Sandra Rosales A. Desarrollo de una propuesta de métrica para evaluar las complejidades en computación cuántica y su equiparable en computación tradicional 2022

Universidad Autónoma de Querétaro

Facultad de Informática

Desarrollo de una propuesta de métrica para evaluar las complejidades en computación cuántica y su equiparable en computación tradicional

TESIS

Que como parte de los requisitos para obtener el Grado de

Maestra en Ciencias de la Computación

Presenta:

Lic. Sandra Samara Rosales Alvarado

Dirigido por:

Dr. Hugo Jiménez Hernandez

Querétaro, Qro. a 13 de enero de 2022

Desarrollo de una propuesta de métrica para evaluar las complejidades en computación cuántica y su equiparable en computación tradicional

por

Sandra Samara Rosales Alvarado

Universidad Autónoma de Querétaro
Facultad de Informática
Maestría en Ciencias de la Computación

Desarrollo de una propuesta de métrica para evaluar las complejidades en computación cuántica y su equiparable en computación tradicional

Tesis

Que como parte de los requisitos para obtener el Grado de

Maestra en Ciencias de la Computación

Presenta:
Lic. Sandra Samara Rosales Alvarado

Dirigida por:
Dr. Hugo Jiménez Hernández

Dr. Hugo Jiménez Hernández
Presidente
M. en C. Fidel González Gutiérrez
Secretario
Dr. Alberto Lara Guevara
Vocal
Dra. Diana Margarita Córdova Esparza
Suplente
Dra. Ana Marcela Herrera Navarro
Suplente

Centro Universitario, Querétaro, Qro.
enero 2022
México

Agradecimientos

En primer lugar, agradezco a la Universidad Autónoma de Querétaro, por darme la oportunidad de formar parte de esta institución educativa. Además, de brindarme el apoyo necesario mediante su facultad de informática, para desarrollar este proyecto de maestría.

Asimismo, un agradecimiento a los docentes y a mi director de tesis el Dr. Hugo, con los cuales tuve oportunidad de trabajar y que me ayudaron a ensamblar este trabajo de tesis.

También agradezco al Consejo Nacional de Ciencia y Tecnología (CONACYT), por el apoyo económico para la realización de este trabajo mediante el *Número de Apoyo:757891*.

Índice general

Índice de tablas

Índice de figuras

Resumen

El siguiente trabajo, surge de la creciente popularidad en el uso de la computación cuántica y sus mejoras prometedoras en tiempo de ejecución. Observando, que actualmente es una realidad tener acceso a una computadora cuántica, al menos si se tiene internet; además, utiliza un lenguaje de programación de fácil manejo, como lo es Python. Por lo cual, ahora la pregunta ¿En qué tipo de computadora conviene ejecutar el código? es una pregunta que se debe tener la posibilidad de contestar. Sin embargo, derivado de esta surgen más preguntas, por ejemplo ¿Cómo medir la complejidad algorítmica cuántica? ¿Existen métricas que permitan esta comparación? ¿Cuál métrica permite evaluar un algoritmo cuántico, para decidir en qué computadora aplicar el algoritmo desarrollado? Entonces, es evidente que no se debe dar por sentado que la computadora cuántica es mejor que la computadora tradicional y solo replantear los " problemas difíciles", para su nueva aplicación. Por lo tanto, se realiza esta investigación acerca de la complejidad algorítmica cuántica, en un modelo de computadora cuántica de Turing (*Quantum Turing Machine*). El cuál, permite trasladar algunas ideas sobre la complejidad algorítmica tradicional a esta nueva manera de computación. También se revisa un modelo que se basa en la experimentación en física cuántica nombrado *Quantum Algorithmic Measurement*, permitiendo generar una propuesta de métrica para la complejidad en computación cuántica. Dicha métrica, es implementada sobre dos algoritmos cuánticos, el algoritmo de Grover y el algoritmo de Deutsch. Estos dan origen a una generación de otros algoritmos fundamentales, no solo para la computación cuántica además también en la computación tradicional, como lo son los algoritmos de búsqueda y consultas en bases

de datos, entre otros. Concluyendo, que esta métrica puede ser útil como una forma de argumentación sobre si es factible, en términos de costo sobre el tiempo y el espacio cuántico, implementar un algoritmo desarrollado para la computación tradicional en el modelo de computación cuántica.

Palabras Clave: cómputo cuántico, complejidad algorítmica cuántica, algoritmo cuántico Deutsch, algoritmo de Grover.

Abstract

The following work starts from the growing popularity of quantum computing and its promising runtime improvements. Observing that currently, it is a reality to have access to a quantum computer, at least if you are having internet. In addition, it uses an easy-to-use programming language, such as Python. Therefore, now the question On what type of computer should the code be executed? It is a question that must be able to answer. However, derived from this others arise questions, for example, How to measure quantum algorithmic complexity? Are there metrics that allow this comparison? Which metric allows a quantum algorithm to be evaluated and decide on which computer to apply the developed algorithm? So, it is evident that you should not take for granted that the quantum computer is better than the traditional computer and only reformulate the " difficult problems", for the new application. Therefore, this investigation about quantum algorithmic complexity is carried out, in a model of a quantum Turing machine. Which allow us to transfer some ideas about traditional algorithmic complexity to this new way of computing. A model that is based on experimentation in quantum physics named *Quantum Algorithmic Measurement* is also reviewed, allowing the generation of a metric proposal for complexity in quantum computing. This metric is implemented on two quantum algorithms, Grover's algorithm and Deutsch's algorithm. These give rise to a generation of other fundamental algorithms, not only in quantum computing but also in traditional computing, such as search algorithms and database queries, among others. Concluding that this metric can be useful as a form of argumentation about whether it is feasible, in terms of cost over quantum time and space, to implement an algorithm developed

for traditional computing in the quantum computing model.

Keywords: quantum computing, quantum algorithmic complexity, measure quantum algorithmic complexity, Deutsch's algorithm, Grover's algorithm.

Capítulo 1

Introducción

Podría creerse que la complejidad algorítmica sólo tiene aportes teóricos, sin embargo muchos de sus resultados abren camino para el desarrollo de aplicaciones, creación de áreas nuevas y en muchos casos, originan la interacción entre áreas (Wigderson, 2019). Además se ha de recordar, que la complejidad algorítmica representa una medida para caracterizar un modelo de cómputo y determinar que tan factible es la solución ante un problema bajo cierto contexto, es decir que un modelo computacional resulte útil ante un problema dado; idea fundamental para cuantificar la posibilidad de resolver problemas sobre un modelo computacional proporcionado.

En particular, el modelo de computación cuántica debe considerar aspectos intrínsecos del modelo base, en este caso la física cuántica, los cuales han de ser medibles para determinar la pertinencia del proceso computacional. Realizando un desarrollo de complejidad algorítmica cuántica se estará más cerca de una elaboración de algoritmos eficientes.

Bernstein and Vazirani (1997) mencionan que "Un argumento fuerte al que se le puede atribuir que la TM[1] no puede resolver muchos problemas realizables, es que se basa en un modelo de

[1]Por sus siglas en ingles Turing Machine

física clásica, mientras que la física actual ha mostrado que el universo que nos rodea es físico cuántico". Para demostrar esta afirmación utiliza la tesis de Church Turing (Deutsch, 1985); con la que construye un modelo análogo al de la máquina de Turing y por lo tanto facilita el manejo de construcciones de algoritmos.

Partiendo de algo conocido, como lo son las funciones booleanas, conceptos como el autómata con su $5-$tupla; *alfabeto, conjunto de estados, estado inicial, estado de parada y tabla de transición* y conceptos abstractos hasta llegar a la complejidad, se elabora de manera teórica una forma de evaluar la eficiencia de un algoritmo desarrollado en la computación cuántica.

Desde esta perspectiva una opción para analizar los algoritmos desarrollados bajo este esquema, puede ser también, contabilizar las compuertas cuánticas utilizadas u operadores unitarios, es decir, las instrucciones que debe realizar el algoritmo para tener éxito.

Otra forma de abordar la eficiencia de un algoritmo desarrollado para la computación cuántica, puede ser por medio de la experimentación. Esto es, dando por hecho que las computadoras cuánticas existen y es posible ejecutar algoritmos sobre ellas. Entonces, se construyen soluciones a problemáticas que residen en la física cuántica, creando un algoritmo o unos algoritmos que proponen una solución.

Esta idea se observa en Aharonov et al. (2021), un trabajo reciente que propone medidas cuánticas algorítmicas, representación que se profundizará en el Capítulo 3. La idea fundamental, es construir un modelo o marco de referencia para evaluar un experimento desarrollado en el campo de la física. Aquí, se agrupa el sistema experimental en dos partes, la primera es el acceso, *access QUALM* y la segunda son, las operaciones ha realizar, *LOCC, local operations and classic communications* . De esta forma, inicia desde la experimentación en física, donde surge este paradigma.

Además, Aharonov et al. (2021) plantean el análisis de la complejidad partiendo de la dificultad para desarrollar un experimento. Aunque, existe un inconveniente implícito, el tratar

de introducir o manipular el entrelazamiento cuántico dentro del sistema de laboratorio, definiciones que se detallarán en el Capítulo 3. Aharonov et al. (2021) encuentran que el entrelazamiento tiene una ventaja exponencial y es distinta a la ventaja exponencial de la caja negra, descrita como *oráculo* por otros autores.

Por otro lado, Watrous (2012) define las clases de complejidad algorítmica cuántica mediante la definición de problema de decisión, nombrado en este trabajo como "el problema promesa" o "*promise problems*" y tomando una idea similar al desarrollo análogo de una máquina de Turing cuántica, definiciones que permiten introducirse en el campo de la teoría de la complejidad computacional cuántica, de forma familiar.

Hay un común denominador en estos trabajos; dar una definición unificada que permita de manera general caracterizar los algoritmos en la computación cuántica, permitiendo de esta forma, generar una clasificación y alcance de los algoritmos desarrollados para la misma. En particular un modo de abordar esta temática, al menos en la teórica de computación, es por ejemplo, mediante la contradicción, esto es suponer que el hecho es verdad y al final encontrar una falacia, es posible elaborar un algoritmo de repetición que al final encuentre un contraejemplo.

Otra manera es producirlo (demostrarlo), es decir elaborar algún tipo de aplicación que se enfoque en un solo problema y desarrollar una solución. Wigderson (2019) describe que la teoría de la computación ha crecido como consecuencia de estas maneras al abordar los problemas en la computación. En particular el crecimiento dado en la complejidad algorítmica, es además por las intersecciones creadas sobre las distintas metodologías.

Entonces, tomando en cuenta los distintos modos para abordar la complejidad cuántica, en este trabajo de tesis se pretende identificar características distintivas que permitan cuantificar un algoritmo sobre un modelo de cómputo cuántico, para realizar desarrollos de manera teórica, sin la dependencia de la realidad, ya que elaborar un modelo permite entender, manipular y

predecir. Por lo cual, se usan algunos problemas desarrollados para la computación cuántica, que actualmente pueden ser implementados en computadoras cuánticas reales y simuladas, es decir, se utiliza una mezcla de distintas técnicas como la implementación, análisis del peor caso, el concepto de notación asintótica. Y por último, se analiza la complejidad de espacio, y tiempo sobre implementación de algoritmos *on-line*.

Sin embargo, hay que mencionar que se dejaran abiertos casos como la complejidad de la información (complejidad de la comunicación, como se realiza el intercambio de información en distintos sistemas), computación no determinista, algoritmos probabilistas, complejidad computacional booleana, programación lineal, la complejidad de polinomios (complejidad aritmética), entre muchos otros conceptos.

1.1 Cómputo cuántico y su complejidad

El modelo de Turing, fue un punto de inflexión en ciencias de la computación como se le conoce hoy en día, ya que inicio un nuevo campo que tiene sus bases en investigaciones como Gödel (1931); Turing (1937); Russell and Whitehead (1997), las cuales abrieron camino haciendo uso de un razonamiento formal matemático. No obstante, todo conocimiento evoluciona, siempre existen planteamientos que limitan un primer paradigma, siendo necesario construir nuevos; en muchas ocasiones, estos nuevos paradigmas pueden estar relacionados, o no.

Como se sabe, Turing, Leibniz, Babbage, Neumann, entre otros, hicieron posible la automatización de procesos como el cálculo de distintas operaciones aritméticas hasta llegar a la creación física de una computadora, como se conocen hoy en día y se utilizan a diario, entre ellas, el celular. Aunque, parece que los avances fueron inmediatos, no significa que fuese simple o trivial ya que otra área necesaria para dicha fabricación fue la física clásica. Sin embargo, este campo ha cambiado, se ha descubierto la física cuántica, utilizando una combinación de ideas sobre la teoría cuántica de Planck, la teoría de fotón de luz por Einstein y del

modelo del átomo de Rutherford (Serway Raymond, 1993; Sanchez Ron, 2001), generando conceptos como la superposición y el entrelazamiento.

Feynman (1985) observó que los algoritmos clásicos que trabajan de forma aleatoria al simular la evolución de un sistema cuántico para n partículas requieren un tiempo exponencial sobre n, así que, planteó algoritmos de computadora que contaran con las herramientas de la "mecánica cuántica" por medio de compuertas, que permitieran el uso de algunos conceptos para poder habilitar una simulación eficiente. No es que la computación tradicional no tenga una manera de resolver este tipo de problemas, pero no de forma eficiente. Estos nuevos conceptos no podían ser simulados, se necesitaba una herramienta que permitiera explotar estos nuevos elementos para la resolución o simulación de problemas en este campo, así es como Feynman en 1987 propone la creación de una computadora que utilice estos conceptos (Kitaev, 1997).

Para el modelo de máquina de Turing una serie de múltiples cadenas e instrucciones, alargan considerablemente el proceso, en cambio esta misma serie de múltiples cadenas e instrucciones en una computadora cuántica se traduce en una sola instrucción. Con el desarrollo de una estructura lógica, Turing generó investigación teórica sobre máquinas con cinta o memoria infinita, solo para mostrar su potencial. Siendo estas ideas las que abren la puerta para una creación física de una aparato que se aproxime a ese planteamiento. Actualmente, existen trabajos que muestran las bondades y capacidades de la computadora cuántica en resolver problemas, donde la computación clásica no puede o es inviable su solución; por ejemplo en Arute et al. (2019) desarrollaron un algoritmo cuántico, revelando una mejora en la velocidad de ejecución.

Ahora, el entendimiento de un nuevo paradigma genera nuevas preguntas, entre ellas ¿es comparable con la computadora cuántica? ¿la computadora cuántica a diferencia de la computación clásica tiene infinita memoria? es decir que la hace tan especial ¿cómo medir la utilidad de un algoritmo en un paradigma cuántico, contra un algoritmo en un paradigma de Turing? En este sentido, una primera aproximación para dar respuesta a estas preguntas, Bernstein and

Vazirani (1997) desarrollan un análogo a una máquina de Turing .

En trabajos recientes como Mora and Briegel (2005); Mermin (2007); Bacon and van Dam (2010); Watrous (2012); Apers and Lee (2020) existen definiciones que también caracterizan a los algoritmos en el cómputo tradicional en términos de sus propiedades como en Davis et al. (1994), que utiliza como criterio la cantidad de recursos. Por otro lado, en Aho and Hopcroft (1974) se introduce el concepto de complejidad algorítmica como otro criterio de caracterización.

Sin embargo, algunas de sus consideraciones pueden ser no relevantes o útiles en un paradigma distinto, como el cómputo cuántico. Una manera distinta de responder a estas cuestiones es aceptar que es un modelo completamente diferente e independiente, por lo cual se deben generar sus propias manearas de evaluación, eso hacen Aharonov et al. (2021).

Ahora bien, el tener una computadora con características que superan, al menos en velocidad a la computadora clásica, no implica que se deba cerrar la discusión acerca de la eficiencia de algoritmos, es decir, se ha comprobado que con un máquina lenta, si se desarrolla un buen algoritmo es posible mejorar el tiempo de ejecución, si eso lo se traslada al nuevo paradigma se pueden aprovechar de mejor manera los recursos que ofrece. Pero para llegar a eso se debe primero desarrollar una forma de evaluación de algoritmos cuánticos.

En matemática existen soluciones a problemas que se pueden desarrollar para ejemplos de tamaño pequeño, pero cuando se intenta resolver para entradas más grandes resulta imposible saber si se obtendrá una respuesta. En ciencias de la computación, se puede considerar resuelta una serie de problemas de un determinado tipo cuando se elabora el algoritmo con su resolución. Es decir, se toma un problema “general” y se trata de dar solución por medio de una algoritmo, si esto sucede es factible afirmar que los problemas subyacentes pueden ser resueltos.

Algunos problemas de este tipo son, el problema de colorear un mapa, el teorema de *Euclides*,

el problema de las $n-$reinas, el agente viajero, entre otros. En particular, hay un convenio general acerca del tipo de problemas que no se puede resolver en menos de un tiempo exponencial, se les llama *problemas intratables*, esto es, que para casos pequeños el algoritmo asociado a la solución del problema funciona adecuadamente, pero conforme incrementa el tamaño de problema este es inviable observar su solución (Aho and Hopcroft, 1974).

Por otro lado, se sabe que la complejidad algorítmica contribuye a mejorar la construcción y elección de algoritmos en el cómputo tradicional. Además, no es necesario implementar el algoritmo para verificar si es correcto, simplemente se analiza y se toma una decisión con base en el resultado del análisis. Entonces, considerando el nivel de aplicación en cada uno de los paradigmas mostrados, existe una forma cuantitativa de caracterizar propiedades de un mismo algoritmo en dos paradigmas que permitan su comparación; y más aún, da información acerca de la factibilidad en el uso de recursos de acuerdo a cada paradigma.

En esta tesis se construye una métrica de complejidad en algoritmos cuánticos, que permita comparar el paradigma de cómputo tradicional y el cuántico, sin tomar en cuenta los recursos disponibles en ambos casos (memoria y/o procesamiento), utilizando exclusivamente el análisis de las distintas complejidades. Por lo tanto, se analiza un algoritmo cuántico mediante el modelo de compuertas cuánticas (Mosca, 2009; Rieffel and Polak, 2011; Coles et al., 2018) y se revisan las distintas formas de análisis sobre la complejidad en el cómputo tradicional.

1.2 Justificación

Un modelo de cómputo permite definir un marco de referencia que delimita la forma de la computación, los recursos a utilizar y en algunos casos el impacto (tiempo y espacio requerido) que tendrá el modelo. Además da la facultad de diferenciar procesos *decidibles*[2].

[2]Decidible: Existe un algoritmo o procedimiento finito que permite determinar si una fórmula es o no válida en el sistema. Un sistema formal es decidible si dispone de un procedimiento general para decidir si cualquier

Entonces, dada la diferencia entre los paradigmas (físico y matemático), cada uno representa dos modelos teóricos que tienen un conjunto distinto de problemas decidibles. A su vez, tendrán un conjunto de problemas equivalentes y otro de problemas computables distintos.

Por lo cual en este trabajo se propone, una métrica sobre un conjunto de problemas comparables (realizables en ambos paradigmas) que cuantifique la factibilidad de un proceso desarrollado, de tal forma que permita equiparar y explotar las características que ofrece cada modelo de cómputo. La métrica ayudará a entender y establecer el marco de referencia dentro sobre estos paradigmas, para luego, contar con un criterio útil y estar en posibilidad de decidir en que paradigma y bajo que argumentación es más útil una u otra aproximación. Algunas de las ventajas de realizar esta métrica incluyen:

- Evitar hacer uso de recursos a ciegas.
- Mejor elección de algoritmos cuánticos.
- Mejor construcción de algoritmos cuánticos.
- También, puede proporcionar un catálogo sobre algoritmos realizados en el cómputo cuántico.

La importancia en la clasificación de algoritmos cuánticos, es comparable con la importancia que tiene la complejidad en el cómputo tradicional, ya que estos análisis permitieron, en su momento, determinar si era o no prudente implementar un algoritmo, que tal vez nunca llegaría a una solución. De modo que, entender mejor este nuevo paradigma, dará herramientas para aprovechar su potencial.

fórmula bien formada dentro del sistema es o no un teorema del mismo Feys (1956).

1.3 Organización de capítulos

En el Capítulo 2, se abordaran las bases para entender el desarrollo de algoritmos cuánticos, así como algunas perspectivas destacadas que abordan la complejidad algorítmica cuántica, desde un punto de vista análogo a la computación tradicional, hasta una idea de experimentación en física. En el Capítulo 3, se describirán los algoritmos que se abordan en esta tesis, así como su complejidad algorítmica desarrollada en el cómputo tradicional además de profundizar en las ideas de Papadimitriou (1993), Bernstein and Vazirani (1997) y Aharonov et al. (2021). En el Capítulo 4, se muestra la implementación de los algoritmo seleccionados, así como el resultado de la propuesta desarrollada y una breve discusión sobre este resultado. Por último, se concluyen algunas propuestas iniciales en el Capítulo 5.

1.4 Hipótesis

Dado un conjunto de problemas realizables en dos modelos de cómputo distinto, mediante una función de complejidad algorítmica espacio temporal se podrán cuantificar las características de dichos problemas con propiedades similares de forma equiparable en computación cuántica y en computación tradicional.

1.5 Objetivo general

Definir una métrica de complejidad espacio temporal en algoritmos cuánticos para comparar y evaluar en modelos de cómputo tradicional y cuántico las propiedades en sus diversos procesos.

1.5.1 Objetivos específicos

- Seleccionar y elaborar una tabla con dos algoritmos significativos en el modelo de referencia y en el modelo cuántico.
- Realizar un análisis comparativos de dos algoritmos significativos.
- Desarrollar e implementar dos algoritmos significativos en el portal "*IBM Quantum Experience*".
- Estudiar y desarrollar criterios que permitan la creación de la complejidad espacio temporal de un algoritmo cuántico.
- Implementar la métrica de complejidad desarrollada sobre dos algoritmos cuánticos significativos.

1.6 Alcances y limitaciones

El siguiente trabajo está sujeto a los siguientes alcances y limitaciones:

1. Dado que se trabaja unicamente con dos algoritmos, la métrica desarrollada no podrá ser una generalización sin un estudio más profundo.
2. Debido al tamaño del proyecto esta métrica solo utiliza el modelo de compuertas cuánticas. Es decir, que el desarrollo de esta función de complejidad hará uso de esta teoría, dejando para estudios futuros el análisis y comparación sobre otros modelos.
3. La función de complejidad aquí desarrollada permitirá dar claridad sobre la factibilidad de un algoritmo sobre estos dos modelos diferentes, sin embargo no contempla una clasificación sobre los tipos de problemas existentes.

Capítulo 2

Antecedentes

Para abordar la complejidad computacional cuántica se deben mencionar los conceptos base de los cuales parte la complejidad computacional tradicional. Por lo cual, en este Capítulo se revisarán las distintas maneras de abordar en análisis de un algoritmo cuántico, así como otras representaciones útiles para desarrollar la propuesta. Además, se mencionan algunos antecedentes de la complejidad que han abierto el camino para ´´mejores" algoritmos y creación de áreas especializadas en la optimización de los mismos.

Actualmente, computabilidad o *computability* es una palabra utilizada de forma cotidiana en ciencias de la computación, se entiende como un proceso tal, que puede implementarse en la computadora, es decir generar su algoritmo y programarlo. Sin embargo, su origen viene de la palabra generada por Turing al responder a las preguntas propuestas por Hilbert en los años 30's.

Hilbert observó que las soluciones "satisfactorias", que permiten determinar si un objeto tiene o no una propiedad particular, proporcionan procedimientos mecánicos (Wigderson, 2019). Turing nombró a estos procedimientos (algoritmos), *procedure decision* y los representó como una máquina de Turing (actualmente un programa de computadora). Los procedimientos que

se detienen en una entrada sobre un tiempo finito son procedimientos decidibles (*decidability*), en caso contrario se habla de indecidibilidad (*undecidability*), enunciados que no tienen un procedimiento sobre un tiempo finito que permita determinar si tiene o no una propiedad característica (Wigderson, 2019).

De esta manera, cuando se habla de computable o no computable se hace referencia de forma implícita, a la existencia de un algoritmo tal, que se detiene en tiempo finito. Sin embargo, existen problemas a los cuales no es posible asociar un algoritmo decidible, como ejemplo *el problema del paro* (Aho and Hopcroft, 1974). Por lo cual, surge la clasificación de problemas en teoría de la computabilidad.

Desde otra perspectiva existe la *complejidad de la prueba*, tratando de responder preguntas en matemáticas como ¿Es posible cuantificar matemáticamente la dificultad de probar varios teoremas? Esta complejidad de la prueba, busca clasificar los teoremas proposicionales (llamados "tautologías") de acuerdo con la dificultad para demostrarlos. De hecho, la complejidad de la prueba tiene una relación similar con la teoría de la prueba de forma similar a la relación entre la complejidad del circuito y la teoría de la computabilidad (Aaronson, 2013).

Una pregunta relevante en la teoría de la computabilidad es $P\ vs\ NP$. Wigderson (2019) explica el corolario $P \subseteq NP$, que ayuda al intento de responder la pregunta, aunque actualmente la pregunta sigue abierta. Algunos problemas clasificados en P que permitirían contestar la pregunta son, coincidencia perfecta (*perfect matching*), dada una gráfica hacer coincidir cada par de vértices por medio de su arista, el palíndromo, el problema de la primalidad, entre otros. Por otra parte, algunos clasificados en NP son, el problema de decidibilidad, es decir determinar si una sentencia matemática es verdadera o falsa, las ecuaciones diofánticas solubles y los teoremas demostrables en aritmética de *Peano*, etc (Wigderson, 2019).

Es importante mencionar que las teorías de la computabilidad y complejidad están estrechamente relacionadas. En teoría de la complejidad, el objetivo es clasificar problemas como

fáciles o difíciles; en cambio la teoría de la computabilidad, clasifica los problemas de acuerdo a la existencia de una solución. Además la teoría de la computabilidad introduce algunos conceptos usados en la teoría de la complejidad, uno de ellos la notación asintótica (Sipser, 2012).

El punto de vista asintótico es inherente a la teoría de la complejidad computacional. En cambio, en la teoría de la computabilidad, la dependencia del tamaño de entrada no existe, simplemente se requiere que los algoritmos se detengan en un tiempo finito. Asimismo es relevante señalar, que gran parte de la metodología se importó a la clasificación de complejidad de problemas (teoría de la complejidad computacional) (Wigderson, 2019).

Una herramienta principal en la computabilidad (teoría de la computabilidad) son las reducciones, técnicas para generar funciones más sencillas pero computables, mientras que el enfoque de la complejidad computacional es la eficiencia (Wigderson, 2019). Si bien aquí se enfoca en la eficiencia del tiempo y espacio, el campo estudia una gran variedad de otros recursos. En esta tesis se trata de relacionar la dificultad computacional cuántica de problemas para los cuales se tienen soluciones (teóricamente).

En particular, la teoría de la complejidad plantea la pregunta: ¿cómo se escalan los recursos necesarios para resolver un problema con alguna medida n de tamaño del problema, ya sea "razonablemente" (como n o n^2), o "irrazonablemente" (como 2^n o $n!$)? En general, se considera eficiente el cálculo cuyo tiempo de ejecución en cualquier entrada de longitud n está limitado por una función polinomial en n (Aaronson, 2013).

Como ejemplo, dos números enteros de n dígitos se pueden multiplicar usando aproximadamente n^2 pasos computacionales, o incluso $n \log \log \log n$ pasos (utilizando métodos de optimización). Por el contrario, el método más rápido conocido para la operación inversa, factorizar un número entero de n dígitos en números primos emplea aproximadamente $2^{n^{\frac{1}{3}}}$ pasos, lo que se considera ineficiente. Cabe mencionar que estas dificultades relativas

de multiplicar y factorizar son la base de la mayor parte de la criptografía que se utiliza actualmente (Aaronson, 2013).

Insistir en que los programas deben terminar después de un período de tiempo razonable, que usan cantidades razonables de memoria, etc., puede parecer correcciones sobre la noción de computación de Turing. Aún así, permite contestar preguntas sobre la evolución en biología, la mecánica cuántica, la física estadística, la economía o la adquisición del lenguaje humano, temas que no tendrían sentido desde el punto de vista de la computabilidad (ya que todos los problemas relevantes son calculables) (Aaronson, 2013).

También la complejidad difiere de la computabilidad en la diversidad de técnicas matemáticas utilizadas: mientras que inicialmente la complejidad (como la computabilidad) se basaba principalmente en la lógica matemática, hoy se basa en la probabilidad, la teoría de números, la combinatoria, la teoría de la representación, el análisis de *Fourier*, entre otros (Aaronson, 2013).

Por otra parte Papadimitriou (1993) menciona, que los problemas computacionales no son solo cosas que deben resolverse, también son objetos que vale la pena estudiar. Además, señala que es posible formalizar y analizar matemáticamente los problemas y algoritmos, por ejemplo, como lenguajes y máquinas de Turing, respectivamente, donde el formalismo no es muy relevante. Asimismo explica, que la computabilidad en tiempo polinomial es una propiedad deseable importante de los problemas computacionales, similar a la noción intuitiva, que se tiene de una solución práctica.

Para resolver un problema mediante una máquina Turing, se debe decidir cómo representar mediante una cadena una instancia del problema. Una vez realizada esta representación, por ejemplo para un algoritmo sobre un problema de decisión, entonces el resultado será una máquina de Turing que decide el lenguaje correspondiente, conforme a la entrada representada por una instancia “si” o “no”. De manera similar, los problemas que requieren una salida más

compleja, se resuelven con una máquina de Turing que calcula la función apropiada de cadenas a cadenas (donde la salida se representa igualmente como una cadena) (Papadimitriou, 1993).

La representación resulta fundamental a la hora de desarrollar una máquina de Turing, por ejemplo representar números en unario[1], en lugar de binario o decimal, podría hacer el trabajo más complicado, ésta requiere exponencialmente más símbolos que la representación binaria. Como resultado, la complejidad de un algoritmo, medida en función de la longitud de entrada en la máquina de Turing, puede parecer falsamente favorable (Papadimitriou, 1993). En general, cuando se analiza una máquina de Turing que resuelve un problema computacional particular, se asume que utiliza una representación de entrada razonablemente concisa.

La teoría de la complejidad es sumamente independiente del problema de la representación de entrada. Ésta podría haberse desarrollado de forma independiente a cadenas y lenguajes. Pero una elección sensata de representación (que en la práctica significa evitar la notación unaria para los números) hace que los resultados de la teoría de la complejidad sean inmediatamente relevantes para problemas reales en la práctica computacional (Papadimitriou, 1993).

Ahora, para el "nuevo" modelo de computación cuántica es suficiente retomar el modelo de Turing y su teoría de la complejidad. Es notable que las máquinas de Turing, son bastante poderosas. Pero, ¿pueden realmente usarse para implementar algoritmos arbitrarios?, esta pregunta la plantea Papadimitriou (1993). Entonces, si se consideran los algoritmos cuánticos como un subconjunto de algoritmos arbitrarios, la pregunta se puede reescribir como ¿puede usarse el modelo de máquina de Turing para implementar algoritmos cuánticos? Para esta tesis bastará abordar ésta última pregunta.

Papadimitriou (1993) afirma que que las máquinas de Turing, son suficientes para implementar algoritmos arbitrarios. Sin embargo, no es posible probar ésta afirmación rigurosamente, porque la noción de un "algoritmo arbitrario" es algo que no es posible definir de forma

[1]Donde 14 es "11111111111111".

simple.

La versión clásica de esta afirmación informal, es conocida como *Tesis de Church*, donde afirma que cualquier algoritmo se puede representar como una máquina Turing (Aaronson, 2013). A principios del siglo XX matemáticos intentaron formalizar la noción de un "algoritmo arbitrario" ideando modelos de computación que finalmente se mostraron equivalentes en potencia computacional a la máquina de Turing (Papadimitriou, 1993).

En investigaciones sucesivas sobre la tesis de Church, científicos informáticos reforzaron la creencia de, que cualquier intento razonable de modelar matemáticamente algoritmos informáticos y su desempeño en el tiempo está destinado a terminar con un modelo de cálculo y costo de tiempo asociado que es equivalente a las máquinas de Turing dentro de un polinomio (Papadimitriou, 1993).

Tomando en cuenta lo descrito anteriormente, suponer que es posible generar otro modelo de computación que sea distinto al modelo de Turing parece un desatino. Sin embargo, puede ser un tema de trabajo sucesivo, ya que no solo se habla de un modelo distinto, si no además de una teoría física diferente.

Un ejemplo de como el módelo de Turing es hasta hoy suficiente, se muestra en el artículo de Bernstein and Vazirani (1997) y es la base para este trabajo, ya que demuestra que existe una *máquina cuántica de Turing*, que abre una puerta hacia el tratamiento viable de la computación cuántica como un análogo a una máquina de Turing clásica.

Además, genera las complejidades utilizadas en muchos otros artículos para clasificar algoritmos cuánticos, algunos de los cuales son (Goldreich, 2008; Arora and Barak, 2009; Watrous, 2012; Montanaro, 2016; Coles et al., 2018; Apers and Lee, 2020). Algunas definiciones utilizadas sobre complejidad algorítmica clásica que ayudan a contraponer la idea de crear una complejidad algorítmica cuántica independiente; dichas definiciones fueron tomadas de (Papadimitriou, 1993; R.C.T., 2007; Sipser, 2012; Sen, 2019).

Aunado a los trabajos realizados sobre el cómputo cuántico y la complejidad algorítmica cuántica existen trabajos relevantes que han propuesto algunas ideas, como por ejemplo la contabilización o medición de la eficiencia por medio de experimentos en la física (Aharonov et al., 2021) o utilizando la contabilización de acceso a estados cuánticos (consultas, "*Quantum query models*") o como en Mora and Briegel (2005) donde la complejidad algorítmica de un estado cuántico se mide a través del tamaño más pequeño del estado de preparación por medio de compuertas cuánticas. Estos estudios sentarán las bases necesarias para el Capítulo 4, en donde se utilizará lo aquí explicado.

2.1 Cómputo cuántico

Actualmente existen distintos tipos de modelos computacionales tradicionales (máquina de Turing de una cinta, máquina de Turing con múltiples cintas, máquinas de acceso aleatorio, autómatas celulares, etc.) los cuales son equivalentes entre ellos. Hoy en día hay modelos computacionales relativos a configuraciones cuánticas, que son funcionales. Por ejemplo, el modelo universal de computadora cuántica topológica propuesta por Freedman, Kitaev, Larsen y Wang (Wigderson, 2019), el cual es la base de proyectos prácticos para la construcción de una computadora cuántica real. Sin embargo, se han generado otros modelos computacionales. La Tabla 2.1 muestra algunos de estos enfoques.

Tabla 2.1: Modelos computacionales relativos a configuraciones cuánticas. Fuente: elaboración propia

Modelo	**Descripción**
Máquina de Turing Cuántica (Deutsch, 1985; Bernstein and Vazirani, 1997)	Se vale del modelo de una máquina de Turing clásica, cambiando la cinta de bits a una cinta de qubits y con amplitudes de transición, en lugar de transición.
Caminatas Cuánticas (Santha, 2008)	Usa el principio de caminatas aleatorias con comportamientos de los sistemas cuánticos.
Basado en mediciones (Wiseman and Milburn, 2009)	Realiza mediciones descriptivas de un solo qubit sobre un estado especial inicial de entrelazamiento.
Topológico (Wang, 2010)	Se basa en el trenzado de un cierto tipo de cuasi-partículas llamadas *anyons*, que pueden surgir en sistemas cuasi-bidimensionales de muchos cuerpos.
Compuertas Cuánticas (Jordan, 2008; Mosca, 2009; Rieffel and Polak, 2011)	Aplica la noción de un circuito lógicas (AND,NOT, etc.), donde las entradas son qubits.
Adiabático (Albash and Lidar, 2018)	Utiliza el concepto de Hamiltoniano cuántico, requiere temperaturas más bajas conforme el tamaño del problema es más grande.
One Clean Qubit (Fujii et al., 2014; Yoganathan and Cade, 2019)	Emplea un qubit en estado puro y n qubits en estado máximo mezclado.

De los modelos mostrados en la Tabla 2.1, el más utilizado actualmente, es el modelo de compuerta cuántica, revisado desde 1994 (Shor, 1994). Su principal característica es proporcionar un marco de referencia equivalente a cómputo clásico, ya que trabaja con la idea de circuitos lógicos y cables, con los cuales se realizan operaciones nombradas, compuertas cuánticas, y al igual que con las compuertas lógicas, donde cualquier función booleana puede ser representada usando compuertas "básicas" como lo son *AND* y *NOT* (universalidad); dado un conjunto de compuertas cuánticas de dos qubits se puede construir cualquier compuerta unitaria (matriz unitaria) y realizar con precisión cálculos arbitrariamente largos (Jordan,

2008).

Ademas, se trabaja con el concepto de funciones reversibles, fundamental para la física cuántica y distinto a las funciones o compuertas lógicas manejadas en el cómputo clásico.

2.1.1 Modelo de compuertas cuánticas

En la mecánica cuántica, el cambio de los estados físicos, o la evolución de un sistema son desafíos de estudio, ya que éstos sistemas cambian con leyes probabilistas. Es decir, que pasar de un estado a otro, ocurre con cierta probabilidad (Yanofsky and Mannucci, 2009). La manera de abordar estos procesos dinámicos físicos fue mediante la representación gráfica o grafo del sistema, donde cada nodo representa una posición de la partícula y el peso de la arista representa la probabilidad de que la partícula se encuentre en ese estado, obteniendo de esta forma su movimiento.

Para observar el cambio de una partícula (objeto, fotón, cuanto, etc.) de un tiempo t a un tiempo $t+1$, suponga que se tienen las matrices de adyacencia que representan la partícula en el tiempo t y $t+1$ respectivamente; bastará con calcular el producto de estas matrices para representar los cambios del sistema en un tiempo k. Véase la Figura 2.1, donde su matriz de representación es:

$$\begin{bmatrix} \frac{1}{\sqrt{2}} & -\frac{i}{\sqrt{2}} & 0 \\ \frac{1}{\sqrt{2}} & \frac{i}{\sqrt{2}} & 0 \\ 0 & 0 & i \end{bmatrix}$$

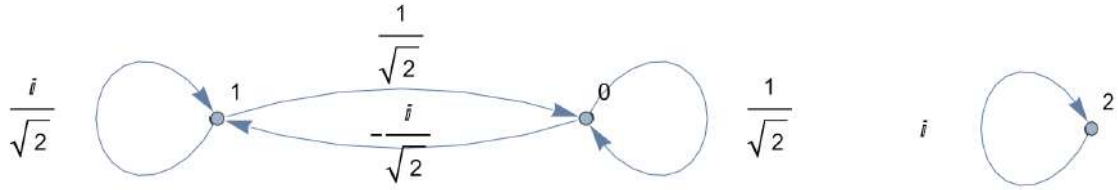

Figura 2.1: Representación gráfica de una partícula en el tiempo. Fuente: elaboración propia

En computación cuántica, esto se traduce en comenzar con un vector par como estado inicial, este será esencialmente la entrada del sistema (valores de entrada). Las operaciones en una computadora cuántica corresponderán a multiplicar el vector por distintos estados (matrices). La salida será el estado resultante después de la aplicación de los distintos estados (multiplicación de matrices resultantes). Esto representa la evolución de un objeto dado un estado inicial. Estas matrices además, tienen la característica de ser reversibles, ayudando a observar el fenómeno físico real en los distintos estados posibles (Chuang and I.L., 2010; Lipton, 2014).

Es decir, si la partícula es un fotón, querrá decir que éste no está en una sola posición, más bien está en muchas posiciones de manera simultánea. Característica conocida como *superposición* (Lanzagorta and Uhlmann, 2008), que se definirá más adelante. En este modelo los bits de información como lo son 0 y 1 para la computación tradicional, se les llama *qubits* su estado puede ser 0 o 1 o ambos simultáneamente (Steane, 1997).

Definición 1 (qubit[2]).
Unidad básica de información en una computadora cuántica. Representada como $|1\rangle$ *o* $|0\rangle$

Definición 2 (estados básicos).
$|0\rangle$ *y* $|1\rangle$*, que simbolizan* $\binom{1}{0}$ *y* $\binom{0}{1}$ *respectivamente vectores ortonormales y ortogonales, que*

[2]La notación $|\rangle$ lleva el nombre de notación de Dirac, que representa vectores unitarios con producto interno de la forma $\binom{a}{b}$

producen una base en $\mathbb{C}^2$ [3].

Definición 3 (superposición).
Combinación lineal de estados básicos de la forma:

$$\alpha_0|0\rangle + \alpha_1|1\rangle \approx \alpha_0\begin{pmatrix}1\\0\end{pmatrix} + \alpha_1\begin{pmatrix}0\\1\end{pmatrix}$$

donde los símbolos α_0 *y* α_1*, serán la amplitud de* $|0\rangle$ *y* $|1\rangle$*, respectivamente.*α_0 *y* $\alpha_1 \in \mathbb{C}$.

Antes de realizar una observación o *medición* al sistema, cada qubit se encuentra simultáneamente en ambos estados básicos con probabilidad proporcional α_0 y α_1 (Jordan, 2008). De tal manera, que la norma es

$$|\alpha_0|^2 + |\alpha_1|^2 = 1$$

Para describir un sistema cuántico con *n qubits* se utiliza el concepto de producto tensorial $|00\rangle = |0\rangle \otimes |0\rangle = \begin{pmatrix}1\\0\\0\\0\end{pmatrix}$.

Definición 4 (n qubits).
Vector de 2^n *dimensiones, cada estado básico corresponde a un vector de la base de* $\mathbb{C}^{2n}$.

Los estados en superposición con $n\,qubits$ son combinaciones lineales de la forma $\alpha_0|0\cdots0\rangle + \cdots + \alpha_{2^n-1}|1\cdots1\rangle$ con $\alpha_i \in \mathbb{C}$ tal que $|\alpha_0|^2 + \cdots + |\alpha_{2^n-1}|^2 = 1$

Definición 5 (medición o medida).
Observación de un sistema o algoritmo cuántico que ocasiona un colapso. El resultado de este colapso se le llama medida o medición.

Por ejemplo, sea el sistema en superposición $\alpha_0|0\rangle + \alpha_1|1\rangle$ al medir, colapsará al estado $|0\rangle$

[3]Nombrado un espacio de *Hilbert*

con probabilidad $|\alpha_0|^2$, o al estado $|1\rangle$ con probabilidad $|\alpha_1|^2$.

Para el caso de n qubits, colapsará al estado $|0\cdots0\rangle$ con probabilidad $|\alpha_0|^2$ de igual forma, a cualquier $i-$ésimo estado con probabilidad $|\alpha_i|^2$.

Otro concepto importante en la computación cuántica son los estados de *entrelazamiento*, que representan la interdependencia que existe entre partículas dentro de la física cuántica, es decir, si dos partículas giratorias están entrelazadas, sus giros están relacionados en todas las direcciones. Este estado no tiene un análogo clásico, por lo tanto, esta es una ventaja cuántica (es decir, donde las computadoras cuánticas funcionan mejor que las clásicas).

También es conocido como *paralelismo cuántico* que nada tiene que ver con el paralelismo en el cómputo tradicional. En el cómputo paralelo se distribuyen procesos de tal manera, que no exista dependencia entre ellos, con la finalidad de que el proceso final no colapse. En cambio, para la computación cuántica la interdependencia permite, por ejemplo, que con la ejecución de una instrucción, se genere n veces el mismo proceso de forma simultánea, este hecho se observa en la Figura 4.1 cuando se aplica la función oráculo a los dos qubits, misma situación ocurre en la Figura 4.6 mostradas en el Capítulo 4 Resultados.

Definición 6 (entrelazamiento (Coles et al., 2018; Rand, 2019)).
Combinación lineal que no puede ser representada como un producto tensorial.

En un estado de entrelazamiento, no será posible modificar el estado de un qubit, sin cambiar los demás. La representación de algunos estados con entrelazamiento son BELL, positivo y negativo que corresponden a las ecuaciones 2.1, 2.2 y 2.3, respectivamente.

$$\frac{|00\rangle + |11\rangle}{\sqrt{2}} \tag{2.1}$$

$$|+\rangle = \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle \tag{2.2}$$

$$|-\rangle = \frac{1}{\sqrt{2}}|0\rangle - \frac{1}{\sqrt{2}}|1\rangle \tag{2.3}$$

En este modelo la manipulación de información es por medio de compuertas lógicas, igual que en el cómputo clásico o tradicional recibe datos de entrada realiza una transformación (función) de la que se obtiene un resultado. La Tabla 2.2, muestra algunas compuertas útiles en la computación cuántica, así como su notación *bra−ket*.

Definición 7 (compuertas cuánticas).
Transformaciones unitarias que tienen inversa sobre qubits y deben cumplir:

1. *Definición del efecto sobre los estados básicos.*
2. *Comportamiento lineal sobre los estados de superposición.*
3. *Ser unitaria y reversible.*

Por ejemplo, sea una función $f(00) = 00, f(01) = 01, f(10) = 10, f(11) = 10$, por lo tanto su comportamiento es de la forma $f(x_1, x_2) = x_1, x_1 \oplus x_2$ donde $\oplus$ denota la suma módulo 2. De hecho, ésta función es una compuerta cuántica conocida como X(NOT), representada por la matriz $\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$ y con notación de Dirac o *bra−ket* mostrada en la Tabla 2.2.

Tabla 2.2: Compuertas cuánticas más utilizadas (Coles et al., 2018; Rieffel and Polak, 2011; Steane, 1997; Lipton, 2014). Fuente: elaboración propia

Nombre	**Operación**
I (Identidad)	$\|0\rangle\langle 0\| + \|1\rangle\langle 1\|$
X (NOT)	$\|0\rangle\langle 1\| + \|1\rangle\langle 0\|$
H (Hadamard)	$\|0\rangle\|+\rangle + \|1\rangle\|-\rangle$
M (Medida)	Esta compuerta, permite observar el resultado, ya que, mientras el proceso esta en ejecución, no son observables los estados sobre qubits.

Una compuerta sumamente importante, ya que es similar al condicionante *if* en la computación tradicional, es la compuerta C−NOT o *Controlled NOT* definida como, $|ab\rangle \Rightarrow |ab\rangle \otimes |a\rangle$ donde $a, b \in \{0, 1\}$. Donde el primer qubit se le llama "control" y el segundo "objetivo". Si el *control*$= 0$, el valor de *objetivo* se mantiene, en caso contrario, si *control*$= 1$, entonces el *objetivo* se niega. La representación ket−bra es $|0\rangle\langle 0| \otimes I + |1\rangle\langle 1| \otimes X$ y la notación matricial se muestra a continuación.

$$CNOT = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \tag{2.4}$$

Existe un método visual útil para representar en forma de diagrama el proceso de concatenación en las compuertas cuánticas, que se muestra en la Figura 2.2. En el diagrama se observan, dos qubits iniciales $|0\rangle$ y $|1\rangle$; cada cable (línea) representa este qubit en el tiempo, sobre el mismo se colocan las compuertas cuánticas que serán aplicadas al qubit correspondiente, al final del cable seleccionado se coloca una compuerta de *medida o medición*, para obtener el resultado, Definición 5.

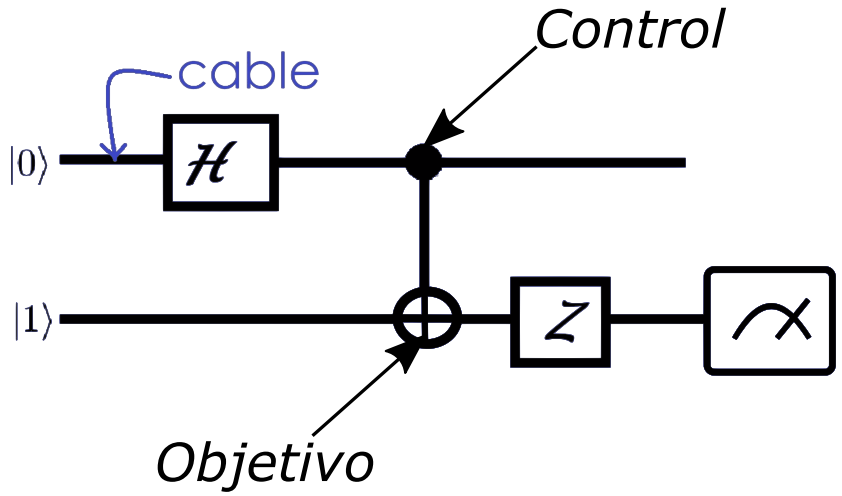

Figura 2.2: Diagrama concatenación de compuertas cuánticas. Fuente: elaboración propia

En particular, en la Figura 2.2 la compuerta C−NOT esta representada por el punto negro sobre el qubit $|0\rangle$ que representa el *control* y sobre el qubit $|1\rangle$ la cruz dentro del círculo señala, el *objetivo*.

2.1.2 Algoritmos cuánticos

Cuando se busca solución a un problema, una manera de abordarlo es mediante un algoritmo para encontrar la(s) solución(es), en computación tradicional se realiza una cadena de instrucciones, como lo son, ciclos, condicionantes, escritura, almacenamiento e impresiones de pantalla, etc. Para el cómputo cuántico es equivalente tener las mismas operaciones por medio de las compuertas cuánticas, convirtiendo la unión de estas estructuras en una posible solución del problema, a esto se le llama algoritmos cuánticos, definición que se desarrolla en el Capítulo 4 Resultados, utilizando la definición desarrollada por Knuth (2011) descrita a continuación.

Definición 8 (algoritmo en el cómputo tradicional).
Proceso, método, técnica o rutina, que contiene un conjunto de reglas secuenciales para resolver un tipo de problemas, esta compuesto por 5 características.

1. *Finito. El algoritmo debe terminar en un número finito de pasos.*
2. *Definitud (preciso). Cada acción que se debe realizar en cada paso, deberá ser especificada de manera precisa.*
3. *Entradas. Puede tener cero o más datos de entrada; la cantidad será asignada antes de iniciar el algoritmo, o de forma dinámica cuando el algoritmo este en ejecución. Estas entradas serán tomadas de un conjunto específico de objetos.*
4. *Salidas. Tiene una o más salidas; la cantidad estará relacionada conforme a las entradas y el tipo de problema a resolver.*
5. *Eficacia. Las operaciones que se realizan deben ser elementales, para que cualquiera pueda realizar una reproducción, por ejemplo, con papel y lápiz, en un tiempo finito.*

Una primera aproximación sobre la definición de un algoritmo cuántico podría ser: proceso que resuelve un problema (de algún área) utilizando comportamientos definidos en física cuántica, el cual recibe n entradas y obtiene al menos una salida. Con base en esta descripción también es posible mejorar la definición antes mencionada, como una concatenación de circuitos cuánticos (Jordan, 2008). En Coles et al. (2018) definen un algoritmo cuántico a partir de tres pasos:

1. Codificación de los datos, que podrían ser clásicos o cuánticos, en el estado de un conjunto de qubits como entrada.
2. Una secuencia de puertas cuánticas aplicadas a este conjunto de entrada qubits.
3. Mediciones de uno o más qubits al final para obtener un resultado interpretable de forma clásica.

La característica esencial de estos algoritmos es aprovechar los comportamientos que existen en la mecánica cuántica, como lo es, la superposición Definición 3 y el entrelazamiento, descritos en las ecuaciones 2.1, 2.2 y 2.3. De lo cual derivan dos conceptos:

Concepto 1 (paralelismo cuántico)**.** *Capacidad de una computadora cuántica para evaluar una función $f(x)$ en los diferentes valores de x de forma simultánea, por medio de una transformación unitaria.*

Por ejemplo, sea una función $f(x) = \{0,1\}^n \rightarrow \{0,1\}$, para el cómputo tradicional se tendría que evaluar la función para cada valor de x, y tal vez ejecutarla de forma simultánea por distintos procesadores. En cambio, con la computación cuántica se realiza una transformación unitaria de la forma $|x,y\rangle \xrightarrow{U_f} |x, y \oplus f(x)\rangle$ para cualquier $x \in \{0,1\}^n$ y $y \in \{0,1\}$, con una sola ejecución se obtiene el valor de $f(x)$. Sin embargo, nuevamente al obtener el valor de la función se deberá recorrer cada valor del vector o conjunto de vectores resultantes, para tal situación se encuentra la interferencia cuántica.

Concepto 2 (interferencia cuántica)**.** *Capacidad de una computadora cuántica para obtener por medio de un operador unitario los resultados de un algoritmo cuántico, en una sola ejecución.*

En la Tabla 2.3 se muestran los algoritmos implementados en el portal IBM (2019) que hacen uso de lo descrito en este Capítulo, para su posterior análisis de complejidad en el Capítulo 4 Resultados. Es importante mencionar que actualmente, en Mosca (2009); Coles et al. (2018); Jordan (2011), se pueden ver implementados y mejorados diversos algoritmos cuánticos.

Tabla 2.3: Algoritmos cuánticos implementados y analizados (Mosca, 2009; Coles et al., 2018; Jordan, 2008).Fuente: elaboración propia

Algoritmo	**Breve descripción**
Algoritmo de Deutsch	Primer algoritmo desarrollado para computación cuántica, que resuelve el decidir si una función es constante
Algoritmo de Búsqueda Grover	Búsqueda de un elemento con una función *oráculo*

2.2 Complejidad cuántica

Antes de intentar contraponer las ideas desarrolladas en la complejidad de la computación cuántica versus la complejidad de la computación tradicional, se debe tener en cuenta los conceptos desarrollados en esta última. La teoría de la complejidad computacional tradicional tiene sus bases en la teoría de la computación, que a su vez se apoya de la resolución en distintas problemáticas en las matemáticas.

Sin embargo, algunos inconvenientes como lo es, la cantidad de tiempo en encontrar la solución de algún problema (en muchos casos la vida de una persona no sería suficiente para observar la solución o respuesta del algoritmo, incluso con métodos más nuevos, como lo es la computación aleatoria o probabilista), originaron la necesidad de clasificar los algoritmos como intratables o tratables. A continuación se explican las definiciones de estas dos áreas para eventualmente contraponerlas y analizar sus distintas características en el Capítulo 4 Resultados, si las hay.

2.2.1 Complejidad algorítmica en el cómputo tradicional

La complejidad algorítmica en la computación tradicional permite conocer la posibilidad de creación de un algoritmo, no de cualquier algoritmo, ya que actualmente, se podría decir que "cualquiera" puede construir un algoritmo, es decir una serie de pasos con los cuales se tiene éxito para resolver algún problema o situación.

Dado que esto no es suficiente, porque no se tienen recursos infinitos como lo planteo Turing (1937), no es posible realizar infinitas operaciones, mucho menos tener un almacenamiento infinito. Por tanto, medir estas características permitirán definir que tan "bueno" es un algoritmo.

En la práctica habrá algoritmos que puedan cumplir con las características descritas en la Definición 8, o no; esto tampoco significará que se desarrollo un "buen" o "mal" algoritmo. Algunos criterios, pueden ser el tiempo que lleva ejecutarlo, expresado como el número de veces que se realiza cada paso; la adaptabilidad de un algoritmo, la claridad y elegancia, etc (Papadimitriou, 1993).

La idea general es tomar un algoritmo, que en principio cumple con las características de la Definición 8, y determinar (medir de alguna manera) sus características de rendimiento, resolviendo de esta forma si un algoritmo es óptimo o no. Para lo cual, se describen las siguientes funciones que establecen límites para una clasificación (Papadimitriou, 1993).

Definición 9 (notación asintótica (Sipser, 2012)).
Sean f, g, h y j funciones tales que, $f, g : \mathbb{Z}^+ \to \mathbb{Z}^+$ y $j, h : \mathbb{N} \to \mathbb{R}^+$.

Se escribe $f(n) = O(g(n))$ (f está en el orden de g), si hay un n_0 y un c tal que, para todo $n \geq n_0$, $f(n) \leq c \cdot g(n)$. Informalmente significa que f crece como g o más lento; cota superior asintótica.

Se escribe $f(n) = \Omega(g(n))$, si hay un n_0 y un c tal que, para todo $n \geq n_0$, $f(n) \geq c \cdot g(n)$. Cota inferior asintótica.

Se escribe $f(n) = \Theta(g(n))$, si $f(n) = O(g(n))$ y $f(n) = \Omega(g(n))$. Las dos funciones crecen asintóticamente iguales.

Se escribe $j(n) = o(h(n))$ si $\lim_{n\to\infty} \frac{j(n)}{h(n)} = 0$, esto es que, para cualquier número $c > 0$, existe un número n_0, tal que $j(n) < c \cdot h(n)$ para todo $n \geq n_0$

Se expresará:
*$f(n)$ es **polinómica**; si existen las constantes A, c tal que $\forall n, f(n) \leq An^c$.*
*$f(n)$ es **exponencial**; si existen las constantes A, c tal que $\forall n, f(n) \leq Ae^{n^c} = A\exp(n^c)$.*

Estas funciones se aplican sobre el algoritmo desarrollado, contabilizando sus instrucciones.

Dado que son funciones generales, significa que pueden representar "casi" cualquier realidad, en particular límites sobre solución a problemas computacionales, por esto su uso colectivo en distintos campos de la teoría de la computación, no solo para la teoría de la complejidad (Wigderson, 2019). Teniendo estas herramientas se puede cambiar él "que tan bueno es" un algoritmo, a cual función representa la eficiencia del algoritmo. En este texto eficiencia representa:

Definición 10 (eficiencia).
Se dice que una algoritmo es eficiente si tiene éxito (resuelve, realiza la acción adecuada dependiendo del contexto) y utiliza la menor cantidad de recursos (memoria y tiempo).

Hasta este momento no se ha hablado de una implementación en algún dispositivo real. Es decir, estas herramientas deben permitir, antes de ejecutar el algoritmo en una máquina real, verificar el tiempo aproximado o cantidad de recursos necesarios para la ejecución de la propuesta de algoritmo, además de poder realizar una prueba de escritorio, esto es, hacer la ejecución para ejemplos elementales con papel y lápiz.

Por otro lado, una manera de analizar la complejidad algorítmica tradicional es por medio de la viabilidad o factibilidad de un problema, sobre todo cuando es factible reducir la cuestión a problemas de decisión, es decir, funciones *booleanas*, dado que las compuertas lógicas se rigen por este mismo modelo.

Definición 11 (factible o factibilidad).
Una función booleana $f = [f_n]$ es factible siempre que la f_n individual es ejecutada por compuertas de tamaño $n^{O(1)}$.

La definición 11 establece un límite para una solución de una función booleana, dicha función es viable trasladarla hacia un problema o algoritmo de decisión, este límite no solo es aplicable al tiempo de ejecución, además puede ir enfocado al espacio de una implementación, y llamarla factible o no factible. Dado que esta tesis se enfoca en un análisis sobre el espacio y el tiempo,

la revisión para problemas factibles, en los términos antes mencionados se deja a futuro.

Sin embargo, también el análisis del problema a resolver es una variante, al cual se puede abordar por medio de la complejidad de polinomios básicos asociando una función $s(f)$, la cual representa el tamaño mínimo del circuito aritmético de un polinomio f, y de esta forma determinar su complejidad (Wigderson, 2019). Aunque esta manera es distinta, ya que se deben clasificar, primero, los problemas en la computación cuántica, cuestión abierta para futuros trabajos.

Por otra parte, las *clases de complejidad*, están definidas como el conjunto de todos los lenguajes o problemas que se pueden resolver sobre una máquina M que opera con procedimiento apropiado para cualquier entrada x, que gasta, a lo más $f(|x|)$ unidades de un recurso específico, está f, es una función que limita la eficiencia. Para definir esta f, se debe tener cuidado, ya que hay infinitas funciones, diferenciales, integrables; en los distintos conjuntos de números, $\mathbb{Q}, \mathbb{C}, \mathbb{R}, \mathbb{Z}, \mathbb{N}$, etc. Entonces se describe una función a partir del tamaño de entrada y hasta el número de pasos en el cálculo.

Concepto 3 (funciones apropiadas de complejidad)**.** *Sea $f : \mathbb{Z}^+ \to \mathbb{Z}^+$. Se dice que f es una función apropiada si $\forall\, n\ f(n+1) \geq f(n)$. Entonces, dada una M_f que sobre alguna entrada x ejecuta una cadena de tamaño $n \cdot f(|x|)$, donde n es un símbolo "casi−blanco", M_f se detiene en $O(|x| + f(|x|))$ pasos y utiliza $O(f(|x|))$ espacio, además de su entrada.*

Además, se debe realizar una selección de un modelo de computación, así como sus particularidades asociadas. Recordando algunos modelos de la computación tradicional como, RAM$_u$ o RAM$_l$[4]. De forma especifica, este trabajo retoma la máquina Turing, TM (Aho and Hopcroft, 1974), que sirven para construir el modelo de máquina de Turing cuántica (QTM[5]), que se revisará en capítulos posteriores.

Para desarrollar una función que permita determinar estos límites, se debe entender que el

[4] $_u$ uniforme o $_l$ logarítmica

[5] Por sus siglas en ingles Quantum Turing Machine

número de pasos que utiliza una máquina antes de entrar en un estado de aceptación o rechazo, es el tiempo computacional o número de cálculo. El número generalmente dependerá del tamaño de la entrada.

Las siguientes definiciones son fundamentales para ciencias de la computación, en particular para la complejidad computacional. Una característica sumamente importante es que no dependen del modelo de computación elegido, mientras sea un modelo polinomialmente equivalente al de una máquina de Turing, esto hace posible trasladar la complejidad sobre distintos modelos de cómputo que son polinomialmente equivalentes. Estas definiciones se enfocan en el problema que se quiere resolver y los recursos que probablemente utilizará en un peor caso (pensamiento contrario).

Definición 12 (P).
Clase de lenguajes/problemas que pueden ser ejecutados/resueltos por una TM determinista (computadora clásica) en tiempo polinomial.

$$\mathbf{P} = \bigcup_k TIME(n^k)$$

Definición 13 (NP).
Clase de lenguajes/problemas que pueden ser ejecutados/resueltos por una TM no-determinista en tiempo polinomial.

$$\mathbf{NP} = \bigcup_k NTIME(n^k)$$

Para observar el comportamiento conforme al espacio utilizado, se utilizan las siguientes definiciones.

Definición 14 (PSPACE).
Problemas de espacio polinómico que son ejecutados en un máquina de Turing determinista.

$$\mathbf{PSPACE} = \bigcup_k SPACE(n^k)$$

Para esta definición su contraparte no-determinista de espacio se denomina **NPSPACE**. Algunos ejemplos se muestran en la siguiente tabla.

Tabla 2.4: Ejemplos de complejidad algorítmica tradicional sobre algoritmos eficientes (Papadimitriou, 1993; Sipser, 2012).Fuente: elaboración propia

Notación asintótica	Clase	Problema
$\mathcal{O}(1)$	**P**	suma de dos números enteros
$\mathcal{O}(n)$	**P**	búsqueda lineal
$\mathcal{O}(n \log n)$	**P**	ordenar un arreglo de tamaño n
$\mathcal{O}n^3$	**P**	multiplicación de matrices
$\mathcal{O}2^k$	**NP**	problema de la satisfacibilidad (SAT)
$\mathcal{O}n!$	**NP**	problema del agente viajero

Para la complejidad algorítmica cuántica, gracias a la creación de un modelo equivalente de máquina de Turing cuántica desarrollado por Bernstein and Vazirani (1997), permiten iniciar este desarrollo con algunos conceptos de la complejidad algorítmica en computación tradicional. En principio los algoritmos seleccionados, que se explican en el Capítulo 3, estarían clasificados con la teoría de complejidad tradicional como NP (Deutsch, caso de entrada suficientemente grande) y NP (Grover).

2.2.2 Complejidad algorítmica en el cómputo cuántico

Generalmente, cuando se habla de eficiencia se hace referencia al tiempo de ejecución, donde cada instrucción se realiza en un tiempo t, por lo cual si se tiene una cantidad de instrucciones suficientemente grande, entonces se ocupara una cantidad de tiempo notable. Sin embargo, en computación cuántica se debe dar por hecho que la eficiencia en tiempo comparado con la computación tradicional es mejor, debido a que fue creada con este fin.

Dar solución a problemas en donde la computación tradicional no tenía la posibilidad de dar un resultado accesible (tiempo y recursos). Esto es, desarrollar una opción que permitiera la ejecución de estos algoritmos en un tiempo conceptual distinto al tiempo conocido y con otro paradigma, la mecánica cuántica. Feynman (1986), señaló que era necesario construir

una computadora cuántica que fuera capaz de simular eficientemente la dinámica de sistemas cuánticos, ya que resultaba imposible ejecutarlo en la computadora tradicional.

Por lo cual, es natural creer que resolverá todos los problemas que una computadora tradicional no puede, además de resolver los problemas conocidos en mucho menor tiempo y con menor cantidad de recursos. Hay que señalar, que sigue siendo necesario delimitar los alcances, porque en la realidad los recursos son finitos, por lo cual la siguiente definición.

Definición 15 (calculo cuántico factible (Lipton, 2014)).
Un cálculo cuántico C en s qubits es factible siempre que $C = U_t U_{t-1} \dots U_1$, donde cada U_i es una operación factible, y s y t están limitados por un polinomio designado por el número de n de qubits de entrada.

La definición anterior permite de forma preliminar, delimitar que características debe cumplir un algoritmo cuántico. En el mismo sentido, en la teoría de la complejidad tradicional desarrollaron conceptos útiles para construcción y análisis de los algoritmos, algunos de ellos explicados en Papadimitriou (1993)y Wigderson (2019). Es posible trasladar estos conceptos casi directamente al modelo de computación cuántica, además funcionan para dar continuidad y solidez a los argumentos aquí mostrados, útiles para la construcción de *métrica para la complejidad algorítmica cuántica*.

Concepto 4 (modelo computacional cuántico)**.** *Recursos desarrollados en la computación cuántica como **qubit, entrelazamiento, superposición**, flujo de trabajo por medio compuertas cuánticas (transformaciones unitarias desarrolladas en los complejos) Capítulo 2.*

Concepto 5 (pensamiento al límite (asintótico))**.** *Estudio de problemas "retadores" para la computación cuántica (instancias sumamente grandes) ayudan a observar las capacidades de la computación cuántica.*

Concepto 6 (pensamiento contrario (adversario))**.** *Imaginar el peor caso y preparase para eso cambiando las restricciones especificas por casos más generales, tal vez para el cómputo cuántico esto no siempre es sencillo ya que pensar de manera general es intentar pensar en un mundo cuántico, pero existirán casos en los que será más sencillo hablar de la complejidad algorítmica cuántica utilizando este método.*

Concepto 7 (clasificación)**.** *Organizar los problemas en clases de complejidad de acuerdo con los recursos cuánticos que se requieren.*

Concepto 8 (reducciones)**.** *Asumir que se puede resolver un problema de manera eficiente, de esta manera se pueden explorar todas las opciones posibles, como lo es el resolver problemas más pequeños que ayuden al problema más general o más grande. Por ejemplo, el algoritmo de Deutsch ilustrado en el Capítulo 3.*

Concepto 9 (completes o completud)**.** *Identificar los problemas más difíciles en una clase de complejidad (definido previamente).*

Concepto 10 (barreras)**.** *Momento en el cual no hay un avance acerca de una solución a un problema, por un periodo largo de tiempo; se analizan las técnicas utilizadas hasta ese momento y se intenta argumentar formalmente que no serán adecuadas para la solución.*

Los conceptos mostrados son una herramienta valiosa para la complejidad algorítmica cuántica, comprobando su utilidad en la teoría de complejidad algorítmica tradicional. Estos pueden ser usados a la hora de desarrollar un algoritmo cuántico, aunque primero se debe elegir el modelo cuántico. Una aproximación teórica desarrollada en Bernstein and Vazirani (1997), demuestra que no es suficiente con una máquina de Turing construida como un análogo cuántico, es decir se necesita una parte determinista, una parte probabilística y una reversible, concluyendo que una máquina de Turing cuántica se define como la unión de distintas máquinas de Turing. Modelo que se usará en esta tesis.

También desarrolla los conceptos base para la construcción de una complejidad algorítmica cuántica, como la mostrada en la definición17, las cuales son sumamente utilizada para clasificar y diferenciar algoritmos cuánticos como en Kempe et al. (2011); Watrous (2012); Farhi et al. (2019); Bencivenga et al. (2021); Brown et al. (2021) y Bouland et al. (2021). Si se desea entender a profundidad se puede revisar (Bernstein and Vazirani, 1997; Lipton, 2014), donde describen las demostraciones de los teoremas utilizados para las definiciones desarrolladas.

Definición 16 (BPP, *bounded error on probabilistic computer in polynomial-time*).

La clase de todos los lenguajes/problemas que se pueden ejecutar/resolver en tiempo polinomial con probabilidad $1/3$ *sobre una máquina de Turing probabilística (Jordan et al., 2018).*

Definición 17 (BQP, *bounded error on quantum computer in polynomial-time*).
La clase de todos los lenguajes/problemas que se pueden ejecutar/resolver de manera eficiente en tiempo polinomial sobre una QTM con un error acotado por la probabilidad de al menos $2/3$ *(Jordan et al., 2018).*

Decir "error acotado" implica formalizar las condiciones bajo las cuales se puede amplificar la probabilidad de éxito sobre un algoritmo cuántico. Se define BQP tanto para funciones como para problemas de decisión.

Definición 18 (BQPTime).
Conjunto de lenguajes que son aceptados con probabilidad $\frac{2}{3}$ *por alguna QTM cuyo tiempo de ejecución sobre cualquier entrada de longitud* n *está limitado por* $T(n)$.

Definición 19 (EQP, *exact on quantum computer in polynomial-time* libre de errores).
El conjunto de lenguajes que son exactamente aceptados por alguna QTM en tiempo polinomial. Es decir, pueden ser resueltos en tiempo polinomial con una probabilidad igual a 1.

Definición 20 (EQPTime).
Conjunto de lenguajes que son exactamente aceptados por una QTM cuyo tiempo de ejecución en cualquier entrada de longitud n *está limitado por* $T(n)$.

Con esta nueva clasificación es posible catalogar los algoritmos elegidos para esta tesis como EQP, para el algoritmo de Deutsch, y el algoritmo cuántico de Grover como BQP.

Es importante mencionar que la teoría de la complejidad en ciencias de la computación surgió en 1960 (Wigderson, 2019) es decir, 20 años después de que Turing desarrollara su teoría que abriría la puerta a la construcción de algoritmos con formalismos matemáticos.

Entonces, notando que la computación cuántica fue propuesta por Feynman (1982) y 15 años después en Bernstein and Vazirani (1997) desarrollan un análogo a la máquina de Turing, para una manipulación más afable de este nuevo paradigma (en realidad, trata de demostrar la falsedad de la tesis de Church-Turing, construyendo una máquina de Turing cuántica), sumando la construcción de máquinas cuánticas reales en Arute et al. (2019), agregando además, los algoritmos concebidos de manera teórica a lo largo de estos años como los implementados en Coles et al. (2018) o los mostrados en Jordan (2011). Posiblemente, sea tiempo de iniciar una breve discusión acerca de la complejidad algorítmica en el cómputo cuántico.

Partiendo del hecho de que la computación cuántica surge de los conceptos en mecánica cuántica, como una posibilidad de simular sistemas cuánticos imposibles o intratables, al menos para la computación tradicional. Y de como surgió la clasificación de algoritmos en la computación tradicional observando los alcances de la misma. Podría pensarse que se deben "conocer" todos los límites de la computación cuántica para tener herramientas suficientes e iniciar con la clasificación de algoritmos para el cómputo cuántico, hecho que es imposible. Sin embargo, algunos problemas en la mecánica cuántica se pueden caracterizar de distintas maneras, haciendo uso de los conceptos antes mencionados.

Un objetivo de la complejidad algorítmica cuántica debe ser permitir generalizar o ampliar esta propuesta de medida no solo para un problema en particular si no, para problemas con similares propiedades. Se asumirá que todos los problemas aquí presentados tienen la misma representación sobre compuertas cuánticas definidas en la sección 2.1.

Una característica principal en algunos algoritmos cuánticos es el llamado *oráculo*, nombrado de esta forma porque permite la ejecución de varios procesos de forma simultánea con solo una aplicación, una primera interpretación mediante una TM, que se retomará en el Capítulo 4 Resultados es:

Definición 21 (TM Oráculo (Atallah and Blanton, 2009)).

Una TM de oráculo es una TM T normal con una cinta de consulta de oráculo adicional, un estado especial $q?$, y dos estados distinguidos etiquetados q_y y q_n. Sea A cualquier lenguaje bajo el alfabeto Σ. Siempre que T entre en el estado $q?$ con alguna cadena $z \in \Sigma^*$ en la cinta de consulta, el control pasa al estado q_y si $z \in A$, o al estado q_n si $z \notin A$. El cálculo continúa normalmente hasta la siguiente vez que la máquina ingresa $q?$. La máquina T con una elección dada por el oráculo A se denota por T^A.

En Montanaro (2016), se realiza una clasificación "informal" para el modelo de computación cuántica, y menciona que existe una herramienta oculta que no se toma en cuenta en los algoritmos cuánticos, el *oráculo*, con la definición 21 antes descrita es posible empezar a tomar en cuenta este proceso, de manera más detallada.

Otra manera de abordar el diseño es observando los problemas para los que "fue diseñada a resolver", por ejemplo *el problema del agente viajero, la factorización de un número de 1000 dígitos, accesibilidad de un grafo, etc* de forma más general se puede intentar, entonces clasificar los tipos de problemas como, problemas de decisión, problemas de optimización, etc.

Pero esta forma de clasificación, requiere entender a profundidad acerca de ese campo, por ejemplo, la física y así determinar que al menos en ese campo la solución es irrealizable. Por tanto, este trabajo será un primer acercamiento a la complejidad algorítmica cuántica utilizando conceptos establecidos en la teoría de la computación.

Si bien, en la teoría de la complejidad computacional existen sentencias que no han sido demostradas, que se utilizan como convenciones que funcionan, aún siguen siendo preguntas fundamentales abiertas en teoría de la computación y matemáticas. La investigación en teoría de la complejidad computacional se centra en estas cuestiones y un primer paso para trabajar en estas preguntas es identificar los problemas, ya sea usando las definiciones descritas anteriormente como NP o NPSPACE o generando otras.

Esta tesis desarrolla elementos para iniciar con un tipo de clasificación en el cómputo cuántico, pero no se enfocará en resolver ese tipo de preguntas abiertas, así que se usarán las convenciones para avanzar en la propuesta de *métrica para la complejidad algorítmica cuántica*.

2.3 Algoritmos seleccionados

Actualmente existen diversas versiones de algoritmos cuánticos, por ejemplo en Coles et al. (2018) recopilan y detallan muchos de ellos. Para esta tesis, la selección está conformada por dos algoritmos que son base para la generación de otros tantos, incluso de áreas nuevas como lo es el aprendizaje automático cuántico (*Quantum Machine Learning*) (Wittek, 2014). El *algoritmo de Deutsch* y el *algoritmo de Grover*.

2.3.1 Algoritmo de Deutsch

David E. Deutsch actualmente es profesor visitante en la universidad de *Oxford*, en donde fundó el *Centre for Quantum Computation, the Clarendon Laboratory*, y es miembro de la *Royal Society*. Fue precursor en el área de computación cuántica, debido a su propuesta de un primer algoritmo que utiliza conceptos de la mecánica cuántica, es decir, el primer algoritmo cuántico desarrollado, conocido como el *algoritmo de Deutsch*.

Este algoritmo es útil para entender los conceptos de superposición y entrelazamiento, ya que, muestra la capacidad de realizar múltiples evaluaciones en una sola ejecución, utilizando la idea de evaluar una función y verificar si ésta es constante o balanceada. Este algoritmo usa un *oráculo* descrito en la definición 21, un operador unitario lineal que es aplicado sobre un estado de superposición.

Objetivo. Decidir si dada una función $f : \{0,1\}^n \to \{0,1\}$ es constante o "balanceada".

Esto es, si la cantidad de 0's y 1's es la misma se le llama *balanceada*, en otro caso es constante (Deutsch, 1985; Mosca, 2009; Rieffel and Polak, 2011; Coles et al., 2018).

Entonces, sea una $f(x) : \{0,1\} \rightarrow \{0,1\}$, la cual se pretende conocer si $f(0) = f(1)$ ó $f(0) \neq f(1)$. En otras palabras, saber si la función es constante o no.

Algoritmo cuántico. Dado un circuito que implementa $|x\rangle|0\rangle \rightarrow |x\rangle|f(x)\rangle$. Deutsch demostró que usando una ejecución de este circuito es posible calcular 2.5.

$$\frac{1}{\sqrt{2}}|0\rangle|f(0)\rangle + \frac{1}{\sqrt{2}}|1\rangle|f(1)\rangle \tag{2.5}$$

La ecuación 2.5, viene dada por la aplicación de una compuerta *Hadamard* 2.2, seguido de una *CNOT* 2.4, vistas en el Capítulo 2. Esto conforma él *oráculo* o transformación unitaria, que se comporta como $U_f|x\rangle|y\rangle \longrightarrow |xy \otimes f(x)\rangle$.

En la Figura 2.3 se muestra la concatenación de compuertas cuánticas necesarias. En términos de matrices la representación matemática e lee de derecha a izquierda y esta representada como: $(H \otimes I)U_f(H \otimes H)|0,1\rangle$, s.

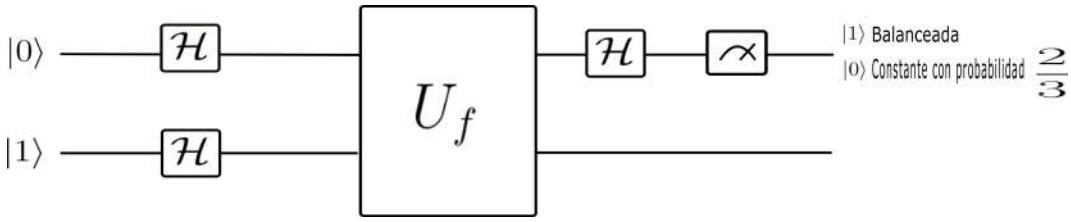

Figura 2.3: Diagrama algoritmo de Deutsch. Fuente:elaboración propia basada en (Yanofsky and Mannucci, 2009)

Se puede observar en el diagrama que, después de tener una superposición se aplica la transformación U_f, quedando una superposición de la forma:

$$(-1)^{f(x)}|x\rangle\left[\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right]$$

Con el primer qubit $|x\rangle$ en estado de entrelazamiento positivo 2.2 se tiene

$$\left[\frac{(-1)^{f(0)}|0\rangle + (-1)^{f(1)}|1\rangle}{\sqrt{2}}\right]\left[\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right] \tag{2.6}$$

Entonces las posibles opciones son $f(0) = f(1)$ o $f(0) \neq f(1)$.

$$\begin{cases} (\pm 1)\left[\frac{|0\rangle+|1\rangle}{\sqrt{2}}\right]\left[\frac{|0\rangle-|1\rangle}{\sqrt{2}}\right], & Si\ f\ es\ constante \\ \\ (\pm 1)\left[\frac{|0\rangle-|1\rangle}{\sqrt{2}}\right]\left[\frac{|0\rangle-|1\rangle}{\sqrt{2}}\right], & Si\ f\ es\ balanceada \end{cases} \tag{2.7}$$

Dado que la compuerta *Hadamard* es reversible, su uso en serie sobre una estado cuántico, devuelve el estado inicial. A continuación algunas de sus propiedades.

$$H|+\rangle = |0\rangle$$
$$H|-\rangle = |1\rangle$$
$$H|0\rangle = |+\rangle$$
$$H|1\rangle = |-\rangle$$

Por tanto, al aplicar una compuerta *Hadamard* al primer qubit de la ecuación 2.7, se reduce a,

$$\begin{cases} (\pm 1)|0\rangle\left[\frac{|0\rangle-|1\rangle}{\sqrt{2}}\right], & Si\ f\ es\ constante \\ \\ (\pm 1)|1\rangle\left[\frac{|0\rangle-|1\rangle}{\sqrt{2}}\right], & Si\ f\ es\ balanceada \end{cases} \tag{2.8}$$

Por ejemplo, si $f(0) = 1$ y $f(1) = 0$, se obtiene:

$$-1|0\rangle\left[\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right]$$

Luego, dejando de lado el signo de (± 1), solo importara el qubit resultante. Si éste se encuentra en el estado $|0\rangle$, será una f *constante* en otro caso será *balanceada*.

Retomando la Figura 2.3, se observa la cantidad total de compuertas cuánticas aplicadas, y el total de valores aplicados. Entonces, al menos, necesita 2 qubits (valores) de entrada. Este algoritmo muestra que los procesos cuánticos no tienen que dar resultados probabilistas, también pueden ser certeros. Además, solamente se ejecuta una vez para obtener el resultado.

En cambio un *algoritmo clásico* quedaría como $return(f(0) == f(1))$. Donde es evidente que para cada valor la función se tendría que evaluar de manera separada (Mosca, 2009; Rieffel and Polak, 2011). Sin embrago, al hablar de complejidades, tal vez no se tengan los mismos resultados.

Análisis sobre algoritmo de Deutsch tradicional vs cuántico

A continuación se presentan las dos versiones del algoritmo a comparar en el Capítulo 4 Resultados.

Algoritmo tradicional: evaluación de una función.

Entradas: una función f y dos valores x_1 y x_2 para evaluar.

Salida: constante o balanceada.

Procedimiento:

```
si (f(x1)==0) { si (f(
   x2)==0){
  print(Constante)
 }else{
  print(Balanceada)}
} else {
 si(f(x2)==0){
  print(Balanceada)
 } else {
  print(Constante)}
}
```

Algoritmo cuántico: evaluación de una función.

Entradas: una función U_f y un qubit $|0\rangle$ y $|1\rangle$.

Salida: constante $|0\rangle$ o balanceada $|1\rangle$.

Procedimiento:

1 $|0\rangle \rightarrow |1\rangle$

2 $\frac{|0\rangle+|1\rangle}{\sqrt{2}} \otimes \frac{|0\rangle-|1\rangle}{\sqrt{2}}$

3 $|x\rangle \left(\frac{|f(x)\rangle-|1\rangle \otimes |f(x)\rangle}{\sqrt{2}}\right)$

4 $(-1)^{f(x)} \left(|x\rangle \left(\frac{|0\rangle-|1\rangle}{\sqrt{2}}\right)\right)$

2.3.2 Algoritmo de búsqueda de Grover

Lov Kumar Grover es un científico en computación, creador del algoritmo de búsqueda en base de datos haciendo uso, al menos teóricamente, de la computación cuántica. En 1996, era el segundo algoritmo más famoso (después del algoritmo de Shor) en computación cuántica. Este permite encontrar un elemento específico dentro de un arreglo desordenado de tamaño N, con una probabilidad de éxito de $\frac{1}{2}$, realizando $O(\sqrt{N})$ operaciones. Note que para el cómputo tradicional, se sabe que el peor de los casos tendría que realizar $O(N)$ consultas. Es

evidente la disminución sobre la cantidad de operaciones en un algoritmo clásico.

Además, se demostró que ninguna máquina de Turing cuántica puede hacerlo en menos de $O(\sqrt{N})$ operaciones (Coles et al., 2018). Sin embargo, este enfoque no solo es exclusivo para búsquedas en bases de datos; Mosca (2009) plantea que, es posible utilizarlo para cualquier problema NP (el conjunto de problemas para los que se puede verificar una solución en tiempo polinomial), definición 13. Por ejemplo, para encontrar una solución a una instancia del problema $3-$SAT.

Algoritmo cuántico (problema de búsqueda). Buscar en una base de datos desordenada de tamaño N, indexados por los números $0, 1, \ldots, N-1$. Por conveniencia se asume que $N = 2^n$ de modo que el índice puede almacenar n bits. Asimismo, se considera que el problema tiene exactamente M soluciones, con $1 \leq M \leq N$ (Chuang and I.L., 2010).

Para el desarrollo del algoritmo (circuito) cuántico se utiliza un qubit auxiliar q y se invierte si dada una función f es igual a 1 para una entrada x. La función está definida en 2.9, donde x es un conjunto de variables binarias y x^* es el elemento buscado.

$$f(x) = \begin{cases} 1 \quad si \quad x = x^* \\ \\ 0 \quad si \quad x \neq x^* \end{cases} \tag{2.9}$$

Por ejemplo, sea $x = (x_1, x_2)$ y se desea encontrar x^* tal que ${x_1}^* = 1$ y ${x_2}^* = 1$. Esencialmente significa que la función $f(x)$ se comporta como una compuerta lógica *AND*, es decir lo puedes resolver con computación tradicional. Sin embargo, existe una versión de la compuerta $C-$NOT vista en el Capítulo 2, mostrada en forma matricial en 2.4, conocida también como "compuerta de Toffoli" (Ying, 2016), que es similar a la compuerta *AND* lógica.

Para este caso, la compuerta toma tres bits de entrada y tres bits como salida. Los primeros 2

bits no se modifican. El tercer bit se invierte si los 2 primeros bits son iguales a 1. De forma general el comportamiento se puede definir como $|x\rangle|q\rangle \longrightarrow |x\rangle|f(x) \oplus q\rangle$, donde $|x\rangle$ es el índice, $\oplus$ denota la suma módulo 2, y $|q\rangle$ es el qubit oráculo, el cual cambia si $f(x) = 1$, en otro caso no lo hace. Se puede verificar si x es una solución de la búsqueda, preparando $|x\rangle|0\rangle$, aplicando el oráculo y verificando si el qubit cambio a $|1\rangle$(Coles et al., 2018).

Para este algoritmo es útil inicializar al qubit oráculo con $\frac{|0\rangle-|1\rangle}{\sqrt{2}}$, el estado de entrelazamiento negativo 2.3. Asimismo, se puede verificar si x es una solución al problema, aplicando el oráculo al estado $|x\rangle\left(\frac{|0\rangle-|1\rangle}{\sqrt{2}}\right)$. Si los estados $|0\rangle$ y $|1\rangle$ se intercambian, indicará que x es una solución al problema quedando $-|x\rangle\left(\frac{|0\rangle-|1\rangle}{\sqrt{2}}\right)$[6], en otro caso, no mostrará ningún cambio. Por tanto, la acción del oráculo quedaría como:

$$|x\rangle\left(\frac{|0\rangle-|1\rangle}{\sqrt{2}}\right) \rightarrow (-1)^{f(x)}|x\rangle\left(\frac{|0\rangle-|1\rangle}{\sqrt{2}}\right) \tag{2.10}$$

Observando que el qubit oráculo no cambia durante la ejecución del algoritmo, y por convención, se puede omitir para simplificar la función.

$$|x\rangle \xrightarrow{O} (-1)^{f(x)}|x\rangle \tag{2.11}$$

Entonces, se dice que el oráculo marca las soluciones al problema de búsqueda, cambiando la fase (girando). Por lo tanto, para un problema de N elementos con M soluciones se necesita aplicar $O(\frac{N}{M})$ veces el oráculo de búsqueda (Chuang and I.L., 2010). En la Figura 2.4 se muestra la implementación a manera de compuertas y funciones descritas en 2.12 y 2.13.

[6]Recuerde que $a - b = (-1)(b - a)$

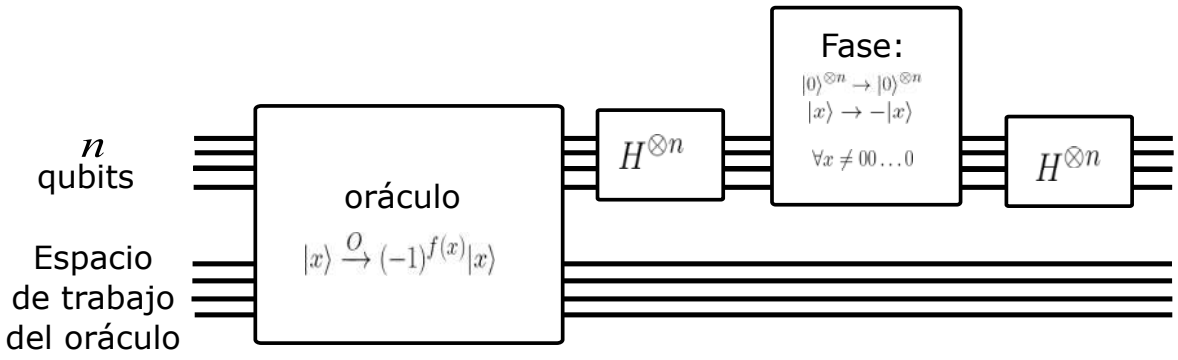

Figura 2.4: Diagrama de algoritmo de Grover. Fuente: elaboración propia basaba en (Chuang and I.L., 2010)

Reescribiendo las funciones 2.9 y 2.11, *oráculo* y *difusor*, respectivamente se obtiene:

$$U_{f_o} : \begin{cases} |x^*\rangle & \rightarrow -|x^*\rangle \\ |x\rangle & \rightarrow |x\rangle \end{cases} \forall x \neq x^* \Rightarrow U_f = 2|x^*\rangle\langle x^*| - I \tag{2.12}$$

Y

$$U_{f_d} : \begin{cases} |0\rangle^{\otimes n} & \rightarrow |0\rangle^{\otimes n} \\ |x\rangle & \rightarrow -|x\rangle \end{cases} \forall x \neq 00\ldots0 \Rightarrow U_{f_d} = 2|0\rangle\langle 0|^{\otimes n} - I \tag{2.13}$$

Dichas funciones se emplean en el Capítulo 4. Primero se realiza una superposición uniforme y al último qubit se invierte a $|1\rangle$, posteriormente se aplica la función *operador Grover* 2.12, R veces con $R \leq \lceil \frac{\pi}{4}\sqrt{\frac{N}{M}} \rceil$, en particular, para el caso $M = 1$, implica $R \approx \lceil \frac{\pi\sqrt{2^n}}{4} \rceil$ veces, por último se realiza una observación para obtener el valor de la posición, si es que el valor fue encontrado.

Análisis sobre algoritmo de Grover tradicional vs cuántico

Dado que existen distintas formas de abordar el problema para computación tradicional, en particular se observará el caso donde el arreglo es desordenado. A continuación se presentan

las dos versiones del algoritmo para la búsqueda de un elemento utilizados en el Capítulo de 4.

Algoritmo tradicional: encontrar la posición del valor x^*, en un arreglo desordenado de tamaño N.

Entradas: un arreglo desordenado de tamaño N y valor buscado x^*.

Salida: posición del valor x^*, si es que se encuentra.

Procedimiento tradicional:

```
pos = 0 //posicion actual en el vector
while pos < N{
 if (vec[pos] == dato){
  Print(Posicion es pos)
 }   else{
  pos = pos + 1}
}
Print(No se encontro el elemento)
```

Algoritmo cuántico: encontrar la posición del valor x^*, en un arreglo desordenado de tamaño N, dado un oráculo U_f (transformación).

Entradas: $N+1$ qubits en estado básico $|0\rangle$ y un *oráculo* de la forma $O|x\rangle|q\rangle = |x\rangle|q \oplus f(x)\rangle$, donde $\forall\ 0 \leq x < 2^n$, $f(x) = 0$, excepto para x_0, tal que $f(x_0) = 1$.

Salida: posición del valor encontrado x^*.

Procedimiento cuántico:

1 $|0\rangle^{\otimes n}|0\rangle$

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

3 $[(2|\psi\rangle\rangle\psi|-I)O]^R \frac{1}{\sqrt{2^n}} \sum_{x=0}^{2^n-1} |x\rangle \left[\frac{|0\rangle-|1\rangle}{\sqrt{2}}\right] \approx |x_0\rangle \left[\frac{|0\rangle-|1\rangle}{\sqrt{2}}\right]$

4 x_0 medida de los primeros n qubits

Capítulo 3

Metodología

La mayoría de los algoritmos elaborados para la computación cuántica han sido desarrollados sobre modelos de computación discreta, es decir que manejan, espacios discretos o trabajan en tiempo discreto o ambos.

Múltiples ideas principales que guían el desarrollo de la computación cuántica tienen sus bases en trabajos sobre la computación reversible (Mosca, 2009). Esta característica es fundamental para el desarrollo de algoritmos cuánticos, porque además de utilizar conceptos de física cuántica, los resultados podrán ser verificables.

La metodología para desarrollar una propuesta que permita evaluar la complejidad de un algoritmo en los distintos paradigmas computacionales (cuántico y tradicional), se resume en el esquema mostrado en la Figura 3.1. Posteriormente se detalla cada una de sus fases.

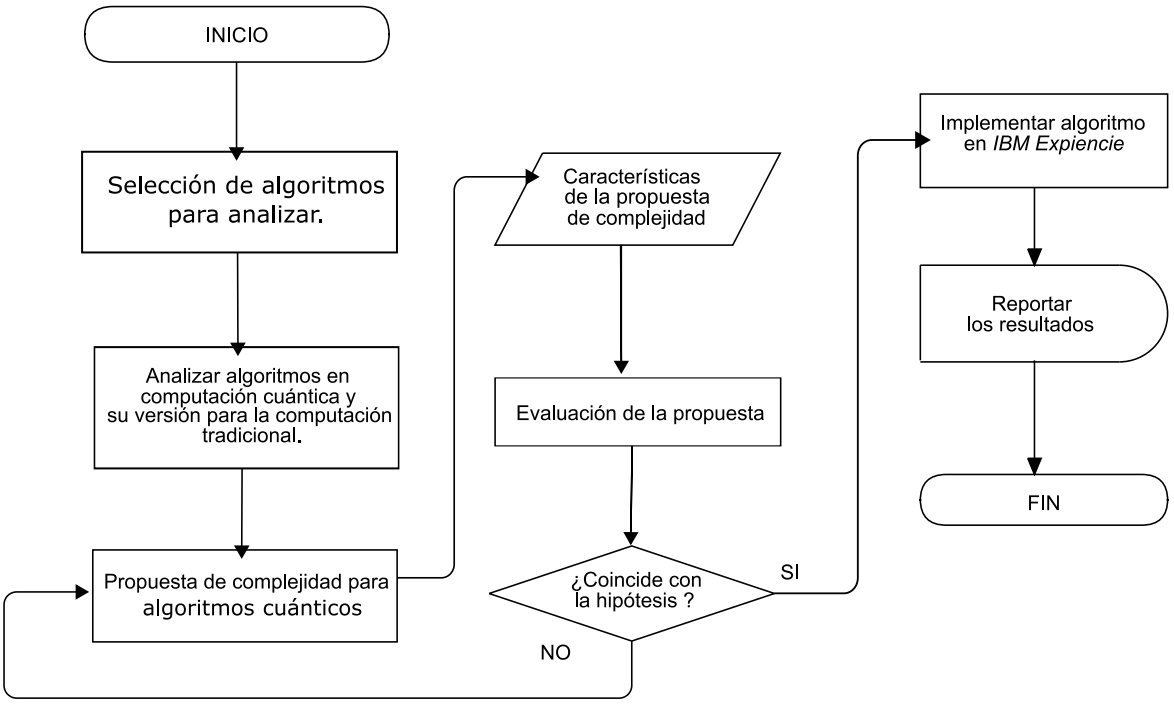

Figura 3.1: Metodología de trabajo. Fuente: elaboración propia

1 Selección de algoritmos para ser analizados: se seleccionaron el *algoritmo de Deutsch* y el *algoritmo de Grover*, de tal forma que los algoritmos puedan ser representados en ambos paradigmas (tradicional y cuántico).

2 Analizar algoritmos en computación cuántica y su versión para la computación tradicional: revisión del estado del arte sobre los algoritmos cuánticos seleccionados, para identificar formas de abordar la propuesta, así como los fundamentos teóricos necesarios para el desarrollo de la misma.

3 Propuesta de complejidad para algoritmos cuánticos: recabar información sobre las diferentes medidas de complejidad actuales, para ser utilizada en la propuesta.

4 Características de la propuesta de complejidad: determinar las características que permitan la aplicación de la propuesta.

5 Evaluación de la propuestas: utilizar la propuesta desarrollada para analizar los algoritmos

seleccionados.

6 Implementar algoritmo en el portal IBM (2019) *Quantum Experience*: se implementan los algoritmos analizados.

7 Reportar los resultados obtenidos: concluyendo los objetivos y corroborando la propuesta para la complejidad algorítmica cuántica, se publicarán los resultados obtenidos en eventos académicos.

3.1 Desarrollo de métrica

En esta sección se presentan de las bases utilizadas para la propuesta, partiendo exclusivamente de tres posturas; (1) en Bernstein and Vazirani (1997) que desarrollan una máquina de Turing cuántica (*QTM*), como una máquina de Turing reversible (*TM reversible*), unitaria y con un proceso llamado "*oráculo*"; (2) desde la parte computacional clásica, en el trabajo Papadimitriou (1993), define formalmente la complejidad para la computación tradicional, y por último (3) en Aharonov et al. (2021) ilustran una manera de abordar el proceso experimental físico como una especie de algoritmo cuántico, siendo esta perspectiva una mezcla entre lo desarrollado en física en teoría de la computación.

3.1.1 Clases de complejidad algorítmica

Retomando lo explicado en la sección 2.2.1 Complejidad algorítmica en el cómputo tradicional, para estar en posibilidad de crear una clase de complejidad algorítmica cuántica, Papadimitriou (1993) determina, al menos se deben tener tres características:

1. Elección del modelo de computación.
2. Modo de computación (tipos de entradas que acepta el modelo).

a) Determinista.

b) No determinista.

3. Revisar cual recurso se desea limitar (recurso que puede ser costoso en un evento, como lo es el *peor caso*).

Recordando los dos modos de computación mencionados son:

Definición 22 (modelo determinista (Aho and Hopcroft, 1974)).
Dada una secuencia de instrucciones siempre obtengo el mismo resultado esperado.

Definición 23 (modelo no determinista (Aho and Hopcroft, 1974)).
Dada una secuencia de instrucciones no siempre obtengo el mismo resultado esperado.

Entonces, para iniciar con la creación de una clase de complejidad algorítmica cuántica:

(i) Se selecciona el modelo de la máquina de Turing como modelo de computación cuántica, el cual utiliza *MT's deterministas reversibles* desarrollado por Bernstein and Vazirani (1997). Con este hecho, se cumple lo mencionado en el Capítulo 2 Antecedentes, acerca de analizar un modelo equivalente a una máquina de Turing.

(ii) El modo de computación será no determinista, dado que el resultado de la ejecución de un algorítmico cuántico, podrá variar. Hecho que se observará en el Capítulo 4.

(iii) Por último,determinar el recurso a limitar, para esta tesis serán el tiempo y el espacio. Recursos que también se limitan en el desarrollo de algoritmos para la computación tradicional. Esto porque en la realidad no se tienen recursos infinitos.

3.1.2 Definición y complejidad de una QUALM

La propuesta desarrollada por Aharonov et al. (2021) abarca la idea de contabilizar el número de operaciones cuánticas locales elementales (por ejemplo, compuertas o medidas) necesarias para implementar el experimento cuántico. De esta manera desarrolla una caracterización de la complejidad de los experimentos cuánticos, desde una perspectiva independiente de los sistemas y aparatos físicos. Este enfoque permite el estudio de experimentos cuánticos desde un punto de vista matemático abstracto, independiente de los detalles físicos precisos.

Al mencionar un experimento físico, se recuerda una especie de interacción entre la naturaleza y un aparato experimental, al cual se tiene acceso solo a un subconjunto de grados de libertad (cantidad limitada de observaciones) que caracterizan al sistema físico. Se inicia con esta idea, en donde existe un experimentador o experimentalista el cual maneja tres subsistemas o registros. El primer subsistema, se nombra "naturaleza" y se denota por **N**; será el registro que la naturaleza mantiene en secreto, y no se tiene acceso directo a él.

Este sistema tiene un espacio de $Hilbert$ asociado que se indica con la misma notación. **N** es el primer subsistema y está contenido en el sistema de laboratorio **L**. Además, se tiene acceso a un sistema denominado, espacio de trabajo **W**, el cual se aprovecha para realizar el procesamiento de los datos cuánticos. En total, el espacio de $Hilbert$ completo es $H = N \otimes L \otimes W$ correspondiente a la descomposición en subsistemas **N**, **L**, **W**. La idea básica es realizar mediciones para leer información de **L** y **W**, pero no directamente de **N**. A continuación, se describe a detalle los protocolos experimentales abstractos y las *mediciones algorítmicas cuánticas* o *QUALM* desarrolladas por Aharonov et al. (2021).

Definición 24 (QUALM, *quantum algorithmic measurement*).
*Un QUALM es el conjunto de compuertas admisibles $\mathcal{G}$ que actúa sobre los registros **L**, **W**, es una secuencia ordenada de símbolos $\mathcal{Q} = (\mathcal{Q}_1, \mathcal{Q}_2, \ldots, \mathcal{Q}_{final})$ del alfabeto $\mathcal{G} \cup \{b\}$, junto con una especificación de subsistemas de entrada y salida $S_{in}, S_{out} \subseteq$**W**. Aquí, se utiliza a $\mathcal{G}$ como un conjunto de símbolos (es decir, cada compuerta se etiqueta con un símbolo distinto),*

de la misma manera b es un símbolo. Cada QUALM tiene un mapa asociado

$$QUALM : \mathcal{LO}(N, L) \rightarrow QuantumCircuits(N \otimes L \otimes W) \tag{3.1}$$

Esta función toma un oráculo de laboratorio LO y genera un circuito cuántico en $\boldsymbol{N} \otimes \boldsymbol{L} \otimes \boldsymbol{W}$*. Específicamente, QUALM (LO) 'compila' un circuito cuántico* $\mathcal{Q} = (\mathcal{Q}_1, \mathcal{Q}_2, \ldots, \mathcal{Q}_{final})$ *donde se reemplaza cada símbolo en* $\mathcal{G}$ *por su compuerta correspondiente, y cada* b *es reemplazado por el superoperador* ε_{NL} *correspondiente al LO.* S_{in}, S_{out} *corresponden a los subsistemas de entrada y salida del circuito resultante, respectivamente.*

En términos menos formales Aharonov et al. (2021) explican, que un QUALM es un circuito cuántico construido a partir de un conjunto de compuertas admisibles, donde el circuito tiene puntos designados para que se inserte un superoperador[1] de oráculo de laboratorio, qubits de entrada y salida especificados. Se puede observar en la Figura 3.2 una representación en forma de esquema de lo que representa un QUALM.

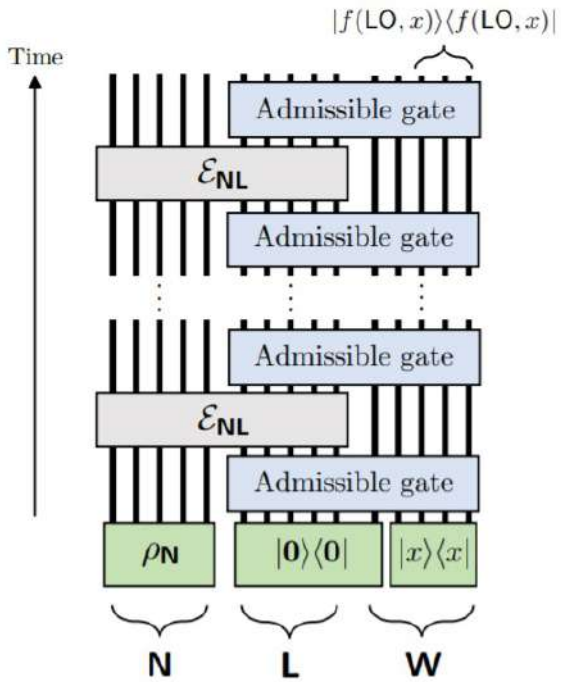

Figura 3.2: Esquema de una QUALM. Fuente: Tomado de (Aharonov et al., 2021)

[1]Transformación unitaria u operador unitario, visto en el Capítulo 3

En la Figura 3.2 la parte azul del diagrama representa la QUALM, los cuadros verdes el estado inicial $\rho_{\mathbb{N}} \otimes |0\rangle\langle 0| \otimes |x\rangle\langle x|$ donde x es una cadena de bits. Los rectángulos grises son el superoperador de oráculo de laboratorio $\varepsilon_{\mathbf{NL}}$ según la Definición 25. Un QUALM (Definición 24) consiste en una secuencia de aplicaciones de las puertas admisibles $\mathcal{G}_i \in \mathcal{G}$ o el superoperador de oráculo de laboratorio ε_{NL}.

La salida del QUALM está en un subsistema S_{out}, indicado por el corchete en la parte superior derecha del diagrama. Si este mapa produce un estado $|f(LO, x)\rangle$ que codifica una función f especificada en oráculos de laboratorio y cadenas de bits iniciales que son aplicados a una "tarea" (ver Definición 26), entonces se dice que una QUALM logra esa tarea.

Tanto los registros de entrada como de salida se pueden generalizar para que sean cuánticos. Desde un punto vista estrechamente relacionado, el circuito obtiene como entrada un estado cuántico y emite otro estado cuántico. Tal proceso genera un canal (o función) cuántico que depende del oráculo cuántico que se utiliza como caja negra. Esto es similar a una QTM que implementa una transformación unitaria desarrollada por Bernstein and Vazirani (1997).

El acceso al oráculo por medio de una función unitaria U puede usarse para implementar varios canales que son funciones de U; por ejemplo, aplicando la compuerta $U - controlled$, U^*, U^T y U^{-1}, o elevando U a alguna potencia fraccionaria, y más. El oráculo de laboratorio modela tanto la memoria del sistema físico, almacenada en **N**, como el superoperador (generalmente desconocido) que aplica el sistema físico.

Definición 25 (oráculo de laboratorio).
*Un oráculo de laboratorio se especifica mediante un par $LO = (\varepsilon_{NL}, \rho_N)$ donde ε_{NL} es un superoperador cuántico (es decir, un mapa positivo de conservación de trazas) sobre $\boldsymbol{N} \otimes \boldsymbol{L}$ y ρ_N es un estado en **N**. El conjunto de oráculos de laboratorio se denota por $LO(\boldsymbol{N},\boldsymbol{L})$.*

Ahora, se define la noción de tarea que corresponde al "problema experimental". El experi-

mentador debe lograr la tarea solo mediante el uso del superoperador de oráculo de laboratorio, junto con las operaciones a su disposición del laboratorio (es decir, las compuertas admisibles en $\mathbf{L} \otimes \mathbf{W}$).

Definición 26 (tarea).
Una "tarea" es una tupla $Task = (S_{in}, S_{out}, f, \mathcal{G})$, asociada a un sistema dado $\mathbf{N} \otimes \mathbf{L} \otimes \mathbf{W}$ (que generalmente es implícito). Aquí, S_{in} es un subsistema de $p-$qubit de $\mathbf{W}$, S_{out} es un subsistema de $q-$qubit de $\mathbf{W}$ y sea f una función

$$f : \{LO_0, LO_1, LO_2, \dots\} \times \{0,1\}^p \to \{0,1\}^q,$$

$\mathcal{G}$ es un conjunto de puertas admisibles sobre $\mathbf{L} \otimes \mathbf{W}$. En el dominio de f, $\{LO_0, LO_1, LO_2, \dots\}$ es un conjunto de oráculos de laboratorio (se denota este conjunto como discreto, pero por supuesto también se puede considerar un conjunto continuo de oráculos de laboratorio como entrada), es decir, un subconjunto de $LO(\mathbf{N},\mathbf{L})$

Observe que además del oráculo de laboratorio, al cual la QUALM solo tiene acceso como una caja negra y que se puede considerar como una especie de "entrada" cuántica para una QUALM, también se le da una cadena de bits adicional que es su entrada clásica.

Cabe destacar que existen tareas con un resultado probabilista, sin embargo para efectos de este análisis no causa ningún cambio. Además se da por hecho que las tareas podrán contener estados cuánticos de entrada y salida.

Para definir la noción de un QUALM que logra (implementa) una tarea, se considera que una *tarea* específica es una función $f : \{LO_0, LO_1, LO_2, \dots\} \times \{0,1\}^p \to \{0,1\}^q$.

Definición 27 (QUALM implementando una tarea).
Sea un QUALM sobre puertas admisibles $\mathcal{G}$ con subsistemas de entrada y salida especificados como S_{in} y S_{out} respectivamente. Éste implementa una tarea $= (S_{in}, S_{out}, f, \mathcal{G})$, si

$$\rho_{S_{out}}(QUALM(LO_i, x)) = |f(LO_i, x)\rangle\langle f(LO_i, x)| \tag{3.2}$$

para todo LO_i, x en el dominio de f (es decir, para todo $LO_i \in \{LO_0, LO_1, LO_2, \dots\}$ y todo

$x \in \{0,1\}^p$). *Se dice que QUALM implementa la tarea con un error de a lo más* ϵ *si para cada entrada* LO_i, x*, se tiene*

$$\| \rho_{S_{out}}(QUALM(LO_i, x)) - |f(LO_i, x)\rangle\langle f(LO_i, x)| \|_1 \leq \epsilon. \tag{3.3}$$

Vale la pena revisar la configuración de los algoritmos cuánticos ordinarios, en los que $f : \{0,1\}^p \to \{0,1\}^q$. Note que, este es un caso particular de la definición 26 al permitir que el conjunto de oráculos de laboratorio en el dominio sea el conjunto vacío. Si el superoperador QUALM es solo un circuito unitario U, entonces la ecuación 3.2 se convierte en

$$tr_{S_{out}} \left\{ U(|x\rangle\langle x|_{S_{in}} \otimes |0\rangle\langle 0|)U^\dagger \right\} = |f(x)\rangle\langle f(x)| \tag{3.4}$$

la cual si se ingresa x en S_{in}, entonces genera $f(x)$ en S_{out}, es decir, una función tradicional.

Finalmente, para las QUALM que generan distribuciones de probabilidad, se puede hablar de una QUALM que logra aproximadamente ciertas tareas. Una métrica de tareas se define de forma natural al considerar la distancia de variación total máxima entre las distribuciones que las dos funciones generan para una entrada dada, maximizada sobre los oráculos de laboratorio y las entradas clásicas en el dominio de la función. Además, se define una manera de complejidad sobre una QUALM, que se revisará a continuación.

Complejidad QUALM

La complejidad de una QUALM está definida como, el número de llamadas al oráculo de laboratorio, más el número de puertas que se aplica en **L**, **W**. Esta es una combinación de complejidad de consulta y complejidad de compuerta. Habiendo definido las tareas y los QUALM, ahora se está en posibilidad de definir la complejidad de QUALM.

Definición 28 (complejidad de compuerta, consulta y del QUALM).
La complejidad de la compuerta de un QUALM dado sobre el conjunto admisible de compuertas $\mathcal{G}$ *es la longitud (es decir, el número de símbolos de* $G \cup \{b\}$*) de la secuencia* $\mathcal{Q}$*,*

menos el número de b símbolos. Denotado por GateComplexity[QUALM], y se nombra como complejidad de la puerta QUALM. De manera similar, la complejidad de la consulta es el número de b's que aparecen en $\mathcal{Q}$, y esto se denota mediante QueryComplexity [QUALM] complejidad de consulta QUALM. La suma

$$GateComplexity[QUALM] + QueryComplexity[QUALM] = |\mathcal{Q}| \tag{3.5}$$

será la complejidad QUALM.

Como es habitual en la complejidad computacional, se está interesado en familias de tareas y QUALM's, donde algún parámetro que dicta el tamaño del problema crece hasta el infinito, y se observa el crecimiento de la complejidad en función de ese parámetro, para esto la siguiente definición.

Observación 1 (complejidad aproximada o asintótica)**.** *La complejidad aproximada de la puerta QUALM, la complejidad aproximada de la consulta QUALM y la complejidad aproximada QUALM de una tarea, respectivamente, se definen tomando el mínimo de todos los QUALM que logran la tarea con la aproximación deseada.*

3.2 Propuesta de modelo

Tomando las perspectivas descritas anteriormente, es posible notar que tienen características comunes. Además retomando la clasificación de complejidad desarrollada con las características mínimas requeridas por Papadimitriou, se esta en posibilidad de generar una propuesta de complejidad algorítmica cuántica. Seguidamente se describe el modelo de máquina cuántica de Turing desarrollado en Bernstein and Vazirani (1997), contraponiendo el modelo de TM.

Una TM M_T es una $7-$tupla $(Q, \Sigma, \Gamma, \delta, q_0, \#, F)$	De acuerdo con Bernstein and Vazirani (1997) una QTM M_Q consta de un triplete (Σ, Q, δ)
Q Conjunto de estados $\|Q\| < \infty$ Σ Alfabeto de entrada $\Sigma \subseteq \Gamma$ Γ Alfabeto de cinta $\|\Gamma\| < \infty$ δ Función de transición $\delta : Q \times \Gamma \longrightarrow Q \times \Gamma \times \{L, R\}$ q_0 Estado inicial $q_0 \in Q$ $\#$ Símbolo en blanco $\# \in \Gamma \backslash \Sigma$ F Estados de aceptación	Σ Alfabeto finito con un símbolo en blanco identificado $\#$ Q Conjunto finito de estados con un estado inicial identificado por q_0 y un estado final $q_f \neq q_0$ δ Función de transición cuántica, tal que $\delta : Qx\Sigma \longrightarrow \tilde{C}^{\Sigma xQx\{L,R\}}$. Donde $\tilde{C}$ es un conjunto que consta de $\alpha \in \mathbb{C}$, tal que existe un algoritmo determinista que calcula la parte real e imaginaria de α dentro de 2^{-n} en el polinomio de tiempo en n.

Esta QTM M_Q tiene una cinta infinita de dos vías, con celdas indexadas por $\mathbb{Z}$ y un cabezal de lectura/escritura que se mueve a lo largo de la cinta. De forma más especifica, la función de transición representa las distintas configuraciones que tendrá la M_Q, empleando un operador de evolución temporal U_{M_Q}. De manera tal que: si M_Q comienza en la configuración c con el estado actual p y símbolo leído σ, después de un paso i, M_Q estará en superposición de configuraciones $\psi = \sum_i \alpha_i c_i$, donde cada α_i distinto de cero corresponde a una transición $\delta(p, \sigma, \tau, q, d)$ y c_i es la nueva configuración que resulta de aplicar esta transición a c.

Más formalmente el operador de evolución temporal U_{M_Q} es $U_{M_Q} : S \longrightarrow S$. Donde S es un espacio de combinaciones lineales finitas de complejos con norma euclidiana y producto interno, que representan las configuraciones de M_Q, en el cual $\psi \in S$ forma una superposición de M_Q. Cabe destacar, que S contiene una base ortononormal, por lo tanto cada

superposición $\psi \in S$ puede representarse como un vector de números complejos indexados por las configuraciones $c_0, \ldots, c_i$. El operador de evolución en el tiempo U_{M_Q} puede ser representado por una matriz cuadrada con columnas y filas indexadas por configuraciones $c_0, \ldots, c_i$, donde el elemento de la matriz en la columna c_0 y la fila c_1 da la amplitud con la cual, la configuración c_0 conduce a la configuración c_1 en un solo paso de M_Q.

Puesto que la selección sobre los recursos a limitar en el modelo cuántico es el *tiempo* y el *espacio*, se desarrolla una métrica particular para cada recurso. En particular un *qubit* es la unidad básica de información del modelo de computación cuántica y contiene mayor información que permite agilizar ciertos procesos, por lo cual será la unidad base. Por otra parte el tiempo, se nombrara t_q, tiempo cuántico y será la unidad de medida. Además se establece la definición de *oráculo* y *algoritmo cuántico*, utilizando los conceptos vistos en los capítulos anteriores.

Tanto para la métrica en el *espacio* y *tiempo*, se toman en cuenta el oráculo, las compuertas cuánticas y la cantidad de compuertas de medida utilizadas, como consecuencia del modelo no-determinista, en este trabajo no se toma en cuanta la probabilidad de éxito, es decir, al menos para estos dos desarrollos de algoritmos cuánticos se da por hecho que tienen éxito. El estudio de esta métrica para casos distintos, se deja a futuro.

Asimismo, para comparar el tiempo, es posible obtener directamente el rendimiento de este en la aplicación de python[2]. Por lo cual, se compara la ejecución de una simulación contra una computadora cuántica real. Por otra parte, revisar el espacio de almacenamiento que conlleva usar una compuerta cuántica con acceso a un oráculo, deberá ser un trabajo posterior, ya que es relevante para el desarrollo de futuros algoritmos cuánticos.

Existen varios conceptos de ambas perspectivas que son útiles para esta tesis. Ya que, de manera natural es posible trasladar algunas ideas a los experimentos realizados en el Capítulo 4

[2]Lenguaje de programación en donde se maneja *Qiskit*

Resultados, es decir por medio de las definiciones 28, 24 y el modelo de (Bernstein and Vazirani, 1997), aplicadas al análisis de los algoritmos de Deutsch 2.3.1 y Grover 2.3.2, se podrá examinar la propuesta aquí mostrada.

Capítulo 4

Resultados y discusión

En esta sección se muestra la propuesta para una complejidad algorítmica cuántica espacio temporal resultante, así como también su aplicación sobre los dos algoritmos mencionados en el Capítulo 3, implementados en el portal IBM (2019). Además, se agrega una breve discusión sobre algunas otras posibles formas de abordar esta temática.

Como se explicó anteriormente esta propuesta permite la evaluación de un algoritmo cuántico sin la necesidad previa de codificarlo sobre algún lenguaje de programación. Siendo útil, al menos, en dos situaciones (i) dado que las computadoras cuánticas no están completamente disponibles, es decir, no es que ahora mismo se pueda tener todo el poder computacional cuántico en casa, actualmente una opción es *IBM Quantum Experience*, sin embargo (ii) tampoco es posible trasladar muchos de los algoritmos cuánticos desarrollados para casos grandes.

Por lo cual, debe existir una opción que permita estudiar sus posibilidades y limitaciones, de esta forma tener expectativas reales sobre el problema a resolver, antes de realizar la implementación. Entonces, para estar en posibilidad de generar una *propuesta de complejidad algorítmica cuántica*, se deben precisar algunas definiciones.

Retomando la definición 8 (algoritmo) descrita en el Capítulo 2, a continuación se reescribe desde una perspectiva útil para un algoritmo cuántico:

Definición 29 (algoritmo cuántico).
Proceso, método, técnica o rutina, que contiene un conjunto de reglas secuenciales, determinadas por el modo de computación cuántica (qubits, compuertas cuánticas, oráculos y compuertas de medida, revisados en la sección 2.1.1), para resolver un grupo de problemas, el cual está compuesto por 5 características:

1. *Finito. El algoritmo debe terminar en un numero finito de pasos.*
2. *Definitud (Preciso). Cada acción que se realizar en cada paso, deberá ser especificada de manera precisa. Es decir, compuertas cuánticas que serán aplicadas.*
3. *Entradas. Puede tener cero o más datos de entrada; la cantidad será asignada antes de iniciar el algoritmo, o de forma dinámica cuando el algoritmo este en ejecución. Estas entradas serán tomadas del modelo computacional cuántico definido en el Capítulo 2.*
4. *Salidas. Tiene una o más salidas; la cantidad estará relacionada conforme a las entradas y el tipo de problema a resolver. Esto es, el empleo de al menos, una compuerta de medida vista en el Capítulo 2.*
5. *Eficacia. Las operaciones que se realizan deben ser elementales para el campo de la física cuántica, es decir sobre espacios de $Hilbert$, utilizando operadores lineales de evolución temporal que pueden ser reproducibles (casos pequeños), por ejemplo, con una computadora tradicional, en un tiempo finito.*

En el Capítulo 2, se señaló el *qubit* (definición 4) como la unidad básica de información en computación cuántica. De tal modo, que la cantidad de qubits utilizados en un algoritmo cuántico, representará el espacio requerido para implementarlo. Además, cuando se hable de tiempo, t_q simbolizará la unidad de tiempo cuántico. También, se hará referencia a un algoritmo cuántico como circuito cuántico y viceversa.

En este trabajo el QUALM es conveniente debido a su perspectiva experimental, en particular se utilizará como una representación de un circuito cuántico bien definido, ideas desarrolladas

tanto en Bernstein and Vazirani (1997) y Aharonov et al. (2021), con compuertas admisibles y tareas por implementar. Marco de referencia útil para utilizar la complejidad desarrollada sobre un QUALM de manera similar hacia un circuito cuántico, definido en el Capítulo 2.

Es importante mencionar, que una QUALM también se desarrolla una complejidad sobre el tipo de acceso y transmisión de información entre ellas, esto cuando se desea trabajar con una red de circuitos cuánticos, en el cual define tres formas: *local-local*, *registro único* y *acceso incoherente* (Aharonov et al., 2021).

Sin embargo, por ahora este trabajo se enfoca, unicamente en la construcción y desarrollo de un algoritmo cuántico, desde el planteamiento del problema real hasta su implementación, con previa evaluación de complejidad algorítmica cuántica asociada, hasta la ejecución del mismo.

4.1 Métrica para la complejidad algorítmica cuántica

Si bien se mostró en capítulos anteriores que existen distintas maneras de clasificar un algoritmo, corroborar que la representación de complejidad, aquí desarrollada es una aproximación útil, es fundamental para su aplicación. Por tal motivo, en seguida se dan las definiciones que caracterizan esta representación de complejidad.

Tomando algunos conceptos del Capítulo 2 como *pensamiento al límite*, *eficiencia*, *pensamiento contrario*, entre otros; se desarrolla la siguiente definición.

Definición 30 (complejidad en algoritmos cuánticos).
La complejidad algorítmica cuántica es una medida de los recursos usados por el algoritmo (también se hace referencia al costo del algoritmo), usualmente medida como una función sobre el tamaño de entrada del algoritmo. La complejidad c_{max} sobre el tamaño de entrada n es tomada como el costo del algoritmo sobre el peor caso para un problema de tamaño n. Por otra parte, la complejidad algorítmica es mínima c_{min}, cuando referente a un problema,

existe una cantidad de recursos mínimos requeridos por cualquier algoritmo desarrollado para resolver el problema.

*Esta definición está relacionada con la **notación asintótica**, explicada en el Capítulo 2 definición 9.*

Notando el hecho de, que tanto en el modelo desarrollado por Bernstein and Vazirani (1997) como el modelo de QUALM desarrollado por Aharonov et al. (2021), manejan operadores lineales como representación del cambio de estados en las entradas de un algoritmo cuántico. Consecuentemente trabajan con espacios de Hilbert, de modo que manejan las mismas propiedades matemáticas. Por lo tanto, es posible trasladar las complejidades desarrolladas en Bernstein and Vazirani (1997) y Aharonov et al. (2021) hacia la complejidad de un algoritmo cuántico, con algunas adecuaciones, así pues para dar claridad al texto se retoman algunas definiciones descritas en el Capítulo 2 como:

TM Oráculo, definición 21 Una TM de oráculo es una TM T normal con una cinta de consulta de oráculo adicional, un estado especial $q?$, y dos estados distinguidos etiquetados q_y y q_n. Sea A cualquier lenguaje bajo el alfabeto Σ. Siempre que T entré en el estado $q?$ con alguna cadena $z \in \Sigma^*$ en la cinta de consulta, el control pasa al estado q_y si $z \in A$, o al estado q_n si $z \notin A$. El cálculo continúa normalmente hasta la siguiente vez que la máquina ingresa $q?$. La máquina T con una elección dada por el oráculo A se denota por T^A.

Oráculo de laboratorio, definición 25 Un oráculo de laboratorio se especifica mediante un par $LO = (\varepsilon_{NL}, \rho_N)$ donde ε_{NL} es un superoperador cuántico sobre $\mathbf{N} \otimes \mathbf{L}$ y ρ_N es un estado en $\mathbf{N}$.

Nótese que, una característica que permanece en estas dos definiciones, son la cantidad inicial de entradas, ya sea con dos estados distinguidos como q_y y q_n o un par de valores de entrada

como $LO = (\varepsilon_{NL}, \rho_N)$, donde ε_{NL} es un superoperador cuántico y ρ_N es un estado en **N**, es decir un vector. Observe que Aharonov et al. (2021), introduce el superoperador cuántico, el cual se usará como operador lineal, de esta forma se afirma que el oráculo trabaja como una transformación unitaria. De modo que, usando el modelo de compuertas cuánticas explicado en el Capítulo 2, la definición de un oráculo será:

Definición 31 (oráculo para computación cuántica).
Operador lineal de evolución temporal (transformación unitaria) U_f, con al menos dos elementos (qubits) de entrada $q_i \in \{0,1\}^n$ y $q_{aux} \in \{0,1\}$, el cual utiliza el paralelismo cuántico (Definición 1) e interferencia cuántica (Definición 2), que evalúa una función $f(x)$ sobre distintos valores de x y accede al resultado de forma simultánea.

Cabe mencionar, que diversas lecturas hablan de no poder observar de forma directa el funcionamiento del oráculo, es decir solo es posible acceder al resultado. Entonces, para abordar la complejidad en tiempo y espacio sobre un oráculo, el interés se enfoca sobre la cantidad de accesos al mismo.

En particular, al plantear *la complejidad espacio temporal sobre un algoritmo cuántico*, se toma la cantidad de *qubits totales* para implementar *el* algoritmo cuántico, agregando los accesos al oráculo. Así en la definición 32 se propone, además contabilizar el uso de compuertas cuánticas, como una forma de medir la complejidad del espacio sobre un algoritmo cuántico, debido a que representan transformaciones lineales unitarias necesarias para ejecutar una tarea.

Entonces, sea una compuerta cuántica C_q, la cual tiene n cables de entrada y m cables de salida, dicha operación cuántica mapea n qubits a m estados. Para determinar el espacio completo de la operación se deberá contar, no solo la cantidad de qubits de entrada, además se debe agregar el espacio requerido para realizar la operación, como se menciono anteriormente son transformaciones conformadas por una cantidad de qubits internos necesarios para su funcionamiento. Por lo tanto, sumar la cantidad de qubits de entrada para la aplicación de

la compuerta, más la cantidad de qubits de trabajo de la misma, representará la cantidad de espacio requerido para una operación (compuerta) cuántica.

Por ejemplo en la compuerta X, se puede observar en su representación *bra−ket*, $|0\rangle\langle 1|+|1\rangle\langle 0|$ mostrada en la Tabla 2.2 que utiliza 4 qubits. Por lo que, la complejidad de espacio para la compuerta cuántica corresponde a la suma de los qubits de entrada (qubits a los que se aplicará dicha compuerta cuántica) más 4_q. Siendo así, al aplicar dicha compuerta sobre un qubit $|0\rangle$, la complejidad de espacio para la compuerta cuántica será igual a 5_q.

Esto se realizará de similar forma para un cambio de fase. Para la complejidad algorítmica cuántica del espacio ESP_q, además se agregan los accesos o entradas al oráculo $ORAC_q$, más una unidad c_m de compuertas de medida MED. Se implementará este análisis para el algoritmo de Deutsch y Grover más adelante.

Definición 32 (complejidad algorítmica cuántica del espacio ESP_q).
Sean los siguientes elementos necesarios para la construcción de un algoritmo cuántico.

$ENTR_q$ *Cantidad de qubits $|0\rangle$ ó $|1\rangle$ de entrada necesarios para el funcionamiento del algoritmo cuántico. En la representación en forma de diagrama (Figura 2.2) cada cable inicial simboliza un qubit por usar.*

$COMP_q$ *Cantidad de qubits necesarios para el funcionamiento de cada compuerta cuántica aplicada, denotadas como $H, X, Z, \ldots, etc$, explicadas en el Capítulo 2.*

$ORAC_q$ *Accesos al oráculo, cada cable de salida indicará un acceso al resultado del oráculo $ORAC$, el total de estos representará un valor acumulado.*

MED_q *Cantidad de compuertas medida o medición aplicadas (MED) en el algoritmo cuántico, se representa por medio de una constante c_m. Por ejemplo, para indicar que se aplican dos compuertas de medida, será $2c_m$*

La suma de estos elementos representa el espacio requerido para un algoritmo cuántico dado.

$$ESP_q = ENTR_q + COMP_q + ORAC_q + MED_q c_m \tag{4.1}$$

Para el desarrollo de la *complejidad de tiempo algorítmico cuántico* se asigna una unidad t_q (unidad de tiempo sobre el cómputo cuántico), por cada acción realizada en el algoritmo cuántico. Es necesario señalar, que tanto para el desarrollo de una QTM de Bernstein and Vazirani (1997), como para una QUALM (Aharonov et al., 2021), no definen una forma de evaluar la complejidad algorítmica cuántica sobre tiempo.

En el caso de la QTM, mencionan que una transición δ de una QTM equivale a la evolución en un solo paso de tiempo T; para algunas configuraciones menciona que $T(n)$ es polinomial, el cual también podrá estar determinado por la longitud de la cadena de entrada n, definiciones *BQPTime* 18 y *EQPTime* 20 descritas en el Capítulo 2. En cambio, para una QUALM menciona que se puede apreciar el crecimiento de la complejidad en función de algún parámetro, por ejemplo el tiempo, por medio de la *observación 1*.

Sin embargo, especificar una forma precisa de obtener los límites y alcances de un algoritmo cuántico, permite definir los requerimientos necesarios y de esta manera, tener la posibilidad de construir u observar las necesidades que deben ser modificadas para el desarrollo del algoritmo. En particular, hacer una diferencia entre estas características como lo son, el tiempo y el espacio proporciona claridad sobre el elemento a mejorar. Para ello la siguiente definición.

Definición 33 (complejidad algorítmica cuántica del tiempo $TIME_{t_q}$).
Los elementos considerados para esta propuesta de complejidad temporal sobre un algoritmo cuántico son:

$\boldsymbol{COMP_{t_q}}$ *Unidad de tiempo t_q por cada compuerta aplicada, denotadas como $H, X, Z, \ldots,$ etc. En el diagrama (Figura 2.2) es posible observar un recuadro con la letra de correspondiente a la compuerta aplicada.*

$\boldsymbol{ORAC_{t_q}}$ *Unidad de tiempo por accesos al oráculo, es decir cada cable de entrada representará una unidad t_q. 5_{t_q} indicará cinco unidades de tiempo, que representan cinco accesos al oráculo de forma simultánea.*

MED_{t_q} *Unidad de tiempo t_q por compuertas de medida o medición aplicadas (definición 5).*

La unidad de tiempo t_q representa una unidad de tiempo abstracta, a la cual se le pueden asociar características particulares, por ejemplo para la computación cuántica, podrá representar un tiempo cuántico de ejecución por instrucción[1].

$$TIME_{t_q} = COMP_{t_q} + ORAC_{t_q} + MED_{t_q} \tag{4.2}$$

Observe que MED_{t_q} contiene las mismas unidades, ya que para efectos del análisis servirá contabilizar el total de las unidades tiempo necesarias para la ejecución de un algoritmo cuántico, incluyendo la compuerta de medida, puesto que en la definición 29, especificada arriba, debe aplicar al menos una compuerta de medida, por lo tanto mínimo tendrá una t_q.

Otra manera de abordar la complejidad, es por medio de la clase de problemas que puede resolver una QTM. De esta forma surge una clasificación de los lenguajes que son aceptados por esta máquina de Turing cuántica como lo es *BQP* 17 y *EQP* 19. Es importante mencionar, que si se desea abordar esta perspectiva con otro modelo distinto a una máquina de Turing, se deberá construir un modelo computacional independiente que represente las peculiaridades de la mecánica cuántica.

Una manera, pudiera ser la abstracción de los problemas de física transformados a problemas de decisión u observar desde un espacio físico experimental cual podría ser el lenguaje de la física cuántica con un enfoque computacional. Sin embargo, la aproximación que existe hasta hoy con una QTM es sumamente conveniente para el crecimiento y consolidación del área.

Lo aquí mostrado, más bien analiza los requerimientos reales necesarios para la ejecución de un algoritmo cuántico sobre las herramientas hoy desarrolladas, como una primera aproximación de una creación de complejidad experimental espacio temporal en algoritmos cuánticos, donde las particularidades de un experimento físico real, no representan ninguna alteración en el desarrollo del algoritmo cuántico.

[1]Si existiese una característica especial o diferente de tiempo puede ser representada con t_q

Asimismo, las características particulares de un lenguaje de programación, tampoco refieren ninguna modificación. Se espera que lo aquí expuesto permita desarrollos de algoritmos cuánticos más precisos y óptimos. Aún existen preguntas abiertas, que mejoraran la comprensión de los alcances y limitaciones de los algoritmos cuánticos.

4.2 Implementación algoritmo de Deutsch

El objetivo de este algoritmo es determinar si la función es constante o balanceada, utilizando un oráculo de la forma $|x\rangle|0\rangle \rightarrow |x\rangle|f(x)\rangle$. Por lo tanto, la implementación de este oráculo sobre IBM (2019) produce el siguiente código 4.1.

Código 4.1: Función óraculo. Representación de la función 2.5, recibe el tipo de caso a evaluar. Fuente: elaboración propia

```
def ora_Deustch(caso):

    circuitCu=QuantumCircuit(2)#,1)

    if caso=='balanceada':
        circuitCu.cx(0,1)
    if caso=='constante':
        output = np.random.randint(2)
        if output==1:
            circuitCu.x(1)

    comp_oraculo = circuitCu.to_gate()
    comp_oraculo.name="Óraculo" #Imprime la caja con óraculo

    return comp_oraculo
    #return circuitCu
```

Posteriormente se agrega la *función óraculo*, código 4.1, a la función principal, código 4.2.

Código 4.2: Función principal de Deutsch. Integración de oráculo al código principal. Fuente: elaboración propia

```
def fun_Deust(caso='aleatoria'):
    """Determina si una función es constante o balanceada.

    Parametros:
        case(str): colocar si la función es balanceada o
           constante

    Salida:
        CircuitoCuántico: Algoritmo cuántico de Deutsch.
        str: Qubit 0 función constante. Qubit 1 función
           balanceada.
        """
    cq=QuantumCircuit(2,1)
    # invertir el segundo qubit para el qubit auxiliar y
       crear
    # el estado de entrelazamiento negativo
    cq.x(1)

    cq.h(range(2))
    #barra de separación
    cq.barrier()

    if caso=='aleatoria':

        aleat = np.random.randint(2)

        if aleat == 0:
            caso ='constante'
        else:
            caso = 'balanceada'
    #Función óraculo
    oraculo = ora_Deustch(caso)#.to_gate()
    cq.append(oraculo,range(2))

```

```
32      #barra de separación
33      cq.barrier()
34      #Aplicación de compuerta hadamard regresar el valor 0 o 1
35      cq.h(0)
36
37      #Compuerta de medida
38      cq.measure(0,0)
39      #Dibujar circuito cuántico
40
41      cq.draw('mpl')
```

El circuito cuántico se muestra en la siguiente Figura 4.1.

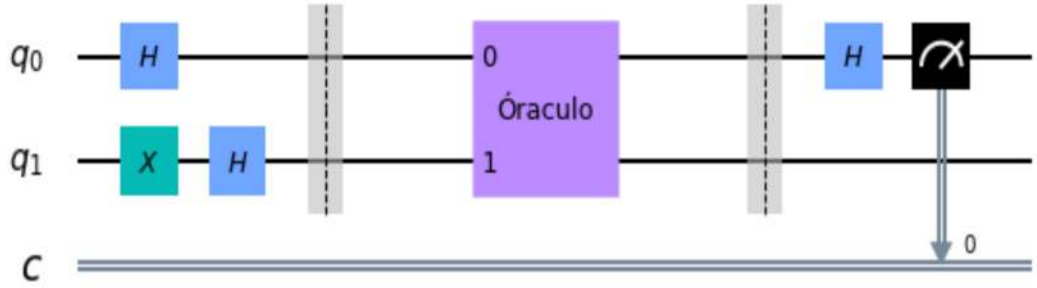

Figura 4.1: Resultado de la ejecución del código 4.2. Fuente: elaboración propia

Para la ejecución sobre un simulador de computadora cuántica se utiliza el siguiente código 4.3

Código 4.3: Ejecución de algoritmo de Deutsch sobre simulador. Se realizan 1024 disparos (shots= 1024) para la comparación con una computadora cuántica real. Fuente: elaboración propia

```
1 d_algCuant.draw('mpl')
2
3 # ejecutar el programa sobre un simulador
4 backend = Aer.get_backend('qasm_simulator')
```

```
#Ya que la salida será determinista, basta una sola ejecución
    (shot) para conseguir el resultado
d_alg_cuant = fun_Deust('balanceada')
inicio = time.time()
trab = execute(d_alg_cuant, backend, shots=1024).result()
fin = time.time()
print(fin-inicio)
salida = trab.get_counts()

plot_histogram(salida)
```

El resultado de ejecutar el algoritmo cuántico de Deutsch sobre un simulador de computadora cuántica, se muestra a continuación en la Figura 4.2. En donde se aprecia un solo resultado, pues al colocar la prueba del código 4.3 con una función *balanceada*. Se obtiene "casi" un algoritmo determinista.

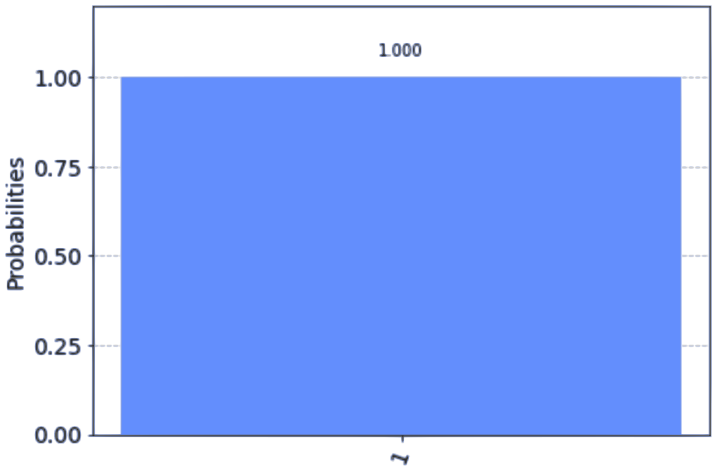

Figura 4.2: Ejecución de algoritmo cuántico sobre un simulador en IBM (2019) *Quantum Experience*. Se realizan 1024 repeticiones, sin mostrar cambios. Fuente: elaboración propia

Por otra parte, en la ejecución sobre una computadora cuántica real, se utilizó el siguiente código 4.4

Código 4.4: Ejecución de algoritmo de Deutsch sobre computadora cuántica $ibmqx2$. Se realizan 1024 disparos (shots= 1024) para la comparación. Fuente: elaboración propia

```
# busco el circuito disponible con menos trabajos
qcomp = least_busy( provider.backends( filters=lambda  x: x.
    configuration().n_qubits == 5 and not x.configuration().
    simulator and x.status().operational==True))
#ibmq_armonk
 #colocamos nombre de la computadora cuantica
#qcomp = provider.get_backend('ibmq_armonk')
#Obtengo el nombre
print("Backend menos ocupado: ",qcomp)
qcomp = provider.get_backend('ibmqx2')
get_ipython().run_line_magic('qiskit_job_watcher', '')

shots=1024
inicio = time.time()
job=execute(d_alg_cuant, backend=qcomp,shots=shots,
    optimization_level=3)
fin = time.time()
print('En computadora cuántica,',fin-inicio)


resultados = job.result()
respuesta = resultados.get_counts()
plot_histogram(respuesta)
```

El resultado de ejecutar el algoritmo cuántico de Deutsch sobre una computadora cuántica con dos *qubits*, se muestra a continuación en la Figura 4.3. En la cual se observa otro resultado, el 0, que representa una función *constante*, con probabilidad baja del $0{,}185\,\%$. Consecuencia de la interferencia cuántica (Definición 2), la cual genera errores en la medición final. Sin embargo, más repeticiones implican menos error, es decir es inversamente proporcional.

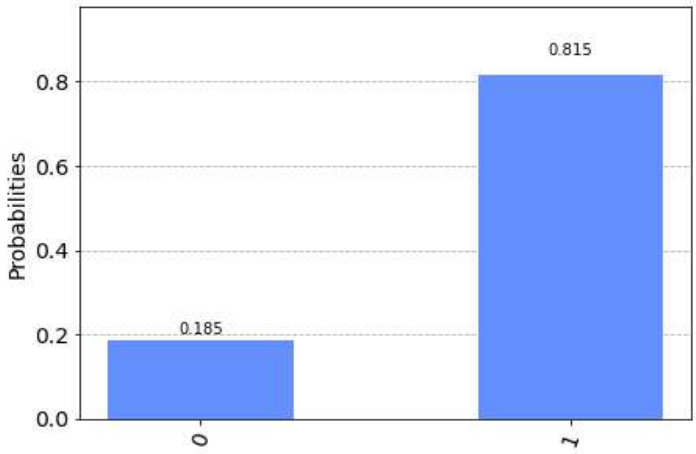

Figura 4.3: Ejecución algoritmo de Deutsch en IBM (2019) *Quantum Experience* con 1024 repeticiones. El 0 representa función *balanceada* y 1 función *constante*. Fuente: elaboración propia

Obteniendo el tiempo de ejecución del algoritmo cuántico de Deutsch, tanto para la computadora cuántica simulada como para la ejecución sobre una máquina cuántica real. Es posible observar y comparar los tiempos de ejecución. En la Tabla 4.1 se muestra la comparación de tiempos.

Aunque la diferencia no es visible sobre la marcha, por medio del dato numérico es posible observar una diferencia en milésimas de segundos, entre el simulador y el computador cuántico real. Con la complejidad temporal $TIME_{t_q}$ explicada anteriormente es posible determinar el tiempo aproximado que tomará si se desea repetir este experimento, el resultado se logrará siempre en un tiempo *constante*.

Tabla 4.1: Tiempos de ejecución con 1024 repeticiones sobre un simulador de computador cuántico y una computadora cuántica real de 3 qubits. Fuente: elaboración propia

Ejecución en simulador (s)	**Ejecución en computadora cuántica** (s)
`0.1311659812927246`	`0.004987239837646484`

Asimismo se puede corroborar que hay una ventaja (mínima) de tiempo, al menos para este algoritmo, sobre una computadora cuántica. Característica, por la cual adquiere su importancia. Ahora corresponde al análisis y aplicación de la propuesta de métrica de complejidad algorítmica descrita en 4.1 sobre el algoritmo de Deutsch.

4.2.1 Evaluación de métrica en algoritmo de Deutsch

Para estar en posibilidad de emplear la métrica, primero se debe conocer la complejidad espacial de las compuertas utilizadas. En particular, este desarrollo usa la compuerta X analizada anteriormente y la compuerta $Hadamard$, esta contiene los estados de entrelazamiento $|-\rangle$ (2.3) y $|+\rangle$ (2.2), los cuales están conformados por $2q$ cada uno. Por lo tanto, la complejidad espacial total de la compuerta $Hadamard$ es de $6q$.

Ahora, tomando la Figura 4.1 y aplicando la propuesta de complejidad del espacio (32) sobre el algoritmo de Deutsch. De forma visual, se obtiene un espacio total de $2q(ENTR_q) + 16q(COMP_q) + 2q(ORAC_q) + 6q(COMP_q) + C_m(MED_q) = 26q + C_m$ observada en la Figura 4.4.

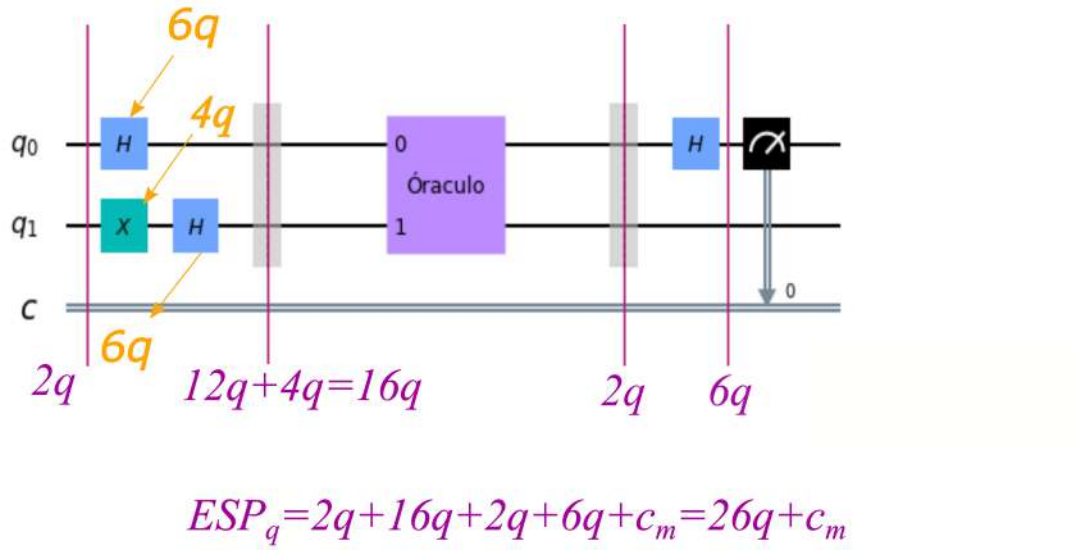

Figura 4.4: Se contabiliza la cantidad total de qubits necesarios para el algoritmo cuántico, usando la definición 32. Fuente: elaboración propia

De igual forma, se toma la Figura 4.1, para el análisis de la propuesta sobre la complejidad de tiempo. Obteniendo como resultado $TIME_{q_t} = 7_{t_q}$, mostrado en la Figura 4.5. Tomando la definición 9 la complejidad $TIME_{q_t} = O(7_{t_q})$, es decir constante.

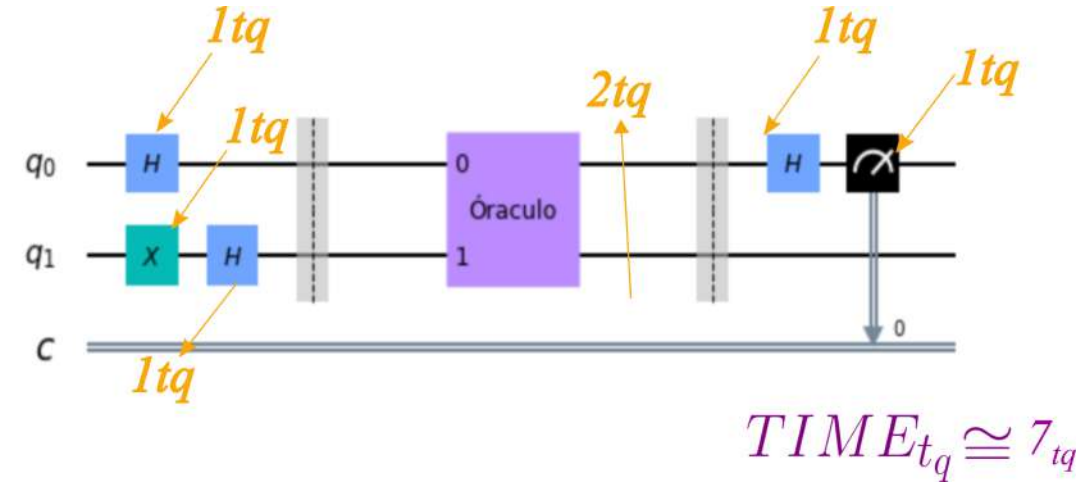

Figura 4.5: Se contabiliza la cantidad de tiempo t_q sobre el algoritmo de Deutsch, utilizando la definición 33. Fuente: elaboración propia

Debido a la simplicidad de este algoritmo podría pensarse que no es necesario realizar un análisis de este tipo dado que es casi determinista. Sin embargo, la importancia de éste es la apertura hacia la generalización sobre la evaluación de más de una función, es decir de n funciones evaluadas con una ejecución. Esta evolución, es conocida como el algoritmo de *Deutsch-Jozsa*. Por lo cual, la métrica aquí aplicada podrá emplearse también a ese desarrollo, la cual crecerá conforme crece la cantidad de n funciones dadas.

4.3 Implementación algoritmo de Grover

Las funciones descritas en el Capítulo 3, función *oráculo 2.12* y función *difusor 2.13* producen los siguientes códigos 4.5 y 4.6 en el lenguaje de programación Qiskit.

Código 4.5: Función oráculo en Grover. Representación de la función oráculo 2.12, el cual recibe el tamaño del arreglo, así como el índice a marcar como el valor encontrado. Fuente: elaboración propia

```
1
2 def fase_oraculo(n, indices_a_marcar, nombre = 'Oraculo'):
3
4     # crea un circuito de n qubits
5     qc = QuantumCircuit(n, name=nombre)
6
7     #crea una matriz identidad con n qubits
8     matriz_oraculo = np.identity(2**n)
9     # marcas con -1 los elementos a buscar
10    for g in indices_a_marcar:
11        matriz_oraculo[g, g] = -1
12
13    matriz_oraculo
14    #Convierte tu matriz (matriz_oraculo) dentro de un
          operador, y lo agregas al circuito cuántico
15    qc.unitary(Operator(matriz_oraculo), range(n))
16
```

```
17     return qc
```

Código 4.6: Función difusor en Grover. Aplicación de la función difusor 2.13, la cual emplea la compuerta Hadamard 2.2 para amplificar el valor correcto. Fuente: elaboración propia

```
1      #Crear un circuito cuántico de n qubits
2      qc = QuantumCircuit(n, name='Difusor')
3
4      # aplicar la compuerta Hadamard a todos los qubits (
           Superposición)
5      qc.h(range(n))
6      # Llamada a la función oraculo aplicada al estado cero
7      qc.append(fase_oraculo(n, [0]), range(n))
8      # Aplicar la  compuerta Hadamard a todos los qubits
9      qc.h(range(n))
10
11     return qc
```

Se realiza un código principal 4.7 que usa el código 4.5 y código 4.6. Como se explicó en la sección 2.3.2, es necesario determinar la cantidad de veces que se deben ejecutar estas funciones en el algoritmo principal para obtener el valor marcado x^*. En particular, esa cantidad se define como una rotación de aproximadamente 90ř, es decir $r = \lceil \frac{\pi}{4}\sqrt{\frac{N}{k}} \rceil$, donde k es el número de elementos a buscar $(indices\_a\_marcar)$ y N es el tamaño del arreglo (elementos totales).

Código 4.7: Función Grover. Algoritmo principal que ejecuta las dos funciones de código 2.12 y código 2.13, recibiendo como entrada el tamaño del arreglo y los índices a buscar. Fuente: elaboración propia

```
1
2      # Crea un circuito cuantico de n qubits con b bits clá
           sicos
3      qc = QuantumCircuit(n, n)
4
5      # Determina r, valor más acercado a 90 grados
```

```
6       r = int(np.floor(np.pi/4*np.sqrt(2**n/len(
            indices_elements_marcados))))
7       print(f'{n} qubits, Estados básicos {
            indices_elements_marcados} marcados, {r} rondas')
8
9       # paso 1: Aplicar compuerta Hadamard a todod los qubits
            estado de superposición
10      qc.h(range(n))
11
12      # paso 2: aplica r veces la fase de oraculo y difusor
13      for _ in range(r):
14          qc.append(fase_oraculo(n, indices_elements_marcados),
                range(n))
15          qc.append(difusor(n), range(n))
16
17      # step 3: aplicar compuerta medida todos los qubits
18      qc.measure(range(n), range(n))
19
20      return qc
21
22  mycircuit = Grover(3, [1])
23  #mycircuit.draw(output='text')
24  mycircuit.draw()
```

La implementación del algoritmo cuántico desarrollado se muestra en la Figura 4.6, en forma de diagrama. Se observa la cantidad de r veces que se ejecutará el conjunto de las funciones oráculo y difusor, así como la compuerta de medición al final, la cual mapea los qubits hacia los bits clásicos c_1, c_2, c_3. La Figura 4.7 muestra el resultado de la implementación en la simulación de computadora cuántica.

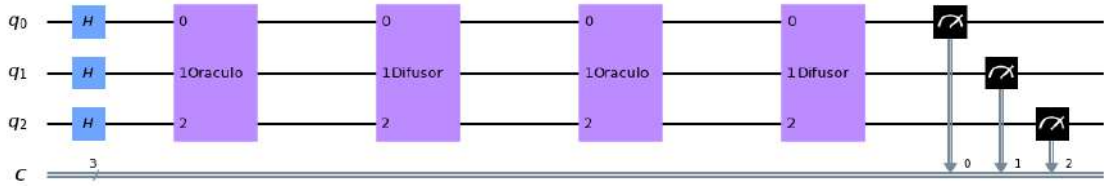

Figura 4.6: Muestra la entradas cuánticas y clásicas (q_i, c_i), las compuertas necesarias y las funciones *difusor* y *oráculo* aplicadas $r = 2$ veces. Fuente: elaboración propia sobre Qiskit en IBM (2019)

Note que en el código 4.7 se utilizó la función Grover con un arreglo de tamaño 3 y el valor 1 como datos de entrada $mycircuitQC = Grover(3, [1])\#Ejecuto$, es decir sobre un arreglo de tamaño 3 quiero conocer la posición del valor 1. En la Figura 4.7 se muestra el resultado de la simulación la cual disminuye el ruido, en consecuencia la probabilidad de que el elemento 1 se encuentre en la posición del arreglo 001 es de $94{,}6\,\%$, para esta ejecución. Sin embargo, para la ejecución sobre una computadora cuántica real mostrada en la Figura 4.8 es un poco distinto.

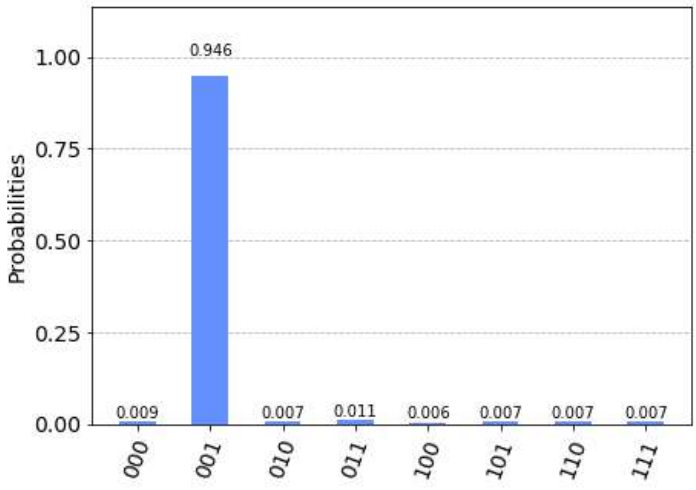

Figura 4.7: Resultado de la ejecución del código realizando 1000 repeticiones, con 3 qubits y 1 como elemento a buscar. Implementado sobre simulador en IBM (2019) *Quantum Experience*. Fuente. elaboración propia

Para la implementación sobre una máquina cuántica, se elige un circuito con 5 qubits. El producto de la ejecución del algoritmo, son mostrados en la Figura 4.8, donde se expone el resultado correcto, con probabilidad $24{,}0\,\%$. Se nota que también crecen otras opciones, no de igual forma que la respuesta correcta. Este comportamiento se debe a la utilización de amplificación de amplitudes, es decir el desarrollo de la función difusor, código 4.6.

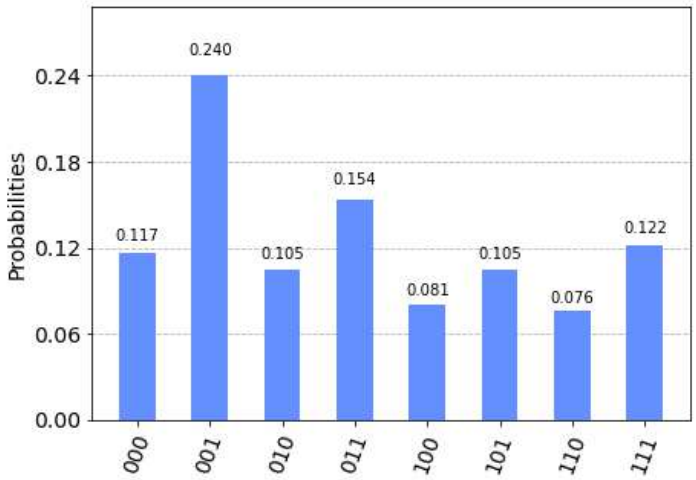

Figura 4.8: Ejecución del código, que realiza 1000 repeticiones, con 3 qubits y 1 como elemento a buscar. Implementado en máquina cuántica '$ibmq\_5\_yorktown$'. Implementado en IBM (2019) *Quantum Experience*. Fuente: elaboración propia.

También para este algoritmo se obtuvieron los tiempos de ejecución sobre la computadora cuántica y el simulador, los cuales ponen en evidencia las limitaciones de este paradigma, debido a la interferencia cuántica, asimismo el canal de comunicación puede determinar un aumento o disminución en el tiempo de acceso al resultado correcto.

Tabla 4.2: Tiempos de ejecución con 1000 repeticiones sobre un simulador de computador cuántico y una computadora cuántica real de 5 qubits. Fuente: elaboración propia

Ejecución en simulador (s)	**Ejecución en computadora cuántica** (s)
`0.03013324737548828`	`2.0418701171875`

Con estos elementos es posible analizar el algoritmo cuántico de Grover implementado, para hacer una comparación.

4.3.1 Evaluación de métrica

Tomando la Figura 4.6 y aplicando la propuesta de complejidad de espacio desarrollada, sobre el algoritmo de Grover de forma visual en la Figura 4.9, se aprecian $3q$ de entrada ($ENTR_q$), 3 compuertas $Hadamard$ aplicadas ($COMP_q$), las cuales son representadas por $3(6q)$, posteriormente se aplica el conjunto de oráculo ($ORAC_q$) y difusor ($COMP_q$) r veces, por lo cual el acumulado está representado como $3q + 2(6q)$, y por último, se aplican $3c_m$ compuertas de medida (MED_q).

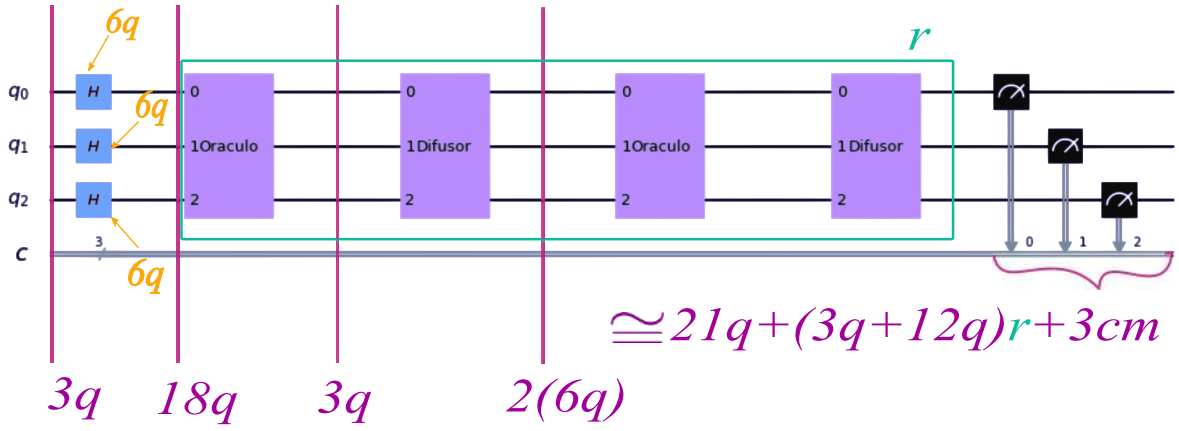

Figura 4.9: Se contabiliza la cantidad total de qubits necesarios para el algoritmo cuántico, usando la definición 32. Fuente: elaboración propia

La función difusor en este caso, representa un cambio de fase, y como se explico anteriormente se considera como una compuerta cuántica, es decir se contabilizará la cantidad de qubits con los que debe trabajar además de los qubits sobre los cuales se aplicará dicha función.

Al observar con detalle la función difusor, tiene una entrada de n qubits, en la implementación $n = 3$, por lo tanto, la complejidad inicial espacial es 3_q. Además aplica 2 veces la compuerta

$Hadamard$ para crear un estado de superposición, en consecuencia se agregan $2(6_q)$, esto se repite una cantidad r para la amplificación de la amplitud. Obteniendo un resultado sobre la función difusor de $3q + 2(6q)$.

Como resultado final, se obtiene una complejidad espacial de $ESP_q = 3_q + 18_q + (3_q + 12_q)r + 3C_m \cong 21_q + (15_q)r + 3C_m$ note que la complejidad tiene una variable r que representa la repetición del oráculo y difusor. En particular, este experimento utiliza $r = 2$, en consecuencia al sustituir este valor, la complejidad espacial queda $ESP_q = 51_q + 3c_m$, de forma general se observa que la complejidad espacial se comporta linealmente, conforme al número de repeticiones r.

Igualmente para el análisis de la propuesta sobre la complejidad de tiempo, se toma la Figura 4.1. Dando como resultado $TIME_{q_t} = 3_{t_q} + (3_{t_q} + 2_{t_q})r + 3_{t_q}$, mostrado en la Figura 4.10. Tomando la definición 9, en la complejidad de tiempo $TIME_{q_t} = O(3_{t_q} + 5_{t_q}r + 3_{t_q})$, se distingue, nuevamente, la variable independiente r, en consecuencia se afirma que es lineal $O(r)$.

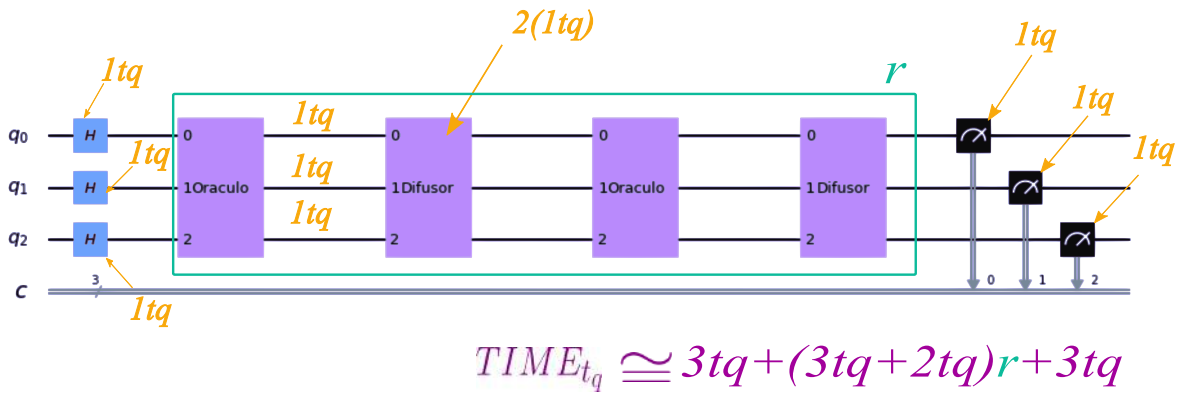

Figura 4.10: Se contabiliza la cantidad de tiempo t_q sobre el algoritmo de Grover, utilizando la definición 33. Fuente: elaboración propia

Para la complejidad temporal $TIME_{q_t}$, de manera similar a la complejidad ESP_q, la función difusor simboliza una compuerta que a su vez emplea la aplicación Hadamard, en consecuencia

se debe contabilizar la cantidad de tiempo t_q por compuerta interna aplicada. A su vez dependerá de la cantidad r (repeticiones del oráculo y difusor) nuevamente, para este caso $r = 2$, por lo tanto $TIME_{q_t} = 3_{t_q} + 5_{t_q}(6) + 3_{t_q} = 36_{t_q}$, es decir esta función crecerá conforme crezca la variable r.

Es evidente que cada uno de los dos desarrollos aquí descritos (Bernstein and Vazirani, 1997; Aharonov et al., 2021), aportan elementos de forma separada en cada área particular (teórica y experimentalmente), sin embargo, al unir las dos ideas es posible generar una complejidad consistente y útil para determinar características necesarias en la implementación de algoritmos cuánticos sobre cualquier lenguaje desarrollado, no solo de manera abstracta si no además de forma experimental, como se mostró en los desarrollos previos.

Se debe mencionar que la base matemática, fortalece el campo. Sobre este desarrollo, mezcla dos teorías fundamentales (matemática y física), para la generación y aplicación de algoritmos cuánticos. Por lo cual, es posible establecer la cantidad de requerimientos sin la necesidad de una realización física, de igual manera permite hacer afirmaciones mediante demostraciones matemáticas.

Un detalle relevante es el oráculo, el cual cuando se analiza su creación es posible conocer de forma aproximada el funcionamiento, pero éste puede no ser significativo para problemas pequeños, como los aquí mostrados. En cambio, existen problemas que podrían necesitar un oráculo con una mayor cantidad de qubits o compuertas de mayor complejidad y no tomarse en cuenta estas estructuras internas del oráculo, podrá derivar en una complejidad no tan precisa, esto dependerá al problema estudiado.

De modo que, es útil (dependiendo de lo que se desea analizar) tomar en cuenta el desarrollo interno del oráculo, ya que determinará de forma más específica la complejidad espacio temporal del algoritmo, hecho actualmente relevante debido a las limitaciones de qubits y espacios reales físicos. Por otro lado, si se desea observar de forma general el comportamiento

de un parámetro en particular, basta con obtener su función y determinar de esta manera un límite haciendo uso de notación asintótica altamente funcional.

Capítulo 5

Conclusiones

En esta tesis se ha mostrado el desarrollo de una métrica para la complejidad algorítmica cuántica espacio temporal utilizando conceptos tanto del cómputo tradicional como del cuántico, en donde la forma de comparación es por medio de la notación asintótica. Asimismo, esta métrica permite determinar el comportamiento de los recursos necesarios con una representación asintótica. Este hecho permite crear algoritmos cuánticos contemplando un crecimiento en la cantidad de datos de entrada.

También se observó, que la aproximación desarrollada de una QTM es sumamente útil, ya que la transición de desarrollos sobre una TM a una QTM, son realizables. Esto es, que si se desea trasladar algoritmos tradicionales conocidos al modelo cuántico, se deberá estudiar a detalle los procesos algorítmicos con ayuda por ejemplo, de la métrica aquí desarrollada, que afirme si es factible o no cambiar de modelo. Considerado como un trabajo a futuro.

La métrica que se propone en esta tesis permite ser comparada con la computación tradicional, en el entendido que se obtiene una función de complejidad. En consecuencia, permite cuantificar el costo de recursos computacionales en ambos paradigmas sobre un grupo común de tareas computable, como se ha mostrado.

En el contexto de la tesis en el Capítulo 4 Resultados se muestran implementaciones en dos paradigmas. Estas implementaciones son comparables en ambos paradigmas. Sin embargo, la optimización para cada uno de estos desarrollos pueden propiciar otro tipo de implicaciones, como el que se vuelvan no comparables. Tal es el caso del algoritmo de Deutsch. Ya que la generalización de este algoritmo, es decir su implementación sobre una mayor cantidad de funciones, muestra el potencial del modelo cuántico al tener propiedades como el paralelismo masivo, superposición de información, por mencionar algunas, evidenciando la ventaja en el paradigma cuántico el cual no tendría un símil con el cómputo tradicional.

Sobre algoritmo de Grover, al ser de naturaleza cuántica resulta complicado compararlo con en cómputo clásico. Por esta razón, una forma de evaluar el algoritmo es la complejidad algorítmica desarrollada, que permite determinar cuán grande es el aporte de un algoritmo sobre este paradigma, sin tener la necesidad de implementarlo sobre una computadora cuántica real.

Como trabajos futuros para lograr una métrica más generalizable es posible abordar la dificultad de plantear los distintos problemas desarrollados en el cómputo tradicional directamente a un entorno cuántico. En este sentido se requiere de trasladar varias ideas de la computación tradicional a la computación cuántica, sin embargo estos son los primeros pasos de un largo camino para una computación que promete una mejora en procesos para compartir, administrar, almacenar y mantener la seguridad de la información. Por lo tanto, es importante seguir contestando preguntas que permitan un avance casi en cualquier dirección del área, para comprender y proponer maneras de aprovechar esta nueva realidad.

Consecuentemente el aporte de esta métrica es notable, ya que la conjetura P=NP, sigue ahí como un zumbido en los teóricos de la computación y saber si la solución de un problema NP se puede resolver con una QTM, puede abrir camino a contestar esa conjetura. Por ahora este será un trabajo pendiente, ya que es evidente que el cómputo cuántico ofrece una alternativa distinta a un modelo clásico de computación y presenta una herramienta para desarrollar

algoritmos distintos a los tradicionales.

Es importante mencionar que si se desea desarrollar una clasificación sobre algoritmos cuánticos requiere responder preguntas fundamentales en la teoría de la computación tradicional, como lo es $P\ vs\ NP$. Si se pretende contestar este tipo de preguntas, se deberá realizar un análisis de los distintos modelos de la computación cuántica y de forma general, realizar una teoría o un modelo universal, que ayude a establecer dicha clasificación.

Lo aquí mostrado también da evidencia de que la comparación en tiempos de ejecución sobre una aplicación no es suficiente para determinar que tan bueno o eficiente es un algoritmo cuántico. En la actualidad existen múltiples desarrollos de algoritmos en diversas tecnologías, mostrando los tiempos de ejecución, o cantidad de recursos usados, información útil, pero rápidamente mejorable, debido a los avances tecnológicos que existen.

Otra perspectiva útil para analizar este modelo cuántico, puede ser con la "teoría algorítmica de la información" desarrollada por el matemático Chaitin (1982), que habla de cómo medir el tamaño de los programas informáticos. Él define la palabra *tamaño* sin referirse a la cantidad de recursos (memoria o tiempo) que puede ocupar una máquina al ejecutar un programa. Trabajo que podrá complementar lo aquí presentado.

Aún queda pendiente, un trabajo que estudie la optimización de los recursos necesarios evidenciados por la métrica aquí expuesta. Sirviéndose de la misma como apoyo para observar los requisitos mínimos de un algoritmo cuántico de forma teórica.

Sobre este camino existirán muchas suposiciones falsas, muchos errores, los cuales serán aportes (en ocasiones mínimos) que ayudarán a desechar un camino o seguir abriendo paso sobre éste. Este trabajo forma parte del camino a recorrer. Como se observó la perspectiva general desarrollada funciona sobre cualquier algoritmo (con un previo análisis) que haya sido desarrollado para la computación cuántica.

Pero sobre este campo todavía hay muchas preguntas por responder por ejemplo ¿Dado que utiliza otras leyes, las leyes cuánticas, se debe medir de la misma manera?, ¿cuales son los nuevos conceptos o parámetros importantes en esta nueva área? ¿Solo los físicos especializados en cuántica pueden responder a estas preguntas? ¿Los cómputologos no tienen posibilidad de hacer afirmaciones sobre estos temas?

Además existen aplicaciones por generar que exploten esta área como lo son; solución a sistemas de ecuaciones, corrección de errores debido al acceso incoherente de información, el método para transmitir información, el paso de mensajes (*message-passing*), la complejidad de la comunicación cuántica (*Quantum communication complexity*), la complejidad de consulta (*Quantum query complexity lower bounds*), entre otros.

Bibliografía

Aaronson, S. (2013). Why philosophers should care about computational complexity. *Computability: Turing, Gödel, Church, and Beyond*, 261:327.

Aharonov, D., Cotler, J., and Qi, X.-L. (2021). Quantum algorithmic measurement.

Aho, A. V. and Hopcroft, J. E. (1974). *The design and analysis of computer algorithms*. Pearson Education India.

Albash, T. and Lidar, D. A. (2018). Adiabatic quantum computation. *Reviews of Modern Physics*, 90(1):015002.

Apers, S. and Lee, T. (2020). Quantum query complexity of edge connectivity. *arXiv preprint arXiv:2011.09823*.

Arora, S. and Barak, B. (2009). *Computational complexity: a modern approach*. Cambridge University Press.

Arute, F., Arya, K., Babbush, R., Bacon, D., Bardin, J. C., Barends, R., Biswas, R., Boixo, S., Brandao, F. G., Buell, D. A., et al. (2019). Quantum supremacy using a programmable superconducting processor. *Nature*, 574(7779):505–510.

Atallah, M. J. and Blanton, M. (2009). *Algorithms and theory of computation handbook, volume 2: special topics and techniques*. CRC press.

Bacon, D. and van Dam, W. (2010). Recent progress in quantum algorithms. *Commun. ACM*, 53:84–93.

Bencivenga, D., Chen, X., and Høyer, P. (2021). Quantum sampling in markov chains.

Bernstein, E. and Vazirani, U. (1997). Quantum Complexity Theory. *SIAM Journal on Computing*, 26(5):1411–1473.

Bouland, A., Fefferman, B., Landau, Z., and Liu, Y. (2021). Noise and the frontier of quantum supremacy. *arXiv preprint arXiv:2102.01738*.

Brown, P., Fawzi, H., and Fawzi, O. (2021). New rényi divergence families defined via convex optimization and their applications.

Chaitin, G. J. (1982). Godel's theorem and information. *International Journal of Theoretical Physics*, 21(12):941–954.

Chuang, M. N. and I.L. (2010). *Quantum Computation and Quantum Information*, volume 52. Cambrige University Press, United Kingdom.

Coles, P. J., Eidenbenz, S., Pakin, S., Adedoyin, A., Ambrosiano, J., Anisimov, P., Casper, W., Chennupati, G., Coffrin, C., Djidjev, H., et al. (2018). Quantum algorithm implementations for beginners. *arXiv preprint arXiv:1804.03719*.

Davis, M., Sigal, R., and Weyuker, E. J. (1994). *Computability, complexity, and languages: fundamentals of theoretical computer science*. Elsevier.

Deutsch, D. (1985). Quantum theory, the church–turing principle and the universal quantum computer. *Proceedings of the Royal Society of London. A. Mathematical and Physical Sciences*, 400:97 – 117.

Farhi, E., Goldstone, J., Gutmann, S., and Zhou, L. (2019). The quantum approximate optimization algorithm and the sherrington-kirkpatrick model at infinite size. *arXiv preprint arXiv:1910.08187*.

Feynman, R. P. (1982). Simulating physics with computers. *Int. J. Theor. Phys*, 21(6/7):467–488.

Feynman, R. P. (1985). Quantum mechanical computers. *Optics news*, 11(2):11–20.

Feynman, R. P. (1986). Quantum mechanical computers. *Foundations of Physics*, 16:507–531.

Feys, R. (1956). José ferrater mora, hugues leblanc, logica matemática. *Revue philosophique de Louvain*, 54(42):335–336.

Fujii, K., Kobayashi, H., Morimae, T., Nishimura, H., Tamate, S., and Tani, S. (2014). Impossibility of classically simulating one clean qubit computation. *Physical review letters*, 120 20:200–502.

Gödel, K. (1931). Über formal unentscheidbare sätze der principia mathematica und verwandter systeme i. *Monatshefte für mathematik und physik*, 38(1):173–198.

Goldreich, O. (2008). *Computational complexity: a conceptual perspective*. Cambridge University Press.

IBM (2019). Liga ibm experience quantum. `https://quantum-computing.ibm.com/composer`.

Jordan, S. (2011). Quantum algorithm zoo. `https://math.nist.gov/quantum/zoo/`.

Jordan, S. P. (2008). Quantum computation beyond the circuit model.

Jordan, S. P., Krovi, H., Lee, K. S., and Preskill, J. (2018). Bqp-completeness of scattering in scalar quantum field theory. *Quantum*, 2:44.

Kempe, J., Kobayashi, H., Matsumoto, K., Toner, B., and Vidick, T. (2011). Entangled games are hard to approximate. *SIAM Journal on Computing*, 40(3):848–877.

Kitaev, A. Y. (1997). Quantum computations: algorithms and error correction. *Uspekhi Matematicheskikh Nauk*, 52(6):53–112.

Knuth, D. E. (2011). *Art of Computer Programming, Volumes 1-4A Boxed Set*. Addison-Wesley Professional.

Lanzagorta, M. and Uhlmann, J. (2008). Quantum computer science. *Synthesis Lectures on Quantum Computing*, 1(1):1–124.

Lipton, R. J. (2014). *Quantum Algorithms via Linear Algebra*.

Mermin, N. D. (2007). *Quantum Computer Science An Introduction*. Cambrige University Press, United States of America, NY.

Montanaro, A. (2016). Quantum algorithms: an overview. *npj Quantum Information*, 2:15023.

Mora, C. E. and Briegel, H. J. (2005). Algorithmic complexity and entanglement of quantum states. *Physical Review Letters*, 95(20).

Mosca, M. (2009). Quantum algorithms. *Encyclopedia of Complexity and Systems Science*, pages 7088–7118.

Papadimitriou, C. H. (1993). Computational complexity.

Rand, R. (2019). Verification logics for quantum programs. *ArXiv*, abs/1904.04304.

R.C.T., Lee; S.S., T. R. C. Y. T. (2007). *Introducción al diseño y análisis de algoritmos: Un enfoque estrategico*. McGraw Hill Interamericana.

Rieffel, E. G. and Polak, W. H. (2011). *Quantum computing: A gentle introduction*. MIT Press.

Russell, B. and Whitehead, A. N. (1997). *Principia mathematica to 56*, volume 2. Cambridge University Press Cambridge, UK.

Sanchez Ron, J. M. (2001). Historia de la fisica cuantica. *Barcelona, Editorial Critica*.

Santha, M. (2008). Quantum walk based search algorithms. In *TAMC*.

Sen, Sandeep y Kumar, A. (2019). *Design and Analysis of Algorithms: A Contemporary Perspective*. Cambridge University Press.

Serway Raymond, A. (1993). Fisica tomo ii.

Shor, P. W. (1994). Polynomial time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM J. Comput.*, 26:1484–1509.

Sipser, M. (2012). *Introduction to the Theory of Computation*. Cengage learning.

Steane, A. M. (1997). Quantum computing.

Turing, A. M. (1937). On computable numbers, with an application to the entscheidungsproblem. *Proceedings of the London mathematical society*, 2(1):230265.

Wang, Z. (2010). *Topological Quantum Computation*, volume 112. Conference Board of the Mathematical Sciences, Washington, DC, american mathematical society edition.

Watrous, J. (2012). Quantum computational complexity. *Computational Complexity: Theory, Techniques, and Applications*, 9781461418:2361–2387.

Wigderson, A. (2019). *Mathematics and Computation*, volume 15. Princeton University Press, United Kingdom.

Wiseman, H. M. and Milburn, G. J. (2009). Quantum measurement and control.

Wittek, P. (2014). *Quantum machine learning: what quantum computing means to data mining*. Academic Press.

Yanofsky, N. S. and Mannucci, M. A. (2009). *Quantum computing for computer scientists*, volume 46.

Ying, M. (2016). *Foundations of Quantum Programming*.

Yoganathan, M. and Cade, C. (2019). The one clean qubit model without entanglement is classically simulable.