Escuela Superior
de Ingeniería y Tecnología
Universidad de La Laguna

Trabajo de Fin de Grado

Desarrollo de microprogramas cuánticos - Algoritmo de Shor

Quantum microprogram development - Shor's algorithm

Sergio Guerra Arencibia

La Laguna, 10 de junio de 2021

D. **Molina Gil, Jezabel Miriam**, con N.I.F. 78507682B profesor Titular de Universidad adscrito al Departamento de Nombre del Departamento de la Universidad de La Laguna, como tutor

D. **Martín Fernández, Francisco**, con N.I.F. 78629638K, gerente de ingenieria del software en investigación IBM, como cotutor.

C E R T I F I C A (N)

Que la presente memoria titulada:

"Desarrollo de microprogramas cuánticos - Algoritmo de Shor"

ha sido realizada bajo su dirección por D. **Sergio Guerra Arencibia**, con N.I.F. 51148679Z.

Y para que así conste, en cumplimiento de la legislación vigente y a los efectos oportunos firman la presente en La Laguna a 10 de junio de 2021

Agradecimientos

En primer lugar agradecer a mis tutores, Jezabel y Francisco, por hacer posible la realización de este aventurado Trabajo de Fin de Grado.

También agradecer a mi familia por su ilimitada paciencia y apoyo.

Licencia

Resumen

El presente Trabajo de Fin de Grado tiene como objetivo realizar una implementación básica del algoritmo de Shor, un algoritmo de gran impacto en los sistemas criptográficos modernos. Para llevar esto a cabo es imprescindible realizar una introducción al paradigma cuántico, explicando todos los conceptos necesarios para una correcta comprensión de la implementación que se presenta posteriormente. Por lo tanto se comenzará presentando elementos básicos como un cúbit o un sistema entrelazado, posteriormente se usarán estos conceptos para construir subrutinas más complejas como la Transformada Cuántica de Fourier. Finalmente se profundizará en el algoritmo de Shor, su implementación y ciertas optimizaciones básicas que se pueden realizar al circuito.
La tecnología fundamental que se utilizará es la herramienta de software cuántico Qiskit. Esta nos permitirá implementar el algoritmo de Shor y ver su funcionamiento. Adicionalmente se ejecutará y se expondrán los resultados de sus ejecuciones.

Palabras clave: Computación cuántica, cúbit, Algoritmo de Shor, factorización de números enteros, Qiskit, IBMQ

Abstract

This end of degree project aims to perform a basic implementation of Shor's algorithm, an algorithm of great importance for modern cryptographic systems. To carry this out, it's essential to make an introduction to the quantum paradigm, explaining all the necessary concepts for a correct understanding of the implementation that is presented later. Therefore we will begin by explaining basic elements as a qubit or an entangled system, later these concepts will be used to build complex subroutines as the Quantum Fourier Transform. Finally, we will delve into Shor's algorithm, its implementation and certain basic optimizations that can be made to the circuit.

The fundamental tecnology used is Qiskit, an open-source framework for Quantum Computing. Qiskit will allow us to implement the Shor's algorithm and see how it works. Additionally, it will be executed and the results of its executions will be studied and exposed.

Keywords: Quantum Computing, qubit, Shor's algorithm, Integer factorization, Qiskit, IBMQ

Índice general

Índice de Figuras

Capítulo 1

Introducción

1.1. La computación cuántica

A principios del siglo XX los principales científicos de la época se encontraban trabajando en diferentes problemas físicos que no parecían tener una solución dentro de las leyes de la física clásica. Estos problemas fueron, principalmente, la ley de la radiación de un cuerpo negro, el efecto fotoeléctrico y el problema de estabilidad y tamaño de los átomos. La resolución de estos problemas trajo consigo el nacimiento de la mecánica cuántica, la cual permitiría el desarrollo de una nueva manera de cómputo basado en ella, la computación cuántica.

La computación cuántica es un paradigma de computación reciente y muy diferente de la computación clásica a la que estamos acostumbrados. Teorizado por primera vez en la segunda mitad del siglo XX, este nuevo paradigma aprovecha, como ya se mencionó, las leyes y principios cuánticos para su funcionamiento, permitiendo así procesar información de una manera radicalmente distinta a como se ha venido haciendo hasta ahora.

Lo que hace tan interesante a este nuevo enfoque, es que un mismo problema tiene una complejidad diferente en computación cuántica y en computación clásica. Esto se puede expresar en términos de modelos de cómputo, un ordenador clásico computa la información de la misma manera que una máquina de Turing, mientras que un computador cuántico lo hace como una máquina de Turing cuántica. Así, algunos problemas que actualmente son intratables de forma clásica se convierten en tratables en el marco de la computación cuántica. Desde luego esto tiene una gran cantidad de ventajas, y, aunque no está exento de algunos inconvenientes, es fácil entender por qué ha causado tanta expectación entre la comunidad científica y técnica, así como en la sociedad en general.

Como es de esperar, este nuevo paradigma de computación conlleva una nueva forma de manipular la información, nuevas puertas lógicas y nuevos algoritmos. Todo esto lleva aparejado la necesidad de invertir esfuerzos en el desarrollo de esta nueva tecnología. En este sentido se enfoca este Trabajo de Fin de Grado, cuyo objetivo fundamental se centra en el estudio y aplicación de esta nueva forma de computación a través del desarrollo del algoritmo cuántico de Shor.

Es importante destacar que un computador cuántico no va a ser superior a un computador clásico en todas y cada una de las situaciones. Para una parte de la computación que

se realiza a día de hoy no presentará ninguna ventaja, de hecho, será menos adecuado dada la mayor complejidad del sistema cuántico. A pesar de esto, existen problemas para los cuales es conveniente realizar una aproximación cuántica y es aquí donde este nuevo paradigma brilla y nos aporta complejidades mucho menores de las que disponemos en el paradigma clásico.

Por tanto es necesario entender estos dos paradigmas como complementarios, ya que la gran mayoría de cómputos hacen uso tanto de la computación cuántica como de la computación clásica.

1.2. Algoritmo de Shor

A lo largo de este trabajo veremos qué es el algoritmo de Shor [9] [14] y cómo realizar una implementación del mismo. Este es uno de los algoritmos cuánticos más conocidos ya que permite resolver el problema de la factorización de números enteros en un tiempo polinomial. Esto implica un gran impacto en muchos campos tecnológicos actuales, principalmente en el campo de la criptografía.

Lograr resolver el problema de factorización de números enteros en un tiempo polinómico hace vulnerables los sistemas criptográficos asímétricos utilizados en la actualidad, como por ejemplo los sistemas RSA, Diffie-Hellman, DSA, etc. [12] [16]

Sabiendo esto se aprecia claramente la principal motivación de este algoritmo y el gran cambio que introducirá en los sistemas modernos.

1.3. Objetivos

Para poder realizar un estudio completo del algoritmo comenzaremos estudiando los conceptos más básicos de la computación cuántica, es decir, qué es un cúbit, las puertas lógicas cuánticas, subrutinas necesarias, etc... Así avanzaremos de manera progresiva a través de este nuevo paradigma hasta poder realizar una implementación del algoritmo de Shor. Es interesante destacar que, como se comentó anteriormente, la implementación a realizar consta de un módulo cuántico y un módulo clásico.

El número a factorizar será el 15. Esta información es determinante porque el circuito cuántico del algoritmo de Shor es específico para el número que se desea descomponer. Esto es una consecuencia de que ciertas componentes del circuito implementan operaciones matemáticas donde interviene el número a descomponer. En este trabajo se factorizará el número 15 y, por lo tanto, si se deseara descomponer uno diferente sería necesario cambiar y adecuar la implementación.

Para complementar esta implementación del algoritmo se realizarán ejecuciones del circuito desarrollado, exponiendo los resultados obtenidos y comprobando así su correcto funcionamiento y validez. Para estas ejecuciones se utilizarán dos aproximaciones diferentes:

- Simulador cuántico
- Computador cuántico

La decisión de utilizar ambas aproximaciones se basa en el valor que aporta cada una de ellas. El simulador permite ejecuciones cómodas, rápidas y precisas, por lo que es idóneo para el proceso de construcción y pruebas del circuito. En cambio, el computador cuántico nos permite ejecutar el circuito en un entorno de real, donde existen factores como el ruido, la calidad de los cúbits, etc.

Como se explica con más detalle en el capítulo 4, para llevar a cabo este trabajo, se hará uso principalmente de la herramienta “Qiskit”. Este es un kit de desarrollo software (*software development kit - SDK*) de código abierto pra trabajo en computadores cuánticos a nivel de pulso, circuitos y módulos de aplicación.

También se ha hecho uso de diferentes herramientas proporcionadas por IBM orientadas también al desarrollo de software cuántico. Estas son principalmente IBM Quantum Experience y los IBM Quantum Services.

Capítulo 2

Estado del arte

La computación cuántica fue teorizada por Paul Benioff en el año 1981 [3]. Desde entonces ha habído una gran cantidad de avances y se ha demostrado que esta nueva manera de computación es una realidad.

En cambio, hasta la fecha la computación cuántica no ha tenido una gran implicación en el mundo real. Los computadores cuánticos que existen actualmente se enfrentan a una gran cantidad de problemas. Algunos de ellos son la reducida cantidad de cúbits que somos capaces de manejar, la calidad de estos cúbits, el bajo tiempo de coherencia cuántica, etc... Esto hace que solo sea posible trabajar con circuitos cuánticos de pequeñas dimensiones, implicando esto la resolucion de problemas de un tamaño pequeño.

Muchas de las grandes empresas del sector tecnológico (IBM [8], Google [5], etc...) están trabajando en el desarrollo de sus propios ordenadores cuánticos, con el objetivo de conseguir la denominada *supremacía cuántica*. Esta consiste en demostrar que un computador cuántico es capaz de resolver un problema que un computador clásico no podría resolver en un tiempo razonable.

Cabe destacar que, actualmente, muchas de las empresas que desarrollan y trabajan con computadores cuánticos ponen sus máquinas a disposición de todo el que quiera ejecutar su software. Esto es lo que permitirá en este trabajo ejecutar el circuito implementado en un entorno real.

Los computadores cuánticos más potentes que se han dado a conocer poseen entre cincuenta y ochenta cúbits. Por ejemplo, IBM, la empresa desarrolladora de las herramientas que se usarán en este trabajo y una de las principales participantes en el desarrollo de este nuevo paradigma, cuenta con varios computadores cuánticos de diferente número de cúbits, el mayor de ellos con sesenta y cinco. Además, las apuestas para los próximos años es que estos números crezcan rápidamente.

A pesar de disponer sólo de estos incipientes computadores cuánticos, la existencia de estos sistemas supone una herramienta de gran utilidad para comenzar a trabajar en el desarrollo de su funcionamiento, implementando y ejecutando algoritmos cuánticos aplicados a unas entradas de un tamaño adecuado.

A continuación veremos en qué punto se encuentra el algoritmo de Shor. Para ello veremos qué números se han logrado descomponer satisfactoriamente a día de hoy y cuales están en el punto de mira.

La primera ejecución exitosa del algoritmo de Shor fue realizada en el año 2001, factorizando el número 15. Fue realizada por un equipo de IBM en un computador cuántico de 7 cúbits. [15]

Posteriormente, en el año 2012 se logró factorizar el número 21. Usando una técnica de reciclado de cúbits se logró utilizar un tercio de los cúbits que generalmente se necesitan para la descomposición. [11]

El intento de descomposición de un número mayor más reciente ocurrió en 2019, donde se intentó factorizar el número 35, aunque no se logró realizar de manera exitosa. [2]

Capítulo 3

Conceptos cuánticos

En este capítulo se presentarán y estudiarán conceptos básicos y fundamentales de la computación cuántica. Estos nos permitirán entender las bases del paradigma, formando la base necesaria para la comprensión del trabajo desarrollado.

Se comenzará hablando de los cúbits y de las puertas lógicas cuánticas de un solo cúbit, estudiando una de las ideas fundamentales de la computación cuántica, la superposición. Posteriormente trataremos los casos en los que tenemos múltiples cúbits, donde aparece otro concepto fundamental, el entrelazamiento cuántico.

Cabe destacar que los conceptos y leyes pertenecientes a la mecánica cuántica son extrañas y contraintuitivas. Pero la raíz de esta extravagancia no es otra que la familiaridad que tenemos con el mundo clásico. Las ideas que veremos en este primer capítulo y a lo largo de este trabajo no son intrínsecamente difíciles, pero debemos estar preparados para pensar en términos que serán completamente diferentes a los que habituamos a usar en cómo entendemos y cómo estudiamos la naturaleza del mundo macroscópico.

3.1. Sistemas cuánticos de un cúbit

3.1.1. Cúbits

Definición teórica

Un bit es la unidad mínima de información utilizada en la computación clásica. Este puede tomar dos valores diferentes, generalmente representados por el 0 y el 1. Así, podemos decir que un bit dado está en el estado 0 o en el estado 1, no hay más información asociada al mismo.

Existen infinidad de sistemas que podrían usarse como bits, por ejemplo una caja (abierta o cerrada), una bombilla (encendida o apagada) o incluso un ser vivo (vivo o muerto). En los sistemas informáticos se implementan los bits como corriente eléctrica, siendo un 1 la presencia de voltaje y un 0 su ausencia.

Los cúbits [4] y la computación cuántica son análogos a los bits y la computación clásica. Estos son la unidad mínima de información usada en este tipo de computación, aunque son algo más complejos que los bits convencionales.

Al igual que los bit, los cúbits son sistemas cuánticos que pueden estar en dos estados diferentes, interpretados como 0 o 1, pero la diferencia principal con los bits es que un cúbit también puede estar en los estados 0 y 1 de manera simultánea, en lo que se conoce como un estado de *superposición* .

Por lo tanto un cúbit puede encontrarse en una combinación lineal de los estados 0 y 1.

$$|\phi\rangle = x\,|0\rangle + y\,|1\rangle \tag{3.1}$$

Esta es la forma estandar de expresar los cúbits y se conoce como notación *bra-ket*. A la hora de leerlo la expresión $|\phi\rangle$ se pronuncia *ket* ϕ.

Esta combinación de estados es lo que se conoce como superposición cuántica. La superposición cuántica es un principio físico que enuncia que un sistema físico puede poseer simultáneamente más de un valor de una cantidad observable.

Prestemos más atención a esto, ya que es un concepto complejo. Anteriormente mencionamos que existen infinidad de sistemas que actúan como un bit, por ejemplo una caja, que puede estar abierta o cerrada. Esta situación sería correcta en la computación clásica, pero si obtuvieramos una caja cuántica y esta actuara como un cúbit nos encontraríamos con que esta puede estar abierta, puede estar cerrada o puede estar abierta y cerrada al mismo tiempo. Cuando la caja se encuentra abierta y cerrada al mismo tiempo tiene sus dos estados posibles *superpuestos*.

Una vez y comprendemos qué es un cúbit y qué es la superposición cuántica podemos pasar al siguiente concepto, que está estrechamente ligado a la superposición y se conoce como *colapso de función de onda*.

Los estados en superposición no son medibles, ya que al tratar de hacerlo el sistema superpuesto colapsa en uno -y solamente uno- de sus estados posibles. Así, cuando nosotros tengamos un cúbit en un estado de superposición (una combinación del estado 0 y del estado 1) y nosotros midamos ese cúbit, solo obtendremos como respuesta uno de estos dos valores, obteniendo la misma información que si midiéramos un bit clásico.

Aunque a priori los bits y los cúbits pueden parecer similares, la diferencia entre estos se encuentra en la información almacenada antes de la observación, ya que un cúbit contiene no sólo un valor, sino que contiene una superposición de estos a lo largo de toda la ejecución, hasta que se colapsa en un valor final al ser medido, momento a partir del cual actúa como un bit clásico. Es decir, antes de la medición son radicalmente diferentes y esto es algo aprovechable a la hora de realizar cómputos.

Es interesante destacar que, al igual que pasaba con los bits, existen una infinidad de elementos que pueden usarse como cúbits, por ejemplo fotones, electrones, diferentes tipos de iones, etc... Basta con que el sistema tenga las características cuánticas necesarias para actuar como un cúbit (por ejemplo la superposición cuántica).

Definición formal

Una vez conocemos el concepto de cúbit y la notación bra-ket, veamos cómo se expresa de manera formal.

$$\begin{aligned} &|x\rangle = \alpha\,|0\rangle + \beta\,|1\rangle \\ &\alpha, \beta \in \mathbb{C} \\ &|\alpha|^2 + |\beta|^2 = 1 \end{aligned} \tag{3.2}$$

Para definir un cúbit se utiliza un espacio vectorial complejo de dos dimensiones, es decir, se definen mediante dos números complejos (amplitudes). Adicionalmente, esta pareja de números debe estar normalizada, valiendo su módulo uno.

Es importante comprender el significado que tienen los valores α y β. Estos representan las probabilidades que tenemos de medir al cúbit en cada uno de sus estados básicos. En otras palabras, $|\alpha|^2$ nos da la probabilidad de que cuando midamos el estado del cúbit este colapse en el ket $|0\rangle$. El valor $|\beta|^2$ es análogo para el ket $|1\rangle$.

Con esta información queda en evidencia la necesidad de que este conjunto de valores, es decir, el vector que representa el estado del cúbit, esté normalizado. Su módulo debe de valer 1 porque estamos hablando de probabilidades, y en todo sistema probabilístico las sumas de todas las probabilidades debe ser igual a 1.

Una vez definidos los cúbits tanto conceptual como formalmente, veamos cómo se expresan los estados básicos (0 y 1):

$$|0\rangle = 1\,|0\rangle + 0\,|1\rangle \qquad \alpha = 1, \quad \beta = 0 \tag{3.3}$$

$$|1\rangle = 0\,|0\rangle + 1\,|1\rangle \qquad \alpha = 0, \quad \beta = 1 \tag{3.4}$$

Como se define en la expresión (3.2), cualquier pareja de números complejos que cumpla las condiciones anotadas anteriormente puede formar un estado válido, por ejemplo:

$$|\psi\rangle = \frac{1}{\sqrt{2}}\,|0\rangle + \frac{i}{\sqrt{2}}\,|1\rangle \qquad \alpha = \frac{1}{\sqrt{2}}, \quad \beta = \frac{i}{\sqrt{2}} \tag{3.5}$$

Notación vectorial

A la hora de trabajar con cúbits y sistemas de cúbits, estos son expresados como vectores, pudiendo así operar y manejarlos de una manera sencilla y cómoda. Los números que forman el vector son los factores α y β.Por ejemplo:

$$|0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \qquad\qquad |1\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \tag{3.6}$$

Estos son los vectores que representan los estados básicos. Como se puede apreciar, estos dos vectores son ortogonales y están normalizados, por lo que son un conjunto de vectores ortonomales, formando así una base.

3.1.2. Representación

Expresar un cúbit como una pareja de números complejos conforma un sistema con cuatro grados de libertad (dos por cada uno de los números complejos). Trabajando con las propiedades que presentan los cúbits, podemos simplificar estos cuatro grados y obtener un sistema de solamente dos grados de libertad.

La restricción de normalización $\alpha^2 + \beta^2 = 1$ nos elimina uno de los grados de libertad. El segundo es eliminado del hecho de que la fase global del estado de cúbit no tiene consecuencia física.

Matemáticamente el resultado es el siguiente:

$$|\psi\rangle = \cos\frac{\theta}{2}|0\rangle + e^{i\phi}\sin\frac{\theta}{2}|1\rangle \tag{3.7}$$

Los dos valores que definen a un cúbit se pueden interpretar como una coordenada polar (módulo y fase), permitiéndonos representarlo en la superficie de una esfera, la esfera de Bloch. Cada punto de su superficie representa una posible configuración de un cúbit, donde los polos de la esfera corresponden a los estados básicos.

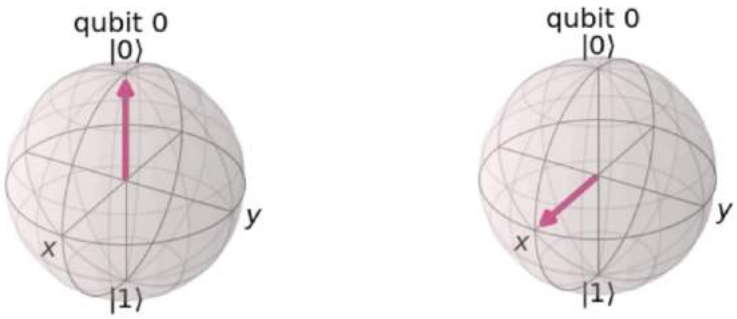


Figura 3.1: Representación de cúbits en la esfera de Bloch

3.1.3. Medición

Como ya sabemos, al intentar medir un sistema (en nuestro caso un cúbit) que se encuentre en una superposición cuántica haremos que este colapse y adopte un valor concreto, siendo este el que obtendremos como resultado de la medición.

Es evidente que a la hora de medir un cúbit debemos medirlo respecto a algo, es decir, debemos medirlo respecto a lo que se conoce como una base computacional. Esta base es una de las cosas más importantes a tener en cuenta a la hora de medir un cúbit y, en términos de la esfera de bloch, una base computacional es un eje del espacio tridimensional en el que esta se representa.

La base computacional que se usa por norma general es la Z, por lo que a lo largo de este trabajo toda medición se realizará sobre esta base. Otras dos bases comunes son la X y la Y. Estas tres bases corresponden con los ejes del mismo nombre con el que acostumbramos a trabajar en los espacios tridimensionales.

Cada base está formada por dos kets diferentes. Los de las tres bases computacionales básicas son los siguientes:

$$\begin{aligned} &Base\ Z : |0\rangle\ y\ |1\rangle \\ &Base\ X : |+\rangle\ y\ |-\rangle \\ &Base\ Y : |\circlearrowleft\rangle\ y\ |\circlearrowright\rangle \end{aligned} \tag{3.8}$$

A parte de medir sobre Z, X o Y, las mediciones se pueden realizar sobre cualquier base que se desee.

3.1.4. Puertas lógicas de un cúbit

Definición y conceptos básicos

El concepto de puerta lógica cuántica es análogo al de puerta lógica clásica. Así, una puerta cuántica no es más que un elemento que permite realizar una operación sobre los elementos de entrada.

Estas puertas cuánticas se expresan como una matriz. Dado que un cúbit es un vector, la aplicación de la puerta al cúbit se realiza mediante la multiplicación matricial de estos dos elementos.

Si quisiéramos aplicar la puerta M al cúbit ψ lo representaremos de la manera siguiente:

$$M\,|\psi\rangle = |\psi'\rangle \tag{3.9}$$

Cabe destacar que no cualquier matriz es una puerta lógica válida. Recordemos que los cúbits siempre deben cumplir la condición de normalización, así que tenemos que asegurarnos de que todas las puertas (matrices) que apliquemos deben de preservar esta restricción, y la forma de hacerlo es hacer uso de matrices unitarias. Una matriz unitaria es aquella que cumple la siguiente condición:

$$AA^* = A^*A = I \tag{3.10}$$

Donde A* es el traspuesto conjugado de A e I es la matriz identidad. Por tanto, una matriz unitaria cumple la condición de que su matriz inversa es igual a su matriz traspuesta conjugada.

También será útil recordar brevemente los autovectores y autovalores. Observando la siguiente expresión:

$$A\,|\psi\rangle = \lambda\,|\psi\rangle \tag{3.11}$$

Donde A es una matriz cuadrada (en lo que nos ocupa A podría ser una matriz unitaria), lambda es un valor y (evidentemente) el $|\psi\rangle$ es un vector. Cuando esta igualdad se cumple podemos decir que λ es un autovalor de la matriz A y que el vector ψ es un autovector asociado a λ.

Trabajando con cúbits, si la matriz A es una puerta lógica (una matriz unitaria) para la que existen valores que cumplan la igualdad (3.10), podemos decir que la aplicación de esa puerta produce el mismo efecto que su correspondiente autovalor λ.

Para visualizar la aplicación de una puerta lógica, podemos pensar que al aplicarla sobre un cúbit este sufrirá una rotación sobre la esfera de Bloch.

En próximos apartados veremos las principales puertas lógicas y también aquellas necesarias para la implementación del algoritmo de Shor.

Puertas de Pauli

Las puertas de Pauli son las puertas cuánticas más básicas. Hay una por cada eje (X, Y y Z), es decir, sobre cada una de las bases computacionales. Cada una aplica una rotación de π radianes sobre uno de estos eje.

La primera de ellas es la puerta X, esta realiza una rotación de π radianes sobre el eje X. Esta rotación conlleva que si aplicamos la puerta X sobre el ket 0 obtengamos el ket 1. Y si lo aplicamos sobre el ket 1 obtenemos el ket 0. Este comportamiento hace a esta puerta análoga a una puerta NOT clásica.

La puerta X tiene asociada la matriz:

$$X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \tag{3.12}$$

Aplicada a los kets de las principales bases computacionales:

	$\|0\rangle$	$\|1\rangle$	$\|+\rangle$	$\|-\rangle$	$\|\circlearrowleft\rangle$	$\|\circlearrowright\rangle$
X	$\|1\rangle$	$\|0\rangle$	$\|+\rangle$	$\|-\rangle$	$\|\circlearrowright\rangle$	$\|\circlearrowleft\rangle$

En segundo lugar tenemos la puerta Y. Realiza una rotación de π radianes sobre el eje Y. Su expresión matricial es de la forma:

$$Y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix} \tag{3.13}$$

Aplicada a los kets de las principales bases computacionales:

	$\|0\rangle$	$\|1\rangle$	$\|+\rangle$	$\|-\rangle$	$\|\circlearrowleft\rangle$	$\|\circlearrowright\rangle$
Y	$\|1\rangle$	$\|0\rangle$	$\|-\rangle$	$\|+\rangle$	$\|\circlearrowleft\rangle$	$\|\circlearrowright\rangle$

Por último tenemos la puerta Z. Totalmente análoga a las dos puertas anteriores, realiza una rotación de π radianes sobre el eje Z.

$$Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \tag{3.14}$$

Aplicada a los kets de las principales bases computacionales:

	$\|0\rangle$	$\|1\rangle$	$\|+\rangle$	$\|-\rangle$	$\|\circlearrowleft\rangle$	$\|\circlearrowright\rangle$
Z	$\|0\rangle$	$\|1\rangle$	$\|-\rangle$	$\|+\rangle$	$\|\circlearrowright\rangle$	$\|\circlearrowleft\rangle$

Puerta de Hadamard

La puerta de Hadamard [7], también conocida como la puerta H, es una puerta cuántica fundamental y que muy probablemente estará presente en todos los circuitos cuánticos que desarrollemos. Lo más importante es que nos permite producir un estado de superposición. Si lo pensamos en términos de la esfera de Bloch, la puerta H nos permite alejarnos de los polos de la esfera.

Su expresión matricial:

$$H = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \tag{3.15}$$

Aplicada a los kets de las principales bases computacionales:

	$\|0\rangle$	$\|1\rangle$	$\|+\rangle$	$\|-\rangle$	$\|\circlearrowleft\rangle$	$\|\circlearrowright\rangle$
H	$\|+\rangle$	$\|-\rangle$	$\|0\rangle$	$\|1\rangle$	$\|\circlearrowright\rangle$	$\|\circlearrowleft\rangle$

Por ejemplo, si aplicamos esta puerta al ket cero obtendremos un estado superpuesto donde la probabilidad de medir el ket cero o el ket uno serán iguales, es decir:

$$|\psi\rangle = \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \tag{3.16}$$

Técnicamente se realiza una rotación de π radianes sobre el eje $\frac{x+z}{\sqrt{2}}$

Esta operación también se puede entender como un cambio entre la base computacional Z y la base computacional X. Es decir, nos permite pasar de una de estas bases a la otra, por ejemplo, al aplicar la puerta H sobre el ket cero nos produce el ket análogo en la base X, el ket positivo.

Puerta de desplazamiento de fase

La puerta de desplazamiento de fase es una puerta parametrizada, es decir, recibe un parámetro ϕ. Por esto también se conoce como la puerta R_ϕ.

El efecto que produce es una rotación de ϕ radianes sobre el eje Z, por lo que ϕ debe ser real. Se expresa de la siguiente forma:

$$R_\phi = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\phi} \end{bmatrix} \tag{3.17}$$

Si prestamos atención a la rotación que produce veremos que la puerta Z es un caso particular de la puerta de desplazamiento de fase donde $\phi = \pi$. Otros casos particulares son la puerta S ($\phi = 2\pi$.) y la puerta T ($\phi = 4\pi$).

3.2. Sistemas de múltiples cúbits

3.2.1. Definición

Dado que un cúbit puede estar en una superposición de sus dos posibles estados, estos se definen usando dos valores pertenecientes a los números complejos. De igual manera, un sistema de múltiples cúbits se define con tantos números complejos como posibles estados tenga el sistema. Así, un sistema con n cúbits, tendrá 2^n posible estados.

Por ejemplo, si tenemos dos cúbits tendremos que describir el sistema con cuatro números complejos.

$$\begin{aligned} &|\psi\rangle = \alpha_1 |00\rangle + \alpha_2 |01\rangle + \alpha_3 |10\rangle + \alpha_4 |11\rangle \\ &\alpha_1, \alpha_2, \alpha_3, \alpha_4 \in \mathbb{C} \\ &|\alpha_1|^2 + |\alpha_2|^2 + |\alpha_3|^2 + |\alpha_4|^2 = 1 \end{aligned} \tag{3.18}$$

Al igual que cuando teníamos un único cúbit, este sistema también se expresa de manera vectorial

$$\begin{bmatrix} \alpha_{00} \\ \alpha_{01} \\ \alpha_{10} \\ \alpha_{11} \end{bmatrix} \tag{3.19}$$

3.2.2. Puertas lógicas de múltiples cúbits

La mayoría de las puertas lógicas de múltiples cúbits son lo que se conoce como puertas controladas.

Las puertas controladas operan sobre varios cúbits, de los cuales uno o más controlan la operación. Es decir, dependiendo del valor de los cúbits de control se realiza o no una operación sobre los otros cúbits, que son llamados objetivo.

Puerta CNOT

La puerta CNOT (Controlled-NOT) afecta a dos cúbits, donde uno de ellos es el cúbit de control y el otro el cúbit objetivo. Si el cúbit de control es el ket uno, se aplica una puerta X (una operación NOT) sobre el cúbit objetivo. En caso contrario no se aplica ninguna operación.

Su matriz es la siguiente, al afectar a dos cúbits es una matriz de dimensiones 4x4:

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \tag{3.20}$$

Cuando estamos en el caso de dos cúbits, observar el efecto de la puerta CNOT sobre los autovectores del sistema puede ser interesante para comprender su funcionamiento. El primer cúbit es el cúbit objetivo y el segundo es de control.

$$|\psi\rangle = \begin{bmatrix} \alpha_{00} \\ \alpha_{01} \\ \alpha_{10} \\ \alpha_{11} \end{bmatrix} \qquad\qquad CNOT\,|\psi\rangle = \begin{bmatrix} \alpha_{00} \\ \alpha_{11} \\ \alpha_{10} \\ \alpha_{01} \end{bmatrix} \tag{3.21}$$

Como vemos, el efecto que realiza la puerta CNOT es intercambiar los valores a_{11} y a_{01}.

Puerta SWAP

La puerta SWAP es una puerta muy simple, ya que nos permite intercambiar el valor de dos cúbits. Es decir, ante la entrada $|\psi_1\psi_2\rangle$ producirá una salida con los valores intercambiados, esto es $|\psi_2\psi_1\rangle$.

Su forma matricial es la siguiente:

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.22}$$

Puerta cambio de fase controlada

La puerta de cambio de fase controlada afecta a dos cúbits e implementa una rotación con control. Es una puerta vital para implementar la Transformación Cuántica de Fourier,

fundamental para muchos algoritmos, entre ellos el algoritmo de Shor. Se expresa de la siguiente manera:

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & e^{i\phi} \end{bmatrix} \tag{3.23}$$

La acción de esta puerta sobre dos cúbits $|\psi_1\psi_2\rangle$, donde ψ_1 es el cúbit de control es la siguiente:

$$\begin{array}{l} |\psi_1, e^{i\phi}\psi_2\rangle \quad si\, |\psi_1, \psi_2\rangle = |1,1\rangle \\ |\psi_1, \psi_2\rangle \quad en\ el\ resto\ de\ casos \end{array} \tag{3.24}$$

Como vemos no es una puerta controlada al uso, ya que solo actúa sobre el estado $|11\rangle$.

3.3. Entrelazamiento cuántico

El entrelazamiento cuántico es un fenómeno en el que los elementos de un sistema no pueden ser definidos individualmente, sino que deben expresarse en conjunto. Es decir, los elementos del sistema están "ligados" y es imposible definir por separado el estado de cada uno de ellos, ya que en la definición de su estado intervienen los otros elementos del sistema.

Un sistema de, por ejemplo, dos cúbits, se puede intentar expresar mediante la definición individual de cada uno de los cúbits. Si tenemos el siguiente estado:

$$|\psi_1\rangle = \frac{1}{\sqrt{2}}|00\rangle + \frac{1}{\sqrt{2}}|01\rangle \tag{3.25}$$

Se puede expresar en base a sus cúbits individuales:

$$|\psi_1\rangle = |0\rangle \otimes (\frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle) \tag{3.26}$$

Ya que hemos expresado el sistema en base a sus elementos de manera individual, por definición, este sistema no se encuentra en un estado de entrelazamiento.

En cambio, intentemos expresar mediante sus cúbits individuales el siguiente estado:

$$|\psi_2\rangle = \frac{1}{\sqrt{2}}|00\rangle + \frac{1}{\sqrt{2}}|11\rangle \tag{3.27}$$

Veremos que no es posible, no hay estados posibles para los cúbits que nos permitan definir el sistema en base a ellos. Por lo tanto el sistema $|\psi_2\rangle$ se encuentra en un estado de entrelazamiento.

Esta situación tiene implicaciones interesantes. Si miramos con atención el sistema $|\psi_2\rangle$, veremos que si medimos uno de los dos cúbits, sabremos automáticamente el estado

del otro (a pesar de que ambos se encuentran inicialmente en un estado de superposición). Es decir, si medimos un cúbit y obtenemos el $|0\rangle$, sabemos que el otro también debe ser el $|0\rangle$ ya que no hay ninguna otra posibilidad (los estados $|01\rangle$ y $|10\rangle$ tienen probabilidad 0). De manera análoga si medimos y obtenemos el $|1\rangle$, el otro tendrá el mismo valor.

Cabe destacar que a pesar de que ambos cúbits estén separados por una distancia arbitrariamente larga, la medición (el colapso) de uno determinará inmediatamente el estado del otro.

El hecho de que un sistema esté entrelazado también tiene implicaciones en su visualización, ya que no podemos usar la esfera de Bloch para mostrar los diferentes cúbits. Si lo intentamos obtendremos que no existen cúbits para formar ese estado, se puede ver en la figura 3.2.

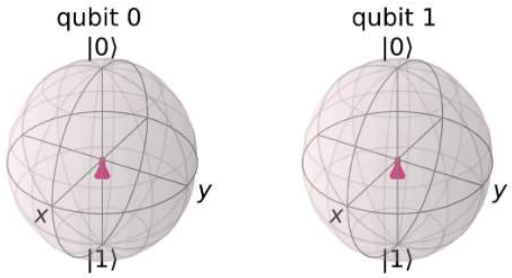


Figura 3.2: Representación de un estado entrelazado en la esfera de Bloch

Como vemos, no existe un estado posible para los cúbits de manera individual y por tanto no se representan.

Para solucionar esto se hace uso de la Esfera Q. En esta esfera las diferentes amplitudes se representan como círculos sobre su superficie, siendo el tamaño y el color proporcionales a su magnitud y fase respectivamente. Así, el estado (3.27) se representa en la Esfera Q como en la figura 3.3.

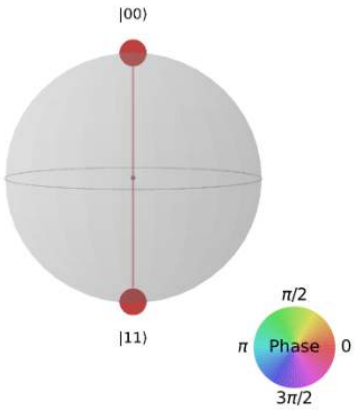


Figura 3.3: Representación de un estado entrelazado en la Esfera Q

Como se ve obtenemos una representación donde hay dos círculos de igual tamaño que representan los estados $|00\rangle$ y $|11\rangle$.

3.4. Contragolpe de fase cuántico

El contragolpe de fase cuántico (*phase kickback*) es un efecto muy importante y usado en una gran cantidad de algoritmos cuánticos. Este efecto ocurre cuando se aplica una puerta lógica controlada (es decir, una matriz unitaria) y la fase (el autovalor) que produciría en el cúbit objetivo se produce en el cúbit de control.

Cuando el cúbit de control es $|0\rangle$ o $|1\rangle$ este efecto no se observa, simplemente se aplica el efecto correspondiente al cúbit objetivo tal y como nuestra intuición nos dice que debe funcionar una puerta controlada.

En cambio, cuando el cúbit de control está en un estdo de superposición (por ejemplo $|+\rangle$), la componente $|1\rangle$ del cúbit aplica el efecto a través de la puerta controlada, pero la componente $|0\rangle$ no lo aplica. Por lo tanto encontramos que solamente una de las componentes del cúbit de control hace que se aplique una puerta sobre el cúbit objetivo. Este hecho hace que el cúbit de control reciba una fase sobre su componente $|1\rangle$.

Para comprender este efecto es esclarecedor ver un pequeño ejemplo. Trabajaremos con la puerta de cambio de fase controlada con un $\phi = \frac{\pi}{4}$ (también conocida como la puerta T).

La puerta T aplica una fase de $e^{\frac{i\pi}{4}}$. Por tanto, cumple la siguiente igualdad:

$$T|1\rangle = e^{\frac{i\pi}{4}}|1\rangle \tag{3.28}$$

Cuando aplicamos una puerta T controlada vemos que si el cúbit de control no está en superposición la aplicación de la fase ocurre con normalidad. Se asume que el cúbit de control es el derecho, es decir, $|x_{objetivo}y_{control}\rangle$

$$\begin{aligned} T|10\rangle &= |10\rangle = |1\rangle \otimes |0\rangle = |10\rangle \\ T|11\rangle &= T|1\rangle \otimes |1\rangle = e^{\frac{i\pi}{4}}|1\rangle \otimes |1\rangle = |11\rangle \end{aligned} \tag{3.29}$$

Ahora veamos qué ocurre cuando el cúbit de control se encuentra en un estado de superposición:

$$\begin{aligned} T|1+\rangle = \frac{1}{\sqrt{2}}(|10\rangle + e^{\frac{i\pi}{4}}|11\rangle) = \\ |1\rangle \otimes \frac{1}{\sqrt{2}}(|0\rangle + e^{\frac{i\pi}{4}}|1\rangle) = \\ |1\rangle \otimes \frac{1}{\sqrt{2}}(|0\rangle + e^{\frac{i\pi}{4}}|1\rangle) \end{aligned} \tag{3.30}$$

En este caso el primer cúbit ($|1\rangle$) sufre la puerta T como una aplicación de una fase global que no es observable, por lo que se queda como $|1\rangle$. En cambio, el cúbit de control recibe esa fase (el autovalor de la puerta T) como fase relativa, ya que solo la experimenta su componente $|1\rangle$ y por tanto no es global.

Así se ve como el cúbit de control de una puerta lógica puede también sufrir modificaciones en su estado.

Otro aspecto importante a destacar es en qué situaciones el cúbit de control recibe la fase de manera completa y en qué situaciones recibe la fase de manera parcial. Como se vio en el desrrollo (3.30), el cúbit de control recibe la fase al completo, es decir, $e^{\frac{i\pi}{4}}$. Esto es consecuencia de que el cúbit de control sea el $|1\rangle$ y pueda extraerse como factor común en la expresión. Veamos qué pasa si el cúbit objetivo no es el $|1\rangle$ sino, por ejemplo, el $|+\rangle$.

$$T\,|{+}{+}\rangle = \frac{1}{2}(|00\rangle + |10\rangle + |01\rangle + e^{\frac{i\pi}{4}}\,|11\rangle) \tag{3.31}$$

Recordemos que la puerta de fase controlada solo aplica su efecto cuando el estado del sistema es $|11\rangle$, por lo que solo una de las cuatro componentes recibe la fase $e^{\frac{i\pi}{4}}$.

La componente $|1\rangle$ del cúbit de control no recibe está vez toda la fase, sino una parte de ella, ya que existen diferentes componentes $|1\rangle$ del cúbit de control y solo algunas contienen esta fase.

Esto será importante ya que en el futuro, cuando queramos hacer uso del contragolpe de fase, querremos que nuestro cúbit de control reciba la fase completa, haciendo una necesidad el hecho de que los cúbits objetivo tengan un valor de $|1\rangle$.

Capítulo 4

Herramientas

4.1. Qiskit

Para la implementación del algoritmo de Shor se usará la herramienta Qiskit [1] [9]. Este es un framework basada en la librería OpenQASM y que fue desarrollada por IBM para el desarrollo de software cuántico. Qiskit se encuentra disponible para diferentes lenguajes de programación, en este trabajo lo usaremos desde el lenguaje Python.

Qiskit permite trabajar a diferentes niveles, tanto a bajo nivel como a alto nivel usando una abstracción que permite su uso a usuarios sin experiencia en cuántica. Así, este framework consta de cuatro componentes principales:

- Terra: creación de circuitos a bajo nivel
- Aqua: herramientas que no usan programación cuántica explícita
- Aer: Simulación de dispositivos cuánticos
- Ignis: herramientas relacionadas con el ruido en los dispositivos

Los circuitos cuánticos implementados en Qiskit pueden ser ejecutados en nuestro propio ordenador, simulando un computador cuántico gracias al componente Aer de Qiskit, o bien en un computador cuántico real ofrecido por IBM. Esta última manera de ejecución se basa en los IBM Quantum Services, donde se ofrecen diferentes máquinas para que los usuarios puedan experimentar con sus circuitos en un computador cuántico real. Como se mencionó en el estado del arte, se ofrecen desde computadores de un cúbit hasta computadores de sesenta y cinco.

A lo largo del trabajo se ha hecho uso de ambas aproximaciones. A medida que los diferentes circuitos se han ido construyendo estos han sido simulados en un computador clásico. Así nos podemos cersiorar de la correcta implementación de los mismos de una manera fácil y rápida. Cuando el algoritmo ha sido implementado y las ejecuciones en el simulador se han mostrado correctas se ha pasado a ejecutar la implementación del algoritmo en un computador ofrecido en los IBM Quantum Services.

El interés de ejecutar nuestros circuitos cuánticos usando tanto simuladores como computadores reales es debido a que los resultados obtenidos no son iguales. En un computador cuántico existen factores como ruido y posibles errores en el cómputo, lo que hará que cuando estudiemos las probabilidades de los posibles resultados veamos estados que no debería ser posible obtener a partir del circuito ejecutado. Evidentemente estos factores no están presentes en el simulador.

Así todo cabe destacar que, como es de esperar, estos resultados producto de factores no deseados aparecen con una probabilidad residual, mucho menos que los resultados correctos. En la sección 4.4 se observa este comportamiento con un sencillo ejemplo.

El computador cuántico usado en el presente trabajo ha sido *ibmq_16_melbourne*, que consta de 15 cúbits

4.2. IBM Quantum Experience

En este trabajo se ha hecho uso de las herramientas proporcionadas por IBM Quantum Experience, es decir, la herramienta IBM Quantum Composer e IBM Quantum Lab.

IBM Quantum Composer es una interfaz gráfica que nos permite construir y visualizar circuitos de manera sencilla, así como mostrarnos información de manera inmediata sobre los autovectores, las esferas de bloch de los diferentes cúbits, etc...

Por otro lado, IBM Quantum Lab permite implementar y ejecutar código como si de un cuaderno Jupyter se tratara.

Ambas herramientas han sido utilizadas para el desarrollo y construcción de todos los circuitos mostrados a lo largo del trabajo, tanto los circuitos de ejemplo como los pertenecientes al algoritmo de Shor

4.3. Puertas lógicas en Qiskit

En la siguiente tabla se muestra la representación de las puertas lógicas en Qiskit. Más concretamente su representación en la herramienta IBM Quantum composer, ya que esta será la que se utilizará en los próximos capítulos de este trabajo.

En las puertas controladas que se han estudiado aparece un cúbit de control y un cúbit objetivo. El primero de estos se representa como un circulo de pequeño tamaño y color entero, el cúbit objetivo se muestra con la puerta lógica que se implementa.

Puerta	Representación
X	
Y	
Z	
H	
R_ϕ	
CNOT	
SWAP	
Fase controlada	

Tabla 4.1: Puertas lógicas y sus representaciones en IBM Quantum Composer

4.4. Circuito cuántico en Qiskit

Utilizando los conceptos presentados se muestra la implementación de un sencillo circuito. Para esto se hará uso de los conceptos explicados en el capítulo anterior y de las herramientas presentadas en el capítulo actual.

En el punto 3.3 se explicó el entrelazamiento cuántico, uno de los principios más importantes de la computación cuántica. Esto nos permitía tener dos cúbits en un estado combinado que no se podía expresar como estados separados. Los estados entrelazados principales y más simples se conocen como *Estados de Bell*:

$$\begin{aligned} |\psi_1\rangle &= \frac{1}{\sqrt{2}}|00\rangle + \frac{1}{\sqrt{2}}|11\rangle \\ |\psi_2\rangle &= \frac{1}{\sqrt{2}}|01\rangle + \frac{1}{\sqrt{2}}|10\rangle \end{aligned} \tag{4.1}$$

Así pues, vamos a implementar un circuito en Qiskit que nos produzca ψ_1. Aplicando una pequeña modificación veremos que también podemos obtener ψ_2. [13]

Nuestro circuito constará de dos cúbits. Por lo que inicialmente el sistema se expresará de la siguiente manera:

$$|00\rangle \tag{4.2}$$

Las puertas necesarias para lograr el entrelazamiento cuántico y llegar al estado de Bell deseado son muy sencillas. En primer lugar aplicamos una puerta Hadamard (la gran mayoría de los circuitos cuánticos comienzan con la aplicación de puertas H dado que nos producen un estado de superposición cuántica) al primer cúbit, así, podríamos expresar nuestro sistema de la forma:

$$|\psi\rangle = \frac{1}{\sqrt{2}}|00\rangle + \frac{1}{\sqrt{2}}|01\rangle \tag{4.3}$$

Como vemos, el segundo cúbit mantiene su valor, ya que en ambos miembros es el $|0\rangle$. En cambio, el primer cúbit puede valer el $|0\rangle$ o el $|1\rangle$ con la misma probabilidad, resultado de la superposición cuántica introducida por la puerta Hadamard aplicada.

A continuación aplicamos una puerta CNOT. El cúbit de control será el primero, es decir, el que se encuentra en un estado de superposición. Tras aplicar esta puerta el sistema queda en el siguiente estado:

$$|\psi\rangle = \frac{1}{\sqrt{2}}|00\rangle + \frac{1}{\sqrt{2}}|11\rangle \tag{4.4}$$

Como vemos, la aplicación de estas dos puertas nos ha conducido directamente a un estado entrelazado y más concretamente a uno de los estados de Bell. Recordemos que en este caso podemos ver fácilmente el entrelazamiento, ya que si por ejemplo el primer cúbit vale 0, el segundo necesariamente tendrá que valer 0 también ya que el ket $|10\rangle$ tiene probabilidad 0. Si el primer cúbit nos diera un uno el razonamiento sería análogo.

En la figura 4.1 se muestra una visualización de la configuración del circuito y los resultados que se espera obtener una vez y se hagan las mediciones al sistema.

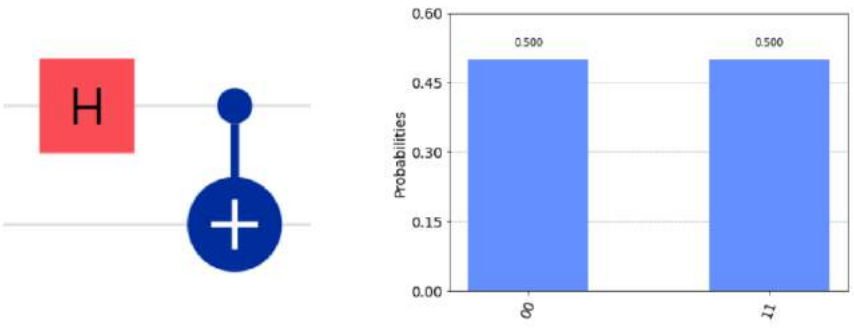


Figura 4.1: Circuito y probabilidades de medición en simulador

Como vemos los únicos estados posibles a medir son el $|00\rangle$ y el $|11\rangle$. Esto ocurre así porque se ha ejecutado en un simulador, cuando lo ejecutemos en un computador cuántico real veremos que los otros dos posibles estados ($|01\rangle$ y $|10\rangle$) pueden ser observados

también, aunque estos presenta una probabilidad mucho menor. Figura 4.2.

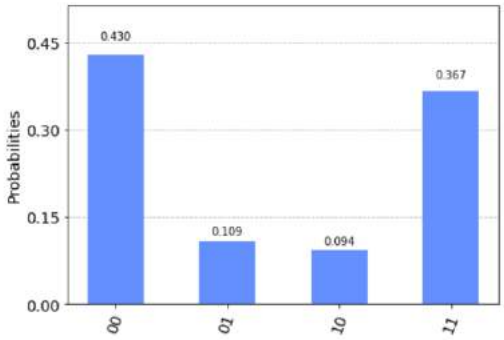


Figura 4.2: Probabilidades de medición en computador cuántico

Para obtener el otro estado de Bell basta con añadir una puerta X en el segundo cúbit. Obteniendo los siguientes resultados

$$|\psi\rangle = \frac{1}{\sqrt{2}}|01\rangle + \frac{1}{\sqrt{2}}|10\rangle \quad (4.5)$$

En la figura 5.2 observamos la ejecución en simulador con los resultados esperados y la ejecución en una máquina real con el ruido y los fallos de ejecución característicos. Figura 4.4.

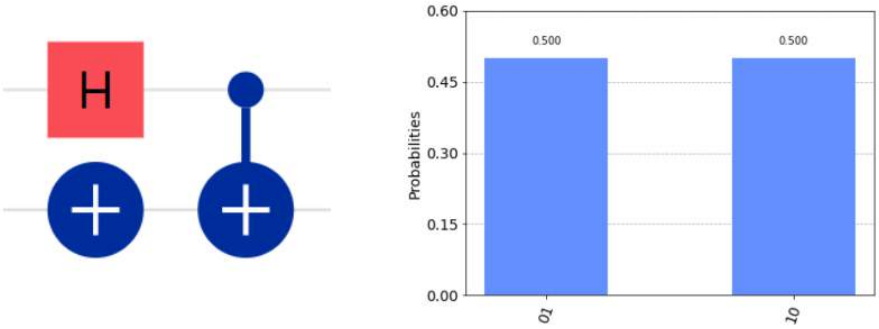


Figura 4.3: Circuito y probabilidades de medición en simulador

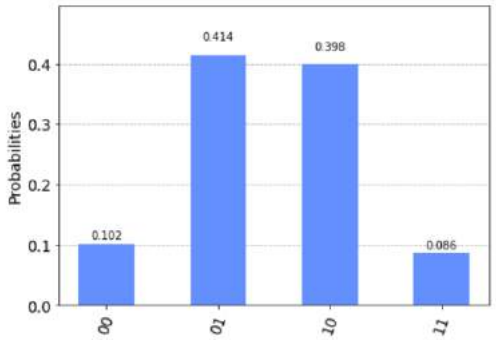


Figura 4.4: Probabilidades de medición en computador cuántico

Capítulo 5

Algoritmo de Shor

5.1. Definición del problema

El algoritmo de Shor [11] se encarga de resolver el problema de factorización de números enteros. Este consiste en descomponer un número entero compuesto en sus factores primos, de tal forma que cuando estos factores son multiplicados entre ellos dan como resultado el número original. Gracias al teorema fundamental de la aritmética, sabemos que esta descomposición es única.

Por ejemplo, el número 21 se descompone en en el número 7 y el número 3. Así, su producto nos da el número original

$$7 * 3 = 21$$

Dentro de la computación clásica este problema es muy costoso computacionalmente, es decir, no existe algoritmo que resuelva el problema de factorización de números enteros en tiempo polinomial.

Uno de los mejores algoritmos clásicos que existente actualmente es el algoritmo de *Criba general del cuerpo de números* (CGCN), un algoritmo sub-exponencial pero super-polinomial, cuyo tiempo asintótico es el siguiente:

$$O(exp((\frac{64}{9}b)^{\frac{1}{3}}(log(b)^{\frac{2}{3}})) \tag{5.1}$$

En cambio, dentro de la computación cuántica sí existe un algoritmo que resuleve el problema de factorización de números enteros en tiempo polinomial, el algoritmo de Shor. El tiempo asintótico del algoritmo de Shor es:

$$O(log(n)^3) \tag{5.2}$$

5.2. Componentes

Comenzaremos estudiando dos componentes necesarias para la implementación de nuestro algoritmo, la Transformada Cuántica de Fourier y la Estimación Cuántica de Fase.

5.2.1. Transformada Cuántica de Fourier

La Transformada Cuántica de Fourier (TCF) [10] es la implementación cuántica de la transformada de fourier discreta aplicada a las amplitudes de una función de onda. Forma parte de muchos algoritmos cuánticos y es una parte fundamental de algoritmo de Shor.

La TCF se aplica a un estado cuántico ψ_1 y produce como resultado un estado cuántico ψ_2. Matemáticamente la transformación se expresa de la siguiente manera:

$$y_k = \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} x_j e^{2i\pi \frac{jk}{N}} \tag{5.3}$$

Al igual que la puerta Hadamard transformaba un estado cuántico en la base computacional Z a un estado en la base X, la transformada de Fourier nos transforma un estado desde la base Z a la base de Fourier.

Para entender cómo funciona la base de Fourier fijémonos en los estados sucesivos que se producen en esta base si representamos los números desde 0 a $2^n - 1$.

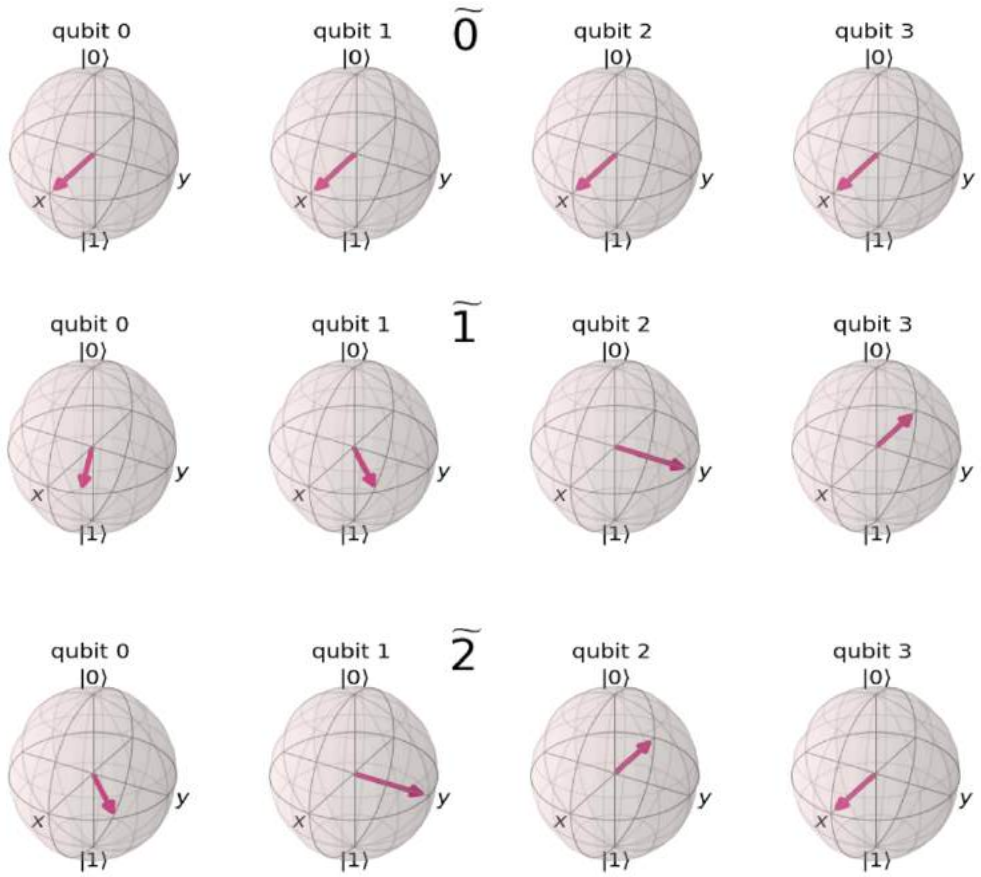


Figura 5.1: Sucesión de estados 0, 1 y 2 en la base de Fourier

El número que se representa determina el ángulo con el que cada uno de los cúbits está rotado alrededor del eje Z. Todos los cúbits comienzan en el estado $|+\rangle$ y se van rotando de manera progresiva. El cúbit menos significativo sufre, en cada incremento del número a representar, una rotación de $\frac{1}{2^n}2\pi$ radianes, de manera que cuando se llegue al último número representable y se vuelva a iniciar la cuenta este cúbit dará una vuelta entera.

Los demás cúbits realizan una rotación con valor del doble respecto al anterior cúbit. Es decir, el segundo cúbit rota $\frac{2}{2^n}2\pi$, el tercero $\frac{4}{2^n}2\pi$, etc... Así, el cúbit menos significativo tiene la menor frecuencia y el más significativo la mayor.

Partiendo de la expresión (5.3) vista al inicio de la sección y trabajando con ella, podemos llegar a una expresión equivalente que nos permitirá implementarla en un circuito cuántico. La nueva forma de la TCF es la siguiente:

$$\frac{1}{\sqrt{N}}(|0\rangle + e^{2i\pi\frac{x_0}{2}}|1\rangle) \otimes (|0\rangle + e^{2i\pi\frac{x_0}{2^2}+\frac{x_1}{2}})|1\rangle \otimes ... \otimes (|0\rangle + e^{2i\pi(\frac{x_0}{2^n}+...+\frac{x_{n-1}}{2})}|1\rangle) \tag{5.4}$$

Donde cada uno de los paréntesis de la forma $|0\rangle + e^x|1\rangle$ corresponde con cada uno de los cúbits, el primer término es el cúbit $n-1$ y el último el primer cúbit. Las operaciones que debemos realizar se pueden implementar usando solamente dos puertas, la puerta H y la puerta de cambio de fase controlada. Cabe destacar que la operación que realiza la puerta H es equivalente a:

$$H|x\rangle = \frac{1}{\sqrt{2}}(|0\rangle + e^{i\pi x})|1\rangle \tag{5.5}$$

Fijémonos en uno de los cúbits para ver cómo transformarlo a un circuito cuántico, en este caso el último término, que corresponde al primer cúbit:

$$\frac{1}{\sqrt{2}}(|0\rangle + e^{2i\pi(\frac{x_0}{2^n}+...+\frac{x_{n-1}}{2})}|1\rangle) \tag{5.6}$$

Parte de esta expresión se corresponde con el efecto de la puerta H mostrado en la ecuación (5.3). Así que si aplicamos esta puerta conseguimos:

$$\frac{1}{\sqrt{2}}(|0\rangle + e^{2i\pi(\frac{x_{n-1}}{2})})|1\rangle \tag{5.7}$$

Es decir, solo con la puerta H obtenemos gran parte de la expresión y el último término de la sucesión de fracciones del exponente. En el caso del primer término con esto sería suficiente para aplicar la expresión, en los demás términos debemos hacer uso de la puerta de cambio de fase controlada.

El siguiente término de la sucesión de fracciones que nos falta por conseguir (el penúltimo) es $e^{\frac{2i\pi x_{n-2}}{2^2}}$, por lo que si usamos una puerta de cambio de fase controlada

donde el cúbit de control es x_{n-2}, el objetivo es el cúbit con el que estemos trabajando y $\phi = \frac{\pi}{2}$, estaremos añadiendo el factor:

$$\frac{2i\pi x_{n-2}}{2^2} |1\rangle \tag{5.8}$$

Donde x_{n-2} aparece porque es el cúbit de control y solo afecta al $|1\rangle$ porque por definición la puerta solo produce su efecto cuando el estado de la pareja de cúbits es $|11\rangle$.

Así, cuando añadimos esta puerta a lo que habíamos obtenido aplicando la puerta H en la ecuación (5.5) tenemos:

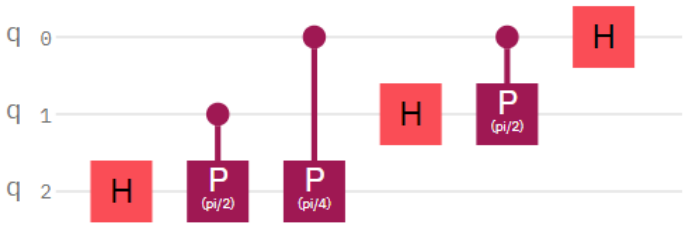
$$\frac{1}{\sqrt{2}}(|0\rangle + e^{2i\pi(\frac{x_{n-2}}{2^2} + \frac{x_{n-1}}{2})}) |1\rangle \tag{5.9}$$

Haciendo un proceso análogo, si aplicamos puertas de cambio de fase controladas sobre todos los cúbits obtendremos la expresión buscada:

$$e^{2i\pi(\frac{x_0}{2^n} + ... + \frac{x_{n-1}}{2})} |1\rangle) \tag{5.10}$$

Para tres cúbits el circuito correspondiente a este primer cúbit sería el siguiente:

Si completamos los otros cúbits haciendo el mismo razonamiento:



Ya tenemos todos los términos de la expresión (5.2). Pero si nos fijamos todavía no está terminado, ya que la el resultado correspondiente al primer cúbit está contenido en el último cúbit. En el caso de 3 cúbits esta es la única discordancia con (5.2) pero, si tuvieramos más cúbits todos los resultados estarían en orden inverso. Por tanto nos falta agregar las puertas SWAP correspondientes.

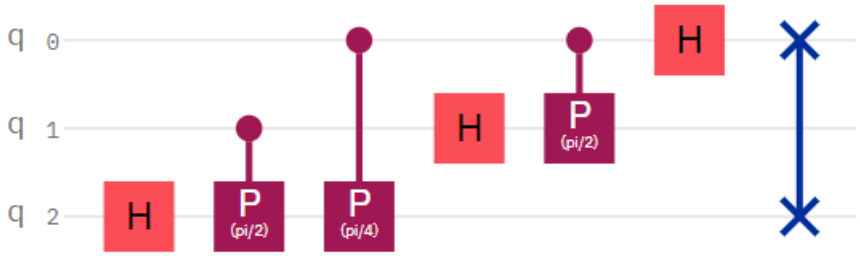


Figura 5.2: Ejemplo con tres cúbits de la TCF

Así logramos implementar satisfactoriamente la Transformada Cuántica de Fourier.

5.2.2. Estimación cuántica de fase

Al igual que la TCF, la Estimación Cuántica de Fase (ECF) es un proceso que aparece en muchos algoritmos cuánticos. Dado un operador (matriz) unitario U, la Estimación Cuántica de Fase se encarga de estimar un parámetro ϕ que aparece en el autovalor aplicado por el operador U. Es decir:

$$U|\psi\rangle = e^{2i\pi\phi}|\psi\rangle \tag{5.11}$$

La idea sobre la que se basa este proceso de estimación de fase es codificar la fase de U ($e^{2i\pi\phi}$) en un registro de t cúbits. Ahora bien, la parte importante es que esta codificación se llevará a cabo haciendo uso de la base de Fourier.

Una vez consigamos esto, tendremos que aplicar la Transformada Cuántica de Fourier Inversa (TCFI) para poder transformar la fase que queremos obtener desde la base de Fourier a nuestra base computacional.

Para lograr volcar la fase de U en nuestros cúbits haremos uso del concepto de Contragole de Fase. Cuando aplicamos una puerta U controlada (C-U), el cúbit de control va a recibir una fase proporcional a la fase aplicada por U, es decir, $e^{2i\pi\phi}$. De hecho, como se explicó en la sección 3.4, si el cúbit objetivo es $|1\rangle$ la fase que recibirá el cúbit de control es exactamente $e^{2i\pi\phi}$. Así, podemos aplicar tantas puerta C-U como necesitemos para codificar la fase ϕ en la base de Fourier.

El circuito contará con un registro de t cúbits que almacenarán la fase en la base de Fourier y un cúbit adicional ψ que será $|1\rangle$ y será el cúbit objetivo de todas las puertas C-U. Por tanto el estado inicial lo podemos expresar como:

$$|0\rangle^{\otimes t}|\psi\rangle \tag{5.12}$$

Posteriormente aplicamos una puerta H a cada uno de los primeros t cúbits.

$$\frac{1}{\sqrt{2^n}}(|0\rangle + |1\rangle)^{\otimes t}|\psi\rangle \tag{5.13}$$

A continuación aplicamos las puertas C-U. El primer cúbit controlará 2^0 C-U, el segundo 2^1 C-U, y así sucesivamente hasta que el último cúbit controlará un total de 2^{t-1} puertas C-U. La expresión matemática del circuito tras este paso es la siguiente:

$$\frac{1}{\sqrt{2^t}}(|0\rangle + e^{2i\pi\phi 2^{t-1}}|1\rangle) \otimes (|0\rangle + e^{2i\pi\phi 2^1}|1\rangle) \otimes (|0\rangle + e^{2i\pi\phi 2^0}|1\rangle) \otimes |\psi\rangle \tag{5.14}$$

Esta expresión se puede simplificar de la siguiente manera:

$$\frac{1}{\sqrt{2^t}} \sum_{k=0}^{2^n-1} e^{2i\pi\phi k}|k\rangle \otimes |\psi\rangle \tag{5.15}$$

Mirando con atención podemos ver que si en la expresión (5.15) sustituimos la sucesión de fracciones de los exponentes por $\phi\, 2^n$ donde $0 < n < t$ obtenemos la expresión (5.4). Es decir, hemos logrado la codificación de nuestra fase en la base de Fourier.

En la TCF el estado del cual se parte y el estado al que se transforma es $|x\rangle$, observable en la expresión (5.4) como una sucesión de fracciones en los exponentes. En el caso de la estimación cuántica de fase disponemos de una expresión equivalente, pero el estado que contiene nuestra expresión es $|2^n\phi\rangle$, como acabamos de ver en el párrafo anterior. Si queremos obtener este valor codificado en la base de Fourier debemos aplicar la Transformada Cuántica de Fourier inversa, que nos permite obtener:

$$\frac{1}{2^n} \sum_{x=0}^{2^n-1} \sum_{k=0}^{2^n-1} e^{-\frac{2i\pi k}{2^n}(x-2^n\phi)}|x\rangle \otimes |\psi\rangle \tag{5.16}$$

La expresión (5.16) tiene su máximo de probabilidad cuando $x = 2^t\phi$. Por lo tanto en este punto el circuito implementado es el siguiente:

$$|2^n\phi\rangle \otimes |\psi\rangle \tag{5.17}$$

Por lo tanto, estamos obteniendo $|2^n\phi\rangle$, donde aparece el factor buscado, es decir, ϕ.

Veamos un circuito de estimación cuántica de fase con un registro t de 3 cúbits donde la puerta U aplicada es una puerta de cambio de fase controlada con $\phi = \frac{\pi}{2}$ (Puerta S) sobre el estado $|1\rangle$, es decir:

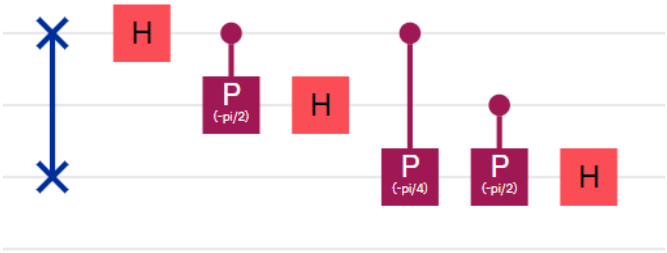
$$S|1\rangle = e^{\frac{i\pi}{2}}|1\rangle \tag{5.18}$$

Dado que esta puerta aplica un cambio de fase de $e^{\frac{i\pi}{2}}$, podemos saber de antemano el valor de ϕ que queremos obtener:

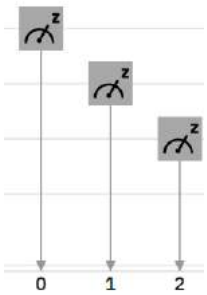
$$\begin{aligned} e^{\frac{i\pi}{2}} &= e^{2i\pi\phi} \\ \phi &= \frac{1}{4} \end{aligned} \tag{5.19}$$

Codificación de fase:

Transformada cuántica inversa:



Medición:



Para la puerta elegida en este ejemplo (puerta S), la medición arroja un valor de 8. Así vemos que:

$$\begin{aligned} 2^3\phi &= 8 \\ \phi &= \frac{1}{4} \end{aligned} \tag{5.20}$$

Por lo tanto, la estimación de fase es correcta respecto a lo que se esperaba obtener en la expresión (5.16).

Algo importante a destacar es que no siempre obtendremos un resultado exacto como el del ejemplo que se acaba de realizar. La precisión del resultado obtenido depende del número de cúbits t que usemos. Cuanto mayor sea este registro mayor será la precisión del resultado.

5.3. Búsqueda de periodo en el algoritmo de Shor

Ahora que conocemos y disponemos de la Transformada cuántica de Fourier, vista en la sección (5.3.1) y de la Estimación cuántica de fase, presentada en la sección (5.3.2), podemos comenzar a estudiar la parte principal del algoritmo de Shor, que resuelve el problema de la búsqueda del período.

El problema de descomposición de números enteros se puede transformar en un problema de búsqueda de período en tiempo polinomial. Por lo que si logramos resolver este segundo problema en tiempo polinomial, habremos logrado un algoritmo para factorizar números enteros de manera eficiente.

La función de la cual debemos obtener su periodo es de la siguiente forma:

$$f(x) = a^x mod N \tag{5.21}$$

Donde a y N pertenecen a los números naturales (enteros positivos), a es menor que N, y no tienen factores comunes.

Para hallar su periodo debemos obtener el entero positivo más pequeño distinto a cero tal que:

$$a^r mod N = 1 \tag{5.22}$$

Para resolver el problema recien planteado debemos hacer uso de la estimación cuántica de fase que vimos anteriormente. El operador unitario U con el que trabajaremos será de la forma:

$$U\left|y\right\rangle = \left|ay \ mod \, N\right\rangle \tag{5.23}$$

Si aplicamos este operador sucesivas veces:

$$U^q\left|y\right\rangle = \left|a^q y \ mod \, N\right\rangle \tag{5.24}$$

Obtenemos una expresión equivalente a la (5.20).

Veamos un ejemplo de aplicación del operador U sobre el ket $|1\rangle$. Usaremos $a = 5$ y $N = 13$:

$$\begin{aligned} U\left|1\right\rangle &= \left|5\right\rangle \\ U\left|5\right\rangle &= \left|12\right\rangle \\ U\left|12\right\rangle &= \left|8\right\rangle \\ U\left|8\right\rangle &= \left|1\right\rangle \end{aligned} \tag{5.25}$$

En este caso tras aplicar cuatro veces el operador unitario U hemos obtenido de nuevo el ket inicial ($|1\rangle$). Por lo tanto, el período para este caso en concreto es 4, es decir, $r = 4$.

Trabajando con los posibles autovectores del operador U, encontramos que una superposición de todos los estados pertenecientes al ciclo es un posible autovector.

$$|\psi\rangle = \frac{1}{\sqrt{r}} \sum_{k=0}^{r-1} |a^k \, mod \, N\rangle \tag{5.26}$$

Al aplicar U a nuestro estado $|\psi\rangle$ descubrimos que su autovalor es 1. Este autovalor no es demasiado interesante y tendremos que trabajar con el estado $|\psi\rangle$ hasta que obtengamos un autovalor más conveniente. Por ejemplo, el siguiente estado $|\psi_s\rangle$ aplica una fase diferente a cada uno de los estados del ciclo:

$$|\psi_s\rangle = \frac{1}{\sqrt{r}} \sum_{k=0}^{r-1} e^{-\frac{2i\pi ks}{r}} \, |a^k \, mod \, N\rangle \tag{5.27}$$

Cuyo autovalor es:

$$e^{\frac{2i\pi s}{r}} \tag{5.28}$$

Donde s es un entero tal que $0 \leqslant s \leqslant r-1$. Este autovalor de U es muy conveniente ya que contiene el período r (necesario para que las diferencias entre las fases aplicadas a todos los estados sean inguales). Además, si sumamos todos los estados $|\psi_s\rangle$ para los posibles valores de s obtenemos:

$$\frac{1}{\sqrt{r}} \sum_{s=0}^{r-1} |u_s\rangle = |1\rangle \tag{5.29}$$

Es decir, todos los estados se cancelan menos el $|1\rangle$. Esto es extremadamente conveniente ya que nos permite aplicar un contragolpe de fase completo.

Dada la forma de la fase (autovalor) aplicada por U ($e^{2i\pi\phi}$), que es exactamente igual al tipo de fase con el que se trabaja en la estimación cuántica de fase, expresión (5.11), y que la superposición de todos los estados $|u_s\rangle = |1\rangle$ (condición necesaria para aplicar procesos que hagan uso del contragolpe de fase), podemos aplicar una estimación cuántica de fase. La estructura del circuito será la de la figura 5.3.

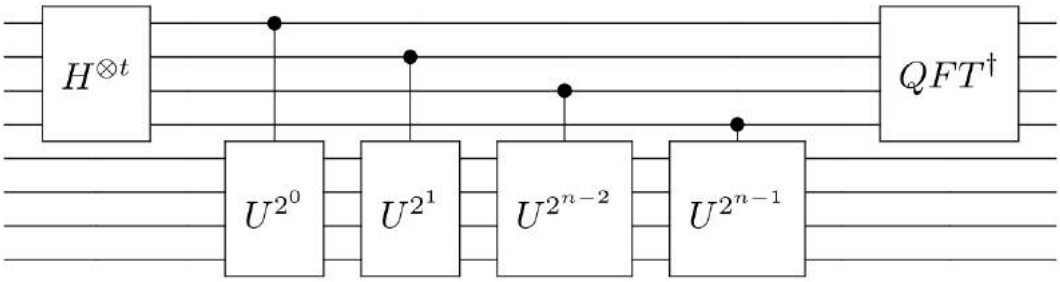


Figura 5.3: Esquema general del Algoritmo de Shor

Lo que mediremos será $\phi = \frac{s}{r}$, que no es lo que buscamos pero sí que contiene r. Para obtener r de esta expresión se aplicará el algoritmo de Fracciones Continuas.

Esta obtención del período r formaría el módulo cuántico del algoritmo de Shor. La obtención de los factores a partir del período forma parte del módulo clásico del algoritmo.

5.4. Obtención de los factores

Una vez hemos obtenido el período de nuestra función (5.21), podemos obtener los factores buscados mediante la aplicación del siguiente proceso:

En primer lugar, debemos recordar la expresión (5.22):

$$a^r mod N = 1$$

De esta igualdad se desprende la siguiente:

$$(a^r - 1) mod N = 0 \tag{5.30}$$

Así sabemos que N es un divisor de $(a^r - 1)$. Esta información es muy valiosa para optener los factores de N

Una condición necesaria para poder seguir con este procedimiento es que r sea un número par. Si esto es cierto podemos expresar $(a^r - 1)$ de la siguiente forma:

$$(a^r - 1) = (a^{\frac{r}{2}} + 1)(a^{\frac{r}{2}} - 1) \tag{5.31}$$

Si r no fuera un número par tendríamos que seleccionar un nuevo valor de a y repetir el procedimiento. Esto habrá de hacerse hasta que obtengamos un período r que sea par.

Una vez tenemos la expresión anterior, se concluye que el máximo común divisor entre N y $(a^{\frac{r}{2}} + 1)$ será un factor de N. Así mismo, el máximo común divisor entre N y $(a^{\frac{r}{2}} - 1)$ nos proporcionará el otro factor, descomponiendo así satisfactoriamente N en sus dos componentes.

Capítulo 6

Implementación del algoritmo de Shor

6.1. Diseño

Una vez que se han expuesto las bases y el funcionamiento del algoritmo de Shor se va a proceder a exponer la implementación y ejecución de un caso concreto del mismo. Todo el código desarrollado se encuentra disponible en un repositorio github [6].

Lo primero es determinar el número N que se desea descomponer. Dada la cantidad y calidad limitada de los cúbits existentes en las computadoras cuánticas actuales el número a descomponer no debe ser demasiado grande (ya se explico en el estado del arte la imposibilidad de aplicar este algoritmo para casos reales).

Para saber cuántos cúbits son necesarios -aproximadamente- para descomponer cierto número podemos aplicar una sencilla regla. Vamos a obtener el siguiente factor:

$$m = log_2(N) \tag{6.1}$$

Este será un factor determinante en la construcción de nuestro circuito. Las puertas controladas U deberán tener m cúbits objetivo. Por otro lado, los cúbits encargados de codificar la fase a estimar en la ECF (cúbits control de las puertas U) deben de ser $2m$ cúbits. Por lo tanto para descomponer un número N necesitaremos un total de $3m = 3log_2(N)$ cúbits.

En este trabajo se implementará el circuito cuántico que descompondrá el número 15, siendo este el producto de los factores primos $3, 5$.

El $log_2(15)$ es $3{,}9$, así que tomaremos como valor m el número 4. Nuestro circuito dispondrá de un total de 12 cúbits, un tamaño válido para los computadores cuánticos actuales.

Ya tenemos fijado nuestro valor N. El valor a debe ser un número natural, menor que N y que no comparta con él ningún factor común. Es posible que escojamos un determinado valor y descubramos que el período r que obtenemos no es par, en ese caso tendremos que escoger un nuevo valor de a.

Para comenzar a trabajar usaremos $a = 7$. Ya que cumple los requisitos propuestos.

6.2. Construcción

En el capítulo anterior se explicaron los diferentes elementos necesarios para poder construir un circuito cuántico que implemente el algoritmo de Shor. La mayoría de ellos los podremos utilizar directamente en nuestro circuito específico para factorizar el número 15.

En primer lugar, la Transformada Cuántica de Fourier Inversa la implementaremos en nuestro circuito de la misma manera que se ha explicado anteriormente, no existe ninguna diferencia entre la TCFI que se implementa en el circuito cuántico para descomponer el número 15 respecto al que se implementa en el circuito cuántico para descomponer el 35.

A parte de la TCFI, gran parte del circuito de Estimación Cuántica de Fase lo implementaremos de la misma manera. Pero aquí, debemos tener en cuenta que este circuito hace uso de la puerta U controlada la cual usamos para estimar la fase deseada. Esta puerta sí que realiza una operación diferente dependiendo del número a factoriar, en el circuito de factorización del 15 se podría usar $a = 7$ y la puerta U correspondiente realizaría la siguiente operación:

$$U\left|\psi\right\rangle = \left|7\psi \, mod(15)\right\rangle \tag{6.2}$$

En cambio, en el circuito de factorización del número 35 usando $a = 4$ la puerta U realiza la operación:

$$U\left|\psi\right\rangle = \left|4\psi \, mod(35)\right\rangle \tag{6.3}$$

Por lo tanto, antes de poder implementar la estimación del perído r (y así, el módulo cuántico del algoritmo) debemos implementar la puerta C-U correspondiente, en este caso, la que realiza la operación (6.2)

Así, el circuito que implementa la operación $U\left|\psi\right\rangle = \left|7\psi \, mod(15)\right\rangle$ es el de la figura 6.1:

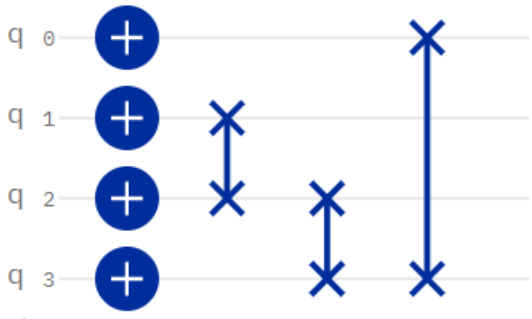


Figura 6.1: Circuito que implementa la operación $U\left|\psi\right\rangle = \left|7\psi \, mod(15)\right\rangle$

Teniendo este circuito, es sumamente sencillo implementar las operaciones sucesivas, es decir, $U_2\left|\psi\right\rangle = \left|7^2\psi \, mod(15)\right\rangle$, $U_3\left|\psi\right\rangle = \left|7^3\psi \, mod(15)\right\rangle$, etc... Simplemente debemos colocar este circuito múltiples veces de manera sucesiva. Por ejemplo, para implementar U_2 construimos el circuito de la figura 6.2.

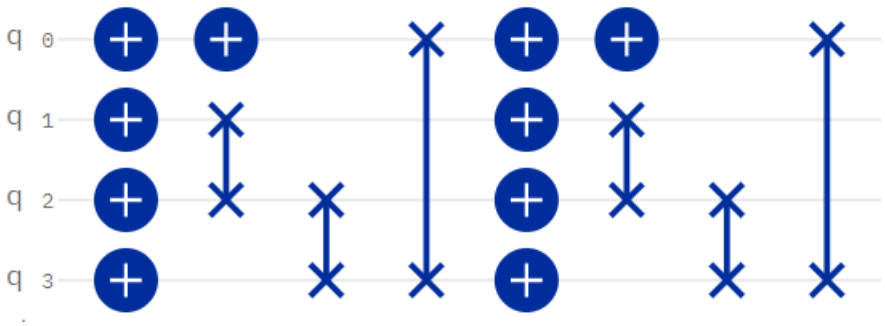


Figura 6.2: Circuito que implementa la operación $U\,|\psi\rangle = |7^2\psi\, mod(15)\rangle$

Una vez tenemos esto claro, la implementación del resto del circuito es inmediata. El circuito constará de un total de doce cúbits. Cuatro de ellos actúan como cúbits objetivo de las puertas U-Controladas, los otros ocho son los cúbits de control de estas. Adicionalmente se dispone de un registro de ocho bits para realizar las mediciones necesarias.

Comenzamos realizando la inicialización de los cúbits de control de las puertas U al ket $|+\rangle$ usando un conjunto de puertas H. Los cúbits objetivo de las puertas U son inicializados al ket $|0001\rangle$.

Posteriormente colocamos las sucesivas puertas U controladas. Estas son colocadas en cascada de la forma en que se explicó en la estimación cuántica de fase (5.3.2).

Finalmente aplicamos la Transformada Cuántica de Fourier Inversa y realizamos las mediciones a los cúbits de interés.

Tras haber implementado estas tres fases del circuito habremos concluido el módulo cuántico de nuestro algoritmo de Shor. La figura 6.3 muestra cómo quedaría el circuito :

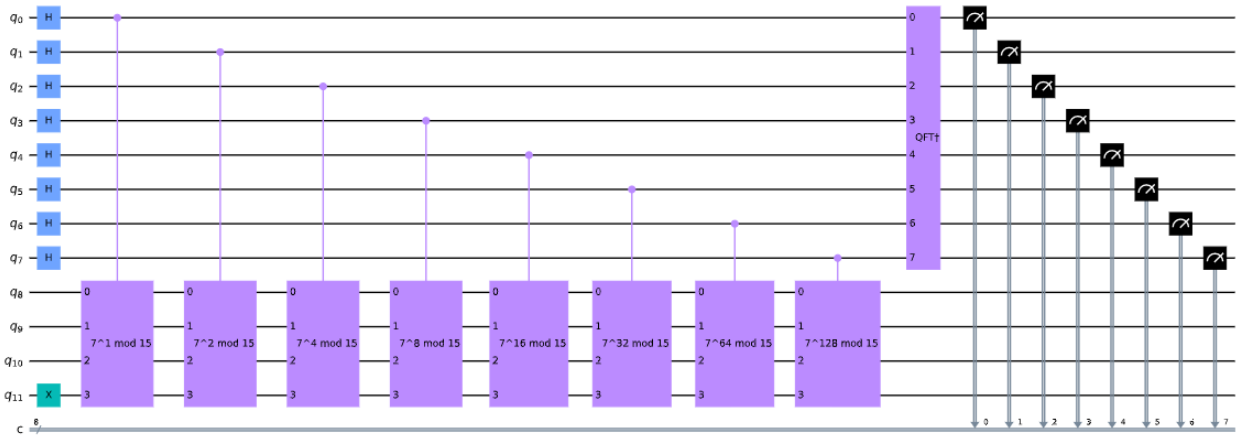


Figura 6.3: Implementación básica del algoritmo de Shor para el número 15

Como se explicó anteriormente, el módulo clásico consta de dos pasos muy sencillos. Estos son: aplicar el algoritmo de fracciones continuas y posteriormente, a través del máximo común divisor obtener los factores.

Indudablemente esta implementación del algoritmo es básica. Existen diferentes maneras de optimizarlo, reduciendo el número de cúbits, el número de puertas empleadas, etc...

Dado que no se ha realizado ninguna optimización, el número de puertas que contiene el circuito es extremadamente elevado. Los módulos $7^x mod 15$ crecen extremadamente rápido. Por lo tanto, con el tamaño actual del circuito no es posible ejecutarlo en un computador cuántico de los IBM quantum services.

Así, vamos a realizar una simple optimización al circuito. Actualmente tenemos una configuración con un alto nivel de redundancia, veremos que de manera muy simple podemos eliminarla y obtener un circuito equivalente con un número de puertas lógicas mucho menor. Tras aplicar esta optimización el circuito se podrá ejecutar en un computador cuántico de IBM sin ningún tipo de incoveniente.

Lo primero que nos reducirá enormemente el tamaño del circuito es el número de cúbits. Como norma general se ha explicado que el número de cúbits necesario para la factorización de un número N es $3m$, donde $m = log_2(N)$.

Debemos recalcar que esto no es un valor exacto, hay situaciones en la que se puede emplear un número menor de cúbits. En este caso en concreto podemos reducir el número de cúbits necesarios a 7 y todavía obtendremos una estimación de fase correcta. Por lo tanto, estamos pasando de tener un circuito de 12 cúbits a un circuito de 7 cúbits, una gran reducción.

Adicionalmente, prestaremos atención a las matrices que son aplicadas con los módulos del tipo $7^n \ mod \ 15$. Los dos primeros módulos aplican matrices que no tienen nada en particular, pero el módulo $7^4 \ mod \ 15$ aplica la matriz identidad. Es decir, la matriz unitaria de este módulo no está realizando ninguna operación real, por lo que es redundante y también la podemos eliminar.

El circuito resultante con estas mejoras se muestra en la figura 6.4.

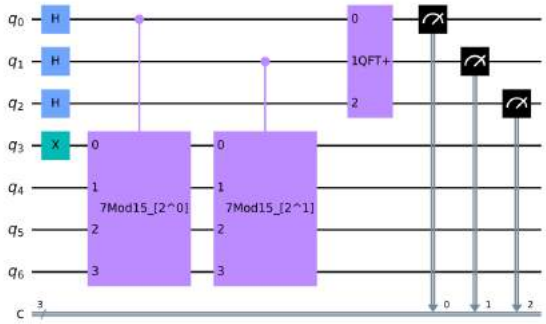


Figura 6.4: Implementación mejorada del algoritmo de Shor para el número 15

6.3. Ejecuciones

A continuación procederemos a ejecutar el circuito mostrado en la figura 6.4.

Comenzaremos utilizando un entorno de simulador local proporcionado por el componente Aer de Qiskit, en este veremos los resultados que se deben medir teóricamente. Posteriormente realizaremos las ejecuciones en el computador cuántico *ibmq_16_melbourne*.

6.3.1. Simulador

Tras comenzar la ejecución los primeros datos que obtenemos son las mediciones realizadas al final del módulo cuántico, es decir, las mediciones de los ocho primero cúbits tras hacerlos pasar por la TCFI. Haciendo una ejecución con 1024 repeticiones obtenemos las probabilidades mostradas en la figura 6.5.

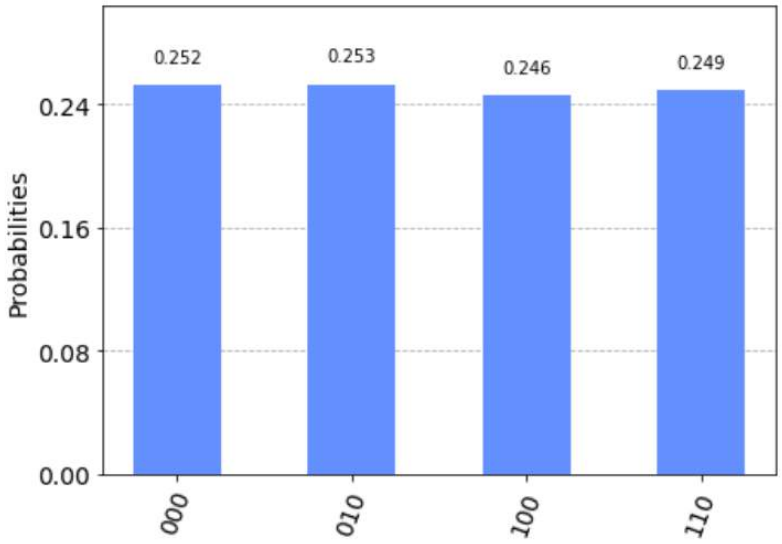


Figura 6.5: Resultado de 1024 ejecuciones en el simulador ibmq_qasm_simulator

Estos resultados han sido obtenidos en un tiempo de 0.0026 segs. Como vemos hay cuatro resultados posibles, correspondientes a los números 0, 2, 4 y 6. Debemos recordar que este valor corresponde con el valor de $\phi = \frac{s}{r}$, por lo que es un valor equivalente a una fracción donde el denominador es el valor de interés.

Es importante apreciar que no todas las ejecuciones nos van a devolver un valor correcto, habrá valores que a pesar de ser obtenidos del circuito de manera correcta no nos sirven para proceder con el algoritmo y tendremos que realizar una nueva ejecución. El más evidente es el número 0, que no nos permite extraer ninguna información. Aunque no es tan evidente como el número 0 el valor 4 tampoco arroja un resultado correcto, esto se debe a que el valor de s y r no son coprimos o a que en vez de r se obtiene un factor del mismo.

Por lo tanto para un resultado correcto necesitaremos medir el valor 2 o el valor 6. Dado que los cuatro valores tienen una probabilidad más o menos homogénea, obtendremos un

resultado correcto aproximadamente la mitad de las veces.

Estas son las probabilidades de las mediciones, a la hora de trabajar no realizaremos una batería de ejecuciones (generalmente suelen hacerse 1024) sino que realizaremos únicamente una y trabajaremos con ese valor si es adecuado. Si no es posible hacer esto, por los motivos explicados en el anterior párrafo, repetiremos la ejecución.

Tras aplicar el algoritmo de fracciones continuas se extrae que el valor del período buscado es 4. Este número es par, por lo que podemos continuar trabajando. Como ya se explicó, si el resultado fuera un número impar tendríamos que escoger un nuevo valor para a donde $a\ != 7$. En este punto conocemos:

$$\begin{aligned} &7^4\ \ mod\ \ 15 = 1 \\ &(7^4 - 1)\ \ mod\ \ 15 = 0 \\ &(7^4 - 1) = (7^2 + 1)(7^2 - 1) \end{aligned} \tag{6.4}$$

Recordando que $(7^2 + 1)(7^2 - 1) = kN$. Aplicamos el máximo común divisor obtenemos los factores buscados, 3 y 5.

6.3.2. Computador cuántico

A continuación repetiremos el proceso, esta vez en un computador cuántico.

Realizando 1024 repeticiones, los resultados obtenidos en el módulo cuántico se pueden ver en la figura 6.6.

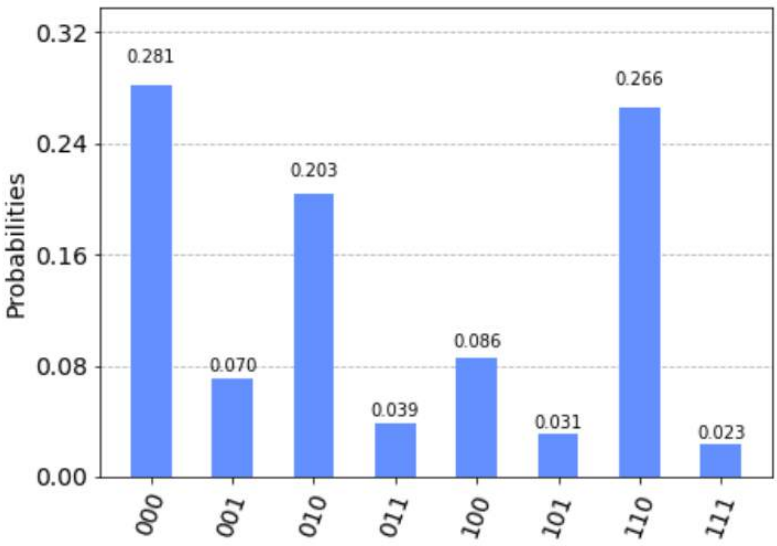


Figura 6.6: Resultado de 1024 ejecuciones en el computador ibmq_16_melbourne

Como vemos, los resultados son bastante diferentes a los que encontramos en el entorno de simulación. El tiempo de cómputo en este entorno real ha sido de $14,2$ segundos.

Como ya se comentó y se explicó y observó en la sección 4.4 vemos que en estas ejecuciones también nos encontramos con la presencia de ruido. A pesar de que teóricamente sólo debería haber 4 resultados medibles vemos como los ocho resultados posibles tienen una probabilidad distinta a cero. Así todo, vemos que los resultados que se deben obtener teóricamente (los números 0, 2, 4 y 6) siguen manteniendo una probabilidad más alta que las otras. Esta tendencia se aprecia especialmente en el 0, 2 y 6.

Como es evidente, esta distribución de probabilidades producirá más ejecuciones erróneas, ya que aunque poco probables, en ocasiones obtendremos resultados incorrectos. Por lo tanto tendremos que hacer un mayor número de ejecuciones para obtener una factorización correcta. En simulador obteníamos resultados válidos aproximadamente la mitad de las veces, aquí en cambio las obtenemos alrededor de un 40-45 % de las veces

Capítulo 7

Conclusiones y líneas futuras

El cometido principal de este Trabajo de Fin de Grado ha sido realizar una introducción comprensible pero fiel a la computación cuántica. Para ello se trabajó con los conceptos más elementales de este paradigma para, poco a poco, ir construyendo sobre ellos un conocimiento más elaborado. Así, hemos logrado estudiar e implementar componentes básicos de muchos algoritmos cuánticos como por ejemplo la Transformada Cuántica de Fourier. Finalmente, se ha presentado el algoritmo de Shor, realizado una implementación cuántica eficiente y sus correspondientes ejecuciones.

La computación cuántica todavía está dando sus primeros pasos y existen una gran cantidad de retos e incovenientes por superar. Pero, a pesar de que aún falta camino para que la computación cuántica resuelva problemas reales, cada vez se consiguen dispositivos con un mayor número de cúbits. Es evidente que está tecnología marcará más pronto que tarde un antes y un después en la computación y en la tecnología moderna.

La descomposición de números en sus factores primos haciendo uso del algoritmo de Shor tiene un gran potencial, aunque también presenta sus propios retos. Tras haber realizado nosotros mismos una implementación del algoritmo de Shor vemos que el mayor inconveniente es necesitar un circuito específico para cada número a factorizar. Ya que, diseñar un circuito específico para cada nuevo número que se desee descomponer no es viable. Adicionalmente la factorización de grandes números también está limitada por el gran tamaño de los circuitos necesarios, es decir, circuitos con una gran cantidad de puertas que no son soportados actualmente por los computadores cuánticos actuales.

Por último, la conclusión más clara que se extrae del trabajo realizado es que el diseño y la implementación de circuitos cuánticos es una tarea ardua y compleja. Es decir, el paradigma cuántico no presenta únicamente limitaciones hardware sino también limitaciones software y conceptuales.

Como posibles líneas de trabajo futuras se propone la mejora del circuito implementado en este trabajo pues, a pesar de haber reducido el número necesario de cúbits y el número de puertas lógicas necesarias, aún existen mejoras que se pueden realizar, como por ejemplo utilizar una Estimación Cuántica de Fase Iterativa (ECFI) en vez de una Estimación Cuántica de Fase común.

Por otro lado sería interesante la implementación de circuitos para factorizar otros números, estudiando la implementación específica de las componentes necesarias o realizar un estudio centrado en cómo diseñar estas componentes para un número arbitrario.

Capítulo 8

Summary and Conclusions

The main purpose of this Final Degree Project has been to achieve an understandable but faithful introduction to quantum computing. For this, we worked with the most elementary concepts of this paradigm to gradually build a better understanding of the matter. This way it has been possible to study and implement basic components of many quantum algorithms such as the Quantum Fourier Transform. Finally, Shor's algorithm has been presented, an efficient quantum implementation and its corresponding executions have been carried out.

Quantum computing is still in its beginnings and there are many challenges and drawbacks to overcome. But, although there is progress to be made until quantum computing can solve real problems, devices with a greater number of qubits are increasingly being developed. It's obvious that this technology will soon mark a before and after in computing and modern technology.

The factorization of numbers into their prime factors using Shor's algorithm has great potential, although it also presents its own challenges. After having carried out an implementation of Shor's algorithm, we see that the biggest drawback is the need of a specific circuit for each number to factorize, because designing a specific circuit for each new number that you want to decompose is not viable. Additionally, the factorization of large numbers is also limited by the large size of the necessary circuits, that is, circuits with a large number of gates that are not currently supported by current quantum computers.

Last but not least, the clearest conclusion to be drawn from the work done is that the design and implementation of quantum circuits is an arduous and complex task. In other words, the quantum paradigm does not only have hardware limitations, but also software and conceptual limitations.

As possible future studies, the improvement of the circuit implemented in this work is proposed because, despite having reduced the necessary number of qubits and the number of logic gates necessary, there are still improvements that can be made, such as using an Iterative Quantum Phase Estimation (IQPE) instead of a common Quantum Phase Estimation.

Also, it would be interesting to implement circuits to factor other numbers, studying the specific implementation of the necessary components or to carry out a study focused on how to design these components for an arbitrary number.

Bibliografía

[1] Abraham Asfaw, Luciano Bello, Yael Ben-Haim, Mehdi Bozzo-Rey, Sergey Bravyi, Nicholas Bronn, Lauren Capelluto, Almudena Carrera Vazquez, Jack Ceroni, Richard Chen, Albert Frisch, Jay Gambetta, Shelly Garion, Leron Gil, Salvador De La Puente Gonzalez, Francis Harkins, Takashi Imamichi, Hwajung Kang, Amir h. Karamlou, Robert Loredo, David McKay, Antonio Mezzacapo, Zlatko Minev, Ramis Movassagh, Giacomo Nannicni, Paul Nation, Anna Phan, Marco Pistoia, Arthur Rattew, Joachim Schaefer, Javad Shabani, John Smolin, John Stenger, Kristan Temme, Madeleine Tod, Stephen Wood, James Wootton. (2017, de Marzo). Learn Quantum Computation using Qiskit. Recuperado de https://qiskit.org/textbook/preface.html.

[2] Amico, M., Saleem, Z. H., Kumph, M. (2019). An Experimental Study of Shor's Factoring Algorithm on IBM Q. arXiv preprint arXiv:1903.00768.

[3] Benioff, P. (1980). The computer as a physical system: A microscopic quantum mechanical Hamiltonian model of computers as represented by Turing machines. Journal of statistical physics, 22(5), 563-591.

[4] Cúbit. (2021, 18 de Abril). Wikipedia, La enciclopedia libre. Recuperado de: https://es.wikipedia.org/wiki/Cúbit

[5] Google. (18 de Julio de 2018) *Cirq*. https://quantumai.google/cirq

[6] Guerra Arencibia, S. (10 de Junio de 2021). *Github* https://github.com/alu0101133201/QuantumMicroprograms

[7] haq Shaik, E., Rangaswamy, N. (2020, October). Implementation of Quantum Gates based Logic Circuits using IBM Qiskit. In 2020 5th International Conference on Computing, Communication and Security (ICCCS) (pp. 1-6). IEEE.

[8] IBM. (7 de Marzo de 2017). *Qiskit*. https://qiskit.org/

[9] de Jesus, G. F., da Silva, M. H. F., Netto, T. G. D., Galvão, L. Q., Souza, F. G. D. O., Cruz, C. (2021). Quantum Computing: an undergraduate approach using Qiskit. arXiv

preprint arXiv:2101.11388.

[10] Koch, D., Patel, S., Wessing, L., Alsing, P. M. (2020). Fundamentals In Quantum Algorithms: A Tutorial Series Using Qiskit Continued. arXiv preprint arXiv:2008.10647.

[11] Martin-Lopez, E., Laing, A., Lawson, T., Alvarez, R., Zhou, X. Q., O'brien, J. L. (2012). Experimental realization of Shor's quantum factoring algorithm using qubit recycling. Nature photonics, 6(11), 773-776.

[12] Mavroeidis, V., Vishi, K., Zych, M. D., Jøsang, A. (2018). The impact of quantum computing on present cryptography. arXiv preprint arXiv:1804.00200.

[13] Singh, P. N., Aarthi, S. (2021, February). Quantum Circuits–An Application in Qiskit-Python. In 2021 Third International Conference on Intelligent Communication Technologies and Virtual Mobile Networks (ICICV) (pp. 661-667). IEEE.

[14] Skosana, U., Tame, M. (2021). Demonstration of Shor's factoring algorithm for N= 21 on IBM quantum processors. arXiv preprint arXiv:2103.13855.

[15] Vandersypen, L. M., Steffen, M., Breyta, G., Yannoni, C. S., Sherwood, M. H., Chuang, I. L. (2001). Experimental realization of Shor's quantum factoring algorithm using nuclear magnetic resonance. Nature, 414(6866), 883-887.

[16] Veliche, A. (2018). Shor's Algorithm and Its Impact On Present-Day Cryptography.