UANL

UNIVERSIDAD AUTÓNOMA DE NUEVO LEÓN

FCFM

FACULTAD DE CIENCIAS FÍSICO MATEMÁTICAS

UNIVERSIDAD AUTÓNOMA DE NUEVO LEÓN
FACULTAD DE CIENCIAS FÍSICO MATEMÁTICAS

Aplicaciones de la Mecánica Cuántica
Proyecto final:
Algoritmo de Shor

Carlos Luna

Nombre:	Matricula:
Giovanni Gamaliel López Padilla	1837522
Ivan Arturo Pla Guzman	1837515

14 de octubre de 2020

Palabras clave:

I. INTRODUCCIÓN

En el área de la matemáticas, la factorización es una técnica que consiste en la descomposición en factores de una expresión algebraica en forma de un producto. El teorema fundamental de la aritmética cubre la factorización de números enteros, este teorema pertenece a la teoría de números. El teorema de factorización afirma lo siguiente:

Teorema 1. *Cada entero positivo tiene una única descomposición en números primos.*

El teorema fue demostrado por primera vez por Euclides, aunque la primer demostración completa apareció en las *Disquisitiones Arithmeticae* de Carl Friedrich Gauss.
El análisis numérico es el estudio de algoritmos, el cual consiste en un conjunto de instrucciones o reglas ordenadas y finitas que permiten realizar una actividad mediante pasos sucesivos con el proposito de resolver problemas matemáticos dentro de la matemática continua, esto da una aproximación numérica. Hay problemas sencillos de calcular como una raiz cuadrada, no tienen soluciones exactas o son costosas en tiempo para obtenerlas.
Para obtener una aproximación útil, se debe conocer la presición de los resultados y cuantos recursos se necesita para lograrlo, estos conceptos cuantificados son conocidos como *presición, convergencia, estabilidad y complejidad.*

La complejidad algorítmica, informalmente, es una medida que permite a los programadores conocer la cantidad de recursos que necesita un algoritmo para resolver un problema en función de su tamaño. El objetivo es comparar la eficiencia de algoritmos a la hora de resolver un problema conocido[2]. Para realizar una clasificación en la complejidad de un algoritmo se usa normalmente la notación asintótica, la cual es que por el número de operaciones básicas ejecutadas por un algirtmo se obtiene una función. Cada una se ve denotada por $T(N)$, donde N es el número de elementos numéricos dentro del algoritmo. Para valores pequeños de N, las constantes que acompañan a los términos de la función T pueden influir de manera significativa al coste tota y con ello obtener conclusiones erroneas respecto a la eficiencia del algoritmo. En cambio este tipo de análisis se realiza con números grandes de N. Deendiendo del análisis que se realice, podemos encontrar diferentes tipos de notaciones. La notación más usada es la llamada O grande y se denota por O.

Definición 1. *Se dice que un algoritmo F tiene una complejidad $O(G(N))$ si existen dos constantes C y N_0 para las que se cumpla $|F(N)| < C \cdot G(N)$ para todo $N > N_0$*

En otras palabras, que el algoritmo F tiene una complejidad $O(G)$ si el número de operaciones necesarias es constante para un número grande de N. Un ejemplo de esta clasificación puede observarse en la siguiente tabla:

Notación	Nombre	Ejemplo de algoritmo
$O(1)$	Constante	Acceso a un elemento de un vector
$O(logN)$	Logarítmica	Búsqueda binaria
$O(N)$	Lineal	Búsqueda secuencial
$O(NlogN)$	Lineal-Logarítmica	Algoritmo de ordenamiento *quicksort*
$O(N^2)$	Cuadrática	Algoritmo de ordenamiento simple
$O(N^3)$	Cúbica	Multiplicación de matrices
$O(2^N)$	Exponencial	Partición de conjuntos

Tabla I. Ejemplos de algortimos numéricos con su clasificación O de complejidad

La mayor parte de los algoritmos de factorización elementales son de proposito general, es decir, permiten descomponer cualquier número introducido, la diferencia entre algortimos es el tiempo que se toman para encontrar la factorización del número dado. El problema de factorizar enteros de tiempo polinómico no ha sido resulto en computación clásica. Esto puede ser de gran ayuda al avance en el ambito de la criptografía, ya que muchos sistemas criptográficos dependen de la imposibilidad de ser resueltos en un tiempo corto.
La complejidad de este problema se encuentra en el núcleo de varios sistemas criptográficos importantes. Un algoritmo veloz para la factorización de enteros significaría que el algoritmo de clave pública RSA es inseguro. Si un número grande, de b bits es el producto de dos primos de aproximadamente el mismo tamaño, no existe algoritmo conocido capaz de factorizarlo en tiempo polinómico. Esto significa que ningún algoritmo conocido puede factorizarlo en tiempo $\mathrm{O}(b^K)$, para cualquier constante k. Aunque, existen algoritmos que son más rápidos que $\mathrm{O}(a^b)$ para cualquier a mayor que 1. En otras palabras, los mejores algoritmos son súper-polinomiales, pero sub-exponenciales. En particular, el mejor tiempo asintótico de ejecución lo contiene el algoritmo de *criba general del cuerpo de números (CGCN)*, que para un número n es:

$$O\left(exp\left(\left(\frac{64}{9}b\right)^{\frac{1}{3}}(logb)^{\frac{2}{3}}\right)\right). \tag{1}$$

Para una computadora ordinaria, la CGCN es el mejor algoritmo conocido para números grandes.

II. OBJETIVO

- Desarrollar el algoritmo de Shor para la factorización de un número dado usando la librería Qiskit en el lenguaje python.

- Calcular la diferencia de tiempos entre el algoritmo de Shor de manera clásica y el algoritmo de Shor usando computación cuántica.

-

III. MARCO TEÓRICO

A. Algoritmo de Shor

1. Transformada de Fourier

IV. RESULTADOS

V. DISCUSIÓN

VI. CONCLUSIONES

VII. CÓDIGO

REFERENCIAS

[1]G. P. Berman, G. D. Doolen, G. V. López, and V. I. Tsifrinovich. Nonresonant effects in the implementation of the quantum Shor algorithm. *Physical Review A - Atomic, Molecular, and Optical Physics*, 61(4):7, 2000.

[2]Luanne S. Cohen and Tanya Wendling. Técnicas de diseño. *Técnicas de diseño*, pages 15–18, 1998.

[3]Edward Gerjuoy. Shor's factoring algorithm and modern cryptography. An illustration of the capabilities inherent in quantum computers. *American Journal of Physics*, 73(6):521–540, 2005.

[4]F. Ghisi and S. V. Ulyanov. The information role of entanglement and interference operators in Shor quantum algorithm gate dynamics. *Journal of Modern Optics*, 47(12):2079–2090, 2000.

[5]Daniel Koch, Saahil Patel, Laura Wessing, and Paul M. Alsing. Fundamentals In Quantum Algorithms: A Tutorial Series Using Qiskit Continued. 2020.

[6]Samuel J. Lomonaco and Louis H. Kauffman. A continuous variable Shor algorithm. pages 97–108, 2005.

[7]Peter W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Journal on Computing*, 26(5):1484–1509, 1997.

[8]Lieven M.K. Vandersypen, Matthias Breyta, Gregory Steffen, Costantino S. Yannoni, Mark H. Sherwood, and Isaac L. Chuang. Experimental realization of Shor's quantum factoring algorithm using nuclear magnetic resonance. *Nature*, 414(6866):883–887, 2001.

[9]Anocha Yimsiriwattana and Samuel J. Lomonaco Jr. Distributed quantum computing: a distributed Shor algorithm. *Quantum Information and Computation II*, 5436:360, 2004.

[10]S. S. Zhou, T. Loke, J. A. Izaac, and J. B. Wang. Quantum Fourier transform in computational basis. *Quantum Information Processing*, 16(3):1–19, 2017.