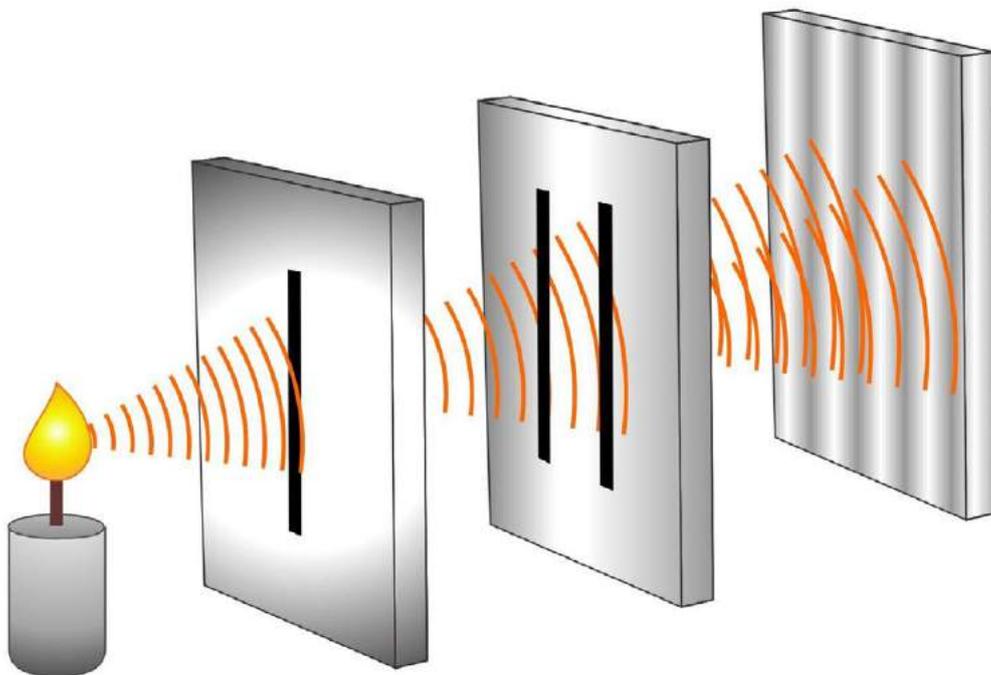


Figura 1. Experimento de la doble rendija de Young.

Fuente: <https://i.pinimg.com/originals/c6/c1/82/c6c182d76768b7cd6a962f4c714498a4.jpg>

Este experimento de forma cruda, tal y como se ha expuesto, genera en la pantalla un patrón de interferencia que, en un principio se debe a la naturaleza ondulatoria de la luz y es que las ondas de luz, al pasar por ambas rendijas, sufren el efecto de la difracción. Este efecto causa que la luz proveniente de un único foco, al cruzar las rendijas, se difracte en ellas y se generen dos focos de ondas. Por ello, después de cruzar las rendijas, existen zonas donde la interferencia entre las ondas provenientes de dichos focos es destructiva, es decir, se generan zonas de oscuridad y otras zonas donde hay interferencia constructiva, donde ambas ondas se combinan y causan la zona de luz.

La teoría de ondas electromagnéticas es lo que explica el patrón de interferencia que se forma cuando la luz monocromática pasa por dos rendijas.

En la figura siguiente, se muestran las distintas configuraciones del experimento de Young.

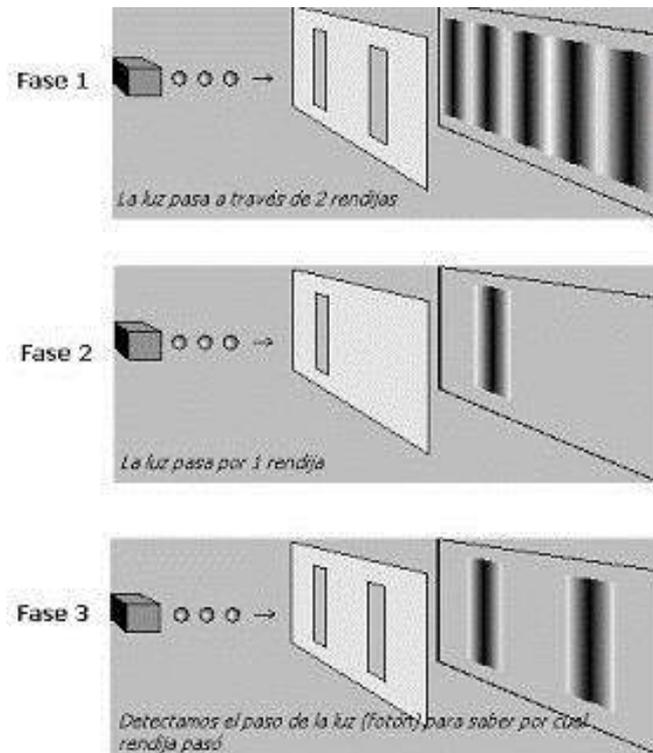


Figura 2. Fases del experimento de Young.

Fuente: http://bp2.blogqer.com/hOOPMnNUd9s/Ry_TrrqXoXI/AAAAAAAAACw/IGZHVafI2u0/s1600-h/Dibuj0.JPG

En la fase 1, la configuración es la expuesta anteriormente, un único foco de luz que pasa a través de dos rendijas y produce un patrón de interferencia en la pantalla.

En cuanto a la fase 2, se utiliza únicamente una rendija y se observa que la luz se muestra en la pantalla con la forma de la rendija, lo cual es totalmente intuitivo.

Sin embargo, en la fase 3, se repite la configuración de la primera fase, pero esta vez se añade un detector de paso de luz con lo que, si el foco emite haces de luz de forma intermitente, dicho detector será capaz de contabilizar el número de haces que han pasado por una rendija y por otra.

En una primera aproximación y contando con que un haz de luz tiene forma de onda, se esperaría que el haz que se emite en cada momento pasara por ambas rendijas, es decir, que la cantidad de haces detectados en una rendija será igual a los detectados en la otra y lo

observado en la pantalla sería lo mismo que en la fase 1, cuando no se detectaba el paso de luz.

De forma contra intuitiva, esto no sucede. En su lugar, lo que se origina es que, al comprobar por qué rendija pasa el haz de luz, se elimina el patrón de interferencia observado en la primera fase y lo que se observa son dos únicas zonas en las que la luz se ha reflejado sobre la pantalla. Estas dos zonas se corresponden con las dos rendijas por las que ha pasado la luz.

Según incrementa el número de haces de luz que se emiten, los detectores de luz contabilizan más o menos el mismo número de haces en una rendija que en otra, lo cual quiere decir que, cuando se utilizan detectores, la mitad de las veces la luz toma el camino de una de las rendijas y la otra mitad pasa por la segunda rendija.

La diferencia entre la fase 1, en la que no existen detectores en las rendijas, y la fase 2, en la que estos se introducen en el experimento, es que en la fase 1 la luz se comporta como una onda dando lugar a interferencias constructivas y destructivas lo cual genera el patrón de interferencia. Mientras que, en la fase 2, al detectar que un haz de luz pasa por una rendija y no por la otra, la luz se comporta como partícula dado que no se dan los efectos de interferencia y se pueden contabilizar el número de fotones que pasan por cada rendija.

Esta diferencia lleva a una situación en la que se pueden distinguir dos manifestaciones físicas distintas de la luz: la onda y la partícula. Cuando se controla la emisión de luz y se detecta el camino que esta toma, la luz aparece en la realidad como una partícula clásica que pasa a través de una de las dos rendijas. En cambio, cuando no hay trazabilidad de la luz emitida por el foco y no se verifica el paso de esta a través de las rendijas, la luz toma la forma de onda y pasa por ambas rendijas al mismo tiempo difractándose y produciendo el efecto de la interferencia ondulatoria.

Existen varias interpretaciones de este experimento, pero, principalmente, lo que da a entender es que la luz tiene distintas formas de manifestarse en la realidad y dichas manifestaciones se basan en la variable de observación y determinación del comportamiento.

Al tratar de cuantificar los haces de luz mediante detectores, la luz se manifiesta como una partícula cuantificable de la que se pueden distinguir propiedades como el spin o la masa. Sin embargo, cuando simplemente se observa qué sucede en el experimento sin detección alguna, la luz se comporta como una onda electromagnética con una frecuencia determinada que se propaga a través de las tres dimensiones espaciales con una velocidad de propagación igual a la velocidad de la luz.

Esto fue muy desconcertante en cuanto a que la variable que provocaba una manifestación física u otra era la observación, el intento de determinación del ser humano. Dicha observación era la causante de que las propiedades ondulatorias como la interferencia y la difracción se perdieran y, en su lugar, se transformaran en propiedades corpusculares coherentes con la lógica clásica como la traza de la partícula.

El experimento no solo se realizó con radiación electromagnética (fotones), sino que también se aplicó con otras partículas de escala atómica como los electrones llegando a las mismas conclusiones.

A partir del experimento de Young, se abrió un nuevo camino de la física en la que la observación o detección de partículas de escala diminuta provocaba una modificación en la manifestación de dicha partícula en la realidad observable por el ser humano, en cuanto a que

los efectos observados en el experimento, manteniendo siempre las características, son diferentes si se observa o no el proceso.

Hamiltoniano cuántico

Después, a mediados del siglo XIX, llegó William Rowan Hamilton con un trabajo que sería decisivo para la mecánica cuántica, el hamiltoniano. [7, 8]

En mecánica clásica, el hamiltoniano describe el estado de un sistema clásico en función de su posición y su momento lineal mientras que, en mecánica cuántica, el hamiltoniano hace referencia a la energía total del sistema como un observable.

De manera formal, el hamiltoniano cuántico se puede definir como un operador autoadjunto en el espacio complejo de Hilbert y con el que se pueden obtener los posibles valores de energía mediante los autovalores del operador hamiltoniano.

Este concepto se utilizó posteriormente junto con la ecuación de Schrödinger para definir la evolución temporal de los estados cuánticos de un sistema.

Cuantización de Max Planck y Albert Einstein

[9] Cuando dos teorías universales, la teoría de la gravitación universal y la teoría de las ondas electromagnéticas, se volvieron insuficientes para determinar ciertos fenómenos como la radiación térmica, es decir, la vibración de las partículas de un cuerpo donde se obtenían cantidades infinitas de radiación, la física conocida hasta el momento no podía explicar con determinación sucesos como este. Fue entonces cuando Max Planck encontró un enfoque que daba una solución plausible ante la incoherencia de la radiación infinita.

El primer paso en el descubrimiento teórico de la mecánica cuántica se dio en el año 1900 con Max Planck, un físico y matemático que contribuyó con grandes avances científicos y que posteriormente fue ganador del Premio Nobel de Física en 1918 por su teoría de la cuántica.

En 1900, Planck explicó la radiación de un cuerpo negro estableciendo la teoría de que la luz emitida y absorbida por dicho cuerpo se daba en forma de cuantos o paquetes de energía. Partiendo de esta idea, Planck llegó al descubrimiento de una constante universal, la constante de Planck, que discretiza las partículas como cantidades de energía y con la que desarrolló un modelo matemático que mantuviera los principios de dicha teoría, llegando a la conocida equivalencia proporcional entre la energía de una partícula y su frecuencia de oscilación donde la proporción es la constante de Planck, también llamada relación Planck-Einstein.

Esta constante universal es de gran relevancia para el resto de la historia de la mecánica cuántica puesto que se define como el cuanto de acción elemental, es decir, la cantidad mínima de acción en un proceso físico envuelta por el producto entre la energía del proceso y el tiempo que tarda en transcurrir. Al hablar de cuantos de energía y de que la constante es la acción mínima, se deduce que la energía de cualquier proceso físico va a ser siempre un múltiplo entero de la constante de Planck.

La teoría de la mecánica cuántica era incompatible con la física clásica pero después de exhaustivas comprobaciones de la constante de Planck se verificó como constante universal y, junto a ella, se incluyeron nuevas unidades físicas como el tiempo de Planck, la longitud de Planck, la masa de Planck y la temperatura de Planck. Todas estas unidades dictaban que las

unidades de medida universales se podían cuantificar siempre como múltiplos de estas nuevas unidades físicas, pasando de un espacio continuo de medida a otro discreto.

Básicamente, estas nuevas unidades de medida universales dadas por Planck se refieren a la mínima cantidad de temperatura, tiempo, longitud y masa que pueden existir en un sistema físico debido a la mecánica cuántica.

Albert Einstein también jugó un papel importante ya que, en 1905, explicó el efecto fotoeléctrico basándose en la hipótesis de Planck y postuló que la luz o radiación electromagnética se podía dividir en un número finito de cuantos de energía, que más tarde serían denominados como fotones, y que se definen como puntos en el espacio en lugar de como ondas transversales. Cuando una fuente emite luz desde un punto, la energía de dicha luz no se distribuye de forma continua, sino que está formada por un número entero de cuantos de energía localizados en el espacio y que se mueven conjuntamente sin dividirse, dando lugar a la posibilidad de absorción o emisión del conjunto completo.

Espacio de Hilbert

[10] El espacio de Hilbert surgió en 1906 cuando el matemático David Hilbert generalizó el concepto de espacio euclídeo extendiéndolo a espacios de cualquier dimensión, incluso infinita.

Su utilidad es simplificar los conceptos de serie de Fourier y transformaciones lineales, por lo que fue de gran contribución para el desarrollo del formalismo matemático de la mecánica cuántica.

Esta formulación permitió trabajar con transformaciones que se denominan operadores y que son los responsables de la evolución en el estado de las funciones de onda de los sistemas cuánticos.

Modelo atómico de Ernest Rutherford

[11] Ernest Rutherford fue un físico británico-neozelandés que propuso una estructura atómica en 1911 a partir de un experimento muy importante y del modelo atómico de Thomson de 1904 por el cual decía que el átomo era una esfera positiva con electrones incrustados en su superficie que neutralizaban la carga eléctrica del átomo.

El experimento revolucionario de Rutherford es el de la lámina de oro [12]. En él, se bombardeaba una lámina de oro de grosor ínfimo con una fuente de partículas alfa o, más concretamente, de núcleos de helio sin envoltura electrónica, únicamente formados por dos protones y dos neutrones y, por tanto, partículas de carga positiva. Rodeando la lámina de oro, se establecía una pantalla circular de sulfuro de zinc con la cual se podían detectar las colisiones de las partículas alfa que pasaban a través de la lámina de oro.

El resultado del experimento fue que la mayoría de partículas alfa provenientes de la fuente atravesaban la lámina de oro sin desviarse puesto que la zona de la pantalla con mayor densidad de impactos coincidía con una trayectoria no perturbada de las partículas emitidas por la fuente. También se observaron otras zonas con una densidad menor de colisiones situadas en los alrededores de la zona de densidad principal, es decir, zonas en las que las partículas alfa se desviaban ligeramente de su trayectoria principal. A su vez, aparecían ciertas

zonas en las que la desviación era muy elevada dado que se detectaban colisiones esporádicas en los laterales de la pantalla. A continuación, en la figura, se puede contemplar la estructura y el resultado del experimento de la lámina de oro.

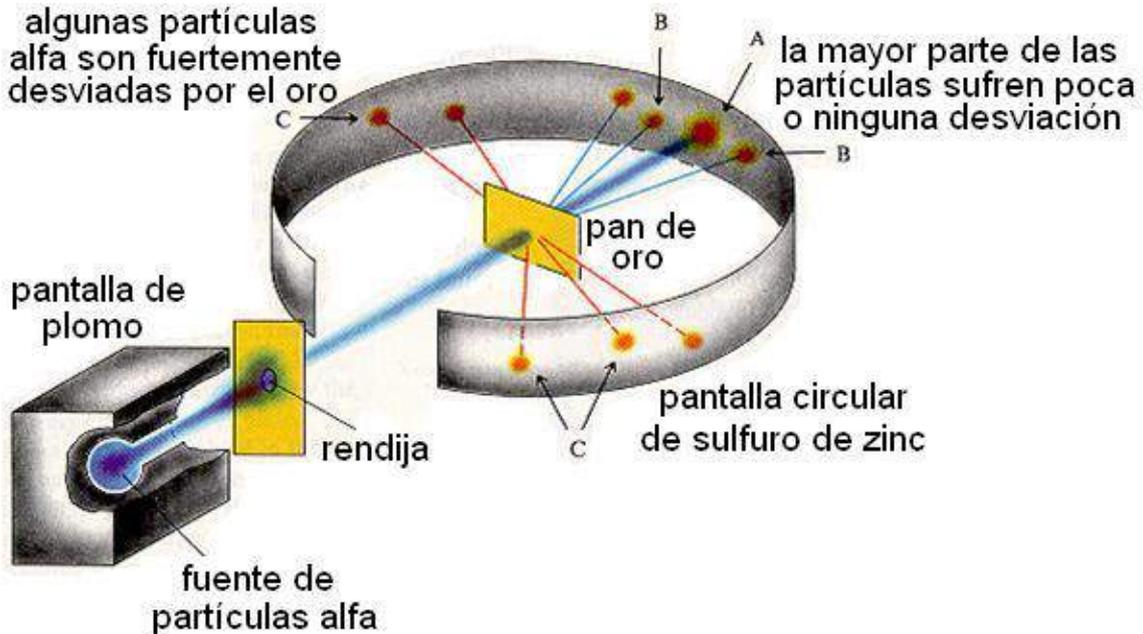


Figura 3. El experimento de la lámina de oro de Rutherford.

Fuente: http://www.gorganica.es/QOT/TO/historia_atomo_exported/img/image11.JPG

Estos resultados supusieron la demostración de características atómicas como que el átomo era en su mayoría espacio vacío, causa por la que gran parte de las partículas alfa traspasaban la lámina de oro sin desviarse, y que, debido a las esporádicas desviaciones ligeras y más fuertes que sufrían dichas partículas, existía un pequeño núcleo positivo en el centro de cada átomo en el cual se concentraba la carga positiva que originaba estas desviaciones y la mayor parte de la masa del átomo. Los electrones, con una contribución másica mínima, eran los que poseían la carga negativa y se encontraban orbitando en el espacio vacío que rodea el núcleo para neutralizar la carga positiva de este.

Sin embargo, las conclusiones derivadas del experimento de Rutherford llevaron a nuevos problemas e inconsistencias que Bohr acabaría resolviendo.

En primer lugar, no se podía explicar cómo se podían mantener las cargas positivas de forma conjunta dentro del núcleo sin que este fuera inestable a causa de las repulsiones eléctricas. Este problema fue resuelto mediante el postulado y el posterior descubrimiento de la fuerza nuclear fuerte, una fuerza fundamental de atracción nuclear que permitió definir el equilibrio dentro del núcleo atómico. Por ello, las fuerzas eléctricas repulsivas que, a priori, impedían la concepción de la concentración de cargas positivas dentro del núcleo se verían neutralizadas por la fuerza de atracción que se generaba dentro del núcleo, llegando a un equilibrio entre ambas fuerzas que daba lugar al núcleo atómico estable observado en el experimento de Rutherford.

Por otra parte, surgía un problema relativo a la electrodinámica clásica por el cual los electrones que giraban alrededor del núcleo disiparían radiación electromagnética haciéndoles perder energía y provocando, así, la caída de estos hacia el núcleo. Las leyes de Newton y las

ecuaciones de Maxwell dictaban que, en caso de que el modelo atómico de Rutherford fuese correcto, los electrones tardarían unos 10^{-10} segundos en caer hacia el núcleo por su pérdida de energía produciendo la inestabilidad del átomo.

Modelo atómico de Niels Bohr

Basándose en la hipótesis de Planck, Niels Bohr desarrolló en 1913 un modelo atómico derivado de los experimentos y de las conclusiones del modelo atómico de Ernest Rutherford, en 1911, pero que resolvía los conflictos con las leyes de Maxwell y de Newton que tenía este modelo [13, 14]. Para ello, también utilizó algunas consideraciones basadas en las investigaciones de Einstein sobre el efecto fotoeléctrico.

La resolución de Bohr ante la inestabilidad del modelo de Rutherford fue que los electrones giraban alrededor del núcleo en unas órbitas determinadas con unas energías concretas proporcionales a la constante de Planck. Dichas órbitas fueron denominadas como niveles de energía y definían la energía de un electrón como una cantidad relativa al nivel de energía en el que se encuentra en lugar de ser continua. Aquí aparece el número cuántico principal n , que hace referencia al nivel de energía en el que se encuentra un electrón dentro de un átomo. Los niveles de energía, al estar cuantizados, se corresponden con números enteros entre 1 y 8.

De esta forma, los electrones, para absorber o emitir energía, se pueden mover entre niveles de energía, necesitando energía para moverse a una capa superior y emitiéndola cuando cae a una capa inferior.

Cuando el electrón permanece en un mismo nivel de energía, posee la energía relativa a este y no se ve modificada. Sin embargo, si un electrón pasa de un nivel de energía a otro inferior, la emisión de energía se produce en forma de fotón tal y como se puede observar en la figura siguiente.

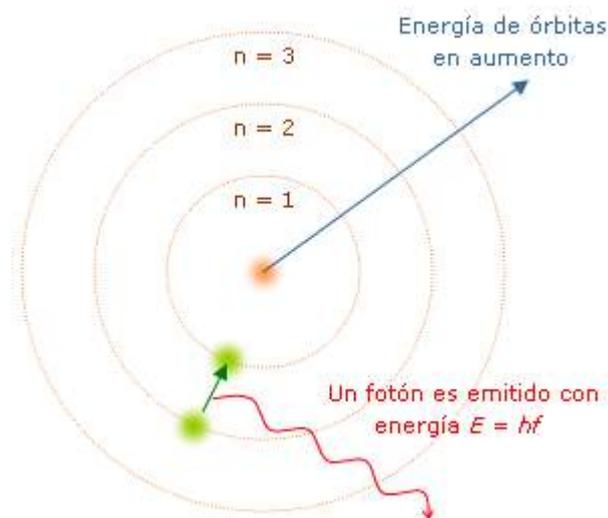


Figura 4. Modelo atómico de Bohr con los tres primeros niveles de energía.
FUENTE: https://commons.wikimedia.org/wiki/File:Modelo_de_Bohr.png

Aquí es donde Bohr incorporó la teoría de Einstein sobre el efecto fotoeléctrico, pues indicaba que la emisión energética del electrón al pasar a un nivel de energía menor era equivalente a

la energía del fotón emitido, hallada como el producto de la constante de Planck por la frecuencia de dicho fotón.

Números cuánticos

Basados en el modelo atómico de Bohr, se empezaron a dar valores discretos a las características físicas u observables que tenían las partículas de magnitud atómica o subatómica, donde dichos valores son los números cuánticos [15].

Para definir la mecánica cuántica, es imprescindible hablar de los números cuánticos ya que son las descripciones dinámicas que se conservan en los sistemas cuánticos y por los que estos se caracterizan al hablar de estados cuánticos de una partícula.

Según se lee en [15], un número cuántico es un autovalor de cada observable del conjunto o sistema. Concretamente, los números cuánticos caracterizan los estados estacionarios de un electrón en un átomo de hidrogenoide. Existen cuatro números cuánticos para describir cualquier sistema de partículas:

- El número cuántico principal n que indica el nivel de energía en el que se sitúa el electrón respecto del núcleo, es decir, la distancia media entre el electrón y el núcleo.
- El número cuántico secundario l que indica el subnivel de energía u orbital del electrón (*s, p, d ó f*), lo que lo distingue de los demás electrones en el mismo nivel de energía n .
- El número cuántico magnético m que hace referencia a la orientación magnética del orbital o subnivel de energía de un electrón.
- El número cuántico de spin s que describe el momento angular del electrón, es decir, el sentido de rotación del electrón sobre sí mismo.

Los números cuánticos son las soluciones estacionarias a la ecuación de Schrödinger dado el principio de exclusión de Pauli, lo cual se introduce en los siguientes apartados.

Hipótesis de De Broglie

Posteriormente, en 1924, se presenta la teoría de ondas de materia de Louis-Victor de Broglie en la que se indaga en la doble naturaleza, ya no solo de la luz, sino de la materia en general [16, 17].

De Broglie postulaba con su teoría una relación entre partícula y onda, y al revés, lo cual quería decir que cualquier partícula tenía una onda asociada y cualquier onda también se podía describir como partícula. El desarrollo de su teoría se basó mayormente en las investigaciones que habían realizado Max Planck y Albert Einstein sobre la física cuántica. Esto le llevó a formular la hipótesis de De Broglie, que aunaba la ecuación de cuantización de la energía de Planck y la ecuación de la relatividad especial de Einstein.

De esta forma, De Broglie dio la equivalencia entre una partícula y su onda de materia asociada basándose en su velocidad y su masa. Concluyó que no solo la luz se comporta de forma cuántica, sino que el resto del mundo también era cuántico, idea que apoyó Einstein pero que puso a muchos otros científicos en su contra.

De Broglie experimentó esta nueva concepción mediante la difracción de electrones replicando el experimento de Young y utilizando en sus cálculos la ecuación, mencionada anteriormente, que define la longitud de onda asociada a una partícula como la constante de Planck dividida por el momento lineal de dicha partícula (producto de su masa y su velocidad). Dicha fórmula es válida cuando la velocidad de la partícula es mucho más pequeña que la velocidad de la luz y funciona para cualquier cuerpo, aunque, para masas muy grandes, la

longitud de onda asociada es prácticamente nula y no se pueden detectar las propiedades ondulatorias.

En 1929, De Broglie recibió el Premio Nobel de Física por la hipótesis expuesta.

Ecuación de Schrödinger

La ecuación de Schrödinger fue de gran importancia en el desarrollo de la mecánica cuántica y fue descrita por Erwin Schrödinger en 1925. [18, 19]

Esta ecuación describe la evolución temporal de la energía de un sistema cuántico con características ondulatorias y no relativistas. Es una analogía de la segunda ley de Newton en la mecánica clásica.

Existen dos definiciones de esta ecuación cuando se habla de dependencia temporal: una general, válida para cualquier sistema que evoluciona en el tiempo y otra no relativista para partículas simples en un campo eléctrico:

Ecuación de Schrödinger dependiente del tiempo (general)

$$i\hbar \frac{\partial}{\partial t} \Psi(\mathbf{r}, t) = \hat{H} \Psi(\mathbf{r}, t)$$

Ecuación de Schrödinger dependiente del tiempo (partícula simple no relativista)

$$i\hbar \frac{\partial}{\partial t} \Psi(\mathbf{r}, t) = \left[\frac{-\hbar^2}{2\mu} \nabla^2 + V(\mathbf{r}, t) \right] \Psi(\mathbf{r}, t)$$

Figura 5. Ecuación de Schrödinger, general y para partículas simples no relativistas.
FUENTE: https://es.wikipedia.org/wiki/Ecuaci%C3%B3n_de_Schr%C3%B6dinger

Para aplicar la ecuación de Schrödinger, es necesario el operador hamiltoniano, expuesto en una sección previa, utilizando las energías cinéticas y potenciales de las partículas que forman el sistema a describir. Al introducir el hamiltoniano en la ecuación, esta se resuelve para la función de onda y con ello se obtienen datos acerca del sistema.

Principio de exclusión de Pauli

A su vez, en el año 1925, Wolfgang Pauli trató de explicar la distribución de los electrones en la estructura atómica razonando, así, la organización de la tabla periódica [20, 21].

Para ello, enunció que dos fermiones no podían tener el mismo estado cuántico dentro de un mismo sistema, es decir, no pueden tener sus cuatro números cuánticos idénticos. A esto se le llamó el principio de exclusión de Pauli. Los fermiones, como el electrón, son partículas con espín semientero por lo que, dentro de un átomo, no pueden existir dos electrones caracterizados por los mismos números cuánticos.

Este principio se puede demostrar utilizando dos funciones de onda que referencian dos electrones en un átomo. Si el estado cuántico de ambos electrones es el mismo, la función de

onda resultante es nula y por ello no existiría. Sin embargo, la energía de la función de onda resultante en un mismo sistema no se anulaba, con lo que se confirmó el principio de exclusión de Pauli.

Principio de incertidumbre de Heisenberg

El principio de incertidumbre de Heisenberg fue enunciado por Werner Heisenberg en 1927 y establece que existe una indeterminación en las magnitudes físicas de una partícula cuántica como son la posición y el momento lineal o velocidad [22].

Este principio introduce lo que se llama el colapso cuántico ya que, si se intenta conocer una de las dos magnitudes, al estar estas en estado de superposición, se colapsa la función de onda que representa la partícula para observar una de las dos propiedades: posición o velocidad. Por ende, al medir una de las dos, se pierde la información relativa a la magnitud no observada y no es posible determinar ambas propiedades físicas al mismo tiempo.

El principio de indeterminación de Heisenberg se expresa de manera matemática con la siguiente fórmula:

$$\Delta x \cdot \Delta p \geq \frac{\hbar}{2}$$

Figura 6. Principio de incertidumbre de Heisenberg.

FUENTE: https://es.wikipedia.org/wiki/Relaci%C3%B3n_de_indeterminaci%C3%B3n_de_Heisenberg

Donde Δx e Δp hacen referencia a la variación que sufre la posición y la velocidad o momento lineal, respectivamente, de la partícula, y \hbar indica la constante de Planck dividida entre 2π .

Ecuación de Dirac

La ecuación de Dirac es de suma relevancia en mecánica cuántica puesto que gracias a ella se describe el suceso del entrelazamiento cuántico. Fue formulada por Paul Dirac en 1928.

Como se dice en [23], la ecuación determina que, sin importar la distancia que separa dos partículas entrelazadas, la conexión cuántica que poseen es instantánea.

Dicha ecuación es una versión relativista de la ecuación de onda de la mecánica cuántica y describe los fermiones (partículas con espín semientero) de acuerdo a la relatividad especial y a los principios de la mecánica cuántica.

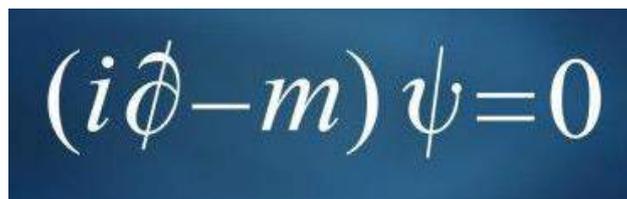
The image shows the Dirac equation, $(i\partial - m)\psi = 0$, written in white text on a dark blue rectangular background. The symbols are in a serif font, with the Greek letter psi (ψ) and the Greek letter delta (∂) being prominent.

Figura 7. Ecuación de Dirac referente al entrelazamiento cuántico.

FUENTE:

https://ichef.bbci.co.uk/news/ws/624/amz/worldservice/live/assets/images/2016/01/21/160121160247_ecuacion_dirac_624x351_bbc_nocredit.jpg

De forma adicional, esta ecuación predice la existencia de antimateria, tal y como se explica en la fuente anterior.

Concepto de ordenador cuántico: Algoritmos de Shor y de Grover

Después de varias décadas, físicos teóricos como Richard Feynmann, Paul Benioff y Charles Bennett propusieron el concepto de computador cuántico entre los 70 y los 80, con críticas por parte de otros científicos.

En 1994, con Peter Shor, llegó el algoritmo cuántico más conocido hoy en día, el algoritmo de Shor para la factorización de números grandes con una ventaja exponencial frente a los algoritmos de factorización clásicos, el cual es capaz de romper los algoritmos de criptografía de clave pública actuales (*RSA*) en un tiempo polinómico. Este algoritmo fue comprobado por *IBM* en 2001 [24], donde consiguieron descomponer el número 15 en los factores 3 y 5 mediante una computadora de 7 qubits, siendo esto un logro simbólico que demostraba que el paradigma cuántico funcionaba como se esperaba.

Después, a finales del siglo XX, Lov Kumar Grover [25] ideó el llamado algoritmo de Grover para la búsqueda en espacios de datos que supone una aceleración cuadrática respecto del algoritmo clásico más eficiente en tiempo de cómputo utilizando un orden logarítmico de almacenamiento, dado que un registro cuántico es capaz de almacenar y tratar un espacio de estados exponencial respecto del número de qubits que lo forma. Dicho algoritmo fue implementado en una máquina real escalable en 2017.

Principalmente estos dos algoritmos fueron los impulsores para la investigación e inversión en computación cuántica dado el potencial que suponen.

3. Introducción a la computación cuántica

Toda esta sección de introducción a la computación cuántica se basa en los contenidos del libro referenciado por [26].

La computación cuántica es un tipo de computación alternativa que se aprovecha de las propiedades de la mecánica cuántica y ofrece características físicas potentes respecto a la computación clásica.

Estas propiedades de la física cuántica permiten obtener muchas ventajas que no se disponen en informática clásica, como puede ser el paralelismo de combinaciones y de aplicación de operaciones. En términos formales, permiten reducir la complejidad computacional y en memoria de muchos algoritmos que pueden llegar a ser irresolubles en tiempo moderado cuando se trata de computación clásica y que, generalmente, se debe al elevado número de variables del problema.

Dichas ventajas se derivan de la representación del estado cuántico, siempre asociado a una función de onda, y del uso de las propiedades de la mecánica cuántica para conseguir una aceleración en la computación.

El espacio de Hilbert fue una generalización matemática importante para describir formalmente la mecánica cuántica. La función de los espacios de Hilbert es expandir las operaciones y tratamientos algebraicos a espacios de dimensión infinita.

Dado que la computación cuántica trabaja con espacios grandes de combinaciones, esta extensión del espacio euclídeo permitió que se expandieran las siguientes operaciones a espacios de dimensión tan grande como se desee: suma directa, producto tensorial, complementos y proyecciones ortogonales.

Análogamente, permitió definir operadores o transformaciones sobre dicho espacio de grandes dimensiones. Esta es la base de la computación cuántica puesto que es la forma en la que se construyen algoritmos cuánticos.

Es importante destacar algunos tipos de operadores como los operadores unitarios, que son invertibles o, en otras palabras, si se tiene el resultado de la transformación de un operador y se aplica un determinado operador (el inverso al original) se obtiene la entrada correspondiente a dicho resultado. También cabe destacar los operadores auto adjuntos que, sencillamente, son su propio operador inverso, es decir, si a un sistema se le aplica dos veces seguidas una transformación auto adjunta, la segunda aplicación anula la primera y el estado del sistema no se vería perturbado.

Los operadores unitarios y auto adjuntos se estudian en una rama de las matemáticas llamada lógica cuántica o *quantum logic*.

Estos operadores son en computación cuántica el equivalente a las puertas lógicas booleanas en computación clásica.

Ahora, se explican detalladamente las propiedades de la mecánica cuántica y sus posibles aplicaciones en la informática o computación.

La superposición de estados en una partícula o sistema de partículas es una propiedad crucial ya que permite la coexistencia de todos los posibles estados en los que puede estar dicha partícula o sistema. En informática clásica se sabe que, si se tienen 8 bits, las posibles

combinaciones son 2^8 , lo cual crea la necesidad de tener 2^8 registros de 8 bits para almacenar todas ellas. Sin embargo, gracias a este fenómeno cuántico, con 8 registros cuánticos de 1 bit y poniéndolos todos ellos en superposición, se almacenan las 2^8 combinaciones distintas únicamente en 8 registros cuánticos de 1 bit. Además, esto también permite analizar o evaluar todas esas combinaciones al mismo tiempo, ya que se utiliza la información cuántica contenida en esos 8 registros, la cual se corresponde con todas las combinaciones binarias posibles.

En la figura a continuación, se muestra un ejemplo de la superposición de dos estados clásicos, arriba y abajo, que pueden representar los valores binarios 0 y 1.

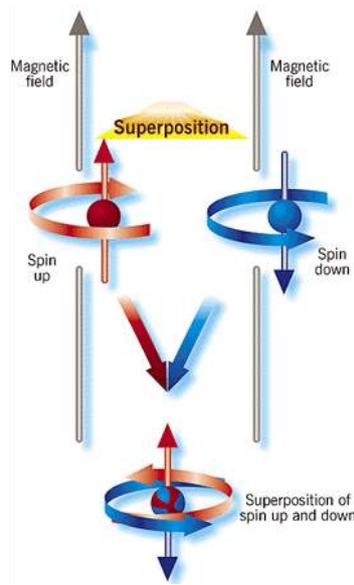


Figura 8. Superposición de dos estados clásicos contenida en un estado cuántico de una partícula.

FUENTE: https://nuevatecnologiasymateriales.com/wp-content/uploads/2019/01/la-superposicio%CC%81n-de-los-estados-cua%CC%81nticos_figura-8-1.gif

La superposición de estados tiene asociada un coeficiente de amplitud que se traduce en probabilidad para cada uno de los posibles estados clásicos. Cuando se habla de superposición uniforme de estados, realmente se hace referencia a que todos los estados poseen la misma probabilidad de colapso cuando se realiza una medición del estado cuántico.

Es importante recalcar la diferencia entre un estado clásico y un estado cuántico. El estado clásico es básicamente un resultado explícito, por ejemplo, un valor numérico binario, mientras que, el estado cuántico puede albergar varios estados clásicos con amplitudes de probabilidad determinadas.

Sin embargo, el estado cuántico no es compatible con el mundo clásico, el mundo macroscópico, por ello, cuando se realiza una medición de dicho estado cuántico, el resultado que se obtiene es un único resultado clásico de todos los posibles estados, concordando con la amplitud de probabilidad de cada uno de ellos.

Con esto, se llega a otra propiedad única de la mecánica cuántica, la observación o colapso del estado cuántico. Una de las cuestiones más filosóficas de la mecánica cuántica es que la medición o la observación del estado cuántico modifica dicho estado proyectando uno de los

posibles resultados clásicos acorde a la distribución de probabilidades que tiene cada uno de estos estados clásicos.

Aquí entra el concepto de aleatoriedad cuántica o *quantum randomness*. En informática clásica, la aleatoriedad no existe y, de hecho, la aleatoriedad que se utiliza en los programas se denomina pseudoaleatoriedad dado que está fundamentada en la frecuencia de reloj y otros eventos que suceden a nivel bajo en el computador. Si dichos eventos fueran conocidos, se podría determinar el resultado pseudoaleatorio que se obtendría. No obstante, en computación cuántica, la aleatoriedad es una propiedad física de la función de onda sin la cual no se podría describir un sistema cuántico y se trata de una aleatoriedad absoluta basada en las amplitudes de probabilidad de cada uno de los autoestados que conforman el estado cuántico, a lo que se llama *quantum randomness*.

Cuando se realizan algoritmos cuánticos, se han de manejar bien las herramientas, métodos y propiedades tal que se pueda obtener un estado o resultado cuántico del que se pueda extraer información de interés teniendo en cuenta que, al realizar una medición, además de obtener solo uno de todos los posibles resultados, también se destruye el estado cuántico observado.

Para retratar gráficamente la explicación del colapso cuántico, en la figura siguiente se muestran tres registros cuánticos en una superposición binaria determinada y lo que produciría la medida de dichos registros con sus respectivas probabilidades.

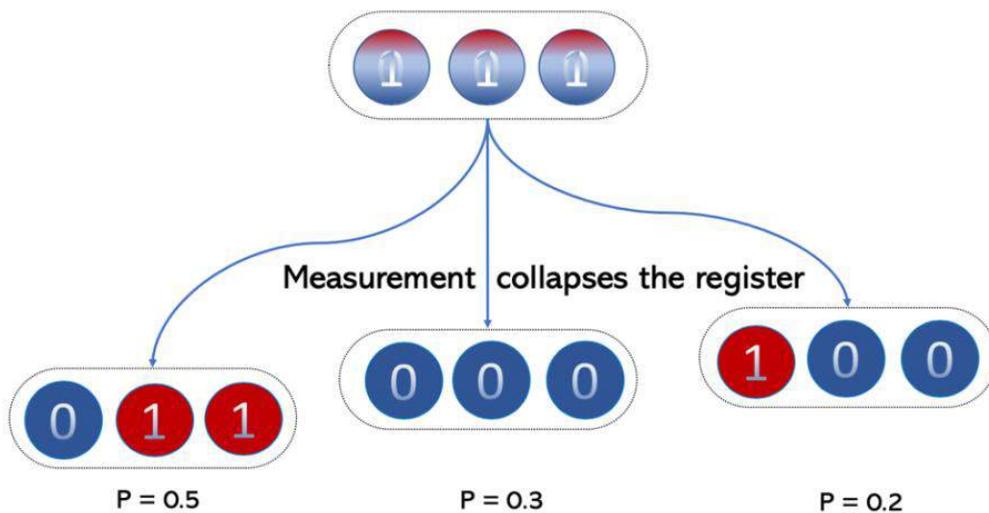


Figura 9. Colapso probabilístico del estado cuántico en un estado clásico.
FUENTE: <https://infofarm.be/wp-content/uploads/2020/02/measurement.png>

El colapso del estado cuántico, al ser probabilístico, implica tener en cuenta las amplitudes de probabilidad de los posibles estados clásicos codificados dentro del estado cuántico. Por ello, se hace uso de otro principio importante de la mecánica cuántica: la interferencia.

La interferencia hace alusión a la interferencia en las amplitudes de la función de onda y esta puede ser interferencia destructiva o constructiva, es decir, la densidad de probabilidad se puede distribuir de tal forma que en algunos puntos se acumule más amplitud y, en otros, dicha amplitud se reduzca o se anule.

Aquí, entran en juego las amplitudes de probabilidad de cada estado clásico que contiene el estado cuántico dado que, según los objetivos del algoritmo, se puede utilizar el fenómeno de la interferencia para destruir determinadas amplitudes de probabilidad acorde a una regla que se podría denominar como regla de filtrado. Esto causa que los estados clásicos que no cumplen la regla eliminarán o reducirán la probabilidad que tienen de aparecer, por lo que su amplitud desaparecerá dentro del estado cuántico y, cuando se realice la medición de este, quedarán únicamente los resultados que sí cumplen la regla.

La interferencia se puede observar físicamente en la figura 2, con las fases del experimento de Young. En concreto, la interferencia ocurre cuando no se utiliza detector/medidor. Otro ejemplo físico de la interferencia se muestra en la siguiente figura, siendo el gráfico rojo la amplitud de probabilidad de cada estado, codificados todos ellos en la función de onda del estado cuántico.

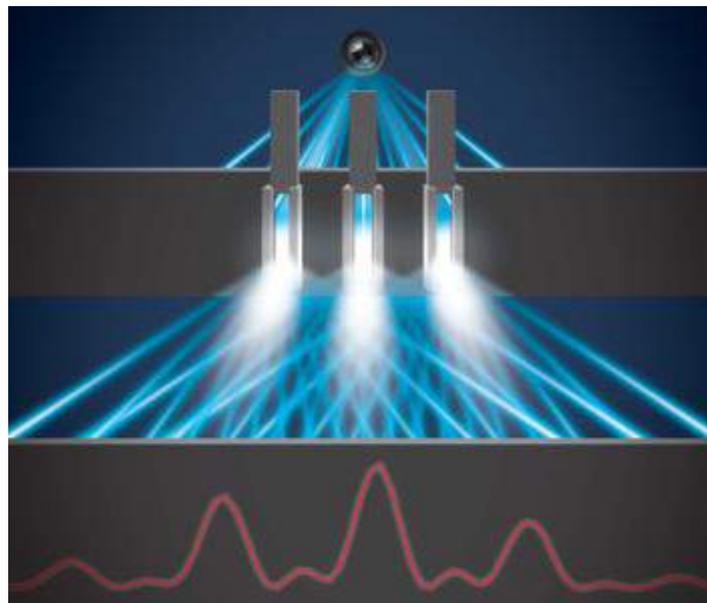


Figura 10. Ejemplo físico de la interferencia cuántica.

FUENTE: https://universodoppler.files.wordpress.com/2010/07/not_a_triple_threat.jpg

La interferencia en la figura anterior es la causa de que ciertos estados aumenten su probabilidad a costa de reducir o anular la probabilidad de otros estados, dando lugar a un nuevo estado cuántico en el que se han filtrado las posibles combinaciones con una función o regla mediante un procesamiento adecuado de la información cuántica.

El entrelazamiento es otro fenómeno cuántico que juega un papel muy importante en el progreso de la computación cuántica. Tal cual dice su nombre, el entrelazamiento cuántico crea una ligadura entre los estados cuánticos de dos o más partículas. Esto quiere decir que el estado cuántico de una partícula es capaz de condicionar el estado de otras.

Cuando una partícula se entrelaza con otra, el sistema que forman no se puede representar por el estado de ambas partículas de forma independiente. El resultado de su entrelazamiento es un sistema cuyo estado es indivisible, a pesar de que dicho sistema esté formado por los estados de dos partículas. El estado cuántico del sistema está condicionado por ambas partículas y, por ello, lo que ocurra en una tendrá lugar en la otra de forma instantánea.

Gracias a este principio de la mecánica cuántica, es posible asociar la información cuántica de un sistema a otro entrelazando ambos sistemas de una manera determinada. Esto permite

ampliar las formas de aprovechar la información codificada en los estados cuánticos de los que, como se ha dicho previamente, no es posible extraer todos los resultados.

La importancia del entrelazamiento cuántico es que se elimina de cierta manera el factor probabilístico independiente en dos o más partículas entrelazadas y crea una coherencia entre el estado de ambas. Para obtener una ventaja cuántica, es necesario el entrelazamiento masivo entre qubits, es decir, se ha de crear una coherencia entre el estado de varios qubits para conseguir que los algoritmos cuánticos sean de utilidad.

La forma de realizar un entrelazamiento cuántico es la que se muestra en la figura a continuación. Dicha figura denota que, una partícula en superposición de estados (0 y 1 al mismo tiempo) controla la inversión del estado de la segunda partícula, que se encuentra en el estado 0 inicialmente. El control de esta inversión funciona de la siguiente manera: si el estado de control es 0, la partícula objetivo no invierte su estado (permanece en 0) y, si el estado de control es 1, dicha partícula sí invierte su estado (cambia a 1). Dado que la partícula de control está en ambos estados a la vez, de la misma forma, la inversión en la partícula objetivo se aplica y no se aplica. De este modo, si la partícula controladora colapsa en el estado 0, la partícula objetivo hará exactamente lo mismo puesto que el estado de control es 0 y su estado no se ha invertido o, en caso contrario, si la partícula de control colapsa en estado 1, la inversión de la partícula objetivo se produce y su estado colapsará también en el estado 1.

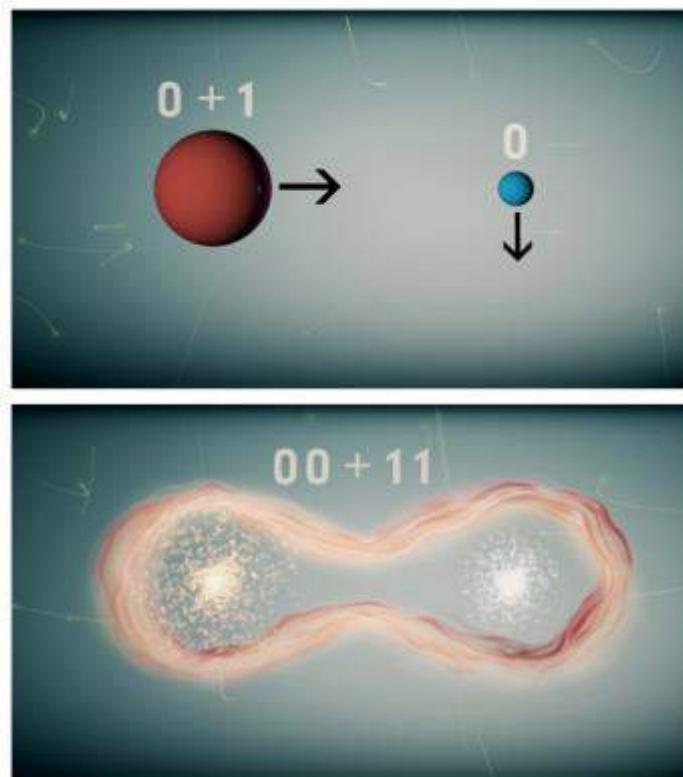


Figura 11. Entrelazamiento del estado cuántico de dos partículas.

FUENTE: <https://d.ibtimes.co.uk/en/full/1470388/difference-between-regular-bits-quantum-bits.png?w=350&f=7ed50ebb8b45a094157f591a0e49d3d4>

Una última propiedad peculiar de la mecánica cuántica es el efecto túnel, el cual permite a las partículas atravesar potenciales de energía sin necesidad de que dicha partícula posea la energía suficiente para superarlos. Dicho de otro modo, es como si una partícula se encontrara ante una barrera de energía que nunca podría saltar, pero dicha partícula espontáneamente

consigue encontrar un túnel que conecta ambos lados de la barrera y permite que esta atraviese sin tener que saltarla. Como información adicional, este fenómeno es por el que se da la reacción de fusión en las estrellas.

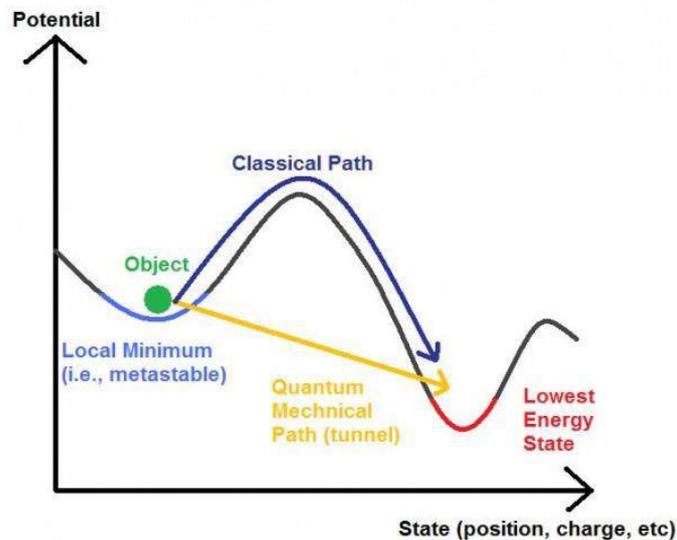


Figura 12. Efecto túnel cuántico.

FUENTE: https://www.physicsoverflow.org/?qa=blog&qa_blobid=16049880102516868939

Este efecto es probabilístico y, por ello, también ayuda a la computación cuántica mediante la búsqueda de máximos y mínimos energéticos. Generalmente, se utiliza en los ordenadores cuánticos adiabáticos para optimizar funciones de múltiples variables.

En la siguiente sección, se distinguirán los tipos de computadores cuánticos existentes donde cada tipo utiliza distintas propiedades de la mecánica cuántica para computar.

Tipos de computadores cuánticos

Cuando se trata de computación cuántica, existen varios tipos de computadores cuánticos a los que se puede acudir dependiendo del problema en cuestión.

En la actualidad, hay tres tipos de computación cuántica: la computación cuántica adiabática, la simulación cuántica y la computación cuántica universal.

En la figura que se muestra a continuación se puede ver una gráfica que relaciona la complejidad de construcción del computador con las distintas capacidades de cómputo. La computación adiabática es la más sencilla de implementar, pero se limita a un tipo específico de problemas. El simulador cuántico contiene dicho tipo específico de problemas y añade algunas características adicionales de computación. Por último, el computador cuántico universal o de puertas permite codificar cualquier tipo de problema, incluyendo las aplicaciones que se pueden realizar en los dos primeros computadores.

Three types of quantum computing

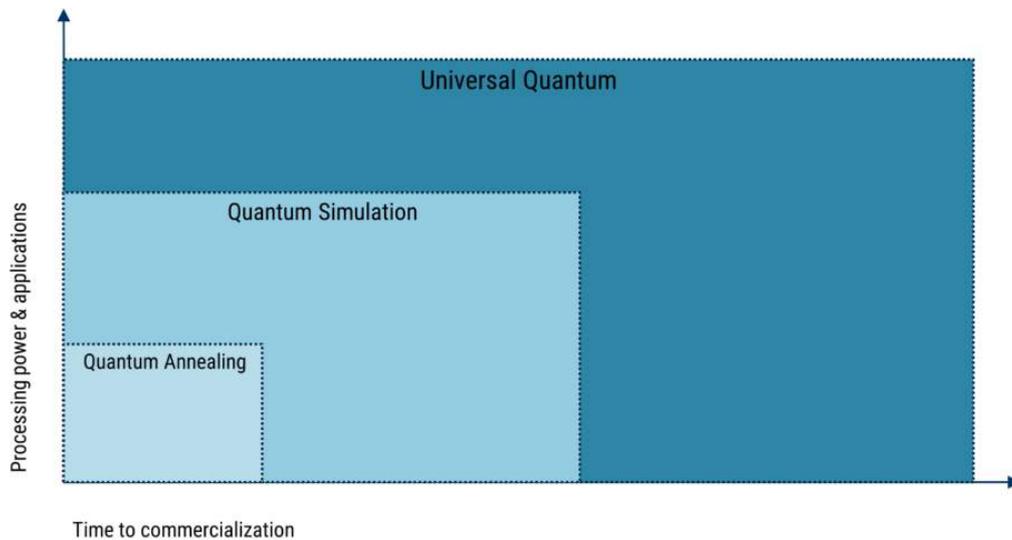


Figura 13. Tipos de computación cuántica según su construcción y sus aplicaciones.

FUENTE: <https://s3.amazonaws.com/cbi-research-portal-uploads/2018/12/06125748/QC-types-chart-1.png>

Como se ha mencionado anteriormente, la computación cuántica adiabática es una forma de computación que está restringida a problemas de optimización. Sin embargo, hoy en día es la más utilizada de las tres a nivel práctico dado que sus limitaciones no son tan exigentes ni impiden, en gran medida, la obtención de resultados válidos.

El proceso que sigue la computación cuántica adiabática es el siguiente. Se inicializa el sistema con una superposición uniforme de todos los estados y, después, el sistema evoluciona acorde a la ecuación de Schrödinger dependiente del tiempo. El sistema cambia las amplitudes de todos los estados en paralelo siguiendo un campo transversal dependiente del tiempo, produciendo el efecto túnel entre estados. Finalmente, el campo transversal se apaga y se espera que el sistema haya alcanzado el estado de mínima energía, lo cual quiere decir que se ha minimizado la función codificada en forma de energías, llamada el Hamiltoniano.

Tal y como dice el artículo [27], muchas empresas han utilizado este tipo de computación para problemas reales.

Como ejemplo, Volkswagen realizó un experimento para reducir el tráfico en la capital de China, Beijing. El algoritmo calculaba con éxito la mejor ruta para cada vehículo y fue ejecutado en las máquinas adiabáticas de *Google* y *D-Wave*.

Otras aplicaciones reales que tiene la computación adiabática están relacionadas con el campo de la economía y de la banca que se basan en optimización de funciones con un gran número de variables.

Este tipo de problemas son ideales para los computadores cuánticos adiabáticos, puesto que optimizan una función de múltiples variables y mínimos locales reduciendo exponencialmente el tiempo de computación respecto de un algoritmo clásico.

En cuanto a la simulación cuántica, se encuentra en un puesto superior a la computación adiabática pues no solo es posible la optimización de funciones, sino que también se puede utilizar en la simulación de procesos cuánticos, físicos, químicos e incluso biológicos.

Dado que estos procesos albergan una gran cantidad de posibles situaciones, la simulación cuántica de estas situaciones permite obtener información sobre dichos procesos y ver cómo afectan los distintos cambios que se puedan producir.

El ámbito más destacado en simulación cuántica es el de la química cuántica. Según se explica en [27], los simuladores cuánticos se podrían utilizar para la simulación del despliegue de proteínas, el mayor problema existente en el campo de la bioquímica.

Finalmente, se encuentra el computador cuántico universal que, tal y como sugiere su nombre, es un computador cuántico que recoge las funciones de todos los computadores anteriores además de ofrecer cualquier otro tipo de computación posible. No está limitado a ningún problema concreto, por lo que permitiría programar cualquier tipo de algoritmo imaginable haciendo uso de las ventajas de la mecánica cuántica.

El computador cuántico universal funciona con puertas lógicas cuánticas que trabajan de una forma similar a las puertas lógicas convencionales, de informática clásica, pero con ciertos matices añadidos.

Las puertas lógicas cuánticas implementan las propiedades de la mecánica cuántica discutidas en la sección previa: superposición, entrelazamiento, interferencia y el colapso o medición del estado cuántico. Al igual que en informática clásica, los algoritmos en computadores cuánticos de puertas se construyen mediante circuitos cuánticos lo cual se asemeja a la construcción de circuitos electrónicos.

En este trabajo, predomina el uso del computador cuántico universal que funciona por circuitos cuánticos construidos con puertas lógicas cuánticas. El algoritmo de *Quantum Counting* que se expone en secciones posteriores se implementa mediante este paradigma de programación cuántica.

Qubits

Con ejemplos anteriores, ya se han expuesto pequeñas nociones de cómo funciona la información cuántica y de su gran potencial. La representación de la información cuántica sigue las mismas líneas que la información en la informática clásica, con código binario, aunque con algunas características adicionales que la hace más versátil, rápida y eficiente.

En informática clásica, la unidad de información mínima es el bit y denota un 0 o un 1 que, a nivel de interpretación abstracta, puede representar un “sí” o un “no”. Según se van formando conjuntos de bits, la información que se puede representar es más compleja. Con 8 bits, es decir, 1 byte, y utilizando la tabla ASCII, se pueden representar caracteres como las letras del abecedario, los números del 0 al 9, símbolos, etcétera.

Asimismo, la computación cuántica posee la misma representación binaria, donde la unidad básica y mínima de información es el cúbit o qubit.

El qubit puede tomar los valores de 0 y 1, al igual que un bit corriente. Sin embargo, el qubit, a diferencia del bit clásico, también puede tomar ambos valores tal y como se explica al principio de la sección de introducción cuando se habla de la propiedad de la superposición de la mecánica cuántica. Cuando el qubit toma ambos valores con sus respectivas amplitudes o probabilidades, se dice que está en estado de superposición, la cual puede ser una superposición uniforme si la amplitud de 0 es la misma que la de 1 o una superposición cualquiera si dichas amplitudes son distintas entre ellas (por ejemplo, 0.7 probabilidad de 0 y 0.3 probabilidad de 1).

Los estados o valores de un qubit o conjunto de qubits también son llamados autoestados o autovectores y se corresponden con los distintos valores clásicos que son capaces de albergar en el estado cuántico. En el ejemplo anterior, con un qubit, los autovectores de este son 0 y 1.

En la siguiente imagen, se muestran gráficamente las diferencias entre un bit clásico y un bit cuántico o qubit. Simplemente, el bit clásico toma el valor 0 o, en su defecto, el valor 1, mientras que, el qubit, puede tomar el valor 0, el valor 1 o una superposición de ambos valores. Además, en este caso la superposición es uniforme puesto que ambos valores tienen la misma probabilidad.

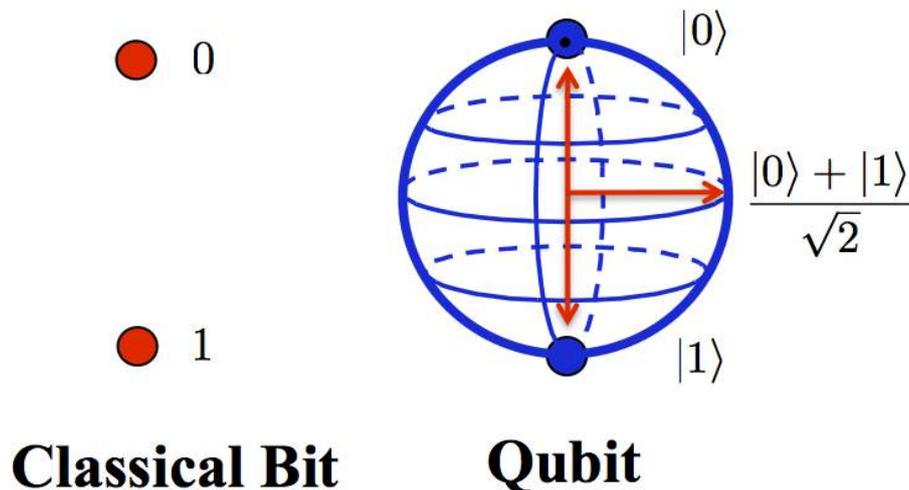


Figura 14. Comparativa entre un bit y un qubit.
 FUENTE: https://www.researchgate.net/figure/Figure-1-Classical-Bit-Vs-Qubit_fig2_308414229

Para la representación de los estados de un qubit o un sistema de qubits, se utiliza la notación de Dirac y se detalla su explicación en la subsección siguiente, pero, a nivel matemático, los qubits pertenecen a un espacio de Hilbert, introducido en el apartado de la historia de la mecánica cuántica.

Quando se habla de amplitudes de probabilidad de 0 y 1 en la superposición de un qubit, esto hace referencia a la probabilidad que existe de que colapse cada uno de estos dos estados. Como se ha dicho anteriormente, el colapso cuántico se da cuando el observador hace una medición del estado cuántico. A pesar de que se conozca el estado cuántico o el algoritmo cuántico que se haya implementado es correcto para obtener la solución deseada, en última instancia se ha de medir el resultado de la computación y esto hace que el estado cuántico desaparezca y colapse en un estado clásico.

Por ello, si medimos el qubit de la figura anterior cuando está en superposición uniforme de estados, obtendremos en un 50% de ocasiones el estado 0 y en el otro 50% el estado 1.

No obstante, antes de realizar la medición y colapsar en uno de los estados, el estado cuántico posee tanto el valor de 0 como el valor de 1 y, por ello, se pueden paralelizar las operaciones con dichos valores. Si, por ejemplo, se aplica la suma binaria de un qubit con otro, estando ambos en superposición uniforme, el estado cuántico resultante contendrá todas las combinaciones posibles de la suma: 0+0, 0+1, 1+0 y 1+1. Pero, cuando este estado cuántico es medido, solo se podrá observar uno de estos posibles resultados.

Para aumentar la capacidad de información y poder resolver problemas más complejos, se procede de la misma manera que en computación clásica. Se agrupan conjuntos de qubits para poder representar números más amplios y, por tanto, de más dimensión.

Siguiendo con el ejemplo expuesto al principio de esta subsección, cuando se agrupan 8 bits, se tiene 1 byte de información, con el que se pueden representar hasta 256 valores diferentes, desde el 0 hasta el $2^8 - 1$. De la misma manera, al agrupar 8 qubits se forma un registro cuántico de 1 qbyte pero, en contraste con la información clásica, un registro de 8 qubits puede almacenar los 256 valores mencionados anteriormente poniendo los 8 qubits en una superposición uniforme de 0 y 1, mientras que, un registro de 8 bits clásico puede tomar solo 1 de los 256 valores posibles.

Esto implica una capacidad de almacenamiento que aumenta exponencialmente con el número de qubits de los que se disponen, es decir, con un registro de n qubits se puede almacenar información relativa a 2^n combinaciones. Además, cada uno de los n qubits es independiente del resto en cuanto a que las operaciones o transformaciones que se les aplica son en paralelo para los n qubits.

A pesar de ello, como se ha dicho con antelación, esta información se encuentra encapsulada en el estado cuántico, el cual no es accesible por métodos tan simples como la medición del estado de los qubits dado el principio del colapso cuántico. Esto es, de las 2^n combinaciones mencionadas en el párrafo anterior, si se realiza una medición de esos n qubits, se obtendrá una única combinación de las 2^n y, por ende, se necesitan técnicas más sofisticadas para aprovechar las ventajas de la computación cuántica.

El algoritmo de *Quantum Counting* explicado en secciones posteriores retrata un ejemplo de cómo se pueden utilizar estas ventajas para conseguir información útil del estado cuántico habiendo implementado el algoritmo que utiliza el paralelismo combinatorio de la computación cuántica.

Adicionalmente, algo que cabe destacar es que un registro cuántico de n qubits se suele utilizar como una variable que representa valores enteros de hasta $2^n - 1$ cuando este registro está en superposición uniforme, es decir, cuando contiene todos los valores en su estado cuántico, desde 0 hasta $2^n - 1$, con amplitudes iguales. Con ello, se aplica una transformación a dicho registro que equivale a hallar los valores de una función con su respectiva variable (representada por el registro cuántico en superposición). Dicha transformación ocasiona que, al tener todos los valores representados en el estado cuántico de la variable de entrada, todos los valores de la función sean calculados aplicando una única vez la transformación equivalente a esta función. Esto es una mejora considerable respecto de la computación clásica, con la que los distintos valores de la función se tendrían que calcular uno a uno, y se denomina como paralelismo cuántico de computación.

Este tipo de transformaciones son los llamados oráculos o cajas negras y se representan por un circuito cuántico en el que se realiza una determinada función la cual, dada una entrada, la transforma de cierto modo y produce la salida correspondiente a la función.

En la figura anterior, donde se comparaba un bit clásico y un bit cuántico, el qubit aparece como una esfera en la cual se mostraba en estado 0 en la parte superior de la esfera y un 1 en la parte inferior. Esta es la representación binaria común que tiene un qubit, sin embargo, cualquier punto en la superficie de dicha esfera se puede representar en el estado cuántico de un qubit. Esta esfera es la denominada esfera de Bloch y se muestra en la figura siguiente.

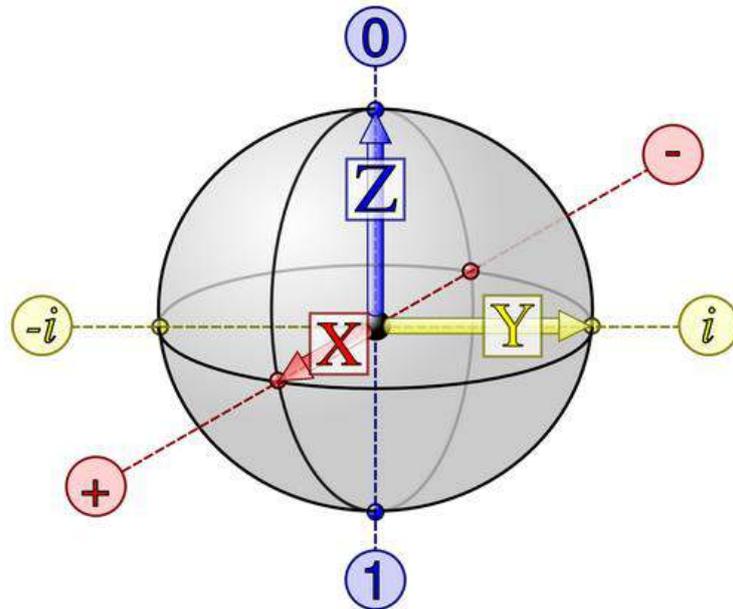


Figura 15. Esfera de Bloch.

FUENTE: <https://francis.naukas.com/files/2014/07/Dibujo20140731-Bloch-sphere-qubit-nature-com.png>

Como se ha dicho, el qubit es una esfera en la que cada punto de su superficie se corresponde con un estado cuántico inequívoco. La esfera de Bloch se representa matemáticamente mediante un espacio complejo de Hilbert de dimensión 2, cuando se trata de un único qubit.

Si se trata de un sistema de N qubits, el estado cuántico que representa es un vector de módulo 1 en el espacio complejo de Hilbert de dimensión 2^N , lo que significa que el sistema es capaz de almacenar información de hasta 2^N estados distintos.

La esfera, al ser un objeto tridimensional, se proyecta en tres ejes, X, Y y Z, tal y como se muestra en la imagen. Esto es de gran relevancia ya que entra en juego otra matriz, la base algebraica de medida. Convencionalmente, se mide en la base Z, pero se puede medir en cualquier otra base de la esfera.

Como ejemplo, si se prepara un qubit en estado de superposición uniforme de 0 y 1, dicho estado se encuentra en el eje X de la esfera de Bloch, es decir, será el estado “+” (también podría ser el estado “-”). Si justo después se realiza una medición en la base Z, como se ha venido diciendo hasta el momento, el colapso del estado cuántico se dará, con un 0.5 de probabilidad respectivamente, el estado 0 o el estado 1.

Sin embargo, dado el mismo supuesto, el qubit está en el estado “+”, cuando se hace una medición de este en la base X, el estado colapsado será el estado “+” con un 100% de certeza.

Si dicho estado se mide respecto al eje Y, sucede exactamente lo mismo que si se mide en la base Z, habrá un 0.5 de probabilidad de obtener i y otro 0.5 de obtener $-i$.

Esta representación tridimensional del estado cuántico es de suma importancia para la obtención de información del mismo ya que los ángulos que posee el estado cuántico respecto de los tres ejes de medida y los parámetros del problema proporcionan información relativa a la respuesta de todas las combinaciones, dando una solución ante el obstáculo del colapso cuántico. Esta técnica es la estimación de fase cuántica y se utiliza en el algoritmo de *Quantum Counting*, se explica detalladamente en su correspondiente sección.

Notación de Dirac

En esta subsección, se introduce la notación algebraica con la que se representa la información cuántica: sus estados con sus respectivas amplitudes de probabilidad. También, se expone la notación matricial y tensorial que no es más que otro tipo de representación más cercana al cálculo.

En la subsección anterior, en la figura que compara los bits clásicos y cuánticos, se puede ver que la notación de 0 y 1 se escribe como $|0\rangle$ y $|1\rangle$ cuando se trata de un estado cuántico. Esta notación es la denominada notación de Dirac o notación bra-ket.

Con ella, se representa el estado cuántico mediante la descripción de su función de onda asociada. Dicha función de onda puede representar cualquier vector unitario que haga referencia a un estado cuántico, pero en computación cuántica, los estados cuánticos que se representan son siempre vectores binarios.

La notación de Dirac tiene propiedades que facilitan los cálculos, operaciones y transformaciones aplicables a una función de onda o estado cuántico.

A continuación, se muestra un ejemplo sencillo de un estado cuántico de un qubit representado con la notación de Dirac el cual hace referencia a la superposición uniforme entre el estado $|0\rangle$ y $|1\rangle$:

$$\frac{|0\rangle + |1\rangle}{\sqrt{2}} = \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle \quad (1)$$

En primera instancia, se separa el coeficiente de amplitud para dar una explicación más clara de su significado referente a cada estado.

Dicho coeficiente representa la amplitud de probabilidad de cada estado cuántico y, a su vez, indica la probabilidad de colapso de cada estado si se eleva al cuadrado el coeficiente asociado al estado. Esta amplitud de probabilidad es un número complejo, aunque, en este ejemplo, la parte imaginaria es nula. Aquí, ambos estados tienen una probabilidad de $\frac{1}{2}$ de aparecer a la hora de realizar una medida de este estado cuántico.

Tal y como se ha expuesto, el vector asociado a la función de onda del estado cuántico siempre es unitario, es decir, su módulo siempre es 1. Esto tiene sentido, pues la suma de las probabilidades de los estados ha de ser siempre 1.

En este caso, el módulo de la función de onda (1) se calcula de la siguiente forma:

$$\sqrt{\left(\frac{1}{\sqrt{2}}\right)^2 + \left(\frac{1}{\sqrt{2}}\right)^2} = 1$$

Análogamente, este estado cuántico de un qubit se puede representar de forma matricial y se corresponde con una matriz columna de dos elementos, donde el primero es la amplitud asociada al estado $|0\rangle$ y el segundo la amplitud del estado $|1\rangle$ de dicho qubit.

Si se quiere representar el estado $|0\rangle$ o $|1\rangle$, estos estados se corresponden con las siguientes matrices columna donde la amplitud de probabilidad es 1 (100% de certeza de obtener el estado) en la posición relativa al estado que se representa:

$$|0\rangle = \begin{pmatrix} P_0 \\ P_1 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix} ; |1\rangle = \begin{pmatrix} P_0 \\ P_1 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

En el ejemplo (1), la matriz que representa su estado cuántico es el mostrado a continuación, siendo P_0 la amplitud del estado $|0\rangle$ y P_1 la amplitud del estado $|1\rangle$ y se relaciona con la función de onda (1) de la siguiente manera:

$$\begin{pmatrix} P_0 \\ P_1 \end{pmatrix} = \begin{pmatrix} 1/\sqrt{2} \\ 1/\sqrt{2} \end{pmatrix} \rightarrow \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle$$

Hasta aquí, se ha expuesto la representación del estado cuántico de un qubit mediante notación de Dirac y notación matricial.

Ahora, se va a representar la función de onda de un sistema compuesto generalizando la notación para n qubits.

Como recordatorio, al igual que un qubit representa 2^1 estados, n qubits llegan a representar 2^n estados. Todos ellos se representan de la misma manera que en el ejemplo (1), sumando todos los estados con sus correspondientes amplitudes.

Para ejemplificar la representación de n qubits, se va a suponer que todos los estados están en superposición uniforme, es decir, cada qubit se encuentra en la superposición descrita por la función de onda (1). Si se quiere representar el estado cuántico del sistema de n qubits, teniendo en cuenta el registro completo en lugar de qubit a qubit, la función de onda queda de la siguiente manera:

$$\frac{|0 \dots 00\rangle + |0 \dots 01\rangle + \dots + |1 \dots 11\rangle}{\sqrt{2^n}} \quad (2)$$

Los distintos estados que componen el estado cuántico son cadenas binarias de longitud n y, al estar todos los qubits en superposición uniforme, el coeficiente referente a la amplitud de probabilidad de cada uno de estos estados también es uniforme, es decir, todos los estados tienen la misma probabilidad de colapsar al realizar la medida del registro de n qubits.

Como se dijo anteriormente, el vector formado por las amplitudes de cada estado ha de ser unitario, por lo que la suma de los cuadrados de todas las amplitudes ha de resultar 1.

En la función de onda (2), el coeficiente de cada uno de los estados es $1/\sqrt{2^n}$ ya que, al existir 2^n posibles estados con la misma probabilidad, el vector ha de ser unitario, por ello se tiene que cada amplitud A es:

$$A^2 * 2^n = 1 \rightarrow A = 1/\sqrt{2^n}$$

Y la probabilidad de obtener cualquiera de los 2^n estados si se mide el registro cuántico (2) se corresponde con el cuadrado de la amplitud: $A^2 = 1/2^n$

Sin embargo, existe una forma simplificada de representar los estados de (2) que, al igual que informática clásica, consiste en representar cada estado como un número entero decimal en lugar de dar la representación binaria. Para el estado cuántico del ejemplo (2), la representación queda de la siguiente forma.

$$\frac{|0\rangle + |1\rangle + |2\rangle + \dots + |2^n - 1\rangle}{\sqrt{2^n}}$$

Siguiendo con la representación decimal descrita, otra forma de expresar la superposición uniforme de n qubits distinta de las vistas anteriormente es mediante un sumatorio. Dado que el estado cuántico de (2) se representa como la suma de todos los posibles estados con su respectiva amplitud de probabilidad, dichos estados y amplitudes se pueden parametrizar como términos de un sumatorio, tal y como se muestra a continuación.

$$\frac{1}{\sqrt{2^n}} \sum_{x=0}^{2^n-1} |x\rangle$$

Esta notación es muy utilizada, pues los algoritmos cuánticos suelen empezar con una inicialización de los registros de qubits estableciéndolos en superposición uniforme de estados.

Para terminar, se representa la función de onda correspondiente a (2) con la notación matricial que, al tratarse de n qubits, es equivalente al producto tensorial de los n qubits, todos en estado de superposición uniforme.

$$\begin{pmatrix} 1/\sqrt{2} \\ 1/\sqrt{2} \end{pmatrix} \otimes \dots \otimes \begin{pmatrix} 1/\sqrt{2} \\ 1/\sqrt{2} \end{pmatrix} = \begin{pmatrix} 1/\sqrt{2^n} \\ \vdots \\ 1/\sqrt{2^n} \end{pmatrix} = \begin{pmatrix} P_0 \\ P_1 \\ \dots \\ P_{2^n-1} \end{pmatrix}$$

Y, al igual que antes, P_x representa la amplitud de probabilidad correspondiente al estado x .

Estas nociones sobre la notación y representación de la información cuántica son suficientes para comprender el desarrollo de las siguientes secciones.

Operadores y puertas lógicas cuánticas

En este apartado, se introducen las puertas lógicas cuánticas y los operadores que provocan cambios en el estado cuántico de uno o varios qubits. Algunas de estas puertas se asemejan a las puertas lógicas clásicas, como la XOR, aunque su funcionamiento es algo distinto en la mayoría de ellas.

A diferencia de las puertas lógicas clásicas, las puertas cuánticas son transformaciones del estado cuántico en un único qubit, es decir, la entrada y salida de la puerta cuántica es el qubit objetivo al que se le aplica dicha puerta. Como ejemplo, la puerta lógica clásica AND recibe dos

o más bits de entrada y produce un único bit de salida. En el caso de las puertas cuánticas, las transformaciones básicas están orientadas a un único qubit, sin embargo, mediante puertas controladas, se construyen transformaciones que permiten realizar operaciones de más de un qubit.

Estas puertas lógicas también forman parte de los operadores, puesto que un operador es cualquier transformación que se pueda dar en un estado cuántico. Dichos operadores son aplicaciones lineales de un espacio de Hilbert sobre sí mismo, es decir, si se aplica un operador sobre un estado cuántico perteneciente al espacio de Hilbert (H), el resultado se corresponde con otro estado cuántico perteneciente al mismo espacio. Por lo que, dado un operador A que actúa sobre un qubit con estado cuántico $|\psi\rangle \in H$ se tiene que:

$$A|\psi\rangle: H \rightarrow H$$

A continuación, se introducen las puertas cuánticas de 1 qubit de entrada. En general, estas se aplican como producto de un operador por el estado cuántico del qubit. La representación de las puertas se define como una aplicación lineal o como una matriz, cuando se trabaja con la notación matricial, expuesta en la subsección previa.

Para comenzar, se tienen las puertas de rotación de 180° respecto a cada eje de la esfera de Bloch: la puerta X cuya función es invertir los estados $|0\rangle$ y $|1\rangle$, también llamada puerta NOT por su capacidad inversora, la puerta Y que invierte la parte imaginaria de las amplitudes de los estados, y la puerta Z que cambia el signo de la amplitud del estado $|1\rangle$. Estas puertas son las llamadas matrices de Pauli y en la figura siguiente se encuentra la representación de la puerta junto con su matriz de transformación.

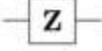
Operator	Gate(s)	Matrix
Pauli-X (X)		$\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$
Pauli-Y (Y)		$\begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}$
Pauli-Z (Z)		$\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$

Figura 16. Puertas lógicas cuánticas X, Y y Z.

FUENTE: https://en.wikipedia.org/wiki/File:Quantum_Logic_Gates.png

Para la aplicación de una puerta a un qubit, se aplica la transformación de la puerta, representada por su correspondiente matriz, al qubit con estado $|\psi\rangle$. Bajo el supuesto de que se tiene un qubit con estado $|\psi\rangle = |0\rangle$, se aplica la puerta X a dicho qubit. El resultado de la transformación será $X|0\rangle$ y se calcula mediante el producto de la matriz X con el de la matriz de estados de $|\psi\rangle$:

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

Tal y como se ha dicho anteriormente, la puerta X invierte la amplitud de probabilidad del estado $|0\rangle$ con la del estado $|1\rangle$. En este caso, el estado del qubit era $|0\rangle$ al 100%, por ello, al aplicar X, se transforma en el estado $|1\rangle$ al 100%, al igual que el funcionamiento clásico de las puertas lógicas.

No obstante, por linealidad, las puertas cuánticas también funcionan para estados en superposición, causando el siguiente efecto, esencial en computación cuántica, al aplicar la puerta X a un qubit en una superposición cualquiera de sus estados $|0\rangle$ y $|1\rangle$:

$$X \begin{pmatrix} P_0 \\ P_1 \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} P_0 \\ P_1 \end{pmatrix} = \begin{pmatrix} P_1 \\ P_0 \end{pmatrix}$$

Otra puerta primordial es la puerta Hadamard, denominada puerta H. Esta transformación cuántica es la que produce el estado de superposición uniforme en un qubit. El operador se define con la siguiente matriz:

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

Al aplicar dicha puerta a un qubit en estado $|0\rangle$, se obtiene el estado de superposición uniforme del ejemplo (1) de la subsección anterior:

$$H|0\rangle = 1/\sqrt{2} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 1/\sqrt{2} \\ 1/\sqrt{2} \end{pmatrix} = \frac{|0\rangle + |1\rangle}{\sqrt{2}}$$

Otras puertas que pueden ser útiles, aunque en este trabajo no se van a utilizar, son las puertas de rotación que permiten rotar el estado cuántico de un qubit un ángulo determinado alrededor de cualquiera de los ejes X, Y o Z.

Las puertas cuánticas básicas están orientadas a transformaciones en el estado de un único qubit, no obstante, para realizar algoritmos cuánticos es necesario que la información se relacione entre los distintos qubits, lo cual se consigue mediante las puertas controladas.

El control de aplicación de una puerta es lo que permite entrelazar la información entre qubits ya que es posible que uno o varios qubits controlen la activación de un operador en otro registro cuántico, condicionando así el resultado mediante el estado cuántico de los qubits que controlan dicho operador.

Un ejemplo de una puerta o transformación controlada es la que se utiliza para entrelazar dos partículas, explicado al inicio de la sección de introducción a la computación cuántica, concretamente en la explicación del entrelazamiento cuántico. Esta puerta es la *Controlled NOT* (CNOT) o *Controlled X* (CX), la cual invierte el estado del qubit objetivo si el estado del qubit de control es $|1\rangle$. Esta transformación es exactamente la equivalente a la puerta XOR clásica, ya que el estado del qubit objetivo será $|1\rangle$ siempre que uno de los dos qubits estén en estado $|1\rangle$, pero no si lo están los dos.

Una puerta también puede estar controlada por múltiples qubits. Siguiendo con el ejemplo anterior, la puerta X puede estar controlada por dos qubits, siendo efectiva la aplicación de dicha puerta cuando ambos qubits de control tienen estado $|1\rangle$. Esta es la puerta de Toffoli, también llamada CCNOT. De esta manera, se puede construir una puerta AND cuántica, ya que el estado del qubit objetivo se invierte cuando ambos qubits controladores se encuentran en estado $|1\rangle$. Si el qubit objetivo se encuentra inicialmente en estado $|0\rangle$, entonces solo cambiará al estado $|1\rangle$ cuando los dos qubits de control sean $|1\rangle$.

La representación gráfica de las puertas CNOT y CCNOT se muestran a continuación, junto con su matriz de transformación correspondiente.

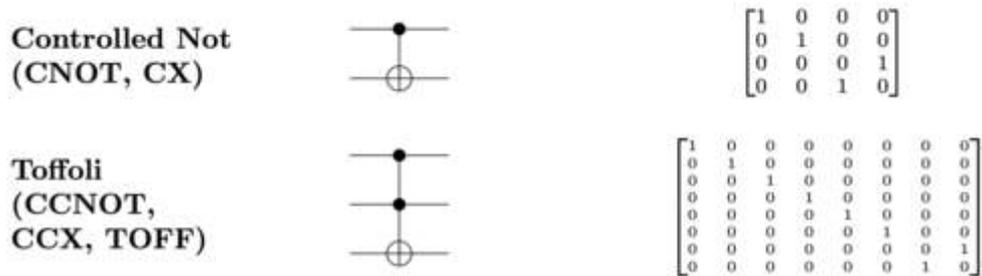


Figura 17. Puertas CNOT y CCNOT.

FUENTE: https://en.wikipedia.org/wiki/File:Quantum_Logic_Gates.png

Una observación importante es que las dimensiones de una matriz de transformación que representa la función de una puerta lógica cuántica depende del número de qubits de entrada que tiene dicha puerta. Las matrices son siempre cuadradas y su dimensión es $2^n \times 2^n$ donde n es el número de qubits de entrada de la puerta correspondiente a la matriz.

Por ejemplo, la puerta CNOT tiene como entrada 2 qubits, por ello, como se observa en la figura anterior, su matriz de transformación es de $2^2 \times 2^2$ que es 4×4 . Y, en cuanto a la puerta CCNOT, al tener 3 qubits de entrada, la dimensión de su matriz asociada es de $2^3 \times 2^3$ que resulta en una dimensión de 8×8 .

Cuando se realiza una operación controlada, los qubits que intervienen establecen una relación de condición en función de su información. La consecuencia directa de este hecho es que los qubits de control asocian a su estado información sobre la transformación que provocan en los qubits objetivo. A esto se le llama asociación de autovalores y se retrata más detalladamente en explicaciones posteriores.

Es importante anotar que todas las puertas cuánticas mencionadas hasta ahora son transformaciones auto adjuntas, además de unitarias. Esto implica que dichas puertas cuánticas son su propia inversa, tal y como se cuenta al inicio de la sección de introducción.

Teóricamente, si una transformación se puede descomponer en un conjunto de transformaciones unitarias, entonces dicha transformación también será unitaria, lo cual se traduce en que, un determinado algoritmo cuántico compuesto por una serie de transformaciones unitarias, se podrá invertir para obtener la entrada en función de la salida deseada tan solo invirtiendo el orden de aplicación de las transformaciones que forman el citado algoritmo.

Esto tiene que ver con la reversibilidad de las puertas cuánticas, ya que se precisa de cierta información para poder revertir un algoritmo y transformarlo en su inverso.

En informática clásica, una puerta AND, por ejemplo, no es reversible, puesto que no se puede descifrar la información de entrada correspondiente a una salida: se sabe que, si la salida es '1', ambas entradas son '1' pero, en el caso de que la salida sea '0', se necesita más información para saber qué valores de entrada han producido dicho resultado.

En contraste, un algoritmo cuántico, si todas las puertas cuánticas que utiliza son unitarias, se cumple que todo el circuito es reversible y, por ello, comentado anteriormente, basta con

conocer el orden de aplicación de las puertas lógicas del algoritmo en cuestión para poder invertirlo.

Las puertas lógicas cuánticas son el desglose con menor nivel de abstracción en la computación cuántica. En la próxima sección, las puertas se agruparán para crear operadores y funciones más complejas con el objetivo de implementar algoritmos útiles algo más generales.

Circuitos cuánticos

Hasta el momento, se han explicado los conceptos, mecanismos y técnicas que se han de tener en cuenta para diseñar algoritmos mediante circuitos cuánticos.

En esta parte, se reúnen muchas de las ideas discutidas hasta ahora como las puertas lógicas cuánticas y las propiedades de superposición, entrelazamiento, interferencia y colapso cuántico. Este conjunto de conceptos es necesario para construir con eficiencia y, sobretodo, eficacia circuitos cuánticos que simulen algoritmos orientados a la resolución de un problema concreto.

Dentro de los circuitos cuánticos, también existen niveles de abstracción, es decir, hay algoritmos cuánticos con un propósito general cuyo circuito está establecido y su implementación es independiente del problema específico que se esté tratando.

Un ejemplo de circuitos cuánticos generales puede ser el circuito relativo a la transformada cuántica de Fourier en el que, únicamente, puede variar el número de qubits sobre los que se realiza esta transformada, pero siempre con una misma rutina de operaciones.

Para introducir los circuitos cuánticos, a continuación, se muestra la forma gráfica de representación de circuitos, donde la profundidad del circuito denota los instantes de tiempo en el que se aplican puertas, y la anchura, cada cable horizontal, el número de qubits existentes.

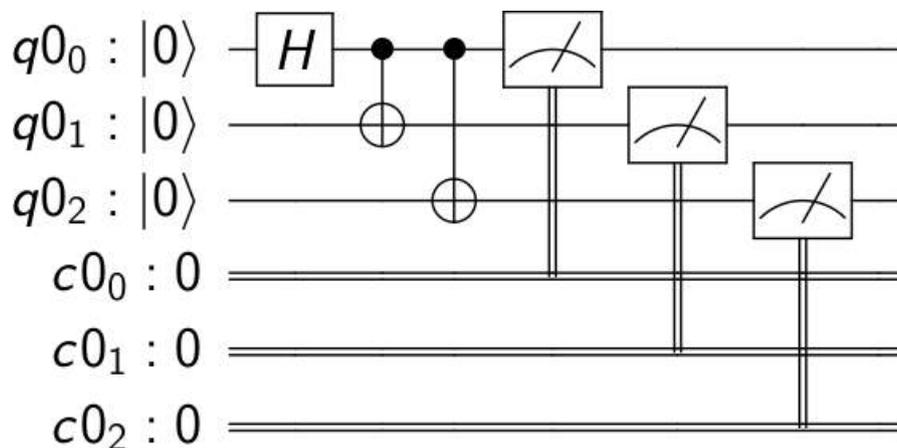


Figura 18. Circuito cuántico de entrelazamiento entre 3 qubits.

FUENTE: https://miro.medium.com/max/699/1*s-5fWmOwUJ4PhM2-1RXr0q.png

El circuito cuántico de la figura consta de 3 qubits, inicializados a $|0\rangle$, y 3 registros clásicos, también inicializados a 0. Está compuesto por una puerta Hadamard (H), dos puertas CNOT y 3 medidas del estado cuántico al final, una por qubit.

La función del circuito es entrelazar los tres qubits: el primer qubit, al encontrarse en estado $|0\rangle$ y tras aplicarle la puerta H, se encuentra en superposición uniforme de $|0\rangle$ y $|1\rangle$. Después, al aplicar las puertas NOT a los otros dos qubit controladas por el primero, como inicialmente

están en $|0\rangle$, cambiarán su estado a $|1\rangle$ cuando el primer qubit sea $|1\rangle$ y permanecerán en estado $|0\rangle$ si el primer qubit está en estado $|0\rangle$.

Por lo tanto, en este circuito, al realizar la medida o colapso del estado cuántico al final, solo habrá dos valores posibles: $|000\rangle$ o $|111\rangle$, que representan el estado de los tres qubits en un único registro general. En concreto, antes de realizar el colapso de los tres qubits, el estado cuántico de estos es el siguiente:

$$\frac{|000\rangle + |111\rangle}{\sqrt{2}} = \frac{1}{\sqrt{2}}|000\rangle + \frac{1}{\sqrt{2}}|111\rangle$$

Por ello, existe un 50% de posibilidades de obtener $|000\rangle$ y otro 50% de obtener $|111\rangle$ cuando se realiza la medida del estado cuántico.

En este caso, el circuito tiene una anchura de 3, dado que se trata de un circuito de 3 qubits, y una profundidad de 3, sin contabilizar los símbolos de medida, ya que la puerta H y las dos CNOT se han de aplicar en un orden secuencial, aunque, seguidamente, se expone otro circuito donde se aprovecha mejor el paralelismo de qubits en cuanto a la aplicación de puertas sobre estos, reduciendo así la profundidad del circuito.

Ahora, se introduce el algoritmo de Deutsch-Josza con su correspondiente circuito cuántico para retratar ciertos métodos y técnicas que, usualmente, se utilizan al construir circuitos relativos a otros algoritmos.

Sea $x = x_3x_2x_1$ donde $x_i \in \{0,1\}$, dada una función booleana $f(x): \{0,1\}^3 \rightarrow \{0,1\}$, el algoritmo cuántico de Deutsch-Josza determina si la función $f(x)$ es constante o balanceada, es decir, si todas las imágenes o salidas de la función devuelven un único valor, 0 o 1, o si la mitad de valores son 0 y la otra mitad son 1.

Hay que recalcar que la utilidad de este algoritmo es únicamente simbólica dado que, sin importar el número de qubits de entrada (la dimensión de x), el algoritmo cuántico reduce exponencialmente el tiempo computacional para resolver el problema, pero, en la práctica, no tiene una utilidad relevante.

Clásicamente, si se tienen 2^n distintas combinaciones, se ha de realizar el cálculo de $f(x)$ para saber si es constante o balanceada que, en el caso peor, requiere calcular $f(x)$ para las 2^n combinaciones posibles. No obstante, el algoritmo de Deutsch-Josza requiere de un solo cálculo o aplicación de $f(x)$ para determinar una solución al problema.

El circuito cuántico equivalente al algoritmo de Deutsch-Josza se muestra en el diagrama siguiente. Se pueden observar dos circuitos donde la única diferencia entre ellos es que los 3 qubits de entrada que simulan la entrada x se juntan en una única línea.

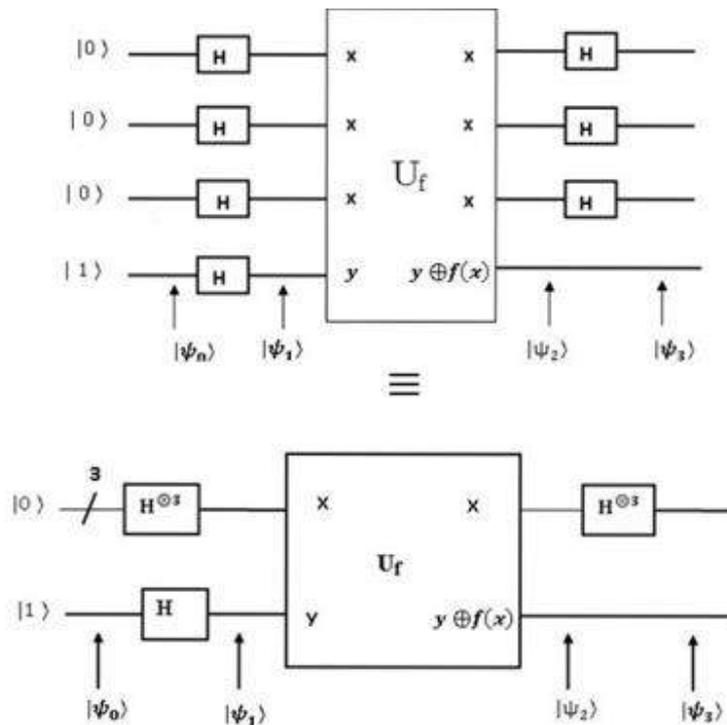


Figura 19. Circuito cuántico para el algoritmo de Deutsch-Jozsa.

FUENTE: https://www.researchgate.net/figure/Circuit-for-implementing-Deutsch-Jozsa-Algorithm_fig1_330291673

Para tener una traza de cómo evoluciona el algoritmo, el seguimiento del estado cuántico del circuito se hace por niveles de profundidad, describiendo la función de onda del estado cuántico después de cada nivel. Aquí, existen 4 niveles de profundidad en los que el estado del sistema de qubits cambia.

Para mayor claridad, en la función de onda se separarán los qubits referentes al registro $|x\rangle$ del qubit objetivo.

Inicialmente, se tienen 4 qubits inicializados, de arriba a abajo, con el estado $|0001\rangle$. Los tres primeros qubits hacen referencia a la variable de entrada x , mientras que el cuarto es el qubit objetivo mediante el cual, en la última etapa, se consigue crear interferencia de amplitudes y dar la solución al problema. El estado en este instante inicial es:

$$|\psi_0\rangle = |000\rangle |1\rangle$$

En primera instancia, se aplican 4 puertas H de forma paralela, una para cada qubit. Esto es, poner en superposición uniforme todos los estados de x y transformar el estado del qubit auxiliar a un estado concreto que determinará la solución en etapas posteriores del circuito.

La diferencia entre inicializar a $|0\rangle$ y a $|1\rangle$ es que, cuando se aplica H a un qubit en estado $|1\rangle$, se tiene una superposición uniforme, pero con una fase negativa en el estado $|1\rangle$ en lugar de positiva como sucede cuando el estado inicial del qubit es $|0\rangle$. Esto provoca que, al volver aplicar H, se pueda distinguir entre el estado $|0\rangle$ y el $|1\rangle$. La función de onda resultante es la siguiente:

$$|\psi_1\rangle = H^{\otimes 3}|000\rangle H|1\rangle = \frac{1}{\sqrt{2^3}} \sum_{x=000}^{111} |x\rangle \left(\frac{|0\rangle - |1\rangle}{\sqrt{2}} \right)$$

Después, se aplica un oráculo sobre todos los qubits, que no es más que un operador unitario compuesto por puertas cuánticas donde la transformación que realiza se da en el qubit objetivo. La entrada de dicho operador es $|x\rangle|y\rangle$, y su salida es $|x\rangle|y\oplus f(x)\rangle$.

Dentro del oráculo Uf , para el algoritmo de Deutsch-Josza, se tiene un circuito cuántico que calcula la función a evaluar. El valor de dicha función se opera con el qubit objetivo $|y\rangle$ mediante una XOR (CNOT).

Dado que la variable $|x\rangle$ está en superposición de todos sus posibles estados y que dicha superposición controla la aplicación de la función $f(x)$, como se ha comentado en la sección de puertas lógicas, los autovalores de la función $f(x)$, que son $(-1)^{f(x)}$ se asocian al registro $|x\rangle$, quedando el estado cuántico como:

$$|\psi_2\rangle = \frac{1}{\sqrt{2^3}} Uf \left(\sum_{x=000}^{111} |x\rangle \left(\frac{|0\rangle - |1\rangle}{\sqrt{2}} \right) \right) = \frac{1}{\sqrt{2^3}} \sum_{x=000}^{111} (-1)^{f(x)} |x\rangle \left(\frac{|0\rangle - |1\rangle}{\sqrt{2}} \right)$$

Al aplicar de nuevo las puertas H al registro de los tres primeros qubits, se realiza la interferencia en función de los valores que toma la función $f(x)$. El resultado de estas transformaciones es:

$$\begin{aligned} |\psi_3\rangle &= \frac{1}{\sqrt{2^3}} H^{\otimes 3} \left(\sum_{x=000}^{111} (-1)^{f(x)} |x\rangle \right) \left(\frac{|0\rangle - |1\rangle}{\sqrt{2}} \right) \\ |\psi_3\rangle &= \frac{1}{\sqrt{2^3}} \sum_{x=000}^{111} (-1)^{f(x)} H^{\otimes 3} \left(\sum_{x=000}^{111} |x\rangle \right) \left(\frac{|0\rangle - |1\rangle}{\sqrt{2}} \right) \\ |\psi_3\rangle &= \frac{1}{\sqrt{2^3}} \sum_{x=000}^{111} (-1)^{f(x)} \frac{1}{\sqrt{2^3}} \sum_{z=000}^{111} (-1)^{xz} |z\rangle \left(\frac{|0\rangle - |1\rangle}{\sqrt{2}} \right) \\ |\psi_3\rangle &= \frac{1}{2^3} \sum_{z=000}^{111} \sum_{x=000}^{111} (-1)^{f(x)+xz} |z\rangle \left(\frac{|0\rangle - |1\rangle}{\sqrt{2}} \right) \end{aligned}$$

Llegado este punto, cuando se mide el registro de los tres primeros qubits, considerando que $|z\rangle = |000\rangle$, si la función $f(x)$ es constante (su imagen es siempre la misma para cualquier x), hay certeza de medir todo $|0\rangle$, mientras que, si es balanceada (la mitad de valores son 0 y la otra mitad 1), hay certeza de que no se medirán solo $|0\rangle$, ya que la amplitud de $|z\rangle = |000\rangle$ es:

$$A_z = \frac{1}{2^3} \sum_{x=000}^{111} (-1)^{f(x)}$$

Por ello, si $f(x)$ siempre tiene el mismo valor (es constante), la A_z vale ± 1 , pues no se anulan los términos del sumatorio y, por tanto, todas las medidas son $|0\rangle$ con una probabilidad de 1.

En caso contrario, si $f(x)$ puede tomar distintos valores, los términos del sumatorio se anulan unos con otros y $A_z = 0$, lo que significa que se tiene certeza absoluta de que no todas las

medidas son $|0\rangle$, es decir, de que existe algún qubit que colapsa en $|1\rangle$ con un 100% de certeza.

En este algoritmo, se utiliza la propiedad de interferencia de la función de onda para obtener propiedades de una función concreta sin necesidad de hacer todas las consultas necesarias clásicamente para demostrar dichas propiedades.

Muchos algoritmos siguen la estructura general de este circuito, un oráculo que implementa la función a evaluar, una superposición uniforme del registro cuántico que representa la variable independiente y un qubit bandera o *flag qubit* en superposición uniforme con una fase negativa en su estado $|1\rangle$, también llamado estado $|-\rangle$, para generar interferencia dependiendo de los resultados de la función.

Ventajas de la computación cuántica

A lo largo de la introducción, se han tratado ejemplos teóricos y prácticos en los que la mecánica cuántica ofrece ventajas computacionales muy determinantes. En esta parte, se recogen dichas ventajas exponiendo la complejidad computacional en relación con el sistema cuántico de computación utilizado.

Una virtud de la computación cuántica es la capacidad de procesamiento de vectores que posee. Hay que tener en cuenta que la capacidad de memoria aumenta exponencialmente con el número de qubits del sistema: añadir un qubit al sistema, implica duplicar el número de estados que se pueden almacenar para ser evaluados en un algoritmo cuántico.

Esto contrasta fuertemente con la informática clásica ya que, mientras que en un registro de n bits se puede almacenar únicamente un número desde 0 hasta $2^n - 1$, en informática cuántica, un registro de n qubits es capaz de almacenar información relativa a los 2^n números representables, a pesar de que dicha información no sea directamente observable.

Otra ventaja es la del paralelismo de operaciones, que permite la aplicación de transformaciones en los qubits de forma paralela, evitando tener que aplicar dichas operaciones de una en una en cada qubit. En un ordenador clásico, se dispone de una ALU (Unidad Aritmético-Lógica) que no permiten paralelizar cálculos a no ser que se disponga de varias ALUs o de una GPU, en la que, además, se han de controlar de alguna manera las dependencias de datos.

Como se ve en el algoritmo de Deutsch-Josza, es posible poner n qubits en superposición uniforme (aplicando n puertas H) en un único ciclo de operación. Y, como este ejemplo, muchas otras transformaciones son paralelizables, con lo que se consigue reducir la profundidad del circuito y, por ende, el tiempo de computación.

Una ventaja significativa es la de la superposición de estados. Dado que un registro cuántico de n qubits puede tomar todos los valores desde 0 hasta $2^n - 1$, dicho registro se puede utilizar como una variable y hacer evaluaciones de una función cuando la variable toma todos los posibles valores, obteniendo información sobre los 2^n valores de la función la cual se almacena en el estado cuántico.

El algoritmo de Deutsch-Josza utiliza esta mecánica pues, dado que el registro $|x\rangle$ toma todos los posibles valores desde 000 hasta 111, la función objetivo $f(x)$ es evaluada con todas las

combinaciones simultáneamente, obteniendo información de los 2^3 valores distintos de $f(x)$ sin necesidad de calcularlos uno a uno.

Esto es una de las principales ventajas de la computación cuántica ya que se puede aumentar el espacio vectorial de combinaciones añadiendo qubits al sistema, lo que implica que las combinaciones posibles de x aumentan exponencialmente y sigue siendo posible realizar las evaluaciones de $f(x)$ para todas las combinaciones posibles en función de la dimensión de x .

Aumentar el espacio de qubits no afecta en la profundidad general del circuito cuántico, pero es cierto que cuanto mayor sea la dimensión de x , más complejo será el circuito cuántico que implementa la transformación de $f(x)$. Con lo cual, es posible que al aumentar el número de qubits para representar una variable x tenga su repercusión en profundidad en cuanto a la construcción del operador que calcula $f(x)$.

A pesar de que el operador de $f(x)$ aumente su profundidad al aumentar el espacio vectorial de x , se calcularán, con cada qubit que se añada al registro x , el doble de valores de $f(x)$ correspondientes a todas las definiciones posibles de x .

Después, gracias a la interferencia de las funciones de onda que representan los estados cuánticos, es posible construir algoritmos cuánticos eficaces con órdenes de complejidad considerablemente menores respecto de cualquier otro tipo de algoritmo con objetivos equivalentes.

Gracias a todo esto, la potencia de cálculo y la escalabilidad de los algoritmos cuánticos son muy superiores a aquellos que se puedan implementar en computación clásica.

Por lo general, el mejor papel que puede desempeñar la computación cuántica es la resolución de problemas en los que el número de combinaciones a evaluar es elevado, como por ejemplo en los problemas NP-completos.

Otros ejemplos en los que hay una gran diferencia en complejidad computacional utilizando algoritmos cuánticos son los problemas de combinatoria referentes a grafos, árboles o a cualquier exploración en espacios de estados, ya que, tal y como se ha visto, la computación cuántica paraleliza comprobaciones y evaluaciones de todos los estados que sea posible representar mediante registros de qubits.

Inconvenientes de la computación cuántica

La computación cuántica ha abierto muchos horizontes en la informática gracias a sus propiedades únicas que paralelizan y aceleran los cálculos.

Sin embargo, la mecánica cuántica también tiene inconvenientes significativos que impiden la construcción de computadores cuánticos físicos de altas capacidades, con un gran número de qubits.

Esto se debe a que el estado cuántico de una partícula se ve perturbado por el medio en el que esta se encuentra. La termodinámica y el resto de partículas del entorno provocan que el estado cuántico se vea corrompido según va transcurriendo el tiempo, tendiendo a perder el estado cuántico y a generar un estado clásico. A esto se le llama decoherencia cuántica y es el mayor inconveniente, hoy en día, a la hora de construir máquinas cuánticas con una gran cantidad de qubits.

A medida que pasa el tiempo, la decoherencia de los estados cuánticos es cada vez mayor puesto que, con las perturbaciones mínimas que sufre un qubit aislado lo más perfecto

posible, los errores se van acumulando. Por ello, cuanto más tiempo pasa desde el inicio del algoritmo cuántico, más inciertos son los resultados finales.

Además, para que el entrelazamiento de información entre qubits sea posible físicamente, los qubits han de estar interconectados entre ellos para poder implementar operaciones de dos qubits o más cuando dichas operaciones son controladas por otros qubits, lo cual también aumenta la decoherencia y la tasa de error de los estados cuánticos.

También, la falta de perfección a la hora de la aplicación de las puertas lógicas cuánticas ocasiona una acumulación de error en el estado cuántico puesto que existen probabilidades no nulas de que la aplicación de una puerta lógica no transforme convenientemente el estado cuántico. Cuantos más qubits de entrada tenga la puerta lógica, menor es la fidelidad de la puerta en cuestión.

La temperatura a la que se encuentran los qubits es uno de los factores determinantes en la decoherencia cuántica. Según el tercer principio de la termodinámica, a medida que la temperatura se va acercando al cero absoluto, es decir, 0 Kelvin, la entropía del sistema tiende a ser constante, por lo que la influencia del entorno en el sistema cuántico también tiende a anularse.

Por esta razón, los computadores cuánticos de la actualidad buscan aislar los qubits de la mejor manera posible, intentando establecer una temperatura lo más cercana al cero absoluto, consiguiendo así, aumentar el tiempo de decoherencia y, como consecuencia, aumentar el tiempo que se puede ejecutar un programa cuántico y obtener resultados con un determinado nivel de confianza.

Cabe recalcar que, si el tiempo de decoherencia es lo suficientemente alto, es posible implementar protocolos de corrección de errores del estado cuántico. No obstante, esto requiere de un sistema de control basado en computación clásica y, a medida que aumentan los qubits del sistema a corregir, la computación de la corrección de errores se hace más y más compleja, tomando así más tiempo para poder corregir los errores lo suficientemente rápido como para hacer ejecuciones sostenibles y confiables según transcurre el tiempo.

Por otro lado, la programación de algoritmos cuánticos va ligada al uso de ciertas técnicas especiales para lograr resultados útiles. Esto se debe a que, como se ha comentado anteriormente, aunque se hagan evaluaciones paralelas de todos los estados posibles, al medir el estado cuántico resultante solo se obtiene uno de esos resultados.

Debido a esto, es necesario emplear propiedades como la interferencia o la estimación de las fases cuánticas, entre otras técnicas, si se desean resultados válidos de algoritmos cuya complejidad se ve reducida al utilizar computación cuántica.

Es importante tener en cuenta estas técnicas adicionales pues la complejidad computacional aumenta debido a ellas. Esto tiene una repercusión relacionada con el tiempo de decoherencia, pues al aumentar la complejidad computacional también aumenta el tiempo necesario para completar la ejecución de un algoritmo, lo cual incrementa la probabilidad de obtener resultados falsificados o corruptos.

4. Estado del arte

Un ordenador cuántico es aquel que utiliza un sistema cuántico de partículas como iones, fotones o átomos de hidrógeno, para programar un algoritmo usando las propiedades de la mecánica cuántica, acelerando así los cálculos de dicho algoritmo.

Para poder construir un computador cuántico, se han de cumplir varios requisitos esenciales con el fin de conseguir un sistema capaz de simular cualquier algoritmo y de obtener resultados correctos con una probabilidad relativamente alta. Según [28], los requisitos necesarios son los siguientes.

El primer requisito es que el sistema permita realizar todas las transformaciones lógicas universales para, de este modo, poder implementar cualquier tipo de función lógica imaginable.

Otro requisito es que se mantenga una coherencia cuántica a medida que el algoritmo se ejecuta en el sistema cuántico, pues la decoherencia generada por las distintas perturbaciones del entorno del sistema produce resultados falseados, con más probabilidad según transcurre el tiempo.

Ha de ser posible, también, realizar medidas de todos los qubits que forman el sistema para poder leer el resultado una vez finalizadas las operaciones del algoritmo cuántico. A su vez, es necesario que el sistema pueda inicializarse para partir siempre de un estado conocido del sistema y construir el algoritmo a ejecutar acorde a ello, haciendo que el sistema evolucione coherentemente con las transformaciones del algoritmo hacia la solución o soluciones buscadas.

Un último requisito es el de la escalabilidad. El sistema ha de soportar extensiones de qubits para poder implementar problemas más complejos computacionalmente y, por ende, más útiles en la práctica.

En la actualidad, existen varios modelos físicos de ordenadores cuánticos universales que cumplen estos requisitos, exceptuando el de la escalabilidad a causa de uno de los inconvenientes de la mecánica cuántica mencionado anteriormente: el aumento de la cantidad de qubits y sus interconexiones supone un obstáculo para la implementación de un protocolo de corrección de errores.

La empresa multinacional de tecnología *IBM* ha intervenido activamente en la construcción de máquinas cuánticas de puertas con un total de 18 distintas desde el 2016, según cuenta el artículo [29], publicado en mayo de 2020. La mayoría de estas máquinas son de uso privado, pero existen otras a disposición de los usuarios, algunas de pago y otras gratuitas.

Algunos de los procesadores cuánticos de uso libre que ofrece *IBM* son el de Melbourne con 15 qubits, o el de Orense o London con 5 qubits, entre otros. Sus topologías de interconexión entre qubits se muestran en la siguiente figura.

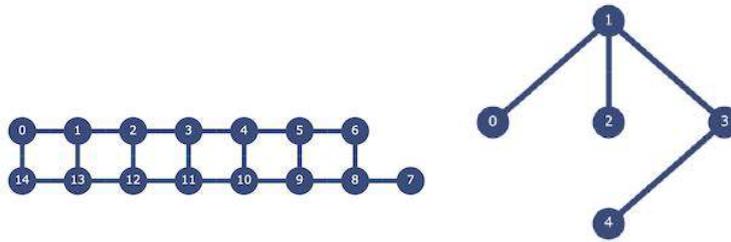


Figura 20. Red de conexiones entre los qubits del IBM Q Melbourne(izquierda) y Orense/London(derecha).
 FUENTE: <https://quantum-computing.ibm.com/docs/manage/backends/>

Los procesadores cuánticos premium o privados de *IBM* son más sofisticados, con un mayor número de qubits. En la cima, se encuentran el de Rochester con 53 qubits, seguido por el de Cambridge con 28 qubits, con sus respectivas topologías de red mostradas en la figura a continuación.

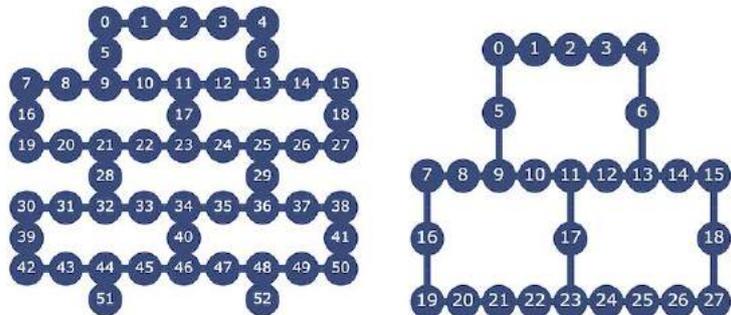


Figura 21. Topología de interconexiones del IBM Q Rochester(izquierda) y Cambridge(derecha).
 FUENTE: <https://quantum-computing.ibm.com/docs/manage/backends/>

Además, en enero de 2019, *IBM* lanzó el primer ordenador cuántico de puertas comercial, accesible para cualquier persona mediante la nube y se trata de un computador de 20 qubits llamado *IBM Q System One*.

Todos estos ordenadores cuánticos están contruidos con qubits superconductores transmon, tal y como afirma *IBM* en [30], los cuales son un tipo de qubit de carga superconductora creados artificialmente con el objetivo de reducir la sensibilidad de estos al ruido de carga. Según enuncia *IBM* en la cita anterior, este tipo de qubits son el fundamento para los ordenadores cuánticos de estado sólido controlados eléctricamente.

La topología de los ordenadores cuánticos utilizada por *IBM* hasta la fecha se basa en dichos qubits superconductores transmon. En la figura siguiente, se puede observar un esquema de cómo están diseñados estos qubits físicamente, los cuales se encuentran interconectados por resonadores de microondas, encargados del direccionamiento y el emparejamiento entre qubits dentro del procesador.

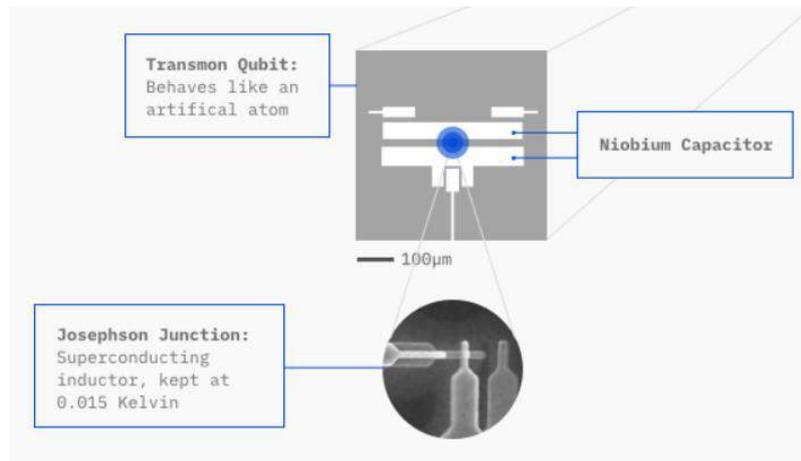


Figura 22. Qubits superconductores transmon de IBM.

FUENTE: <https://www.ibm.com/quantum-computing/learn/what-is-ibm-q/>

Acorde a [31], la computadora cuántica *IBM Q System One* tiene una tasa de error menor al 2% para las puertas cuánticas de 2 qubits y menor del 1% para puertas de 1 qubit.

En artículos recientes como [32], *IBM* presenta la escala de los nuevos ordenadores cuánticos en los que están trabajando.

En este mismo año, 2020, lanzan el *IBM Quantum Hummingbird* de 65 qubits para miembros de *IBMQ Network*. Para el año 2021, planean presentar el *IBM Quantum Eagle* con 127 qubits en el que, debido a su diseño, han conseguido equilibrar la conectividad entre qubits y la reducción de error permitiendo implementar el código de corrección de errores hexagonal pesado.

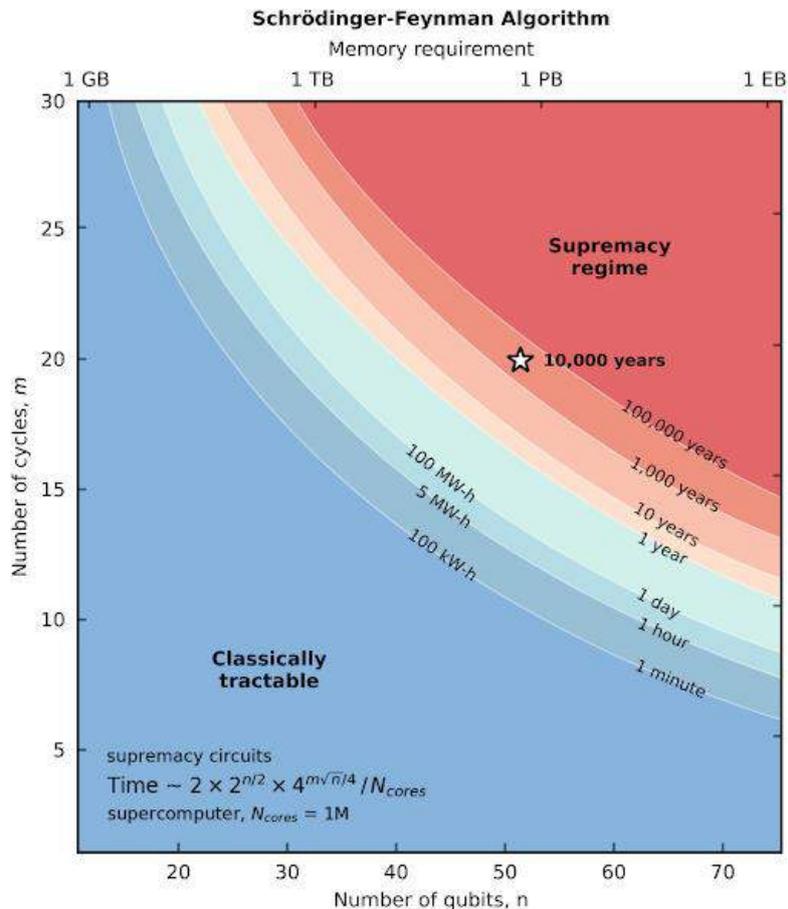
Después, en 2022, introducirán el *IBM Quantum Osprey* de 433 qubits con controles más eficientes y con una infraestructura física criogénica con el objetivo de evitar mantener el rendimiento y reducir el ruido.

Finalmente, en 2023, *IBM* planea lanzar el procesador cuántico *IBM Quantum Condor* con 1121 qubits en el que se tratarán los errores cometidos en los anteriores modelos y, dada la necesidad de aislamiento que requiere un computador de estas magnitudes, también presentan un nuevo refrigerador llamado *Goldeneye*, diseñado para 1 millón de qubits y que se irá testeando con todos los computadores cuánticos mencionados, incrementando más y más la escala de qubits hasta llegar a esta cifra.

Otra corporación relevante en computación cuántica a día de hoy es *Google* que, en septiembre de 2019, declaró que habían alcanzado la supremacía cuántica.

La supremacía cuántica hace referencia a alcanzar la capacidad de procesamiento cuántico tal que sea posible resolver un determinado problema que es casi irresoluble en tiempo por los ordenadores clásicos. En el gráfico de la figura inferior, se observan las características que definen la supremacía, donde la estrella denota la supuesta supremacía alcanzada por *Google*.

Simplemente, los problemas cuya resolución se demoraría 100.000 años o más en un computador clásico de 1 millón de núcleos serían el último nivel de supremacía computacional, teniendo en cuenta, también, la restricción de memoria.



Google afirmó que su máquina cuántica de 54 qubits (53 operativos) llamada *Sycamore* resolvía un problema de generación de números aleatorios que, para el supercomputador más potente construido hasta la fecha: *Summit*, desarrollado por *IBM*, llevaría alrededor de 10.000 años, mientras que el computador cuántico *Sycamore* resolvía los cálculos en unos 200 segundos [33].

Cuando se filtró la noticia, *IBM* se mantuvo escéptico ante la supuesta supremacía declarada por la compañía cuando, días después, la rechazó dado que el tiempo estimado para que el *Summit* resolviera el cálculo se reducía a dos días y medio (donde *Google* establecía 10.000 años) puesto que el problema se podía reestructurar de una forma más óptima para reducir con creces el tiempo de cómputo [34].

Esto supone que, por el momento, la supremacía cuántica no ha llegado ya que un requisito es que el programa que se ejecute ha de ser prácticamente irresoluble para cualquier ordenador o superordenador clásico, en cuanto a tiempo se refiere. Sin embargo, dos días y medio, a pesar de que exista una gran diferencia respecto a los 200 segundos del programa cuántico, no supone un tiempo muy largo de cómputo y, por ello, no se puede hablar de supremacía cuántica.

No obstante, independientemente de alcanzar o no la supremacía como concepto, esto sigue suponiendo un gran paso adelante para la computación cuántica, pues un ordenador cuántico de escasos qubits, ha sido capaz de superar en tiempo con un factor de más de 600 al supercomputador con más capacidad de procesamiento que existe, manteniendo la afirmación de *IBM* de que la tarea se puede ejecutar en dos días y medio en *Summit*.

La arquitectura física del *Google Sycamore* es la mostrada en la figura siguiente. Los qubits que forman el computador cuántico son del tipo de los de *IBM*, qubits transmon superconductores, aunque esta vez la topología del ordenador es una malla en la que cada qubit dentro de la malla se empareja con otros 4, como se observa en la figura.

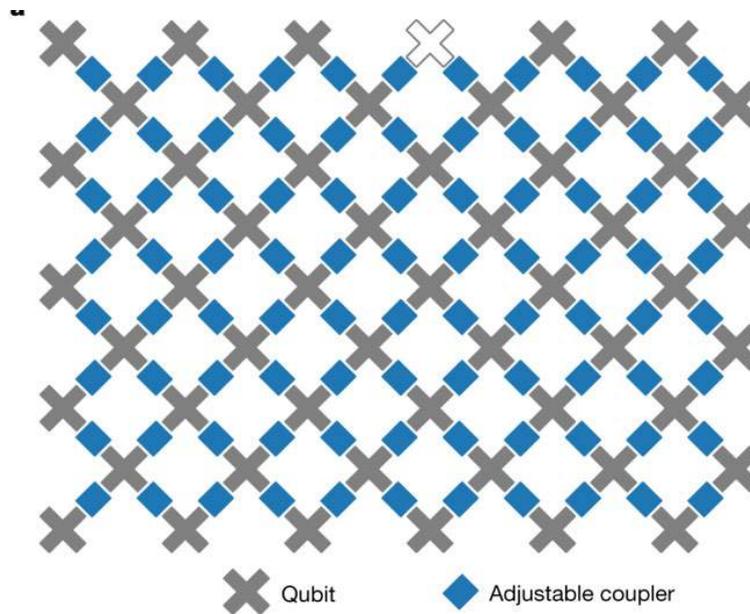


Figura 23. Topología de interconexión de qubits del Google Sycamore.
FUENTE: <https://www.nature.com/articles/s41586-019-1666-5>

El *Sycamore* tiene una tasa de error en las operaciones cuánticas, según afirma el líder del hardware cuántico de *Google* en [35], de alrededor de 0.5% y sugiere que para conseguir un protocolo de corrección de errores efectivo los qubits se ha de reducir dicha tasa en un factor de más de 5.

El hecho de que, en la actualidad, no sea posible reducir la tasa de error en los ordenadores cuánticos hasta un umbral que permita construir una máquina cuántica tolerante a fallos sitúa la computación cuántica en la era llamada NISQ (*Noisy Intermediate-Scale Quantum technology*), una era en la que los ordenadores cuánticos son de escala intermedia y no tolerantes al ruido, es decir, con un número moderado de qubits y con una tasa de error relativamente alta.

Por otro lado, dado el obstáculo de la tolerancia a fallos, la compañía tecnológica *Microsoft* está trabajando en un modelo de qubits topológicos que, básicamente, son estructuras artificiales no fundamentales que actúan como qubits, es decir, son capaces de albergar un estado cuántico y, en general, con mayor fidelidad que los qubits superconductores o las trampas de iones.

El objetivo de los qubits topológicos es deshacerse de las tasas de errores en la aplicación de puertas lógicas, en la observación del estado cuántico y en la generada por la decoherencia. En general, la meta es lograr un sistema cuántico escalable.

Un reciente artículo, publicado por *Microsoft* a finales de marzo de 2020, habla de un gran avance en los qubits topológicos que se basa en los fermiones de Majorana, un fermión que es su propia antipartícula, lo cual presenta características exóticas como la neutralidad de carga

cuando se trata de una cadena de fermiones de Majorana. Estas características dan lugar a mejoras en las tasas de error de las puertas cuánticas y en el tiempo de decoherencia.

Microsoft comenta en [36] que un qubit topológico se construiría organizando varios cables de escala nanométrica que contienen “*Majorana zero modes*” (MZMs) en un entorno con características especiales: temperaturas bajas, campos magnéticos y materiales adecuados. Afirman que el hardware cuántico topológico es robusto en cuanto a ruido se refiere, lo que permitiría la escalabilidad en el número de qubits del computador. Para más información sobre los MZMs, véase [37].

En lo que a computador cuántico físico se refiere, *Microsoft* todavía no dispone de uno. Sin embargo, ha desarrollado un lenguaje de programación cuántica *open source*, *Q#*, que permite crear algoritmos cuánticos y ejecutarlos en un simulador local al cliente o bien en *Azure Cloud*, donde se dispone de mayor capacidad de procesamiento.

A pesar de no disponer de un ordenador cuántico físico, *Microsoft* tiene una arquitectura teórica que describe la estructura de un computador cuántico tolerante a fallos. Dicha arquitectura conecta el lenguaje de alto nivel, *Q#*, con un compilador del mismo para transformar las sentencias del programa en operadores que, mediante una FPGA, se reproducirán en el ordenador cuántico topológico, siendo esta capaz de devolver los resultados medidos cuando el cómputo finalice. Esta arquitectura se puede ver en detalle en el siguiente esquema.

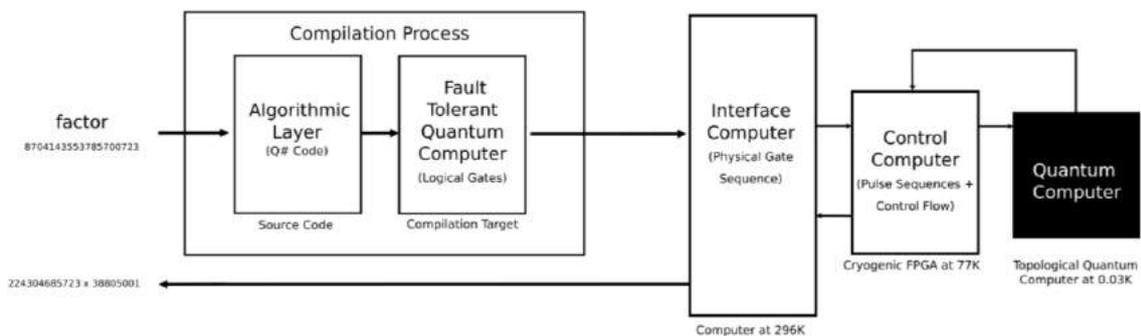


Figura 24. Arquitectura teórica del ordenador cuántico de Microsoft.

Algo que cabe destacar es que, el 14 de agosto de 2020, se publicó el artículo [38] que hace referencia a un nuevo algoritmo, desarrollado por integrantes de la Universidad de Sídney, que encaminaba la computación cuántica hacia una computación libre de errores.

Dicho algoritmo, en términos generales, trata de describir el ruido cuántico de forma precisa y eficiente mediante matrices de correlación entre qubits interconectados, llegando al origen de los errores y permitiendo implementar un protocolo de corrección de errores eficaz y escalable. La descripción detallada del algoritmo se puede encontrar en [39].

Para acabar con las arquitecturas cuánticas, cabe mencionar la empresa de *Rigetti Computing* que posee un ordenador cuántico de circuitos electrónicos superconductores a disposición de muchas herramientas de desarrollo como *1QBit* o *Forest SDK*, entre muchos otros.

Su ordenador cuántico más reciente es el de *Aspen-8* con 30 qubits, donde la duración media de ejecución está entre 19 y 28 microsegundos (μs), lo cual hace referencia al tiempo de decoherencia, y la fidelidad de puertas cuánticas es de 99.8% para puertas de 1 qubit y de 96% para puertas de 2 qubit, tal y como declaran en su página principal [40].

En cuanto a aplicaciones reales, actualmente existe una gran cantidad de aproximaciones teóricas sobre la construcción de algoritmos cuánticos capaces de resolver problemas prácticos que requieren un elevado número de cálculos. Muchos de ellos son algo tan simple como problemas de combinatoria que, mediante computación clásica, su resolución solo llega a ser posible mediante algoritmos de inteligencia artificial. No obstante, dada la situación de la computación cuántica, no es posible implementarlos eficazmente.

En el campo de la inteligencia artificial, concretamente en la clasificación de imágenes, investigadores de *Google* han demostrado cómo la computación cuántica puede ayudar a clasificar imágenes de 28x28 píxeles iluminadas por un único fotón. Demuestran que es posible conseguir, al menos, un 41.27% de precisión frente al 20% que proveen los algoritmos clásicos cuando se trata de clasificar una imagen de 28x28 con el primer fotón que impacta en ella.

Según [41], publicado el 14 de agosto de 2020, los investigadores de *Google* afirman que no es necesario iluminar una imagen con múltiples fotones a la vez para producir interferencia.

Adicionalmente, en el área de la inteligencia artificial, existen múltiples aplicaciones de la computación cuántica para la estimación de hiperparámetros para algoritmos de *Machine Learning*. También, para la construcción de memorias asociativas y redes neuronales cuánticas gracias al perceptrón cuántico.

Otra aplicación en la que es relevante la mecánica cuántica es la ciberseguridad, puesto que el principio del colapso cuántico hace que cualquier medición de la información cuántica destruya su estado originalmente creado.

Por esta razón, como dice el artículo [42], China implementó un sistema cuántico de distribución de claves por satélite para garantizar al 100% una transmisión de información segura. Este sistema se utiliza actualmente en el Banco Industrial y Comercial de China (ICBC) cuando se trata de comunicación entre ciudades distantes, por ejemplo, del noreste al noroeste de China, donde la comunicación puede ser intervenida por un tercero.

Hoy en día, la ciberseguridad cuántica tiene cada vez más impacto por todas las ventajas que ofrece y es cada vez más utilizado en ciertas telecomunicaciones. Además, cabe añadir que el proceso de criptografía cuántica se puede implementar en la realidad utilizando infraestructuras existentes, como cables de fibra óptica aplicando el mecanismo de entrelazamiento de fotones.

Las finanzas son un área en la que la computación cuántica es de gran utilidad dada la gran cantidad de variables. Como se expone en [43], los algoritmos cuánticos son aplicables a servicios financieros como la simulación o la optimización, además de problemas de aprendizaje automático.

Como ejemplo de problemas financieros, en [44] se realiza la estimación de valores referentes al riesgo de crédito con el algoritmo de *Quantum Amplitude Estimation*, con un gráfico relativo a la distribución de pérdidas.

Por último, en el estudio [45] se recogen las aceleraciones en tiempo que supone la computación cuántica frente a algoritmos clásicos para problemas asociados a estructuras de datos.

Por ejemplo, para problemas de modelos de consulta en bases de datos, utilizando el algoritmo de Grover, se consigue una aceleración cuadrática donde la complejidad computacional pasa de $\Theta(n)$ a $\Theta(n^{1/2})$. Otro ejemplo es el conocido algoritmo de Shor basado en la búsqueda de periodicidad en una secuencia, el cual supone una aceleración exponencial respecto de algoritmos clásicos, de $\Theta(2^n)$ a $\Theta(n)$, cuando los datos de entrada tienen una estructura especial. Este último algoritmo es conocido por su poder potencial de romper el algoritmo de cifrado RSA en unas horas, utilizado actualmente para la ciberseguridad en internet.

5. Herramientas para el desarrollo de programas cuánticos

Para crear programas cuánticos, hoy en día, se disponen de librerías y de lenguajes de programación que permiten construir circuitos cuánticos. En [46], se pueden encontrar todos los detalles de las distintas herramientas junto con los desarrolladores de estas. Algunas de ellas serán explicadas a continuación mencionando sus características más relevantes.

Según la herramienta, el nivel de abstracción en la programación es mayor o menor pues algunas ofrecen algoritmos completos, como puede ser la transformada cuántica de Fourier (QFT), y otras solo permiten codificar algoritmos mediante puertas lógicas básicas, como las mencionadas en la sección de introducción.

Algunas de las herramientas poseen una API (*Application Programming Interface*) que permite, tras codificar un programa, ejecutarlo en una máquina cuántica real enviándolo a través de internet. Esto es posible si los desarrolladores de la herramienta en cuestión tienen un ordenador cuántico físico accesible para el público.

Este es el caso de *IBM* con su proyecto *Qiskit*, que es un módulo de *Python*, *Swift* o *Java* mediante el cual es posible codificar algoritmos cuánticos con un nivel de abstracción bajo y, posteriormente, ejecutarlos en un simulador (local o remoto) o directamente en uno de los computadores cuánticos accesibles de *IBM*. Además, si se desea un nivel de abstracción algo más alto, *IBM* también ofrece otras librerías como la de *Qiskit Aqua* que contiene funciones de algoritmos cuánticos básicos parametrizados. Más adelante, se indaga más profundamente sobre *Qiskit* y sus funcionalidades.

Otro proyecto que dispone de una máquina cuántica física a disposición de usuarios es *Forest*, que pertenece a la compañía de *Rigetti Computing*. *Forest* es un proyecto *open source* que utiliza *Python* como lenguaje de programación anfitrión. En general, permite construir circuitos cuánticos con puertas básicas, aunque también ofrece procedimientos de más alto nivel con el paquete *Grove* y está basado en el conjunto de instrucciones de *PyQuil*. Véase [47] para más información sobre *Forest* y *PyQuil*.

Un último kit de desarrollo con acceso a máquinas cuánticas reales es el de *Cirq* y se trata de un proyecto desarrollado por *Google* que, al igual que *Qiskit* y *Forest*, utiliza *Python* como lenguaje de programación anfitrión para la creación de circuitos cuánticos.

En [48], se pueden encontrar cantidad de especificaciones de *Cirq* como pueden ser tutoriales, detalles sobre las máquinas cuánticas de *Google* y de cómo ejecutar un algoritmo de *Cirq* en las mismas.

Por otro lado, existen kits de desarrollo de software cuántico que no ofrecen la posibilidad de ejecutar programas cuánticos en máquinas reales. En su lugar, poseen simuladores que permiten, mediante computación clásica, imitar las características de la computación cuántica, lo cual es muy costoso en tiempo y memoria para los ordenadores convencionales, pero, a pesar de ello, estos kits dan la posibilidad de comprobar el funcionamiento de algoritmos cuánticos de unos pocos qubits y, en general, testear el paradigma cuántico.

Un proyecto relevante que todavía no se encuentra en posesión de un ordenador cuántico es el de *Microsoft Quantum Development Kit (MQDK)*. Como su nombre indica, es un kit de desarrollo de software cuántico lanzado por *Microsoft* a finales de 2017 que, a diferencia de

los proyectos anteriores, posee un lenguaje de programación cuántico multiparadigma llamado *Q#*, en honor al lenguaje *C#*.

Este lenguaje de programación cuántico tiene una gran cantidad de librerías y, por ende, de herramientas. Ofrece métodos cuánticos de todo tipo de niveles de abstracción, desde puertas lógicas básicas hasta procesos útiles en inteligencia artificial cuántica como *Quantum Machine Learning*.

Por último, se introduce *Amazon Braket*, un servicio de la compañía *Amazon* con el que, mediante *Python*, se crean programas cuánticos que se pueden simular o bien ejecutar en ordenadores cuánticos de terceros como el de *Rigetti*, *IonQ* o, el computador cuántico adiabático de *D-Wave*. Las características completas de *Amazon Bracket* se pueden encontrar en [49].

A continuación, en las dos subsecciones siguientes, se estudian con más detenimiento los kits de *Qiskit* y *Microsoft Quantum Development Kit* dado que son lenguajes de programación cuántica sofisticados y, además, con los que se implementa y se testea el algoritmo de *Quantum Counting* expuesto en secciones posteriores.

Microsoft Quantum Development Kit

Como se ha comentado en el apartado anterior, *MQDK* es un conjunto de herramientas *open source* que ofrece *Microsoft* para el diseño de programas cuánticos con puertas y que, además, posee su propio lenguaje de computación cuántica llamado *Q#*, por lo que no es necesario utilizar un lenguaje de programación auxiliar que actúe como anfitrión del programa cuántico.

Para utilizar *Q#* y poder codificar programas, es suficiente con disponer del editor gratuito *Visual Studio Code* y basta con instalar la extensión de *Q#*. De manera adicional, *Jupyter Notebooks* incluye *Q#* como lenguaje, permitiendo crear archivos híbridos de documentación y código en *Q#*.

Todos los datos y características sobre *MQDK* que se desarrollan a lo largo de este apartado se extraen de la documentación que ofrece *Microsoft* en [50].

Antes de comenzar el análisis de *Q#*, cabe resaltar que *Microsoft* está trabajando constantemente en *MQDK*, añadiendo nuevos paquetes con funciones más sofisticadas y modificando matices del mismo lenguaje.

En primer lugar, *Q#* es un lenguaje de programación de tipado fuerte, es decir, es muy rígido en cuanto a los tipos de variables, de funciones y de parámetros. Se trata de un lenguaje de alto nivel, aunque también ofrece funcionalidades de nivel más bajo, como por ejemplo puertas cuánticas primitivas.

A nivel genérico, existen dos tipos de variables, las constantes, no modificables a lo largo de la ejecución del programa, y las mutables, que sí es posible modificar. Después, tiene los tipos de variables comunes como números enteros (*Int*), números reales (*Double*), etcétera. Otros tipos de variables orientadas a la computación cuántica son los tipos de *Qubit*, referente a la instanciación de un qubit, *Result*, que es la estructura de datos en las que se puede almacenar un estado cuántico medido, y *Pauli*, que define la base en la que se aplican las operaciones y en la que se puede realizar la medida del estado cuántico. Además, *Q#* implementa la

funcionalidad de tipos de usuario, permitiendo al programador definir tipos de datos o incluso tipos orientados a cabeceras de operaciones o funciones.

Para declarar varias instancias de un mismo tipo se utilizan arrays, que se definen comúnmente con corchetes delante del tipo de la variable. Sin embargo, los arrays que se pueden definir únicamente pueden ser de una dimensión, es decir, no es posible definir matrices con filas y columnas, aunque sí se puede definir una matriz ordenando todas las filas seguidas como si fueran una sola, teniendo como resultado una única dimensión.

Q# también permite la definición de funciones como en cualquier otro tipo de lenguaje, a lo que se le suma la creación de operaciones las cuales se corresponden con la construcción de operadores cuánticos o transformaciones sobre el espacio de estados de un registro de qubits, aunque no es estrictamente necesario que intervengan qubits. Las operaciones se diferencian con las funciones en que no tienen valor de retorno, pues se trata únicamente de un procedimiento que aplica ciertos cambios sin devolver ningún valor adicional más que la transformación. Además, las operaciones pueden definirse como controlables y/o adjuntas, lo que significa que, al construir un determinado operador controlable y adjunto, es posible instanciarlo convenientemente como controlable por otros qubits y como adjunto o invertible, lo que implica poder revertir la transformación en un momento dado.

Para la simulación de la computación cuántica, se puede utilizar tanto el simulador local que, con ciertas restricciones, permite ejecutar en el PC privado hasta algoritmos con 25 o 30 qubits, como el simulador de *Azure*, que ofrece hasta una instanciación de 40 qubits. Existen, adicionalmente, otros simuladores como el *ToffoliSimulator* que solo permite operaciones con puertas X, CNOT o múltiple CNOT, con el que es posible instanciar, según asegura la documentación, millones de qubits. O un simulador de recursos: *ResourcesEstimator*, que estima el número de puertas y qubits que requiere el algoritmo cuántico en cuestión.

Cabe recordar que *Microsoft* todavía no dispone de una máquina cuántica real en la que se puedan ejecutar los algoritmos desarrollados en *Q#*.

Ahora, una vez introducidos los conceptos básicos del lenguaje, se exponen las herramientas que ofrece *Microsoft* con *Q#*, empezando por las más comunes, siguiendo con librerías de bajo nivel de abstracción y finalizando con un escalado hacia funcionalidades con niveles de abstracción más altos. Todas estas librerías y paquetes que ofrece *Q#* se encuentran en la documentación de la API en [51].

En primer lugar, *Q#* provee funciones matemáticas básicas como valores absolutos, funciones trigonométricas, fracciones continuadas, logaritmos, máximo, mínimo y, en general, operaciones matemáticas fundamentales. También ofrece operaciones lógicas binarias como desplazamiento de bits, puertas lógicas clásicas como AND, XOR, etcétera. Luego, incluye una librería de conversión entre tipos y otras para realizar diagnósticos y aserciones de sentencias, como desigualdades y comprobaciones de ciertas características en estructuras de datos, por ejemplo, en arrays. Y, entre otras, funciones que implementan operaciones en bucle según los parámetros deseados.

Después, *MQDK* ofrece varias librerías con operaciones primitivas. Las operaciones primitivas hacen referencia a las puertas lógicas cuánticas fundamentales: H, Z, NOT, CNOT, CCNOT, rotaciones parametrizadas de fase cuántica respecto de los ejes, y muchas más, como la operación de medida M. Adicionalmente, proveen librerías de preparación de estado cuántico,

con las que se pueden evitar ciertos bucles de inicialización de qubits o aplicar una operación iterativamente acorde a parámetros que definen la condición de terminación.

En un nivel algo más alto de abstracción, se encuentran las representaciones más abstractas de los registros cuánticos como, por ejemplo, representar un registro de qubits como números enteros en lugar de cadenas binarias, definiéndolo como *Little endian* o *Big endian*. También se disponen de operaciones aritméticas como suma o multiplicación de registros de qubits y otras operaciones de colapso que devuelven en forma de número entero el resultado del registro medido. Esto permite trabajar con datos más manejables por el programador sin necesidad de que este tenga que codificar los métodos correspondientes para la representación de la información.

Después, en otro nivel de generalización, *Microsoft* ofrece métodos estándar como la creación de oráculos controlados, composición de operadores, la transformada cuántica de Fourier, y algoritmos de estimación de fase y de amplificación de amplitudes, entre los más importantes, junto con el algoritmo híbrido *Variational Quantum Eigensolver* (VQE) que combina la computación clásica y la cuántica optimizando recursos.

Cabe mencionar otras herramientas como la simulación de Hamiltonianos, la simulación de propiedades dinámicas (sistemas adiabáticos) y métodos para química cuántica, un ámbito cada vez más en auge.

Por otro lado, *MQDK* dispone de una librería de corrección de errores con varios métodos para el tratamiento de la decoherencia y la tasa de error del estado cuántico. Otro paquete relevante para el entorno de computación es el de *Environment*, que ofrece información sobre el número de qubits del sistema de los que se puede disponer de forma auxiliar, para utilizar una sola vez, o de forma permanente, utilizados hasta el final del programa.

En la última capa de abstracción, se encuentran librerías de inteligencia artificial con algoritmos como *Random Walk*, algoritmos de optimización, métodos para clasificadores y gradientes, junto con otro paquete de datasets con los que se pueden realizar entrenamientos de *machine learning*.

Con este breve análisis de las herramientas de *Microsoft Quantum Development Kit*, se puede ver que *Q#* es un lenguaje de programación cuántico completo y versátil, con muchos niveles de abstracción y con una gran variedad de opciones a la hora de programar. Además, el lenguaje es intuitivo y accesible para usuarios con un nivel medio de conocimiento en programación y en computación cuántica.

Es importante incidir en que *Microsoft* sigue construyendo nuevos métodos y librerías útiles para relajar la tarea de la programación cuántica de bajo nivel. Además, cuando dispongan de un ordenador cuántico y lo enlacen los programas escritos en *Q#*, este kit de desarrollo software puede llegar a ser muy potente dada la cantidad de librerías y algoritmos que ofrecen.

Qiskit

Como se ha introducido previamente, *Qiskit* es un entorno de trabajo *open source* desarrollado por *IBM* para la codificación, simulación y ejecución de circuitos cuánticos. Dicho entorno funciona con el lenguaje de programación *Python* y *Qiskit* es un módulo a importar en un código con extensión *.py* para desarrollar programas cuánticos de puertas. Al igual que *Q#*, *Qiskit* también está disponible en la aplicación de *Jupyter Notebooks*.

Toda la documentación relativa a *Qiskit* y de la que se extrae la información recogida en este apartado se encuentra en [52].

Esta herramienta de desarrollo de software cuántico se divide en cuatro elementos que trabajan a nivel de circuitos, pulsos y algoritmos. A continuación, se describen las características más sobresalientes de cada uno de los elementos que forman *Qiskit*, de menor a mayor nivel de abstracción. Para más detalles, se puede acudir a la referencia del párrafo anterior.

El primer módulo es *Qiskit Terra*, en el cual se apoyan el resto de elementos. Es el módulo con el nivel de abstracción más bajo donde los programas cuánticos están codificados a nivel de puertas y, más allá, a nivel de pulsos, que es el equivalente físico de la aplicación de una puerta cuántica.

Dado que se trata de la capa de abstracción más baja, en ella se llevan procesos de optimización del algoritmo cuántico a ejecutar acorde a las restricciones del dispositivo físico objetivo. Y, en cuanto a las ejecuciones remotas, *Qiskit Terra* se encarga de la comunicación entre el *backend*, la máquina física en la que se ejecutará el programa, y el usuario que desarrolla dicho programa.

Qiskit Terra, además, se subdivide en otros 6 módulos: *Circuit*, referente a la instanciación de circuitos cuánticos y sobre los que se implementan las operaciones para construir algoritmos cuánticos; *Pulse*, el nivel más bajo de abstracción donde las puertas cuánticas se traducen en los respectivos pulsos físicos a aplicar en el ordenador cuántico; *Transpiler*, una herramienta diseñada para obtener circuitos equivalentes más eficientes del mismo algoritmo dadas las limitaciones de la decoherencia cuántica y la tasa de errores en las puertas; *Providers*, el módulo encargado de la comunicación entre el *frontend* y el *backend* para facilitar la selección del dispositivo final de ejecución con ciertos parámetros, como el número de repeticiones, y para la obtención de resultados por parte del usuario; *Quantum_info*, con tareas que ofrecen información sobre estimación de medidas y generación de operaciones y estados cuánticos, diseñado para algoritmos más complejos; y, por último, *Visualization*, que contiene métodos para graficar el circuito cuántico físico que representa un determinado algoritmo.

Después está el elemento *Qiskit Aer*, cuyas herramientas proveen entornos de simulación para circuitos cuánticos con distintas características parametrizadas como la configuración de ruido, acercando la simulación de un programa cuántico a una ejecución en un ordenador físico real.

Qiskit Aer se compone de los tres siguientes simuladores en *backend*: *QasmSimulator*, que permite la ejecución ideal y ruidosa de circuitos cuánticos y que, a su vez, incluye simulaciones de un algoritmo con distintas representaciones como *statevector*, donde el estado cuántico se trata como un vector de estados, o *matrix_product_state*, que utiliza la representación

matricial expuesta en la introducción; *StatevectorSimulator*, que ofrece una simulación ideal del circuito y retorna el vector de estados cuánticos final, sin realizar ninguna medida; y *UnitarySimulator* que, igual que el anterior, es una simulación de ejecución ideal pero retorna la matriz unitaria final correspondiente al estado cuántico del circuito, también sin medidas de este.

En el siguiente lugar, se encuentra *Qiskit Ignis*, cuyo objetivo es aliviar los errores caracterizándolos de la mejor manera posible y tratando de mejorar la efectividad de las puertas al máximo. En especial, este elemento está diseñado para el desarrollo y experimentación de correctores de errores cuánticos en cualquiera de sus aspectos.

Existen tres tipos de experimentos que se pueden realizar con *Qiskit Ignis*. El primero, *Characterization*, diseñado para medir parámetros como el ruido y parámetros del Hamiltoniano como tasas de interacción y errores en puertas. El segundo, *Verification*, que verifica aplicaciones de puertas y resultados de circuitos que, en general, sirve para obtener información de la fidelidad de las puertas. Y, el tercero, *Mitigation*, que son rutinas de corrección en circuitos las cuales son aplicables a otros conjuntos de resultados del mismo *backend*.

Dados estos experimentos, el código de *Ignis* se descompone en tres bloques: el bloque de circuitos, donde se encuentran la lista de circuitos para experimentos concretos con un pequeño conjunto de parámetros variables por el usuario, luego, el bloque de *Fitters* o adaptadores, donde los resultados de los experimentos con *Ignis* son analizados y se adaptan según el modelo físico del experimento, y, por último, el bloque de filtros que, según el experimento, los *Fitters* pueden devolver un objeto *Filter* el cual se puede utilizar para mitigar errores en otros experimentos.

En el último nivel de abstracción, se encuentra el módulo de *Qiskit Aqua* donde se recogen métodos y algoritmos que permiten codificar programas cuánticos útiles en el mundo real, orientados a los campos en los que la computación cuántica juega un papel importante como en la química, en la optimización, en inteligencia artificial y en finanzas. Además, existen múltiples tutoriales orientados a la introducción de la computación cuántica en estos campos.

Los métodos de *Qiskit Aqua* requieren algo más de conocimiento y habilidad en computación cuántica. Permite que los programadores puedan contribuir con programas y algoritmos cuánticos de alto nivel.

La librería de circuitos de *Qiskit* se puede consultar en [53]. Dicha librería recoge, en términos generales, puertas cuánticas primitivas, puertas generalizadas, puertas booleanas básicas, circuitos básicos como la QFT, circuitos aritméticos como sumadores o comparadores, entre otros.

Para finalizar, es importante destacar que los códigos cuánticos escritos con *Qiskit* se pueden ejecutar en la máquina cuántica real de *IBM*, además de poder simularlo de forma local o remota. Por ello, el algoritmo cuántico desarrollado en este proyecto se programa con *Q#* y con *Qiskit* con el fin de realizar una comparativa entre ambos códigos y, de forma adicional,

mediante *Qiskit* poder ejecutarlo en un computador cuántico real para discutir los resultados obtenidos.

6. Quantum Counting: Algoritmo de búsqueda cuántico

En esta sección, se explica rigurosamente el desarrollo teórico para la implementación del algoritmo de *Quantum Counting* o cuenta cuántica. El estudio de este algoritmo está inspirado por la experiencia previa en computación cuántica, en concreto con el algoritmo de Grover, y únicamente se ha utilizado la información de [54] para construir el programa.

El *Quantum Counting* es un algoritmo cuántico que permite contar el número de soluciones que tiene una determinada función booleana sin necesidad de tener que comprobar, una a una, las salidas de cada posible entrada de dicha función.

Dada una función $f: \{0,1\}^n \rightarrow \{0,1\}$, donde la entrada a la función es una cadena binaria de dimensión n y la salida de la función es 0 o 1, el algoritmo de *Quantum Counting* tiene como meta contar el número de cadenas binarias de entrada x que satisfacen $f(x) = 1$. De manera formal, este algoritmo tiene como salida $|M|$ donde $M = \{x \mid f(x) = 1\}$ con $x \in \{0,1\}^n$ dada una función objetivo $f(x)$.

Mediante este algoritmo cuántico, es posible obtener la solución a este problema con una complejidad computacional menor que la que requeriría el procedimiento de evaluación de cada una de las combinaciones codificando la función objetivo a evaluar como una función oráculo dentro de un circuito cuántico.

Hoy en día, existen muchos problemas donde el espacio de combinaciones es muy amplio, tanto que da lugar a problemas irresolubles en un tiempo moderado. Un tipo de problemas que poseen esta característica son los problemas *NP-complete* y hacen referencia a problemas no resolubles en tiempo polinómico, en el caso de que *NP* fuera distinto a *P*, lo cual es desconocido en la actualidad. Esto se desarrolla con mayor detalle en la sección 8.

Es importante destacar que buscar el número de combinaciones de entrada de una función cuya salida es '1' no es más que un proceso abstracto. En la realidad, existen una gran cantidad de aplicaciones prácticas, como las explicadas en la sección 9, en las que la resolución del problema reside, a nivel bajo, en la búsqueda del número de entradas que satisfacen una condición determinada, codificada en la función objetivo, como la presencia de determinados elementos en una base de datos, una función de verificación de problemas *NP*, algoritmos de aprendizaje automático utilizando umbrales, etcétera.

El algoritmo de Grover y la transformada cuántica de Fourier para la estimación de fase cuántica, hace posible realizar una implementación de este algoritmo más óptima en tiempo respecto de la fuerza bruta clásica.

En las siguientes secciones se describen meticulosamente cada uno de estos procesos que, organizados, dan forma al algoritmo de *Quantum Counting* clásico, es decir, utilizando la estimación de fase o *Quantum Phase Estimation*.

Algoritmo de Grover

El algoritmo de Grover es un algoritmo cuántico de búsqueda en conjuntos de datos no necesariamente estructurados. Su objetivo es la búsqueda de elementos con características genéricas en un espacio de soluciones potenciales. En general, toda la información sobre el

algoritmo de Grover ha sido estudiada del libro de introducción a la computación cuántica en [26].

Este algoritmo de búsqueda es el núcleo del algoritmo de *Quantum Counting*, pues es el que consigue encontrar las soluciones dada una función objetivo, la cual caracteriza los elementos buscados dentro del conjunto.

A nivel general, el algoritmo de Grover utiliza el mecanismo de evaluar una función objetivo, codificada como un oráculo, con todas las posibles combinaciones de la variable de entrada, representada por un registro de qubits en superposición uniforme. Tras haber evaluado la función, todos aquellos elementos característicos que cumplen la función objetivo son marcados. Por último, se amplifica la amplitud de probabilidad de los elementos marcados aplicando otro operador llamado operador de difusión.

Este proceso, aplicado un determinado número de veces, provoca que los elementos marcados, correspondientes con las soluciones buscadas, tengan una probabilidad muy alta de colapsar cuando se realiza la medición del estado cuántico, ya que sus amplitudes han sido amplificadas en el proceso de Grover hasta el máximo posible.

De manera formal, la definición del problema es muy parecida a la explicada en la introducción del *Quantum Counting*: dada una función $f: \{0,1\}^n \rightarrow \{0,1\}$, el algoritmo de Grover busca los x tal que $f(x) = 1$ donde x es la codificación binaria de una solución válida.

Para la explicación del algoritmo de Grover se va a suponer que solo existe una única solución, por lo que solo existe un valor de x tal que $f(x) = 1$.

Para comenzar, se definen dos tipos de estados acorde a si $f(x) = 1$, que son los estados correctos o los estados solución, o si $f(x) = 0$, que hacen referencia a los estados incorrectos. Con esto y dado que el espacio de entrada está en superposición uniforme de todos los estados, se puede definir la siguiente función de onda, donde M hace referencia al número de soluciones y N al número de combinaciones posibles. Aquí, $M = 1$ por la suposición anterior de que existe una única solución.

$$|\psi\rangle = \sqrt{\frac{M}{N}} |\psi_{CORRECTO}\rangle + \sqrt{\frac{N-M}{N}} |\psi_{INCORRECTO}\rangle$$

Como $|\psi\rangle$ es unitario, sumar el cuadrado de sus amplitudes ha de resultar en 1. Por ello, se puede establecer que $\sin \theta = \sqrt{\frac{M}{N}}$, $\cos \theta = \sqrt{\frac{N-M}{N}}$ y que, según la función objetivo, el valor correcto es $|1\rangle$ y el incorrecto $|0\rangle$, entonces la base de medida es $\{ |\psi_{CORRECTO}\rangle, |\psi_{INCORRECTO}\rangle \}$ donde $f(|\psi_{CORRECTO}\rangle) = |1\rangle$ y $f(|\psi_{INCORRECTO}\rangle) = |0\rangle$.

Ahora, se distinguen dos transformaciones unitarias principales: U_f , que es el oráculo de la función objetivo con el que se marcan los estados solución utilizando un qubit bandera (b), y $U_{\psi_{\perp}}$, que hace referencia al operador de difusión con el que las amplitudes de los estados marcados se amplían. Las transformaciones de estos dos operadores son las siguientes:

$$U_f: |x\rangle |b\rangle \rightarrow |x\rangle |b \oplus f(x)\rangle, \text{ con } |b\rangle = \frac{|0\rangle - |1\rangle}{\sqrt{2}}$$

$$U_{\psi_{\perp}} = H^{\otimes n} U_{0\perp} H^{\otimes n}, \text{ donde } U_{0\perp}: \begin{cases} |x\rangle \rightarrow -|x\rangle & \text{si } x \neq 0 \\ |0\rangle \rightarrow |0\rangle & \text{si } x = 0 \end{cases}$$

Con esto, al aplicar a $|\psi\rangle$ el oráculo de la función objetivo, U_f , se tiene la función de onda resultante que se muestra a continuación, donde las soluciones correspondientes con el estado $|1\rangle$ ahora poseen una amplitud negativa y los estados que resultan en $|0\rangle$ tienen una amplitud positiva, con lo que los estados solución quedan marcados en este paso y es posible diferenciarlos de los estados incorrectos.

$$U_f|\psi\rangle = -\sin(\theta) |1\rangle + \cos(\theta) |0\rangle$$

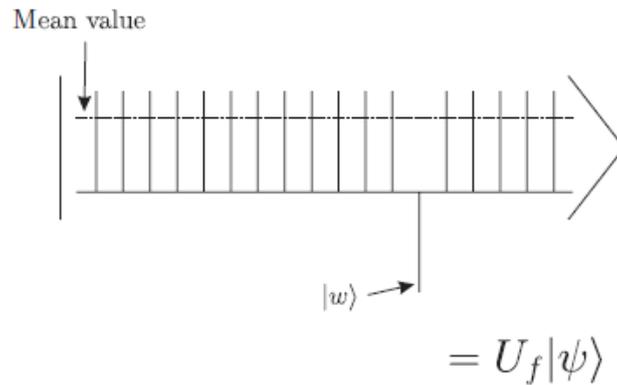


Figura 25. Aplicación del operador U_f a la función de onda $|\psi\rangle$ en estado de superposición con w estado solución.
FUENTE: [26] Grover algorithm

La conversión a amplitud negativa de los estados solución se debe a que el operador U_f se define de forma similar al oráculo del algoritmo de Deutsch-Josza explicado en la sección de circuitos cuánticos de la introducción. Esto es, U_f utiliza un qubit bandera o *flag qubit* en estado $|-\rangle$ con el que, al evaluar la función $f(x)$ con todos los estados posibles de x , los valores de x con $f(x) = 1$ adquieren el autovalor negativo correspondiente al qubit bandera, mientras que, si $f(x) = 0$, el autovalor asociado a dichos estados es la unidad, lo cual significa que los estados incorrectos no sufren transformaciones, consiguiendo así distinguir los estados solución.

Después, se tiene $|\psi_{\perp}\rangle$ que representa el vector ortogonal de $|\psi\rangle$. Si se cambia de base a la base de $\{|\psi\rangle, |\psi_{\perp}\rangle\}$, la función de onda anterior queda como:

$$U_f|\psi\rangle = \cos(2\theta) |\psi\rangle - \sin(2\theta) |\psi_{\perp}\rangle = -\sin(\theta) |\psi_{CORRECTO}\rangle + \cos(\theta) |\psi_{INCORRECTO}\rangle$$

Ahora, si se aplican rotaciones del estado cuántico de la función de onda alrededor del vector $|\psi_{\perp}\rangle$ con el operador de difusión $U_{\psi_{\perp}}$, el resultado es la amplificación de amplitud de los estados cuya amplitud es negativa, es decir, aquellos estados que se corresponden con soluciones del problema habrán adquirido una amplitud negativa. Este método de amplificación de amplitud se denomina inversión sobre la media, dando lugar al siguiente resultado.

$$U_{\psi_{\perp}}U_f|\psi\rangle = \sin(3\theta) |\psi_{CORRECTO}\rangle + \cos(3\theta) |\psi_{INCORRECTO}\rangle$$

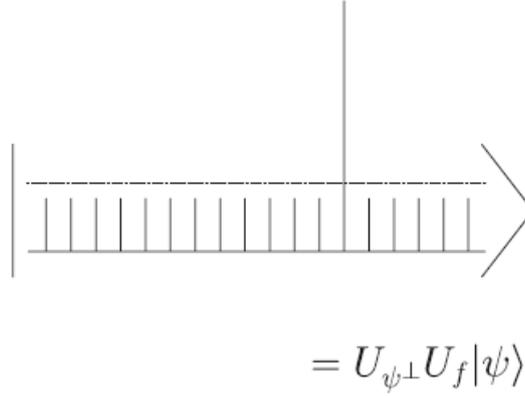


Figura 26. Estado de $|\psi\rangle$ tras aplicar U_f y $U_{\psi\perp}$ donde el estado amplificado se corresponde con el estado solución.
FUENTE: [26] Grover algorithm

De forma general, si este proceso se repite k veces, la función de onda resultante se corresponde con:

$$(U_{\psi\perp} U_f)^k |\psi\rangle = \sin((2k + 1)\theta) |\psi_{CORRECTO}\rangle + \cos((2k + 1)\theta) |\psi_{INCORRECTO}\rangle$$

Y, para maximizar las amplitudes de probabilidad de los estados correctos, se ha de aplicar este procedimiento un número de veces k dependiente de la proporción de soluciones existentes en el espacio de estados, la cual es desconocida a priori.

El algoritmo de Grover para la búsqueda en espacios de estados sigue los siguientes pasos:

1. Inicializar el registro de $n+1$ qubits, correspondientes al registro cuántico n del espacio de estado y al qubit bandera, al estado $|0\rangle^{\otimes n+1}$.
2. Aplicar $H^{\otimes n+1}$ al registro $|0\rangle^{\otimes n+1}$ y Z al qubit bandera para conseguir el estado siguiente, donde $N = 2^n$:

$$|\psi\rangle = \frac{1}{\sqrt{N}} \sum_{x=0}^{N-1} |x\rangle \left(\frac{|0\rangle - |1\rangle}{\sqrt{2}} \right)$$

3. Aplicar el operador de Grover en el siguiente orden: $U_f H^{\otimes n} U_{0\perp} H^{\otimes n}$
4. Repetir el paso 3 $3 \left\lfloor \frac{\pi}{4} \sqrt{N/M} \right\rfloor$ veces, siendo $M =$ número de soluciones.
5. Medir el estado del registro de n qubits para colapsar una de las soluciones posibles.

Para el algoritmo de *Quantum Counting*, lo que se utiliza del algoritmo de Grover es el llamado operador de Grover G , compuesto por los dos operadores principales previamente descritos:

$$G = H^{\otimes n} U_{0\perp} H^{\otimes n} U_f = U_{\psi\perp} U_f$$

Cuando existe un único x que cumple $f(x) = 1$, entonces basta con medir el registro de la variable de entrada al final del proceso para obtener dicho valor de x , pues las amplitudes de probabilidad tras el algoritmo estarán concentradas en dicha solución única.

Suponiendo que existen varias soluciones candidatas que cumplen la función objetivo, al realizar una medición después de aplicar el proceso anterior, se obtiene una única solución del espacio de soluciones con una probabilidad muy alta o, en algunos casos, con certeza absoluta.

Sin embargo, a no ser que se realicen varias repeticiones del proceso y que el espacio de soluciones sea lo suficientemente pequeño, no se obtendría por completo dicho espacio.

Si lo que interesa es conocer el número de elementos que son soluciones potenciales en un tiempo de computación reducido, el algoritmo de Grover no es suficiente para obtener todos los posibles candidatos a solución cuando el espacio de búsqueda es muy grande.

Por ello, para desarrollar el *Quantum Counting*, se necesita conocer cuántos elementos han sido marcados en el proceso del algoritmo de Grover.

Dado que la imagen de la función objetivo $f(x)$ es 1 en los estados solución y 0 en los estados incorrectos, esto genera lo que se llama el ángulo de Grover, definido como θ , el cual tiende a crecer cuantas más soluciones existan dentro del espacio de estados. En el esquema siguiente, se puede observar dicha rotación al aplicar el operador G .

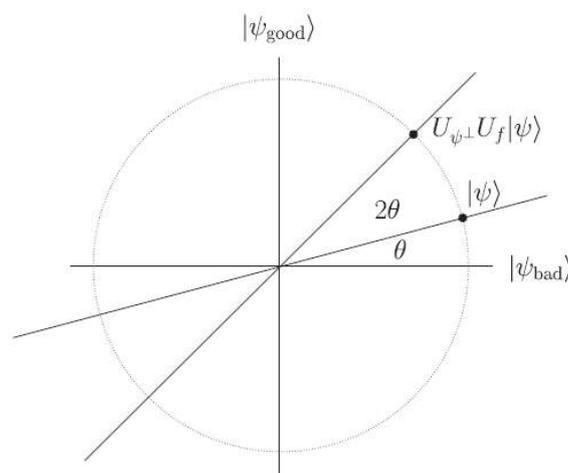


Figura 27. Ángulo del operador de Grover, donde $|\psi_{good}\rangle$ hace referencia al espacio de soluciones.
FUENTE: [26] Grover algorithm

Cuando se aplica el operador G , el espacio de Hilbert de n dimensiones formado por la variable de entrada x induce una rotación θ en relación a la proporción del número de estados solución existentes en la función $f(x)$. Esta rotación se corresponde con el autovalor del operador G y, tal y como se observa en la última función de onda y en la figura anterior, se relaciona con el número de soluciones mediante la siguiente ecuación:

$$\theta = \sin^{-1} \left(\sqrt{M/N} \right)$$

Dicho ángulo θ es el que permite conocer el número de soluciones totales M pues, al tener que $N = 2^n$ es el número total de estados, si se conoce θ , se podría hallar el valor de M .

Ahora bien, para conseguir el valor de θ , entra en juego la estimación de fase cuántica que utiliza, a su vez, la transformada cuántica de Fourier.

Transformada cuántica de Fourier

Para explicar la estimación de fase cuántica o *Quantum Phase Estimation* es necesario introducir el concepto de transformada cuántica de Fourier o *Quantum Fourier Transform (QFT)*.

La transformada de Fourier es una herramienta matemática que transforma una función definida en el dominio temporal en una función equivalente en el dominio de la frecuencia. Existe tanto la transformada continua de Fourier como la discreta, pero en informática se utiliza la transformada discreta puesto que la continua no es posible codificarla.

La definición de transformada discreta de Fourier es la siguiente:

$$X_k = \frac{1}{N} \sum_{n=0}^{N-1} x_n \cdot e^{-\frac{2\pi i}{N}kn}$$

En computación cuántica, existe un proceso similar conocido como *QFT* que se asemeja a la transformada discreta de Fourier con la única diferencia de que la información de la frecuencia en la *QFT* se encuentra en la fase cuántica.

La definición formal de la transformada cuántica de Fourier es la siguiente, donde $M = 2^m$:

$$QFT_M: |x\rangle \rightarrow \frac{1}{\sqrt{M}} \sum_{n=0}^{M-1} e^{2\pi i \frac{x}{M}n} |n\rangle$$

Como se puede observar, existe una clara correspondencia entre la transformada discreta de Fourier y el operador que implementa la transformada cuántica de Fourier. En esta última, el valor de entrada x queda codificado en la fase cuántica de m qubits los cuales representan M estados distintos.

El circuito cuántico relativo a la *QFT* se muestra en la figura a continuación. Está compuesto por puertas Hadamard y rotaciones controladas por los qubits con índice mayor al del qubit objetivo. Esto último es de gran importancia puesto que, si se utilizan n qubits, la frecuencia codificada en la fase cuántica del qubit con el índice más pequeño tendrá una precisión de n dígitos decimales binarios.

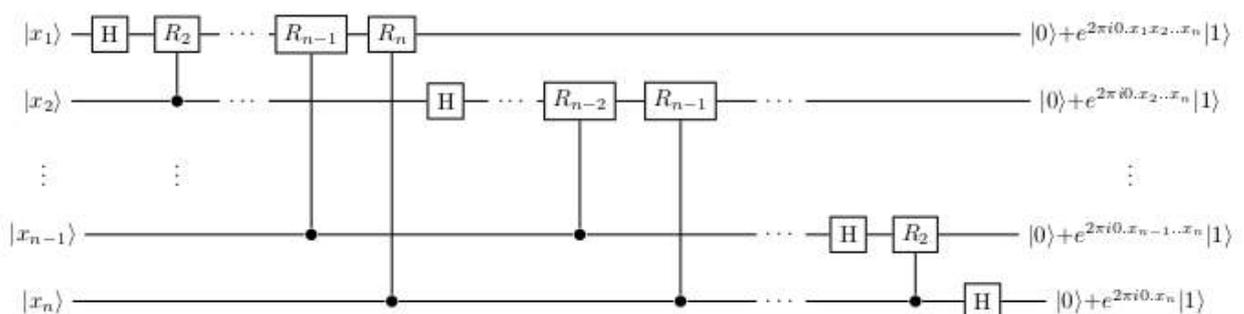


Figura 28. Circuito de la transformada cuántica de Fourier.

FUENTE: https://es.wikipedia.org/wiki/Transformada_cu%C3%A1ntica_de_Fourier

Para el caso de la estimación del ángulo θ , correspondiente al autovalor necesario del operador de Grover G , es necesario el circuito inverso, pues dicho ángulo quedará codificado en la fase cuántica de los qubits que controlan el operador G mediante el mecanismo de asociación de autovalores visto en el apartado de la introducción.

Con el fin de decodificar θ , es necesario obtener la información de la frecuencia de forma que se puedan medir los registros de la transformada $|x_1...x_n\rangle$ y, para ello, basta con aplicar el circuito cuántico a la inversa del representado en la figura superior, lo cual se define como la transformada cuántica inversa de Fourier. Su definición formal es la siguiente:

$$QFT_M^{-1}: \frac{1}{\sqrt{M}} \sum_{n=0}^{M-1} e^{2\pi i \frac{x}{M} n} |n\rangle \rightarrow |x\rangle$$

Es importante destacar que la fase de la transformada se corresponde con un ángulo multiplicado por π , por lo que este es un número decimal escrito en binario y toma valores reales en el intervalo de $[0,1)$. Sin embargo, al utilizar m qubits para hallar dicha fase, no se tienen todos los valores existentes entre 0 y 1, sino que dicho intervalo se divide en 2^m partes iguales, ofreciendo 2^m valores distintos en $[0,1)$.

Por ejemplo, si se tienen dos qubits para realizar la estimación de una determinada fase, al aplicar la transformada inversa, las posibilidades que ofrecen estos dos qubits son los números decimales en base binaria (0'00, 0'01, 0'10, 0'11) o, en base decimal, (0'00, 0'25, 0'50, 0'75).

De aquí, se puede deducir la resolución de la fase que se obtiene tras aplicar QFT^{-1} utilizando m qubits. Con 2 qubits, se tiene una resolución de 2^{-2} , por lo que m qubits ofrecen una resolución de 2^{-m} .

Esta resolución es importante a la hora de obtener la fase θ del operador de Grover ya que, si no se dispone de la suficiente precisión, el valor estimado distará demasiado del valor auténtico de θ . Dicha precisión va en relación al tamaño del espacio de estados y del espacio de soluciones del problema en cuestión, lo cual se trata con detenimiento en el siguiente apartado.

Quantum Phase Estimation

El algoritmo de la estimación de fase cuántica es la última pieza para construir el algoritmo de *Quantum Counting*, cuyo objetivo era agilizar la cuenta del número de soluciones de una función que representa la definición de un determinado problema.

Hasta el momento, se han definido dos componentes esenciales que se van a ensamblar en el *Quantum Phase Estimation*: el operador de Grover (G) y la transformada cuántica de Fourier (QFT).

Para obtener el número de soluciones de una función cualquiera, era necesario estimar el valor del ángulo θ correspondiente al autovalor del operador de Grover. Con este fin, se introduce la QFT , la cual permite obtener el valor de la fase cuántica codificada en la amplitud del estado $|1\rangle$ de cada uno de los qubits que intervienen en la estimación de θ .

Ahora, es necesario una técnica con la que se pueda codificar dicha fase θ en los qubits que realizan la estimación. El mecanismo que se utiliza es la asociación de autovalores mediante la aplicación de operadores controlados que, como se ha mencionado anteriormente, se corresponde con la misma técnica del algoritmo de Deutsch-Josza.

Lo que consiguen los operadores controlados por otro qubit es que, al igual que las puertas controladas de las que se habla en la introducción, el qubit controlador active o inactive la aplicación del operador objetivo dependiendo de si dicho qubit posee el estado $|1\rangle$ para activarlo o el estado $|0\rangle$ para inhabilitar su aplicación.

Esto implica que el operador se activa únicamente cuando el qubit de control se encuentre en estado $|1\rangle$, por lo que el qubit de control tendrá asociado a la amplitud de su estado $|1\rangle$ la fase correspondiente al autovalor del operador que controla. Por otro lado, el estado $|0\rangle$ del

qubit de control no tiene ninguna fase asociada de dicho operador puesto que, cuando se da esta situación, el operador controlado no se aplica y es como si al estado $|0\rangle$ se asociara el autovalor de un operador identidad (no causa modificación alguna) que es 1, lo cual no tiene repercusión.

En el caso del operador de Grover, el autovalor asociado es el ángulo θ que da la información necesaria para obtener el número de soluciones de la función que define el problema. Como el qubit controlador se inicializa con una superposición uniforme de $|0\rangle$ y $|1\rangle$, entonces, al transformar dicho operador en un operador controlable, se obtendría el autovalor codificado en la fase del estado $|1\rangle$ del qubit de control, quedando dicho qubit en el estado siguiente:

$$\frac{1}{\sqrt{2}} (|0\rangle + e^{2\pi i\theta} |1\rangle)$$

Este estado es muy parecido al estado de entrada que requiere la *QFT* inversa para poder descodificar el valor de θ . Sin embargo, se necesita un formato algo más específico dado que, como se contempla en la figura del circuito de la *QFT* o en su definición formal, los qubits a los que se les aplica la *QFT* han de codificar en su fase las décimas binarias correspondientes a la precisión deseada.

En el ejemplo anterior con un único qubit de control, al aplicar la *QFT* sobre este, únicamente se podrá estimar θ con dos valores en binario, $0'0$ o $0'1$, que en decimal equivale a $0'0$ o $0'5$, lo cual es insuficiente en una gran mayoría de los casos, puesto que solo se distinguen dos valores posibles.

Por ello, surge la necesidad de utilizar varios qubits para poder estimar más décimas binarias y así obtener una precisión mayor. Hay que destacar que cada qubit de control que se añada duplica la precisión de la estimación pues el espacio de estados crece de forma exponencial con el número de qubits.

Además, es necesario que cada qubit que se añada tenga el formato adecuado tal que se estime una décima binaria extra del autovalor del operador, en este caso el ángulo θ . Para conseguir esto, el qubit que hace referencia al decimal en la posición n -ésima ha de controlar la aplicación del operador G un número de veces igual a 2^{n-1} .

En la siguiente figura, se muestra el circuito correspondiente a la estimación de fase cuántica donde se recogen todos los matices discutidos hasta el momento: los qubits de control, el operador unitario controlado U , y la *QFT* inversa que transforma la fase de U asociada a los qubits de control en un valor medible al final del algoritmo.

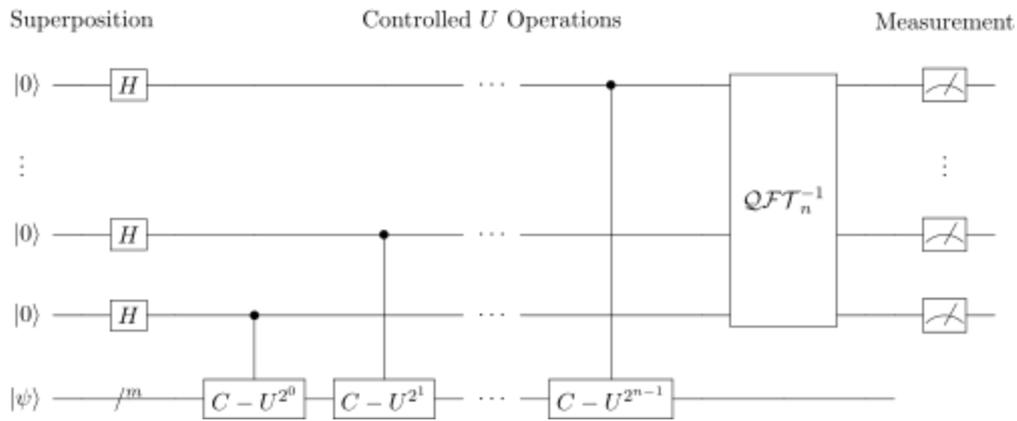


Figura 29. Circuito de Quantum Phase Estimation dado un operador objetivo U .
 FUENTE: https://en.wikipedia.org/wiki/Quantum_phase_estimation_algorithm

En lo que respecta a la implementación del *Quantum Counting*, se tiene que el operador unitario U que se muestra en la figura anterior es el operador de Grover G del cual se obtiene el valor θ que despeja el número de soluciones de la función objetivo $f(x)$. Por lo que, tras aplicar el circuito de la estimación de fase, se podrá medir dicho valor y hallar la cardinalidad del conjunto definido al inicio de la sección:

$$|M| \text{ donde } M = \{x \mid f(x) = 1\} \text{ con } x \in \{0,1\}^n$$

Quantum Counting: Búsqueda del número de soluciones

Una vez explicados todos los procesos necesarios para implementar el algoritmo de *Quantum Counting*, ya se puede describir el circuito cuántico equivalente a dicho algoritmo con su correspondiente fórmula para la interpretación de los resultados obtenidos.

En esta sección, se procede a la integración de todos los elementos necesarios para construir el algoritmo correctamente alrededor de la función objetivo a evaluar.

Primero, se tiene el operador de Grover (G) con el correspondiente operador oráculo U_f , el cual es un circuito cuántico que codifica la función objetivo como una transformación mediante puertas lógicas cuánticas universales. Posteriormente, dentro del operador G se tiene el difusor de Grover $U_{0\perp}$, que es siempre el mismo independientemente de la función objetivo.

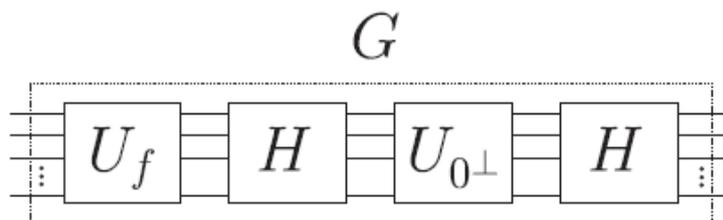


Figura 30. Operador genérico de Grover.
 FUENTE: [26] Grover algorithm

Después, como se dijo en el apartado del algoritmo de Grover, dicho operador G causa una rotación θ la cual determina el número de elementos que cumplen $f(x) = 1$, por lo cual interesa obtener este autovalor del operador G . Para conseguir esto, se utiliza el algoritmo de *Quantum Phase Estimation* que, mediante el control de un determinado número de qubits sobre operadores de G , obtiene la fase correspondiente a la rotación θ .

Por último, con la fase codificada en los qubits de control, se aplica la transformada cuántica inversa de Fourier para que, al medir los qubits de control con la fase θ asociada, sea posible obtener dicho ángulo.

El circuito completo para el algoritmo de *Quantum Counting* se muestra a continuación y está construido con el circuito relativo al algoritmo de estimación de fase, expuesto en la subsección anterior, donde los operadores unitarios controlados son los operadores de Grover.

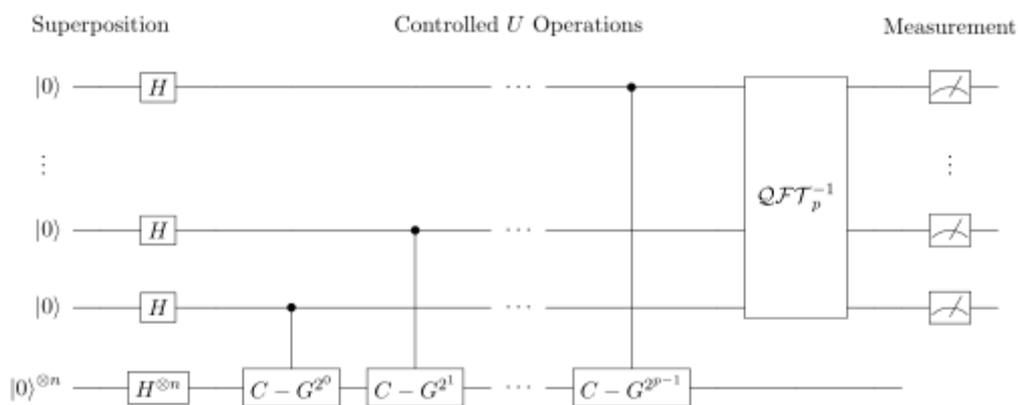


Figura 31. Circuito para el algoritmo de Quantum Counting.
 FUENTE: https://en.wikipedia.org/wiki/Quantum_counting_algorithm

Tal y como se ha detallado en la subsección de *Quantum Phase Estimation*, el número de qubits utilizados para la estimación del ángulo θ son los que determinan la precisión o resolución del número decimal binario obtenido en el resultado final.

Con p qubits, se tiene una resolución de 2^{-p} en el intervalo de $[0,1)$ o, lo que es lo mismo, dicho intervalo se divide en 2^p partes iguales, donde cada valor denota un ángulo determinado separado del anterior y siguiente por un valor de 2^{-p} .

Por esta razón, si se tiene un valor de p lo suficientemente grande para tener una buena estimación de θ , se podrá contar con exactitud el número de elementos que cumplen $f(x) = 1$. Sin embargo, cuantos más qubits de estimación o de control se añadan al algoritmo, mayor será la complejidad computacional de este pues, en última instancia, se ha de aplicar 2^{p-1} veces el operador G en el circuito cuántico, lo cual aumenta la profundidad del circuito.

Para finalizar el algoritmo y obtener el número de soluciones existentes tal que $f(x) = 1$, se ha de tener en cuenta que, al realizar la medición del registro cuántico de estimación tras aplicar la QFT inversa, los valores que se obtienen con una alta probabilidad son $\theta = |x_1 \dots x_p\rangle$, donde x_1 es el bit decimal más significativo y x_p es el menos significativo. Por ello, para obtener el valor del ángulo real $\theta \in [0,1)$ se ha de dividir el resultado de dicho registro cuántico $|x_1 \dots x_p\rangle$ entre la precisión total, es decir, 2^p .

Con esto y con la ecuación, expuesta en la subsección del algoritmo de Grover, que relaciona el ángulo de Grover θ con el número de soluciones M existentes en el espacio total de N estados, se llega a la siguiente fórmula:

$$\frac{\theta\pi}{2^p} = \sin^{-1} \left(\sqrt{M/N} \right)$$

Utilizando esta ecuación y habiendo estimado la rotación θ por el algoritmo de *Quantum Counting*, ya es posible hallar el número de soluciones M y resolver el problema. Despejando la fórmula anterior se tiene que:

$$M = N \cdot \sin^2 \left(\frac{\theta\pi}{2^p} \right)$$

Por lo que, al finalizar la ejecución del algoritmo cuántico de recuento de soluciones, se miden los registros cuánticos de estimación, se obtiene el valor de θ relativa al operador de Grover que contiene la función objetivo y se calcula M con esta fórmula.

Complejidad computacional

En lo que respecta a la complejidad computacional del *Quantum Counting* por el método de estimación de fase, se ha de analizar cada uno de los procedimientos que componen el algoritmo y sus correspondientes complejidades en tiempo de ejecución.

Para ello, se tiene en cuenta la profundidad o *depth* del circuito cuántico que forma el operador en cuestión y se relaciona dicha profundidad con el número de ciclos de aplicación de puertas cuánticas necesario, expresando la complejidad con la notación *big O* en base a dicha profundidad de puertas.

Algo que cabe destacar es que se podrían separar las puertas cuánticas de un qubit y las de dos qubits puesto que suponen un tiempo de aplicación distinto, sin embargo, en el análisis siguiente dichas puertas se consideran iguales.

Para el circuito del *Quantum Counting*, se tiene un circuito de estimación de fase compuesto por operadores de Grover controlados y por una transformada cuántica de Fourier de p qubits.

En primer lugar, se tiene el operador de Grover G , el cual se descompone en otros operadores más sencillos de la siguiente manera:

$$G = H^{\otimes n} U_{0\perp} H^{\otimes n} U_f$$

Aquí, se pueden distinguir n operaciones H relativas a la transformación de Hadamard en n qubits lo cual tiene una profundidad de 1 puerta, puesto que las n puertas H se aplican en paralelo.

Después, se tiene el operador $U_{0\perp}$ que tiene una profundidad total de 3 puertas y una de ellas es de control de n qubits sobre el qubit bandera, mencionado en la sección del algoritmo de Grover. Este operador, al utilizar una puerta multicontrol, incrementan la profundidad dado los registros auxiliares que se requieren para aplicarla, causando que una de las 3 puertas contabilizadas aumente la profundidad en un orden de n puertas.

Por último, está el operador U_f que es el operador oráculo correspondiente a la función objetivo que se va a evaluar. La profundidad de este operador es dependiente de la complejidad de la función objetivo pues se ha de construir un circuito cuántico con las transformaciones necesarias para implementar las operaciones que realiza dicha función.

Por lo tanto, la profundidad total de G se corresponde con:

$$Depth(G) = n + 2 + Depth(Uf)$$

En segundo lugar, se encuentra el número de operadores de Grover controlados que se han de ejecutar en el circuito. Este número está determinado por la cantidad de qubits de control o de estimación que se utilizan para obtener con precisión el parámetro θ del operador de Grover.

A nivel general, el número de operadores de Grover de los que se compone el circuito, según el algoritmo de estimación de fase, se calcula de la siguiente manera, teniendo en cuenta que se utilizan p qubits de precisión para la estimación:

$$Cantidad\ de\ operadores\ G = \sum_{n=0}^{p-1} 2^n \in O(2^p)$$

Al ser una suma, para el orden de complejidad computacional basta con quedarse con el término más grande que, en este caso, sería un orden de 2^{p-1} operadores G .

En tercer y último lugar, está el circuito cuántico que representa la transformada cuántica de Fourier o QFT . Este circuito también depende del número de qubits de precisión utilizados pues está compuesto de puertas H y de puertas de rotación controladas, las cuales se han de aplicar con un orden específico.

No obstante, este orden tiene pocas restricciones y es posible conseguir un paralelismo en las operaciones sobre los p qubits, llegando a tener una profundidad en el circuito de:

$$Depth(QFT) = 2 + p$$

Por lo tanto, la profundidad total del algoritmo de *Quantum Counting* que denota su complejidad es la siguiente, donde p es el número de qubits de estimación y n es el número de qubits utilizado para representar el espacio de estados del problema:

$$Depth(QC) = (2 + p) + (n + 2 + Depth(Uf)) \cdot \sum_{n=0}^{p-1} 2^n \in O((n + Depth(Uf)) \cdot 2^p)$$

Complejidad en memoria

En cuanto a la complejidad en memoria del *Quantum Counting*, se distinguen dos registros cuánticos principales: el registro relativo a la estimación de fase y el que simula el espacio de estados con los que se evalúa la función objetivo.

Primero, se define el espacio de estados con los que interesa analizar la función objetivo y obtener la cantidad de estados solución existente en dicho espacio. Para ello, es necesario conocer con qué rango de valores se va a evaluar la función para definir el espacio de estados acorde a este rango.

Si se quiere que la función sea evaluada con cadenas binarias de longitud n , el registro cuántico que representa el espacio de estados tendrá $n+1$ qubits, pues se ha de añadir el qubit bandera para aplicar el operador de Grover, además de algunos qubits auxiliares temporales para realizar alguna operación relativa a la función objetivo.

Después, para la estimación del autovalor θ , se tiene un registro cuántico de tantos qubits como precisión se requiera en la estimación.

Como se ha comentado en las subsecciones previas, el número de qubits p , referente a la precisión necesaria para hallar la cantidad de soluciones existentes en $f(x)$, depende del tamaño del espacio de estados, es decir, de n .

Esto se debe a que, si en cadenas binarias de tamaño n solo existe una única cadena tal que $f(x) = 1$, entonces se ha de disponer suficiente precisión como para que el algoritmo pueda detectar que existe dicha solución. Por ello, la precisión adecuada sería justamente la dimensión del espacio de estados n , sin embargo, la precisión exacta que se necesita para estimar correctamente el número de soluciones M es:

$$p = \frac{2^n}{M}$$

Por lo tanto, la anchura del circuito relativa a la complejidad en memoria del algoritmo es la siguiente:

$$Width(QC) = p \cdot (n + 1) \in O(p \cdot n)$$

Simulación del algoritmo en Q#

En esta sección, se comenta la implementación y los resultados del algoritmo de *Quantum Counting* con el lenguaje de programación cuántico Q#, de *Microsoft Quantum Development Kit*.

Utilizando las herramientas que ofrece Q#, se ha programado el circuito relativo al *Quantum Counting* clásico explicado anteriormente. Dicha implementación se ha generalizado para permitir evaluar varias funciones objetivo $f(x)$, parametrizando el oráculo que implementa dicha función, la dimensión del espacio de estados n y el número de qubits de estimación p .

Primero, se han inicializado los qubits que evalúan la función $f(x)$ y los qubits referentes a la estimación de fase, cada uno con una superposición uniforme $|+\rangle$, excepto el qubit *flag* inicializado a $|-\rangle$.

Además, los registros cuánticos, tanto el del espacio de estados como el de estimación de fase, se han representado como números enteros en lugar de una cadena binaria para facilitar la aplicación de operadores y simplificar la obtención de resultados. Esto, se ha realizado con el tipo *LittleEndian* definido en el paquete *Microsoft.Quantum.Arithmetic*, el cual recibe un vector de qubits y transforma en un entero la representación binaria en *Little Endian* de los autoestados de los registros.

El operador de Grover ha sido implementado con los respectivos operadores unitarios que lo definen, donde el único operador que puede variar, como se ha dicho, es el oráculo U_f que representa la función objetivo.

$$G = H^{\otimes n} U_{0\perp} H^{\otimes n} U_f$$

El número de operadores controlados de Grover (G) es dependiente del parámetro p , puesto que se comienza aplicando un operador *Controlled* – G y se finaliza con 2^{p-1} operadores $C - G$.

Posteriormente, se utiliza el método de *QFTLE* de la librería *Microsoft.Quantum.Canon* para aplicar la transformada inversa de Fourier al registro cuántico en *Little endian* que estima el ángulo de Grover.

Por último, tras todo el proceso anterior, se mide el registro referente a la estimación de la fase de Grover como un número entero, se resetean los registros cuánticos a $|0\rangle$ y, con los resultados obtenidos, se aplica la ecuación relativa al cálculo del número de soluciones M , donde θ es el valor medido del registro de estimación, N es el número total de estados (2^n) y p es el número de qubits que forman el registro de estimación:

$$M = N \cdot \sin^2\left(\frac{\theta\pi}{2^p}\right)$$

Este proceso se repite iterativamente con la intención de agrupar varias muestras y poder observar la certeza en las soluciones obtenidas.

A continuación, se muestran los resultados de distintas pruebas del algoritmo con 100 ejecuciones por cada prueba, variando la función oráculo, el tamaño del espacio de estados n y el número de qubits de precisión p . Se conoce previamente el número de soluciones de dichas pruebas para comprobar el funcionamiento del algoritmo. Es importante mencionar que las funciones de las pruebas son booleanas, por lo que las operaciones que aparecen se corresponden con funciones lógicas: *AND* representado como un producto “.”, *OR* como una suma “+” y *XOR* como una suma exclusiva “ \oplus ”.

Para la primera simulación, se tienen los siguientes parámetros y su respectiva salida:

- $n = 7 \rightarrow x \in \{0,1\}^7; N = 2^7 = 128$
- $p = 6; p = 5; p = 4$
- $f(x) = \bar{x}_6 \cdot x_5 \cdot x_4 \cdot x_3 \cdot x_2 \cdot x_1 \cdot x_0 \rightarrow M = 1$ solución

```

With precision p = 6 and state space n = 7:
Number of solutions M = 0; Times obtained: 7/100
Number of solutions M = 1; Times obtained: 88/100
Number of solutions M = 3; Times obtained: 2/100
Number of solutions M = 8; Times obtained: 1/100
Number of solutions M = 11; Times obtained: 1/100
Number of solutions M = 23; Times obtained: 1/100

With precision p = 5 and state space n = 7:
Number of solutions M = 0; Times obtained: 1/100
Number of solutions M = 1; Times obtained: 97/100
Number of solutions M = 11; Times obtained: 1/100
Number of solutions M = 117; Times obtained: 1/100

With precision p = 4 and state space n = 7:
Number of solutions M = 0; Times obtained: 51/100
Number of solutions M = 5; Times obtained: 35/100
Number of solutions M = 19; Times obtained: 6/100
Number of solutions M = 40; Times obtained: 5/100
Number of solutions M = 64; Times obtained: 1/100
Number of solutions M = 88; Times obtained: 1/100
Number of solutions M = 109; Times obtained: 1/100

```

Figura 32. Salida de la ejecución del Quantum Counting programado en Q# con $N = 128$, $M = 1$, $p = \{6,5,4\}$.

Como se observa, las salidas obtienen con una probabilidad alta el valor de M correcto para la función oráculo definida en la parte superior cuando se tienen 6 o 5 qubits de precisión, sin embargo, con 4 qubits la precisión no es suficiente como para detectar el ángulo de Grover

referente a $M = 1$ en un espacio de $N = 128$, por lo que, para $p = 4$, la salida del programa es errónea y las probabilidades se condensan en los dos primeros valores de M representables por 4 qubits dado el espacio de estados N : $M = 0$ y $M = 5$.

En la siguiente simulación, se tienen los siguientes parámetros con una función booleana algo más elaborada y la correspondiente salida:

- $n = 4 \rightarrow x \in \{0,1\}^4; N = 2^4 = 16$
- $p = 7; p = 6; p = 5$
- $f(x) = x_0 \cdot x_3 + (x_1 \cdot x_2 \oplus x_0 \cdot x_1) \rightarrow M = 7 \text{ soluciones}$

```

With precision p = 5 and state space n = 4:
Number of solutions M = 0; Times obtained: 1/100
Number of solutions M = 1; Times obtained: 1/100
Number of solutions M = 2; Times obtained: 1/100
Number of solutions M = 4; Times obtained: 1/100
Number of solutions M = 5; Times obtained: 8/100
Number of solutions M = 6; Times obtained: 61/100
Number of solutions M = 8; Times obtained: 23/100
Number of solutions M = 10; Times obtained: 1/100
Number of solutions M = 11; Times obtained: 1/100
Number of solutions M = 16; Times obtained: 2/100

With precision p = 6 and state space n = 4:
Number of solutions M = 0; Times obtained: 1/100
Number of solutions M = 4; Times obtained: 1/100
Number of solutions M = 5; Times obtained: 1/100
Number of solutions M = 6; Times obtained: 14/100
Number of solutions M = 7; Times obtained: 72/100
Number of solutions M = 8; Times obtained: 7/100
Number of solutions M = 9; Times obtained: 1/100
Number of solutions M = 10; Times obtained: 1/100
Number of solutions M = 14; Times obtained: 1/100
Number of solutions M = 15; Times obtained: 1/100

With precision p = 7 and state space n = 4:
Number of solutions M = 4; Times obtained: 1/100
Number of solutions M = 5; Times obtained: 1/100
Number of solutions M = 6; Times obtained: 6/100
Number of solutions M = 7; Times obtained: 85/100
Number of solutions M = 8; Times obtained: 6/100
Number of solutions M = 13; Times obtained: 1/100

```

Figura 33. Salida de la ejecución del Quantum Counting con $N = 16$, $M = 7$, $p = \{7,6,5\}$.

Con estos resultados, se puede decir que, para un factor de 7/16 soluciones, 5 qubits de estimación no son suficientes para determinar el número de soluciones exactas para la función objetivo anterior. En su lugar, con $p = 5$, se obtienen con una probabilidad alta un número de soluciones de $M = 6$ y $M = 8$, exactamente los dos valores entre los que se encuentra el M correcto.

Al realizar la simulación del programa con $p = 6$, se aprecia una probabilidad de 72/100 de obtener la solución correcta, mientras que con $p = 7$, dicha probabilidad aumenta hasta 85/100. La mayor parte de la probabilidad restante se encuentra concentrada en los valores contiguos a la solución correcta, tal y como se puede ver en la figura.

Simulación del algoritmo con Qiskit

Para la ejecución del algoritmo de *Quantum Counting* clásico con *qiskit* en *Python*, *IBM* ofrece el código de dicho algoritmo directamente programado en [55], por lo que se utilizará dicho programa ya que, tras analizar el código, el procedimiento es exactamente el mismo que el programado en *Q#*.

La función oráculo de prueba que viene codificada en el algoritmo de *qiskit* se caracteriza por ser una función con 5 de 16 soluciones. Los qubits de precisión utilizados para la estimación del ángulo θ de Grover son 4, es decir, la transformada cuántica de Fourier se aplica a 4 qubits para obtener dicho ángulo.

Al ejecutar el programa cuántico en el simulador de *Qasm*, el resultado obtenido concuerda con el número de soluciones correcto y es el siguiente, con su respectivo error:

$$\text{Theta} = 1.96350$$

$$\text{No. of Solutions} = 4.9$$

$$\text{Error} < 2.85$$

Sin embargo, al ejecutar el mismo algoritmo con 3 qubits de precisión en lugar de 4, la estimación del número de soluciones no es correcta dado que con 3 cifras decimales binarias no es posible estimar el θ correspondiente a 5/16 soluciones, obteniendo los siguientes resultados con un error elevado en la estimación:

$$\text{Theta} = 4.71239$$

$$\text{No. of Solutions} = 8.0$$

$$\text{Error} < 6.00$$

Por lo que se puede ver, el número de qubits de estimación es muy dependiente de la proporción de soluciones existente en el espacio de estados de la función objetivo.

En el caso de que los computadores cuánticos reales dispusieran de una gran cantidad de qubits y de que el error de estos fuese ínfimo, se podría disponer de un número suficiente de qubits de precisión para abarcar cualquier función objetivo independientemente de la proporción de soluciones de esta.

Cuando se trata de simuladores cuánticos, se puede establecer una cantidad de qubits de precisión suficientemente grande como para obtener el parámetro buscado con una buena estimación. No obstante, hoy en día no es posible proceder de esta manera en una máquina cuántica, por lo que se ha de ajustar al máximo el uso de qubits de estimación y eso genera incertidumbre debido a que el problema a resolver es justamente la obtención de dicha proporción de soluciones de la función objetivo.

Ejecución en el computador cuántico de IBM

En esta sección, se explican los resultados de la ejecución del algoritmo *Quantum Counting* clásico proporcionado por *IBM*, en la referencia mencionada en la sección anterior en la que se simulaba dicho algoritmo.

A raíz de lo discutido en el apartado anterior, al intentar ejecutar el algoritmo con 4, 3 y 2 qubits de estimación en la máquina cuántica *IBMQ Melbourne* de 16 qubits, el programa devuelve un error de exceso de profundidad del circuito cuántico:

'Job has failed: Circuit runtime is greater than the device repetition rate. Error code: 8020.'

Lo cual quiere decir que el tiempo de ejecución del circuito *Quantum Counting* clásico con 4, 3 e incluso 2 qubits de precisión excede el tiempo de decoherencia de la máquina cuántica, a pesar de haber reducido los qubits de estimación al máximo.

Esto supone que el algoritmo clásico de la cuenta cuántica de soluciones no es viable para los ordenadores cuánticos de la actualidad.

Por esta razón, surge la motivación de desarrollar un algoritmo de *Quantum Counting* simplificado con la intención de reducir el número de puertas controladas del circuito y la profundidad de este, tratando de obviar el uso del *Quantum Phase Estimation* para estimar el número de soluciones de una determinada función objetivo.

En la sección siguiente, se estudia otro método que cumpla los requisitos mencionados para simplificar el algoritmo y permitir su ejecución de una forma más eficiente y sencilla.

7. Simpler Quantum Counting

A pesar de que el *Quantum Counting* clásico explicado en el apartado anterior reduzca la complejidad computacional frente a algoritmos clásicos que abordan el mismo problema, no se puede implementar actualmente para espacios de estados muy grandes dado que no es posible mantener la coherencia cuántica en las máquinas cuánticas *NISQ* de hoy en día el tiempo suficiente como para realizar el algoritmo de *Quantum Phase Estimation (QPE)*.

Esto se debe a que el algoritmo de *QPE* requiere un número elevado de operadores de Grover controlados aplicados secuencialmente, lo cual aumenta exponencialmente la profundidad del circuito cuántico según se añaden qubits de estimación o precisión, además de tener que aplicar posteriormente la transformada cuántica de Fourier. La consecuencia de esto es que, al utilizar una gran cantidad de puertas cuánticas controladas y aumentar la profundidad del circuito que implementa el algoritmo, las tasas de error en los qubits incrementan considerablemente dando lugar a medidas falseadas con una probabilidad muy alta, como se ha observado y discutido en apartados anteriores.

En su lugar, existen otros algoritmos que no utilizan la estimación de fase clásica para hallar la solución al problema y, en general, se tratan de métodos estadísticos de estimación de amplitudes. La clave de estos algoritmos es que utilizan un menor número de operadores controlados por qubits y la profundidad del circuito resultante se ve reducida a grandes rasgos, lo cual es el principal problema para los computadores *NISQ*.

La inspiración para el desarrollo del algoritmo de *Simpler Quantum Counting* son de fuentes en las que se expone un método de estimación de amplitudes que no utiliza la estimación de fase cuántica basada en la *QFT*. Las principales fuentes de información para la implementación del algoritmo simplificado son [56, 57].

El desarrollo y la implementación del *Simpler Quantum Counting* que se expone en este apartado se trata de un algoritmo cuántico donde los resultados obtenidos tras realizar la computación cuántica tienen un tratamiento clásico posterior que manipula dichos resultados transformándolos en la solución al problema abordado: contabilizar el número de soluciones de una función booleana.

Como se dijo con antelación, el algoritmo de Grover es un proceso de amplificación de amplitudes acorde a una función objetivo codificada como un operador cuántico. Gracias a este proceso, es posible amplificar la amplitud de probabilidad de los estados solución de una función y, por tanto, contar el número de soluciones existentes en esta para resolver el problema del *Quantum Counting*.

En el algoritmo de *Quantum Counting* clásico, los operadores de Grover G se aplican de manera controlada por los qubits de estimación, necesitando 2^{p-1} operadores G controlados por el qubit de estimación en la posición p aplicados secuencialmente. Dicho operador se ha definido anteriormente como:

$$G = H^{\otimes n} U_{0\perp} H^{\otimes n} U_f = U_{\psi\perp} U_f$$

Para evadir la elevada cantidad de operadores controlados G que requiere la estimación de la fase tradicional, se propone otro proceso de estimación de la fase θ que, como se ha demostrado anteriormente, es el parámetro necesario para hallar el número de soluciones y en el que se basa el algoritmo de Grover de amplificación de amplitudes. Para recordar, dicho parámetro definía la ecuación siguiente con la que se encuentra el número de soluciones M :

$$M = N \cdot \sin^2(\theta)$$

En el apartado donde se describe el algoritmo de Grover, se detalla el proceso mediante el cual las amplitudes de los estados solución son amplificadas en cada iteración de G . Teniendo la función de onda $|\psi\rangle$, mostrada a continuación, como superposición uniforme de todo el espacio de estados, las amplitudes resultantes de dicha función de onda tras aplicar k veces G son:

$$|\psi\rangle = \sqrt{\frac{M}{N}} |\psi_{CORRECTO}\rangle + \sqrt{\frac{N-M}{N}} |\psi_{INCORRECTO}\rangle$$

$$G^k |\psi\rangle = \sin((2k+1)\theta) |\psi_{CORRECTO}\rangle + \cos((2k+1)\theta) |\psi_{INCORRECTO}\rangle$$

Como se puede observar, las amplitudes de los estados se ven modificadas en función de k y de θ , donde el parámetro que se desea estimar es este último dado que k hace referencia al número de veces que se aplica el operador G , por lo que es un parámetro conocido.

Esto permite estimar dicho parámetro θ variando el número de iteraciones (k) de G y muestreando de forma paralela las funciones de onda resultantes, una para cada k . El número de muestras o medidas que se realizan para cada k se denota como N_k .

El método consiste en construir un circuito cuántico de $(n+1) \cdot K$ qubits, es decir, replicar K veces el espacio de estados n de la función objetivo (junto con el qubit bandera), y aplicar los operadores G a cada uno de esos K circuitos modificando el parámetro k . Al realizar esto, se colapsan las funciones de onda y se comprueba si el estado colapsado forma parte del espacio de soluciones. Si este proceso se repite N_k veces, se obtiene el número de veces que ha colapsado un estado correcto en esas N_k repeticiones, con lo que se puede aproximar la probabilidad de obtener cualquiera de los estados solución. Dicha probabilidad está representada por la amplitud de los estados correctos vista en la función de onda anterior: $\sin((2k+1)\theta)$.

Por ejemplo, si el número de subcircuitos es $K=3$ y la función objetivo $f(x)$ tiene como entrada $x \in \{0,1\}^n$, para cada uno de ellos se aplica el operador G un determinado número de veces con su respectiva función oráculo $f(x)$. Sean las iteraciones de Grover $k=1, 2$ y 3 , se tienen las tres funciones de onda siguientes con las amplitudes amplificadas de los estados correctos:

$$k=1 \rightarrow G^1 |\psi\rangle = \sin(3\theta) |\psi_{CORRECTO}\rangle + \cos(3\theta) |\psi_{INCORRECTO}\rangle$$

$$k=2 \rightarrow G^2 |\psi\rangle = \sin(5\theta) |\psi_{CORRECTO}\rangle + \cos(5\theta) |\psi_{INCORRECTO}\rangle$$

$$k=3 \rightarrow G^3 |\psi\rangle = \sin(7\theta) |\psi_{CORRECTO}\rangle + \cos(7\theta) |\psi_{INCORRECTO}\rangle$$

Ahora, al medir las tres funciones de onda se obtiene un estado correcto con una probabilidad igual al cuadrado de la amplitud correspondiente a los estados correctos. Suponiendo que el número de repeticiones de este proceso es N_k , al acabar cada repetición se realiza una medición y se comprueba si el estado colapsado forma parte del espacio de soluciones.

Al final de estas N_k muestras, se tiene que una cantidad de muestras medidas igual a M_k han sido estados solución, con lo que se puede aproximar la probabilidad de obtener un estado

correcto. Dicha probabilidad se corresponde con el cuadrado de la amplitud de los estados correctos para un k concreto que, de manera formal, da lugar a la siguiente ecuación, semejante a la fórmula para hallar el número de soluciones expuesta con antelación:

$$\frac{M_k}{N_k} = \sin^2((2k + 1)\theta)$$

De esta ecuación, se despeja el parámetro θ :

$$\theta = \frac{\arcsin\left(\sqrt{M_k/N_k}\right)}{2k + 1}$$

Con todos los θ calculados para cada k , se puede establecer un proceso estadístico que permite obtener una estimación de θ más precisa teniendo en cuenta el factor por el que se multiplica en cada circuito K .

Para calcular cada θ , se ha de tener en cuenta que el algoritmo de Grover se caracteriza por ser un proceso periódico de modificación de amplitudes, tal y como se puede deducir de las funciones de onda. Esto es, la amplitud de los estados solución se ve modificada en función de $\sin((2k + 1)\theta)$, la cual es una función periódica, por lo que $\exists k$ tal que $(2k + 1)\theta > \pi/2$.

Por ello, es importante recalcar que, al utilizar el arco seno para hallar θ , esta función siempre devuelve un ángulo perteneciente al intervalo $[-\pi/2, \pi/2]$, es decir, al primer y tercer cuadrante trigonométrico. Además, la raíz cuadrada a la que se aplica el arco seno se va a considerar siempre positiva, por lo que el intervalo final queda reducido al primer cuadrante: $[0, \pi/2]$.

Sin embargo, al aplicar k veces el operador de Grover, el ángulo θ puede ser cualquiera en el intervalo $[0, 2\pi)$ que define la circunferencia completa. Esto es crucial ya que, a causa del factor de división $2k + 1$ que se observa en la ecuación anterior, el resultado del arco seno se divide por dicho factor y, por ello, se ha de conocer el ángulo real en referencia a la circunferencia completa y no solo al primer cuadrante. Aún más allá, si los k a evaluar son lo suficientemente grandes como para que el ángulo $(2k + 1)\theta$ supere una vuelta completa, el intervalo $[0, 2\pi)$ no basta para despejar correctamente el parámetro θ .

En resumen, es necesario transformar el ángulo obtenido con el arco seno, perteneciente al intervalo $[0, \pi/2]$, en el ángulo correspondiente a la rotación total que suponen k iteraciones de Grover, incluyendo el número de vueltas completas de dicha rotación en caso de que las hubiera.

Para esto, se parte de la suposición de que la proporción de soluciones de la función objetivo ha de cumplir que $M/N \leq 1/4$ con el objetivo de que una única aplicación de G , es decir, para $k = 1$, dé como resultado un ángulo perteneciente a $[0, \pi/2]$, consiguiendo así que no haya que hacer una transformación de dicho ángulo al aplicar el arco seno para despejar el valor de θ relativo al subcircuito $k = 1$.

Esto se demuestra de la siguiente manera: dado que, para $k = 1$, la amplitud relativa a los estados correctos se corresponde con $\sin(3\theta)$, es necesario que se cumpla $3\theta \leq \frac{\pi}{2}$ para que la rotación total pertenezca al intervalo $[0, \pi/2]$ y no se tenga que recalculer el valor del ángulo real tras aplicar la función del arco seno, lo cual implica que $\theta \leq \frac{\pi}{6}$, por lo tanto, utilizando la

ecuación que despeja el número de soluciones M , se tiene la siguiente restricción en la proporción de soluciones del espacio de estados:

$$\sqrt{M/N} = \sin(\theta)$$

$$\sqrt{M/N} \leq \sin(\pi/6) = 1/2$$

$$M/N \leq 1/4$$

Después de esta suposición, tras haber conseguido el primer valor aproximado de θ , el resto de valores se pueden hallar haciendo uso de dicho θ ya que este valor permite conocer, con una buena aproximación, el número de vueltas y el cuadrante en el que cae cada uno de los ángulos restantes obtenidos por el arco seno según el número de iteraciones k , obteniendo, finalmente, la rotación total de cada uno de los K circuitos que, a su vez, ofrece el valor estimado de θ para cada uno de ellos.

Es de relevancia mencionar que, en el caso de que no se cumpla dicha restricción en la proporción de soluciones, también es posible determinar la rotación total del primer valor de θ a partir del arco seno estableciendo que dicha rotación cae en el segundo cuadrante pudiendo, así, utilizar dicho valor de θ junto con k para estimar el resto de rotaciones. No obstante, esto no suele ser usual ya que la restricción anterior se cumple prácticamente para cualquier función objetivo que haga referencia a un problema real de contabilización de soluciones.

A continuación, se muestra el circuito cuántico referente al algoritmo del *Simpler Quantum Counting* donde, como se ha dicho hasta ahora, k es el número de operadores de Grover G aplicados secuencialmente, $n+1$ es el número de qubits necesarios para representar el espacio de estados de la función objetivo contabilizando el qubit bandera, y las puertas H y Z son las transformaciones correspondientes a la inicialización del algoritmo de Grover.

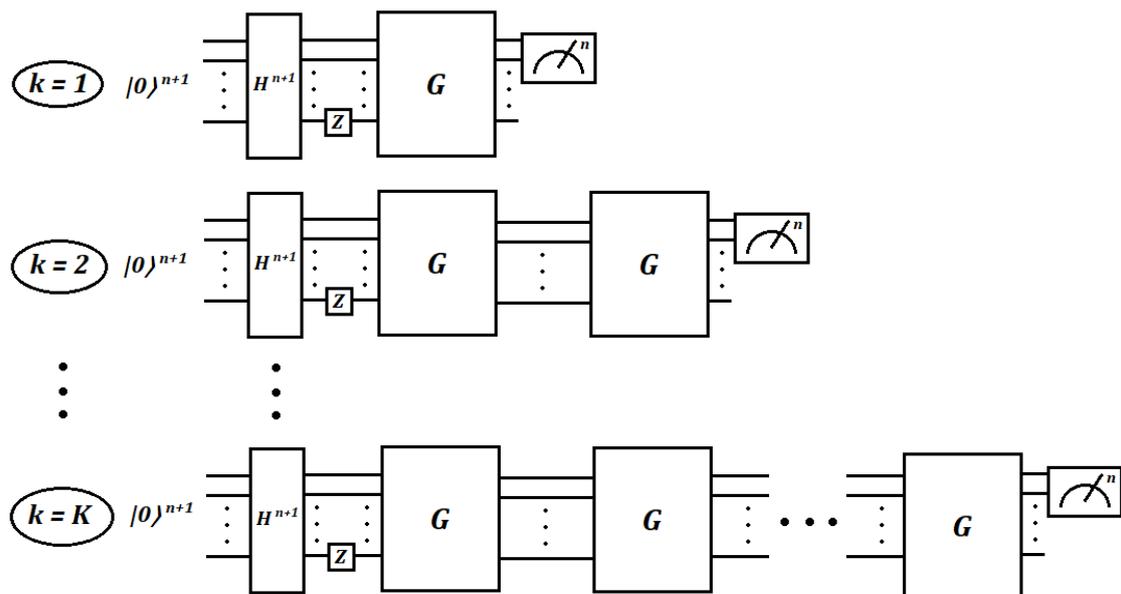


Figura 34. Circuito cuántico del *Simpler Quantum Counting*.
FUENTE: Diseño propio.

Si bien se necesita una precisión más grande del primer valor de θ , en el caso de que la proporción de soluciones en el espacio de estados sea relativamente pequeña, el número de operadores de Grover aplicados secuencialmente puede escalar de forma exponencial (2^k) en lugar de lineal (k), causando una mayor amplificación de amplitudes de los estados solución y permitiendo una estimación más precisa.

Además, el circuito se puede optimizar tal que se pueda aprovechar al máximo el paralelismo cuántico en cuanto a la aplicación de operadores G , lo cual se consigue replicando, por ejemplo, el circuito de $k = 1$ un número de veces proporcional al circuito de profundidad máxima $k = K$. Con ello, se pueden obtener más muestras de $k = 1$, en este ejemplo, y lograr una mayor precisión en la estimación de θ del circuito correspondiente a dicho k .

Teniendo todo lo anterior en cuenta, el algoritmo del *Simpler Quantum Counting* sigue el siguiente proceso:

Dada una función objetivo $f(x)$, codificada como un circuito cuántico dentro del operador de Grover G (función oráculo), cuya entrada $x \in \{0,1\}^n$ es una cadena binaria de longitud n , se tiene que $N = 2^n$ es el número total de estados de entrada de $f(x)$ y M es la cantidad de dichos estados que satisfacen $f(x) = 1$ y, además, se cumple que $M/N \leq 1/4$.

Se establece un K referente al número de réplicas deseado del espacio de estados N , un N_k que denota el número de muestras o mediciones de cada circuito replicado, un M_k para contabilizar el número de soluciones obtenidas de cada circuito en N_k muestras y una escala de las iteraciones de Grover k a realizar en cada réplica, en este caso la escala es lineal (como se aprecia en la figura anterior). Aquí, el número de muestras N_k es igual para todos los subcircuitos, por lo que se definirá como L .

Entonces:

1. Para L repeticiones en cada subcircuito k :
 - Inicializar el registro de $n+1$ qubits al estado $|0\rangle^{n+1}$.
 - Aplicar $H^{\otimes n+1}$ al registro $|0\rangle^{n+1}$ y Z al qubit bandera para conseguir el estado:

$$|\psi\rangle = \frac{1}{\sqrt{N}} \sum_{x=0}^{N-1} |x\rangle \left(\frac{|0\rangle - |1\rangle}{\sqrt{2}} \right)$$

- Aplicar k veces el operador de Grover: $G^k |\psi\rangle$
 - Medir $G^k |\psi\rangle$ excluyendo el qubit bandera, comprobar si el estado colapsado es una de las soluciones y, en caso afirmativo, incrementar el contador M_k .
2. Con el subcircuito $k = 1$, hallar la primera estimación de θ con el resultado M_1 , sabiendo que no es necesario rectificar el valor del arco seno:

$$\theta_1 = \frac{\arcsin\left(\sqrt{M_1/L}\right)}{3}$$

3. Calcular las rotaciones del resto de subcircuitos con sus respectivos k utilizando θ_1 para rectificar el ángulo obtenido por el arco seno:

$$\theta_k \approx (2k + 1)\theta_1 \rightarrow \text{Número de vueltas y cuadrante de } \theta_k$$

4. Hallar las estimaciones de los θ restantes con sus correspondientes M_k rectificando el ángulo obtenido por la función arco seno tal que se calcule la rotación completa:

$$\theta_k = \frac{\text{RECTIFICAR} \left(\text{arc sin} \left(\sqrt{M_k/L} \right) \right)}{(2k + 1)}$$

5. Aplicar un proceso estadístico a todos los θ_k para obtener una mejor estimación de θ . En este caso, simplemente se utiliza la media aritmética:

$$\theta = \frac{1}{K} \sum_{k=1}^K \theta_k$$

6. Por último, utilizar la ecuación siguiente para obtener el número de soluciones M :

$$M = N \cdot \sin^2(\theta)$$

Complejidad computacional y en memoria

En lo que respecta a la complejidad en tiempo de ejecución, al igual que en el *Quantum Counting* clásico, esta se mide mediante la notación *big O* en función de la profundidad del circuito.

Como se observa en el circuito del *Simpler Quantum Counting*, la complejidad depende de la profundidad del operador de Grover G y del número máximo de aplicaciones secuenciales K de dicho operador.

Dado esto y según el análisis realizado en el *Quantum Counting* clásico, el operador de Grover tiene la siguiente profundidad:

$$\text{Depth}(G) = n + 2 + \text{Depth}(Uf)$$

Y, al aplicar un máximo de K operadores G secuenciales en uno de los subcircuitos utilizados, la profundidad del algoritmo *SQC* queda como:

$$\text{Depth}(SQC) = K \cdot (n + 2 + \text{Depth}(Uf)) \in O \left(K(n + \text{Depth}(Uf)) \right)$$

En comparación con la complejidad del algoritmo clásico que utiliza la *QFT*, esta se ve reducida debido al factor K que se reemplaza por el factor 2^p relativo al número de qubits de estimación en el *Quantum Counting* clásico.

En cuanto a complejidad en memoria del *Simpler Quantum Counting*, como se ha demostrado en la explicación del algoritmo, al multiplicar el espacio de estados de $n+1$ qubits, contando con su qubit bandera, en K circuitos distintos para realizar la estimación, la complejidad en memoria es la siguiente, correspondiente con la anchura del circuito:

$$\text{Width}(SQC) = K \cdot (n + 1) \in O(K \cdot n)$$

Dependiendo del valor que tome K , es decir, del número de subcircuitos de estimación, el algoritmo *SQC* tendrá incluso un número menor de qubits que en la versión clásica en el caso de $K < p$.

Simulación del algoritmo en Q#

Seguidamente, se detallan una serie de pruebas del algoritmo con distintas funciones objetivo y variando los parámetros de N , M y k . Dichas pruebas se han realizado codificando en Q# el programa correspondiente al proceso anterior.

Cabe decir que, aunque no se expongan todas las pruebas realizadas del algoritmo, este se ha testeado con muchos otros parámetros y funciones objetivo, obteniendo salidas correspondientes a una buena estimación del número de soluciones. En la mayoría de los casos, esta estimación coincide con el resultado exacto del número de soluciones al redondear al número entero más próximo.

La primera prueba, al igual que en la versión clásica, se trata de una prueba conceptual con una función objetivo sencilla. Sus características son las siguientes:

- $n = 7 \rightarrow x \in \{0,1\}^7; N = 2^7 = 128$
- $f(x) = x_5 \cdot x_4 \cdot x_3 \cdot x_2 \cdot x_1 \cdot \bar{x}_0 \rightarrow M = 2 \text{ soluciones}$
- Número de muestras $L = 50$, $K = 2$ y número de iteraciones de Grover $k = 1$ y $k = 2$.

```
For m = 1: 7/50  
For m = 2: 16/50  
Theta est: 0,12404258898973505; Estimated number of solutions: 1,9593996997675787
```

Figura 35. Salida de la ejecución del Simplex Quantum Counting con los M_k , la estimación de ϑ y el número estimado de soluciones.

En la ejecución anterior, se obtienen los M_k relativos a $k = 1$ y $k = 2$, es decir, a dos subcircuitos distintos con k iteraciones de Grover. Siguiendo el proceso del algoritmo, al medir 7 de 50 veces un estado correcto con $k = 1$ y 16 de 50 veces con $k = 2$, se obtienen dos valores para la estimación de θ que, al realizar la media entre ambos, resulta el número estimado de soluciones 1,9594 equivalente a la respuesta correcta: 2 soluciones de 128.

A continuación, respetando una de las pruebas llevada a cabo en el *Quantum Counting* clásico, se procede a realizar un testeo del algoritmo simplificado con los mismos parámetros:

- $n = 4 \rightarrow x \in \{0,1\}^4; N = 2^4 = 16$
- $f(x) = x_0 \cdot x_3 + (x_1 \cdot x_2 \oplus x_0 \cdot x_1) \rightarrow M = 7 \text{ soluciones}$
- Número de muestras $L = 10, 20, 100$ y número de iteraciones $k = 1, k = 2$ y $k = 3$.

Dado que estos parámetros no cumplen la restricción de $M/N \leq 1/4$, se ha modificado levemente la función objetivo y se ha incrementado el espacio de estados n , quedando de la siguiente manera:

- $n = 5 \rightarrow x \in \{0,1\}^5; N = 2^5 = 32$
- $f(x) = x_4 \cdot (x_0 \cdot x_3 + (x_1 \cdot x_2 \oplus x_0 \cdot x_1)) \rightarrow M = 7 \text{ soluciones}$

Al duplicar el espacio de estados, que equivale a añadir un qubit, y modificar la función objetivo tal que sea necesario que el valor de x_4 del qubit añadido valga 1, se mantiene el número de soluciones M habiendo incrementado el espacio de estados con el objetivo de cumplir el requisito mencionado: $M/N = 7/32 \leq 1/4$.

```
For m = 1: 99/100  
For m = 2: 45/100  
For m = 3: 11/100  
Theta est: 0,4895197540919183; Estimated number of solutions: 7,074880389884189
```

```

For m = 1: 20/20
For m = 2: 6/20
For m = 3: 4/20
Theta est: 0,5170078938235817; Estimated number of solutions: 7,818048441407784

```

```

For m = 1: 10/10
For m = 2: 7/10
For m = 3: 2/10
Theta est: 0,4895734374190825; Estimated number of solutions: 7,076306205166316

```

Figura 36. Salida del *Simpler Quantum Counting* para un número de muestras $L = 100, 20, 10$, con los M_k , las estimaciones de ϑ y el resultado estimado de soluciones.

Como se aprecia en los resultados obtenidos de la ejecución, a pesar de haber variado el número de muestras reduciéndolas hasta 10 muestras por subcircuito, el número estimado de soluciones para la función $f(x)$ que ofrece el algoritmo de *Simpler Quantum Counting* para cada número de muestras L se corresponde con una buena aproximación al número de soluciones existentes en el espacio de estados.

En el caso de $L = 100$ y $L = 10$, el redondeo de la estimación del número de soluciones al número entero más próximo es el relativo al M correcto, es decir, 7. Mientras que, si se redondea la estimación obtenida para $L = 20$, se obtiene $M = 8$, lo cual se debe a que, en las medidas realizadas en el subcircuito $k = 2$ ($m = 2$ en la figura), los estados colapsados no se corresponden con un estado solución el número de veces referente a su amplitud de probabilidad y, por ende, la estimación de θ para ese subcircuito dista algo más del θ real de la función objetivo, aunque dicha estimación sigue estando cerca del resultado correcto de M .

Simulación del algoritmo con Qiskit

Al igual que en la subsección anterior, se ejecutan pruebas del *Simpler Quantum Counting* programado con *Qiskit* en *Python* utilizando el simulador de *IBM* de *Qasm*.

El hecho de ejecutar las pruebas del algoritmo en simuladores implica que los resultados que se obtienen son los ideales, por lo que simular las mismas pruebas llevadas a cabo en la simulación con *Q#* significa obtener prácticamente los mismos resultados.

Con *qiskit*, las funciones objetivo utilizadas para las pruebas son funciones simples con un espacio de estados pequeño para poder comparar los resultados obtenidos en la simulación con los resultados de la ejecución en la máquina cuántica de *IBM*.

La primera prueba consta de un único subcircuito, es decir, solo se obtiene una única estimación de θ , y se construye con los siguientes parámetros, los cuales cumplen la restricción de $M/N = 1/4 \leq 1/4$:

- $n = 2 \rightarrow x \in \{0,1\}^2; N = 2^2 = 4$
- $f(x) = x_1 \cdot x_0 \rightarrow M = 1$ solución = '11'
- $k = 1$ y $L = 1024$

La salida del programa al enviarlo al simulador de *Qasm* es:

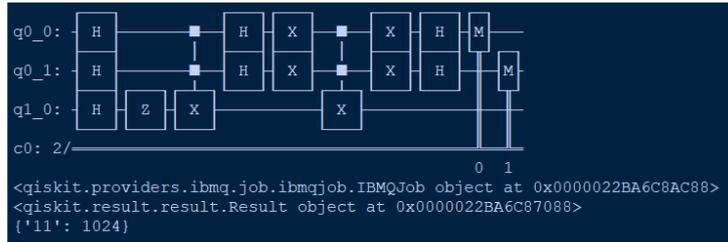


Figura 37. Salida de la simulación del Quantum Counting con $L = 1024$, $k = 1$, donde $M_1 = 1024$.

En el resultado, se obtiene un 100% de las ocasiones el estado '11', es decir, $M_1 = 1024$. Tras realizar el tratamiento clásico con este resultado utilizando las ecuaciones correspondientes al algoritmo para obtener el número de soluciones, se tiene que el número de soluciones obtenido es:

$$\theta = \frac{\arcsin\left(\sqrt{M_1/L}\right)}{2k+1} = \frac{\arcsin\left(\sqrt{1024/1024}\right)}{3} = \frac{\pi/2}{3} = \frac{\pi}{6}$$

$$M = N \cdot \sin^2(\theta) = 4 \cdot \sin^2(\pi/6) = 4 \cdot 0.5^2 = 1 \text{ solución}$$

Otra prueba de simulación es la que se define con los parámetros a continuación:

- $n = 5 \rightarrow x \in \{0,1\}^5; N = 2^5 = 32$
- $f(x) = x_2 \cdot x_1 \cdot x_0 \rightarrow M = 4 \text{ soluciones} = x \in \{??111\}$
- $k = 1$ y $L = 1024$

Donde la salida de dicha prueba es:



Al sumar todas las veces que se ha obtenido una combinación correspondiente a un estado solución, tal y como dicta el algoritmo, se obtiene el M_1 que en este caso es 790. Entonces, igual que antes, el número de soluciones que determinaría el algoritmo es:

$$\theta = \frac{\arcsin\left(\sqrt{790/1024}\right)}{3} = 0'357460779$$

$$M = N \cdot \sin^2(\theta) = 32 \cdot \sin^2(0'357460779) = 3'9177 \text{ soluciones} \sim 4 \text{ soluciones}$$

Por lo tanto, se puede concluir que el algoritmo es efectivo para pruebas sencillas en las que, como se puede ver, no es necesario utilizar más de una estimación de θ . Con lo cual, si se trata de ejecuciones con funciones objetivo con un espacio de estados grande y se utilizan más circuitos estimadores de θ , el algoritmo también ofrecerá resultados con buena aproximación al número de soluciones real.

Ejecución en el computador cuántico de IBM

En cuanto a la ejecución en las máquinas cuánticas de *IBM*, se han utilizado principalmente dos de dichas máquinas: *IBMQ Melbourne*, de 16 qubits, y *IBMQ Santiago*, de 5 qubits.

Como la tasa de errores de estas máquinas es bastante elevada, se ha tratado de reducir la función a analizar, como se dijo con antelación en la primera prueba de simulación con *qiskit*.

Utilizando exactamente los mismos parámetros de dicho test, mostrados nuevamente a continuación, los resultados proporcionados por sendos ordenadores cuánticos son:

- $n = 2 \rightarrow x \in \{0,1\}^2; N = 2^2 = 4$
- $f(x) = x_1 \cdot x_0 \rightarrow M = 1$ solución = '11'
- $k = 1$ y $L = 1024$

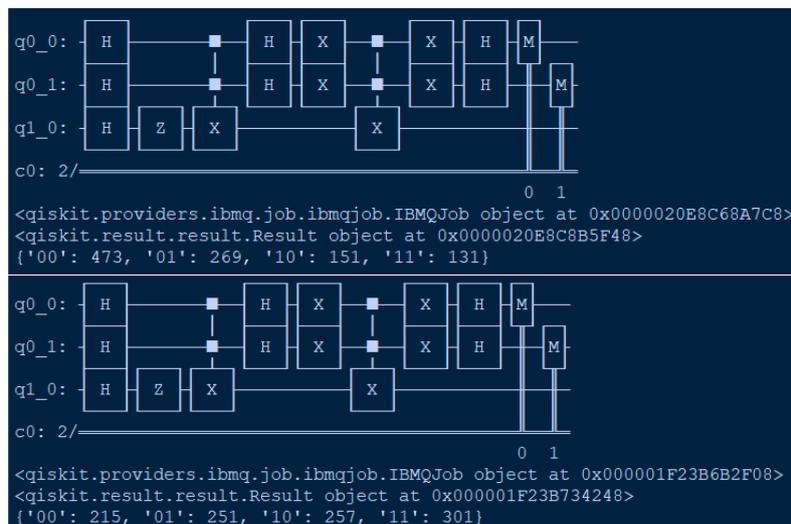


Figura 38. Salidas de las ejecuciones en las máquinas cuánticas de *IBMQ Melbourne*(arriba) e *IBMQ Santiago*(abajo).

Al contrastar los resultados obtenidos por los ordenadores cuánticos físicos con los obtenidos en simulación en el apartado anterior, se puede ver que existe una gran cantidad de ruido a la hora de colapsar el estado cuántico. El resultado de este mismo circuito en la simulación ha sido un 100% el estado '11', mientras que, en la ejecución en máquina real, se puede ver que las tasas de error en los qubits son tales que, en *IBMQ Melbourne*, ni siquiera se obtiene mayoritariamente el estado '11' a diferencia de *IBMQ Santiago*, que es capaz de obtener el estado correcto la mayoría de las iteraciones.

A pesar de que en la ejecución del circuito en *IBMQ Santiago* tenga la mayoría (301 veces) de colapsos del estado correcto, tampoco es útil puesto que el algoritmo propuesto está basado en la probabilidad de colapso de estados correctos según la función de onda resultante tras aplicar ciertas iteraciones de Grover y, dado que dichas probabilidades de colapso se falsean tanto en la máquina real, no es posible obtener una aproximación del número de soluciones para la función objetivo.

Si se dispusiera de un ordenador cuántico con tasas de errores menores, el algoritmo *Simpler Quantum Counting* se podría implementar obteniendo resultados más coherentes y mejor aproximados que resuelvan el problema. No obstante, utilizando un computador cuántico de más capacidad con ruido, se podría modificar el algoritmo incluyendo corrección de errores y obtener resultados viables, aunque el coste en qubits sería mayor, dependiendo del código de corrección de errores que se emplee, con un mínimo del doble de qubits.

Conclusiones

El algoritmo, como se puede ver, carece de puertas controladas más allá de las que se utilizan dentro del operador de Grover, lo cual es una ventaja muy importante a la hora de ejecutar el algoritmo en una máquina cuántica real, como se ha explicado al inicio de la sección.

Adicionalmente, este algoritmo simplificado es muy versátil en cuanto a eficiencia a causa de la capacidad de paralelismo que posee. Además de poder multiplicar el número de muestras en cada subcircuito cuántico k en función de la profundidad del subcircuito k más grande, es posible llevar el paralelismo al punto de poder ejecutar el algoritmo con distintos k en distintas máquinas cuánticas, juntando los resultados en el análisis posterior para la estimación de θ .

La precisión de la estimación de θ para sacar el número existente de soluciones M depende del número de muestras N_k (L en la descripción anterior) de cada subcircuito y del número de subcircuitos K a los que se aplican distintas secuencias de operadores G , sin embargo, al utilizar el algoritmo de Grover para la amplificación de amplitudes, las probabilidades de colapso de los estados solución incrementan y, con un número de muestras no muy elevado, es posible obtener una distribución de probabilidad adecuada para una buena estimación de amplitudes.

A pesar de conseguir estas ventajas, los ordenadores cuánticos utilizados para testear el algoritmo todavía no son lo suficientemente efectivos como para poder obtener resultados satisfactorios. Sin embargo, si se utilizara alguna máquina cuántica con tasas de error lo suficientemente pequeñas, el *Simpler Quantum Counting* sería perfectamente implementable y eficaz en cuanto a una buena estimación se refiere. Además, como se ha demostrado, dicho algoritmo reduce considerablemente la profundidad de los circuitos cuánticos y de los operadores controlados respecto del *Quantum Counting* clásico.

8. Aplicaciones prácticas

Principalmente, la computación cuántica funciona eficientemente en problemas donde los espacios de estados son grandes y se requiere de un número elevado de comprobaciones para resolver el problema.

Este tipo de problemas están normalmente asociados a problemas *NP* o *Non-deterministic Polynomial time* [58], los cuales son problemas de decisión donde es posible comprobar si una decisión específica es correcta en tiempo polinómico, no obstante, a día de hoy no se conoce ningún algoritmo de orden de complejidad polinomial que permita resolver este tipo de problemas. Esto es un problema abierto de gran importancia en matemáticas ya que si $P = NP$ existiría un algoritmo que podría resolver el problema en tiempo polinomial.

Como ejemplo, tomado de [59], un problema *NP* sería el *SAT*, que hace referencia al problema de satisfacer una función booleana pues para una entrada $x \in \{0,1\}^n$ de $f(x)$ se requiere comprobar 2^n combinaciones, mientras que, dada una entrada concreta de dicho conjunto, es posible corroborar en tiempo $O(n)$ si dicha entrada es o no una combinación que satisface $f(x)$, es decir, que cumple $f(x) = 1$. En específico, *SAT* es un problema *NP-Completo*.

Los problemas *NP-Completo* se definen como aquellos problemas de decisión en los que, si un problema $A \in NP$ es reducible a B polinomialmente y B se puede resolver eficientemente, entonces B se puede utilizar para resolver A de forma eficiente. Dicho de otro modo, si es posible resolver B eficientemente entonces cualquier A reducible de forma polinomial a B automáticamente se puede resolver de manera eficiente. Esto implica que este tipo de problemas son los más complejos de resolver en *NP*.

Un ejemplo de problema *NP-Completo*, también en [59], sería A : *detectar ciclos hamiltonianos dado un grafo G y B : problema del viajante (detectar un ciclo hamiltoniano en G con el mínimo coste)*. Aquí, es posible reducir A a B polinomialmente estableciendo que los costes de las aristas son 0 si existe arista entre dos vértices y 1 si no existe arista, con lo que A se resuelve si el coste total es 0, es decir, existe un ciclo hamiltoniano cuando se tiene un coste nulo en el problema del viajante (*TSP*).

El algoritmo de Grover permite resolver algunos tipos de problemas de naturaleza *NP-Complete* mediante comprobaciones en espacios de posibles soluciones que, dada su ventaja cuadrática en complejidad computacional, supone una mejora notable en cuanto a posibles algoritmos clásicos.

Por otro lado, en un nivel mayor de complejidad, se encuentran los problemas *NP-Hard* en los que se cumple que, para todo problema perteneciente al conjunto *NP*, si dichos problemas se pueden reducir a polinomialmente a otro problema, este problema será, como mínimo, igual de difícil que los problemas del conjunto *NP*, pero no pertenece necesariamente a dicho conjunto.

Los problemas de satisfactibilidad booleana *SAT* son problemas *NP* de tipo *hard* puesto que es un problema *NP-Completo* y automáticamente es un problema *NP-hard*, mientras que, el problema del viajante (*TSP*) es un problema *NP-hard* pero no pertenece al conjunto *NP* ya que no se trata de un problema de decisión.

El esquema de los problemas *NP* según el tipo y su orden de complejidad se puede observar en el esquema siguiente, llamado diagrama de Euler.

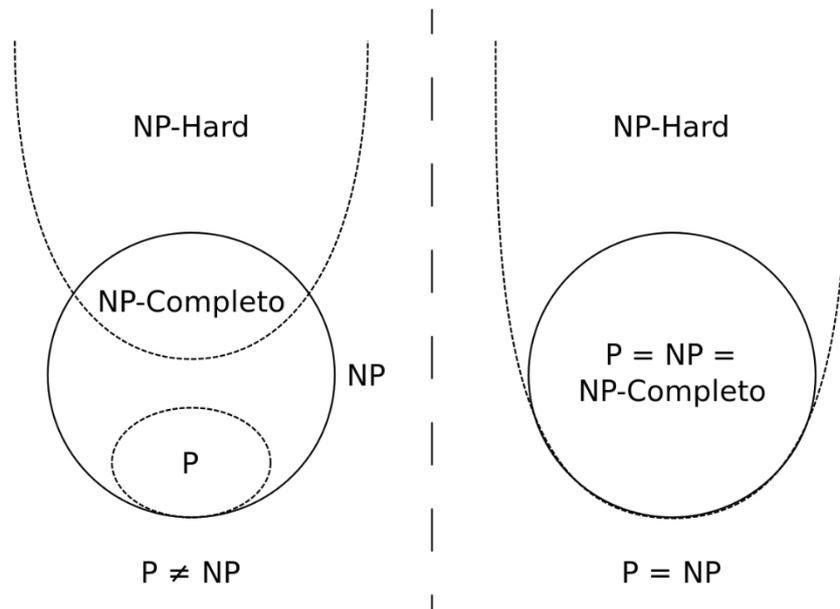


Figura 39. Diagrama de Euler: Familias de problemas en orden de complejidad ascendente.
FUENTE: <https://es.wikipedia.org/wiki/NP-hard>

En los ejemplos de ejecución del *Quantum Counting*, se aborda de cierto modo un problema *SAT* dado que la función objetivo es una función booleana de la cual se busca el número de soluciones distintas existentes o, en otras palabras, se busca el número de combinaciones que satisfacen dicha función booleana. Sin embargo, como se trata de buscar el número de combinaciones que satisfacen una función booleana, realmente el problema abordado es *#SAT* correspondiente a un problema tipo *#P-complete*.

Los problemas *#P-complete*, como se dice en [60], son problemas que cumplen dos características: i) el problema está en *#P* cuando se trata una cuenta de los caminos de una máquina de Turing no determinista de tiempo polinomial, ii) el problema está en *#P-hard* cuando se puede realizar una reducción de Turing en cualquier problema *#P*, lo cual se trata de un algoritmo en el que se hacen una cantidad polinomial de llamadas a una subrutina para el problema dado y fuera de dichas llamadas usa tiempo polinomial.

Este tipo de problemas tienen su propia clase de complejidad en teoría de complejidad computacional y también se clasifican como problemas de decisión.

Algunos de los problemas concretos de tipo *#P-complete* son [61]: el problema ya mencionado de contar combinaciones que satisfagan una fórmula booleana (*#SAT*), cuántas formas existen de colorear un grafo *G* con *k* colores o el problema de contar los *perfect matchings* existentes en un grafo bipartito, que se traduce en establecer un conjunto de aristas adyacentes de todos los vértices existentes en el grafo.

Un problema *#P-complete* es el llamado equilibrio de Nash o equilibrio del miedo [62] que, en teoría de juegos, es un concepto de solución para juegos multijugador en los cuales se asume

que cada jugador conoce y desempeña la mejor estrategia conociendo, además, las estrategias de los demás. En este ámbito, un equilibrio de Nash implica que cada uno de los jugadores seguirá su estrategia de minimización de pérdidas conociendo que el resto hará exactamente lo mismo y no tiene ningún beneficio cambiar dicha estrategia en el caso de competición, no obstante, es posible obtener, si todos cooperaran, beneficios más altos para cada uno de los participantes.

En el campo de la economía, el equilibrio de Nash es un tipo de competencia imperfecta entre empresas que compiten en el mercado por un mismo bien y pueden elegir la cantidad del bien a producir para maximizar las ganancias.

El *Quantum Counting* se puede aplicar al equilibrio de Nash por su poder de resolución de problemas *#P-Complete*, ayudando a economistas que buscan equilibrios en teoría de juegos.

Otra aplicación principal del *Quantum Counting* es la relativa al cálculo del número de iteraciones de Grover para el problema de la búsqueda no estructurada en un espacio de estados.

Tal y como se ha descrito en la sección del algoritmo de Grover, al utilizar este algoritmo cuántico para resolver el problema de la búsqueda en un espacio de estados, el proceso requiere conocer la proporción de soluciones existentes para maximizar la amplitud de los estados correctos y, de la misma manera, minimizar la probabilidad de medir un estado incorrecto.

Dicha proporción indica el número de veces k que se ha de aplicar el operador de Grover G para aumentar al máximo las amplitudes de probabilidad de los estados correctos acorde a la ecuación siguiente:

$$k = \left\lceil \frac{\pi}{4} \sqrt{N/M} \right\rceil$$

Por lo tanto, si se obtiene el número de soluciones M mediante el algoritmo de *Quantum Counting* entonces se podrá determinar el número de iteraciones de Grover necesarias para completar la tarea de la búsqueda de soluciones en espacios de estados de forma directa y correcta, causando que el colapso de la función de onda tras el algoritmo de Grover maximice las probabilidades de obtener uno de los estados correctos existentes en el espacio de estados.

Una de las aplicaciones más relevantes que podría tener el *Quantum Counting*, como se demuestra en [57], sería la estimación de amplitudes de un determinado estado cuántico, dado que con este algoritmo se halla una estimación del parámetro θ referente a las amplitudes de la función de onda introducida en el apartado del algoritmo de Grover:

$$|\psi\rangle = \sqrt{\frac{M}{N}} |\psi_{CORRECTO}\rangle + \sqrt{\frac{N-M}{N}} |\psi_{INCORRECTO}\rangle$$

La cual se puede representar de la siguiente manera, teniendo en cuenta que los estados correctos son aquellos que cumplen $f(x) = 1$ y los incorrectos tienen como salida $f(x) = 0$:

$$|\psi\rangle|f(x)\rangle = \sqrt{a} |\psi_{CORRECTO}\rangle|1\rangle + \sqrt{1-a} |\psi_{INCORRECTO}\rangle|0\rangle$$

Por lo tanto, al estimar el número de soluciones M existente en el espacio de estados N , en el fondo se está estimando la amplitud a de la función de onda anterior.

Esto ofrece la posibilidad de adaptar el *Simpler Quantum Counting* para la creación de un algoritmo de estimación de amplitudes cuánticas: *Quantum Amplitude Estimation*, un algoritmo de gran relevancia en la actualidad dadas las limitaciones que tiene utilizar el *Quantum Phase Estimation*.

Algunas de las utilidades del *Quantum Amplitude Estimation* es el testeo de hardware cuántico, es decir, para comprobar que los estados cuánticos físicos se producen de forma correcta, con sus respectivas amplitudes teóricas, en máquinas cuánticas reales. También tiene aplicaciones en criptografía cuántica en lo que a distribución de claves cuánticas seguras independientes del dispositivo se refiere.

Cabe mencionar que el algoritmo de *Quantum Amplitude Estimation* procede a descifrar las amplitudes de los estados cuánticos contenidos en una función de onda. Esto es importante ya que, a nivel clásico, cuando se procede al colapso de una función de onda, únicamente se obtiene un estado de todos los codificados en el estado cuántico. Utilizar el algoritmo de estimación de amplitudes implica recoger información del estado cuántico, es decir, de todas las posibles combinaciones a las que se aplica una determinada transformación o función.

9. Líneas futuras

En el campo de la computación cuántica, al ser una tecnología emergente en la actualidad dados los avances realizados y por realizar en las arquitecturas de computadores cuánticos, existen una gran cantidad de aplicaciones reales costosas en tiempo computacional cuando de informática clásica se trata, donde los circuitos cuánticos entran en juego relajando dicha complejidad computacional y permiten obtener resultados para problemas complejos en tiempos asequibles.

Según todo lo expuesto en las secciones del *Quantum Counting* y de las aplicaciones prácticas, hay mucha diversidad de trabajos futuros con utilidad en el mundo real o, incluso, en lo que respecta a la mejora de algoritmos y arquitecturas cuánticas.

La principal aplicación, mencionada en las conclusiones del *Simpler Quantum Counting*, se basa en adaptar dicho algoritmo desarrollado para obtener un algoritmo de *Quantum Amplitude Estimation*, cuya importancia se fundamenta por su necesidad en otros programas cuánticos.

Como se demostró con antelación, un algoritmo de estimación de amplitudes cuánticas es un algoritmo primordial para casi cualquier programa cuántico dado que su propósito es obtener, de manera aproximada, las amplitudes asociadas a los estados cuánticos de una función de onda.

Esto es de gran relevancia debido a que muchos algoritmos cuánticos requieren de una rutina final en la que se han de descodificar las amplitudes del estado cuántico resultante tras una serie de transformaciones, relativas al programa cuántico en cuestión, con la intención de obtener los resultados que dictan, en general, cómo se ha modificado el estado cuántico y porqué.

Conociendo las amplitudes, los parámetros iniciales utilizados y ciertas propiedades de la función de onda final, se puede abstraer la solución del problema programado mediante un procesamiento posterior de esta información, de forma similar al algoritmo *Simpler Quantum Counting*.

Por ello, una línea futura sustancial de este trabajo es habilitar una adaptación del *Simpler Quantum Counting* a un algoritmo de *Quantum Amplitude Estimation*, superando el algoritmo de *Quantum Phase Estimation* en recursos necesarios, tiempo de ejecución y tasas de error.

Otro trabajo futuro puede ser el estudio de las máquinas cuánticas físicas actuales para modelar un nuevo *Simpler Quantum Counting* capaz de ejecutarse en dichas máquinas obteniendo resultados más precisos, siendo este tolerante al ruido y a la decoherencia cuántica de algún modo.

Esto es necesario si se desea que el algoritmo sea eficaz en las máquinas cuánticas públicas de la actualidad, dado el contraste entre los resultados de las ejecuciones del *Simpler Quantum Counting* en las máquinas físicas de *IBM* y los resultados de las simulaciones del mismo.

De cualquier manera, si los ordenadores cuánticos mejoran sus tasas de error, esto no sería necesario puesto que, como muestran los resultados en simulación, el algoritmo devuelve con

buena precisión el número de combinaciones que satisfacen la función objetivo sin necesidad de aumentar significativamente el número de muestras de cada subcircuito.

Adicionalmente, es posible construir funciones oráculo mucho más complejas, pasando de funciones de lógica booleana, como las implementadas en las pruebas del *SQC*, a funciones aritméticas.

Por la similitud que tienen los circuitos cuánticos con los circuitos electrónicos con los que se construyen las unidades aritmético-lógicas (*ALU*), se puede proceder a construir un circuito cuántico haciendo uso de la lógica con la que se construyen sumadores, multiplicadores, etcétera.

De esta manera, se puede construir un circuito cuántico cuya función oráculo sea una transformación que equivalga a la aplicación de una función o ecuación, con una o varias variables, y que se evalúen paralelamente todas las combinaciones posibles mediante la propiedad de la superposición, obteniendo como resultado el valor de la variable o variables que satisfacen dicha ecuación.

Comúnmente, las funciones aritméticas se componen de operaciones entre números reales, incluyendo las variables que contienen. Hasta ahora, a lo largo de todo este proyecto, se ha trabajado únicamente con números enteros, no obstante, es perfectamente posible representar números reales en registros cuánticos con una determinada precisión, análogamente a los tipos de datos *float* y *double* referentes a la computación clásica.

Por ejemplo, si se tiene un registro de 16 qubits, se pueden reservar 8 qubits para la parte real de la variable y otros 8 para la parte fraccionaria, pudiendo representar números reales discretos en el intervalo $[0, 2^8) = [0, 256)$ donde la precisión es de 2^{-8} , llegando a representar $2^{16} = 65536$ cifras distintas que se evaluarán de forma paralela, debido a la superposición de estas 2^{16} combinaciones, en el circuito cuántico relativo a la función oráculo. El intervalo, además, se puede modificar añadiendo constantes a la función en cuestión para mover el intervalo al rango que se desee.

En el caso de que se quieran utilizar más variables para evaluar una ecuación multivariable, se necesitan tantos registros cuánticos como variables. Cuando esto sea posible en un ordenador cuántico y las tasas de error sean controlables, los algoritmos cuánticos serán capaces de evaluar todas las combinaciones de valores que puedan tomar las distintas variables en una ecuación que modela un problema y dar la solución buscada en tiempos de cómputo menores, aprovechando el paralelismo cuántico.

Concretando, esta representación de números reales y la construcción de circuitos cuánticos aritméticos da lugar a múltiples aplicaciones futuras como la del desarrollo de un algoritmo de *Quantum Hyperparameter Estimation* referente a la estimación de hiperparámetros en *machine learning*, similar a la que se estudia en [63], donde se expone una librería de *Python* de estimación de hiperparámetros para modelos de inteligencia artificial.

Este algoritmo se fundamenta en la minimización de una función de pérdida o *loss function* relacionada con el modelo de aprendizaje automático a entrenar, equivalente a una función multivariable, teniendo en cuenta múltiples parámetros iniciales como la tasa de aprendizaje (*learning rate*), número de neuronas por capa, número de capas de la red neuronal, factor *gamma*, entre otras.

Mediante el algoritmo de Grover, es posible encontrar el valor de dichos parámetros que minimizan la función de pérdida del modelo, codificando dicha función en la transformación unitaria relativa al operador U_f del operador de Grover.

Utilizando el *SQC* para obtener el número de soluciones que satisfacen la minimización de la función de pérdida, se puede hallar el número de iteraciones de Grover que se han de aplicar en el circuito cuántico para maximizar la probabilidad de obtener los valores de los hiperparámetros que satisfacen la función oráculo correspondiente a la *loss function*, la cual se puede programar como una inequación mediante un circuito comparador (*GEL*).

O bien, otra forma de abordar la estimación de hiperparámetros sería emplear directamente el *Quantum Counting* para calcular el número de combinaciones que satisfacen la minimización de la función de pérdida, lo cual queda pendiente de estudio.

Por último, algunos trabajos futuros interesantes en el campo de la computación cuántica que no se han tratado en este proyecto son los relacionados con *Quantum Machine Learning* o aprendizaje automático cuántico donde el paradigma de inteligencia artificial para el aprendizaje es distinto al que se utiliza en informática clásica y, según muchos artículos como [64] o [65], es un área prometedora donde la computación cuántica puede suponer muchos avances innovadores en redes neuronales cuánticas o redes de convolución, entre otras técnicas.

Agradecimientos

Nada es eterno, la vida es cambio y adaptación y este recorrido es el ejemplo de confirmación, tu camino varía sin cesión. Al finalizar etapas, miras atrás y recuerdos destapas. El tiempo empleado termina perdiendo significado, se contrae en memorias que cuentan historias símbolo de sabiduría después de sudar la gota fría. Varias gotas frías en una aunada que, ahora, se siente como sed saciada. Indaga más de cerca, la sed se mitiga temporalmente, pero otra vez pide agua tu mente. Los transcurros lo requieren, así como los cercanos que te quieren.

Aquí acaba una etapa de las más enriquecedoras de mi vida. Este camino elegido hace cuatro años, con sus altibajos, finaliza con este proyecto en el que he puesto todo mi empeño y pasión. Este período me ha traído grandes experiencias, conocimientos y personas inolvidables, cada una a su manera, lo que hace todavía más excepcional el camino.

He de dar gracias por haber conocido a mis amigos de la *ETSISI UPM*, gente con la que se ha podido contar desde el minuto uno. Gracias Katarina, Claudia, Tarek y Daniel por el apoyo, entre muchos otros que podría nombrar.

Agradecer a mi familia y a mis amigos de Perales del Río por aguantar quejas, emociones y chapas universitario-informáticas, además de otros momentos trascendentales vividos con vosotros. Gracias por estar Henar, Miguel y Virginia.

Por último, agradecer a mi tutor, Giannicola, y a Rafael y Ulises por introducirme en el mundo de la mecánica y computación cuántica que, en el momento en el que me involucré en él, fui consciente de una de mis pasiones y aspiraciones personales.

El futuro puede ser ambicioso, si no lo fuera, el ser humano seguiría viviendo en un foso.

Bibliografía

2. CONTEXTO HISTÓRICO DE LA MECÁNICA CUÁNTICA

Historia de la mecánica cuántica

[1] https://es.wikipedia.org/wiki/Historia_de_la_mec%C3%A1nica_cu%C3%A1ntica

[2] <https://quefuerteeslaciencia.com/2016/10/21/una-breve-historia-de-la-mecanica-cuantica/>

Cronología de la mecánica cuántica

[3] https://particleadventure.org/other/spa_history/quantumt.html

[4] https://www.ecured.cu/Mec%C3%A1nica_cu%C3%A1ntica

Experimento de Young

[5] https://es.wikipedia.org/wiki/Experimento_de_Young

[6] <http://fisicaymasalla.blogspot.com/2006/12/el-experimento-de-la-doble-rendija-o-de.html>

Hamiltoniano cuántico

[7] https://es.wikipedia.org/wiki/William_Rowan_Hamilton

[8] [https://es.wikipedia.org/wiki/Hamiltoniano_\(mec%C3%A1nica_cu%C3%A1ntica\)](https://es.wikipedia.org/wiki/Hamiltoniano_(mec%C3%A1nica_cu%C3%A1ntica))

Cuantización de Max Planck y Albert Einstein

[9] <https://www.sciencekindle.com/es/origen-de-la-fisica-cuantica/>

Espacio de Hilbert

[10] https://es.wikipedia.org/wiki/Espacio_de_Hilbert

Modelo atómico de Ernest Rutherford

[11] https://es.wikipedia.org/wiki/Modelo_at%C3%B3mico_de_Rutherford

[12] http://www.qorganica.es/QOT/T0/historia_atomo_exported/l114.htm

Modelo atómico de Niels Bohr

[13] <https://www.geoenciclopedia.com/modelo-atómico-de-bohr/>

[14] https://es.wikipedia.org/wiki/Modelo_at%C3%B3mico_de_Bohr

Números cuánticos

[15] https://es.wikipedia.org/wiki/N%C3%BAmero_cu%C3%A1ntico

Hipótesis de De Broglie

[16] <https://www.bbvaopenmind.com/ciencia/grandes-personajes/louis-de-broglie-el-principe-de-la-cuantica/>

[17] https://es.wikipedia.org/wiki/Dualidad_onda_corp%C3%BAsculo

Ecuación de Schrödinger

[18] <https://www.bbvaopenmind.com/ciencia/grandes-personajes/schrodinger-un-cuántico-tras-el-secreto-de-la-vida/>

[19] https://es.wikipedia.org/wiki/Ecuaci%C3%B3n_de_Schr%C3%B6dinger

Principio de exclusión de Pauli

[20] https://es.wikipedia.org/wiki/Principio_de_exclusi%C3%B3n_de_Pauli

[21] <http://hyperphysics.phy-astr.gsu.edu/hbasees/pauli.html>

Principio de incertidumbre de Heisenberg

[22] http://es.wikipedia.org/wiki/Relaci%C3%B3n_de_indeterminaci%C3%B3n_de_Heisenberg

Ecuación de Dirac

[23] <https://bibliozacut.wordpress.com/2017/07/20/la-ecuacion-mas-bonita/>

Concepto de ordenador cuántico

[24] https://es.wikipedia.org/wiki/Algoritmo_de_Shor

[25] https://en.wikipedia.org/wiki/Lov_Grover

3. INTRODUCCIÓN A LA COMPUTACIÓN CUÁNTICA

Introducción a la computación cuántica universal o de puertas

[26] *An introduction to quantum computing*

<http://mmrc.amss.cas.cn/tlb/201702/W020170224608149125645.pdf>

Tipos de ordenadores cuánticos y aplicaciones

[27] <https://www.cbinsights.com/research/report/quantum-computing/>

4. ESTADO DEL ARTE

Requisitos de una máquina cuántica

[28] https://es.wikipedia.org/wiki/Computaci%C3%B3n_cu%C3%A1ntica

Ordenadores cuánticos de *IBM*

[29] <https://www.cnet.com/news/IBM-now-has-18-quantum-computers-in-its-fleet-of-weird-machines/>

[30] <https://www.ibm.com/quantum-computing/learn/what-is-IBM-q/>

[31] <https://newsroom.ibm.com/2019-03-04-IBM-Achieves-Highest-Quantum-Volume-to-Date-Establishes-Roadmap-for-Reaching-Quantum-Advantage>

[32] <https://www.itwarelatam.com/2020/09/17/IBM-quantum-condor-1000-qubits-para-el-2023/>

Supremacía cuántica de *Google*

[33] <https://www.nature.com/articles/s41586-019-1666-5>

[34] https://www.abc.es/ciencia/abci-Google-dice-haber-demostrado-supremacia-cuantica-201910231056_noticia.html

Sycamore de *Google*

[35] <https://www.wired.com/story/quantum-computing-here-but-not-really>

Arquitectura topológica cuántica de *Microsoft*

[36] <https://cloudblogs.microsoft.com/quantum/2020/03/27/new-physics-discovery-microsoft-quantum-topology-with-a-twist/>

[37] <https://iopscience.iop.org/article/10.1088/1126-6708/2004/12/032/pdf>

Algoritmo para la computación cuántica libre de errores

[38] <https://singularityhub.com/2020/08/14/new-algorithm-paves-the-way-towards-error-free-quantum-computing/>

[39] <https://www.nature.com/articles/s41567-020-0992-8>

Computador cuántico de Rigetti

[40] <https://www.rigetti.com/>

Aplicaciones existentes implementadas con computación cuántica

[41] <https://venturebeat.com/2020/08/14/Google-researchers-use-quantum-computing-to-help-improve-image-classification/>

[42] <https://www.newscientist.com/article/2229673-china-has-developed-the-worlds-first-mobile-quantum-satellite-station/>

[43] <https://arxiv.org/abs/2006.14510>

[44] <https://arxiv.org/abs/2006.14510>

[45] <https://www.microsoft.com/en-us/research/blog/quantum-speedups-for-unstructured-problems-solving-two-twenty-year-old-problems/>

5. HERRAMIENTAS PARA EL DESARROLLO DE PROGRAMAS CUÁNTICOS

Lista de herramientas de desarrollo en computación cuántica

[46] https://en.wikipedia.org/wiki/Quantum_programming

Rigetti Computing

[47] <https://pyquil-docs.rigetti.com/en/1.9/basics.html>

Google Cirq

[48] <https://cirq.readthedocs.io/en/stable/Google/devices.html#sycamore>

Amazon Braket

[49] <https://aws.amazon.com/es/braket/features/>

Microsoft Quantum Development Kit

[50] <https://docs.microsoft.com/en-us/quantum/?view=qsharp-preview>

[51] <https://docs.microsoft.com/en-us/qsharp/api/qsharp/?view=qsharp-preview>

IBM Qiskit

[52] <https://qiskit.org/documentation/index.html#>

[53] https://qiskit.org/documentation/apidoc/circuit_library.html

6. ALGORITMO QUANTUM COUNTING

Proceso para el *Quantum Counting*

[54] https://en.wikipedia.org/wiki/Quantum_counting_algorithm

Quantum Counting proporcionado por Qiskit IBM

[55] <https://quantum-computing.ibm.com/jupyter/user/qiskit-textbook/content/ch-algorithms/quantum-counting.ipynb>

7. SIMPLER QUANTUM COUNTING

Estimación de amplitudes sin estimación de fase

[56] <https://link.springer.com/article/10.1007/s11128-019-2565-2>

Simpler Quantum Counting de Scott Aaronson y Patrick Rall

[57] <https://arxiv.org/abs/1908.10846>

8. APLICACIONES PRÁCTICAS

Tipos de problemas según su complejidad

[58] <https://www.britannica.com/science/NP-problem>

Ejemplos de problemas *NP*, *NP-Complete*

[59] http://cms.dm.uba.ar/academico/materias/1ercuat2019/optimizacion/Clase_08.html

Definición y características de problemas *#P-complete*

[60] <https://en.wikipedia.org/wiki/%E2%99%AFP-complete>

Ejemplos de problemas *#P-complete*

[61] <https://en.wikipedia.org/wiki/%E2%99%AFP-complete#Examples>

[62] https://es.wikipedia.org/wiki/Equilibrio_de_Nash

Estimación de hiperparámetros en *Machine Learning*

[63] <https://towardsdatascience.com/automate-hyperparameter-tuning-for-your-models-71b18f819604>

Aplicaciones prácticas futuras de *Quantum Artificial Intelligence*

[64] <https://www.nature.com/articles/s41598-020-67018-1>

[65] <https://arxiv.org/abs/2005.04316>