Casa abierta al tiempo
UNIVERSIDAD AUTÓNOMA METROPOLITANA
Unidad Iztapalapa

DIVISIÓN DE CIENCIAS BÁSICAS E INGENIERÍA
Posgrado en Ciencias (Física)

Algoritmos Cuánticos

TESIS PARA OBTENER EL GRADO DE:
MAESTRO EN CIENCIAS (FÍSICA)
QUE PRESENTA

Fís. Didier Gamaliel Buendia Ortiz

Matricula: 408035938
Correo: didier_45@hotmail.com

Asesores: Dr. Isaac Pérez Castillo
Dr. Norberto Aquino Aquino

Jurado:

Dr. Moisés Martínez Mares	Presidente
Dr. Norberto Aquino Aquino	Secretario
Dr. Carlos Francisco Pineda Zorrilla	Vocal

Iztapalapa, Ciudad de México a 22 de abril del 2025

Agradecimientos

¿Por qué camino se difunde la luz y se esparce el viento solano sobre la tierra?
Job 38:24

Quiero iniciar con las más sinceras palabras: Mi agradecimiento al Pueblo de México, quien, con su arduo trabajo, genera los recursos que permiten la existencia de instituciones de educación superior gratuitas. Agradezco a la Universidad Autónoma Metropolitana por abrir sus puertas para permitirme retomar mi camino como estudiante y como científico, así como al Centro de Investigaciones Avanzadas, del Instituto Politécnico Nacional, en donde adquirí disciplina y habilidades de trabajo indispensables para desarrollarme plenamente en la maestría. Y, por supuesto, a la Facultad de Ciencias de la Universidad Nacional Autónoma de México, que me dió los fundamentos, el grado de Licenciatura y la experiencia de una vida dedicada a la investigación científica. Agradezco al Consejo Nacional de Humanidades, Ciencias y Tecnologías (CONAHCYT), hoy Secretaría de Ciencia, Humanidades, Tecnología e Innovación, por la beca que me permitió continuar con mis estudios

Además, quiero expresar mi gratitud con cada una de las personas que ha colaborado, de manera directa, con mi proyecto personal. El camino, lejos de ser sencillo, resultó tortuoso y considero que, de no existir tantos amigos que en un momento u otro me extendieran la mano, jamás habría llegado hasta donde me encuentro.

Agradezco a las personas que me hospedaron en su casa, para permitirme continuar con mis estudios. A Araceli Medina Espinoza y Roberto García Madrid, por confiarme las llaves de su casa para cualquier emergencia o eventualidad. Mi gratitud eterna a la familia Figueroa, quienes me hicieron sentir un miembro de su familia y me asignaron un lugar muy especial en su hogar.

A mis alumnos, y a sus padres, quienes con sus aportaciones auspician mis estudios. A la Universidad Nacional Autónoma de México, así como a la Escuela Superior de Turismo del Instituto Politécnico Nacional, a la Universidad Victoria, al grupo educativo UCΔP,así como al Instituto Isaías, por abrirme sus puertas y permitirme trabajar en sus instalaciones durante mi tiempo libre.

A mis amigos, que jamás dejaron de creer en mí y que siempre han estado ahí para darme ánimos en los momentos más difíciles, así como para compartir los más dulces. Araceli Ferrusca Lira. Ana Grissel Ángel Padilla. Anabel Nieves Cabrera. Gloria Ocho Hernández. Guillermo Adrián Gutiérrez, Anayeli Hernández Rivera. Ariana Wendoli Arizmendi Muñóz. Nelly Paola Guerrero Luna. Yokinaro Raymundo Arredondo. Eunice Pérez Pérez. Juan Gabriel Hernández. Francisco León. A mis amigos del CINVESTAV: Janeth, Ariadna, Ramón y Alex y muy especialmente a Edzna. No encuentro las palabras que representen el agradecimiento, por guiarme desde el Cinvestav hasta la UAM. Tambien al Dr. Luis Odín Estrada Ramos, que me permitió laborar como su ayudante en la última etapa de esta tesis y me permitió vivir la experiencia de mi vida, mi anhelo mas profundo, el sueño que jamás creí ver realizado.

Mi agradecimiento también para las personas que me han inspirado: a Nanci, quien creyó durante mucho tiempo no sólo que yo era bueno, sino que era el mejor. Ahora me toca creerlo a mi. Como he dicho en otro lugar, si tuviera que escribir todas las razones por las que estoy agradecido contigo, esta tesis fácilmente duplicaría su extensión. Es complicado ignorar a M.S., sin embargo, a preferido permanecer en anonimato por razones personales, pero admito que he contraído una deuda que jamás podré pagar. Finalmente, a Katerine García Martinez, que ha permanecido como una excelente amiga y me ha permitido conocer áreas de

las matemáticas con las que jamás habría soñado.

Al Dr. Isaac Pérez Castilo y al Dr Norberto Aquino Aquino, así como a su equipo de trabajo, por sus enseñanza y por la dirección de este trabajo. El método del Dr. Isaac me ha permitido desarrollar el presente trabajo al tiempo que cultivo otras habilidades que me permiten desarrollarme a nivel intelectual, más allá de la labor científica, como ser humano. Gravias por ello

A los sinodales, Dr. Moisés Martínez Mares, Dr. Carlos Francisco Pineda Zorrilla y Dr. Norberto Aquino Aquino, cuyas exhaustivas revisiones y comentarios no solo permitieron que este trabajo alcanzara su presentación final, sino que también guiaron y motivaron el desarrollo del mismo.

Finalmente, agradezco a mi familia, quienes siempre han alentado mis sueños. Ramsés Lara Ortiz. Irving Antonio Lara Ortiz. Roxana Isela Ortiz Medina. Alejandra Medina Espiniza. Jorge Ortiz Trejo. Iris Sarai Buendia Ortiz. Dafne Aline Buendia Ortiz. Alejandra Patricia Ortiz Medina, querida madre esta tesis es por ti, que hoy retomas tus sueños inspirándome para alcanzar los los míos.

Didier Gamaliel Buendia Ortiz
Universidad Autónoma Metropolitana, Unidad Iztapalapa. Abril 2025

Índice

Lista de figuras

Lista de tablas

Resumen

En este trabajo se discuten los principios básicos de la computación cuántica, explicando con considerable detalle el proceso de evaluación de funciones y algunos algoritmos cuánticos bien conocidos: Deutsh, Deustsh Jousza, Simmon, Estimacion de Fase Cuántica,Transformada Cuántica de Fourier, Grover y Shor. Se destaca la importancia de la tesis de borrado de Landauer, que establece el costo energético de la pérdida de información. Se considera importante debido a que las compuertas cuánticas son unitarias y, por tanto, físicamente reversible. La abrumadora mayoría de los autores consideran la equivalencia entre reversibilidad lógica y reversibilidad física, imponiendo la necesidad de la primera dentro del modelo de la computación cuántica.

1 Introducción

La comprensión científica del cosmos y de los fenómenos que tienen lugar dentro de él requiere de muchas habilidades, pero entre todas ellas destaca la capacidad para preparar, observar, registrar e interpretar un conjunto de eventos y ordenarlos dentro de un marco coherente. Hoy en día la sociedad reconoce el papel fundamental de la información y del tratamiento de esa información como base para la comunicación y el descubrimiento, por lo que resulta natural observar que los métodos empleados por los científicos que se dedican a la reconstrucción de los fenómenos naturales también han evolucionado hasta abarcar los métodos computacionales de simulación

1.1. Planteamiento del problema

Simular un proceso natural significa poder reproducir de manera artificial sus propiedades y su evolución dinámica en el tiempo. Esto se hace con mayor eficacia en un entorno en donde todas las variables externas se encuentran bajo un riguroso control. El primer ejemplo de una simulación es, en el sentido más amplio posible, un modelo matemático preciso, es decir, una función que relacione la información que se conoce sobre un sistema de interés con un determinado conjunto de variables y ecuaciones, seguido de una solución analítica o numérica. El conjunto resultante de relaciones matemáticas, o la computadora con el programa destinado a resolverlas, puede llamarse *simulador*. Con un simulador de este tipo, el experimentador puede, en principio, probar el comportamiento del sistema real en condiciones bastante generales, inicializar su estado casi a voluntad, hacer predicciones y comprobar nuevas hipótesis, con las únicas limitaciones de la validez del modelo inicial y, posiblemente, el poder de cómputo a su alcance.

La última frontera en simulación científica está representada por la mecánica cuántica, la teoría microscópica más exitosa. Los modelos descritos en el marco de la mecánica cuántica suelen ser conceptualmente problemáticos y técnicamente exigentes, lo que ha motivado el desarrollo de herramientas y métodos durante el último siglo, pese a lo cual, se ha vuelto más claro que la distancia que existe entre el progreso actual y las simulaciones cuánticas completas de muchos cuerpos difícilmente podrá cubrirse si solo se cuenta con el apoyo de las máquinas de computación clásicas. Hay varias razones, pero la más destacada y esencial radica en la escala exponencial del tamaño, en términos de tiempo y recursos de memoria, de los problemas numéricos a los que se enfrentan los investigadores cuando intentan reducir los sistemas físicos reales a sus constituyentes fundamentales. Las limitaciones de los métodos de simulación clásicos se manifiesta especialmente cuando las fuertes correlaciones cuánticas entre los subsistemas se vuelven dominantes para determinar las propiedades del objeto bajo investigación, que es el caso en muchos problemas interesantes y generalmente abiertos.

Muchos problemas en la física tienen soluciones que se encuentran, en primer lugar, realizando modelos simplificados que permiten eludir algunos detalles que complican excesivamente la solución, pero que en esencia no cambian la física del fondo. A medida que las técnicas y herramientas matemáticas mejoran, es posible que pueda permitirse trabajar con modelos menos simples, pero que reflejen de manera más fiel la

realidad. Las aplicaciones pueden requerir un nivel de detalle que los modelos más sencillos pasan por alto, pero los modelos más complejos pueden resultar duros de tratar a mano. La computadora demostró ser un excelente aliado en el tratamiento de problemas, tanto por la velocidad con la que se obtienen soluciones, como por el nivel de detalle que proporcionan.

Ello conduce a la idea, generalmente atribuida a a Feynman[90] (con contribuciones tempranas de Y. I. Manin[179] y P. Benioff[22]) de que las propiedades íntimamente cuánticas de la naturaleza requieren una máquina de computación completamente cuántica para ser descritas de manera eficiente, de manera que se han propuesto, en las últimas décadas, los simuladores cuánticos como la puerta de entrada a nuevo enfoque para la computación cuántica. Un simulador cuántico es un sistema físico descrito por las leyes de la mecánica cuántica que tiene la capacidad, bajo condiciones controladas, de imitar el comportamiento dinámico de otro modelo físico cuántico. Los simuladores cuánticos surgen del pleno entendimiento de que todo cómputo es, en sí mismo, un proceso físico, por lo que toda máquina creada para calcular, es decir, cualquier computadora, también puede describirse de esa forma. Este enfoque lleva a concebir un sistema capaz de procesar información cuántica, como si sus propios estados internos obedecieran a las leyes que gobiernan la evolución de otro sistema considerado como objetivo.

La historia de la computación cuántica es relativamente reciente, pero en sus casi 4 décadas de existencia ha logrado un desarrollo formidable, en buena medida motivado por la intimidante capacidad potencial de las computadoras cuánticas para resolver problemas que hasta hace poco se consideraba impráctico, en el sentido de que aunque la forma de resolverlos era conocida, el tiempo que le tomaría a las computadoras clásicas excede cualquier medida de razonable o concebible o bien, demanda más recursos que los disponibles en todo el universo. Surge la necesidad de analizar la distinción entre la posibilidad de encontrar una solución que no requiera más que el continuo trabajo repetitivo que puede realizar una máquina, y la factibilidad de hacerlo, porque bien podría ser que no termine de hacerlo.

Los algoritmos cuánticos son más que adaptaciones de la multitud de instrucciones que ya existen para las computadoras clásicas. Cuentan con un conjunto de herramientas novedoso y a veces, no muy intuitivo. Precisa también de una lógica ligeramente distinta a la usual, aunque es capaz de realizar todas las tareas que una computadora clásica haría. Debido a esto, los algoritmos cuánticos conocidos son, a día de hoy, muy pocos y la mayoría no parecen resolver problemas que tengan relación directa con la vida cotidiana. Son, en realidad, el resultado de pensar en cómo una computadora cuántica puede demostrar ventaja de esta forma o de esta otra

En su estado actual de desarrollo, cuenta con capacidades limitadas debido a la extrema sensibilidad de los sistemas físicos que le constituyen. Debido a que se trata de una idea bastante reciente, existen diferentes implementaciones, sin que alguna haya mostrado una ventaja definitiva.

A lo largo de este trabajo se presentan algunos conceptos elementales en computación cuántica.

1.2. Objetivo general.

- Estudiar los algoritmos cuánticos esenciales.

1.3. Objetivos específicos.

- Revisión bibliográfica del estado actual de la computación cuántica.
  - Investigar las nociones básicas de la computación clásica, sus capacidades y limitaciones
  - Entender los elementos generales de la computación cuántica
  - Conocer algoritmos y circuitos cuánticos paradigmáticos
  - Comprender los pilares de la computación cuántica
- Construir circuitos cuánticos elementales con compuertas en IBM Quantum Composer

1.4. Metodología

Existe abundante material bibliográfico y artículos sobre computación cuántica. Se hará una revisión exhaustiva de libros de texto y material especializado. Se participará en talleres de cómputo cuántico realizados por diferentes instituciones.

1.5. Estructura de la tesis.

Este trabajo está distribuido en siete capítulos, incluyendo esta introducción. El capítulo dos contiene los elementos básicos de la computación clásica y explora los antecedentes históricos y motivaciones que llevaron al desarrollo de la computadora convencional. Se presentan dos modelos de computación: La máquina de Turing y el modelo de circuito. Se discuten conceptos elementales de lógica binaria, complejidad computacional, la relación entre energía y computacional que, finalmente, llevaran a conocer las limitaciones de la computación clásicas y entender las motivaciones detrás del desarrollo de la computación cuántica. En el capítulo tercero se presentan, de forma bastante superficial, algunos tópicos de la mecánica cuántica, entre ellos, su formulación axiomática y un par de experimentos que, a pesar de no ser los primeros cronológicamente en conocerse, son los que mejor representan la distinción entre el comportamiento clásico y el cuántico. En el capítulo cuatro se presenta el modelo de computación cuántica, sus elementos, componentes y algunas herramientas. El quinto capítulo describe algunos algoritmos cuánticos elementales, seguido de un capítulo entero dedicado al algoritmo de Shor. El capítulo final contiene una breve discusión sobre las implicaciones de la controversia en torno a la tesis de Landauer.

2

Computación Clásica

Las computadoras son parte importante de la vida cotidiana actual. Es tal su influencia y la necesidad que se tiene hoy de ellas, que es difícil imaginar el mundo si repentinamente se tuviera que prescindir de ellas. Se sabe que, como disciplina, la computación fue introducida a mediados del siglo XX, pero los principios subyacentes, sus fundamentos lógicos y matemáticos, se desarrollaron siglos atrás. Los antecedentes resultan difíciles de rastrear porque se encuentran ligados al desarrollo de las primeras máquinas construidas para la ejecución de cálculos aritméticos, algunas de las cuales son bien conocidas y cuya concepción o realización están asociadas con el nombre de alguna persona bien identificada. Sin embargo, el hecho de que las computadoras hayan alcanzado el estado actual se debe a la lenta y no siempre bien planeada evolución de ideas y propuestas experimentales debidas a muchas personas, que trabajaban en áreas del conocimiento muy distintas entre sí. Matemáticas, lógica, mecánica, ciencia de materiales, electrónica y programación se han fusionado en una disciplina nueva, que toma elementos, notación, aplicaciones e ideas de sus progenitoras.

2.1. Introducción

La computación ha permitido que los avances científicos, industriales y comerciales alcancen niveles imposibles de lograr en cualquier otra época de la historia. Gracias a la ayuda que proporciona el procesamiento automático de datos, muchas empresas funcionan de manera eficiente, lo que permite un importante ahorro en tiempo y otros recursos. En casos extremos, puede considerarse que hay actividades que se vuelven impensables sin la continua vigilancia de una computadora, como el programa espacial. La ubicuidad de las computadoras parece hacer innecesaria la presentación de una definición aquí. Después de todo, la palabra computadora, tomada como adjetivo, hace referencia a la capacidad de computar o calcular. Si se emplea como sustantivo, se refiere a las máquinas que pueden servir para realizar operaciones matemáticas, sin embargo, hoy en día poseen capacidades mucho más amplias, de manera que resulta posible que la propiedad más importante de las computadoras resulte ser su generalidad. Gracias a ella, las computadoras pueden ser usadas tanto en la realización de cálculos científicos, como en el procesamiento de información de comercios, controlar el tráfico aéreo, controlar misiones espaciales, investigación y desarrollo de nuevos materiales, en el campo de la educación y en muchas otras áreas, cada una de las cuales recibe un gran impulso al implementar las herramientas que la computación provee. Aunque esta revolución ocurre desde mediados del siglo pasado, las motivaciones detrás del desarrollo de máquinas que sean útiles para realizar cálculos son mucho más antiguas. Como disciplina, la computación no se define únicamente en términos de las máquinas que realizan operaciones, sino que trasciende para tratar de responder a la pregunta fundamental: «¿Qué puede ser (eficientemente) automatizado?» [65]. En la búsqueda de esa respuesta se encargará de estudiar los procesos algorítmicos que describen y transforman información, considerando la teoría que lo sustenta, la manera en analizan los problemas, la forma en que se diseñan, proponiendo esquemas para cuantificar su eficiencia, estudiando la manera en que llega a implementarse y las áreas en donde puede aplicarse.

El objetivo de la computación no puede entenderse sin explicar con cierto detalle el significado de los llamados «procesos algorítmicos». La aparición y desarrollo de tales procesos parecen motivados por el aburrimiento que producen las tareas monótonas y repetitivas. Tras realizar un cierto esfuerzo intelectual para resolver un problema de cierta categoría y volver a realizar una y otra vez un nuevo esfuerzo intelectual para resolver otros problemas semejantes, los seres humanos tienden a considerar la posibilidad de ahorrarse el esfuerzo en el futuro mediante la invención de un procedimiento de resolución automática para todos los problemas que sean del mismo tipo.

Inicialmente, es posible que se requiera un esfuerzo intelectual adicional. Este esfuerzo puede llegar a ser bastante notable, pero, a partir del momento en que se obtenga el procedimiento automático, no hará falta volver a preocuparse por encontrar solución a los problemas semejantes. En cada ocasión que se haga frente a un problema similar, será posible resolverlo sin apenas pensar o realizar un nuevo esfuerzo intelectual, únicamente hará falta que se sigan cuidadosamente las instrucciones del procedimiento. Mientras esto se hace, es claro que no se necesitará de iniciativa, creatividad o imaginación alguna, porque la aplicación ciega de instrucciones unívocas, ordenadas y mecánicas será suficiente para tratar el problema. Tales métodos, que permiten la solución automática de un conjunto de problemas, reciben el nombre de métodos recursivos [6], procesos algorítmicos o, simplemente, algoritmos. Intuitivamente, los algoritmos son un conjunto de instrucciones para la solución de problemas. Pero, si buena parte del campo de estudio de la computación está dedicada a los algoritmos, conviene presentar una definición que precise esta noción. Un algoritmo tiene las siguientes características[286]:

1. Se denominan datos de entrada a la información que se le asigna para trabajar. Un algoritmo puede operar utilizando cero o más datos.
2. Tras realizar el trabajo, es necesario que devuelva al menos un resultado, al que se denomina salida.
3. Es necesario que pueda especificarse a través de una cantidad finita de pasos.
4. Por definición, es necesario que cada paso pueda, al menos en principio, ser realizado por una persona capacitada y sin más equipo que lápiz y papel.
5. La ejecución del algoritmo debe terminar dentro de un lapso de tiempo finito

La descripción completa de un algoritmo requiere especificar, además de la tarea a realizar y la secuencia de instrucciones que realiza, un análisis en el que se indiquen las razones que garantizan que la secuencia de instrucciones logra completar la tarea.

Los algoritmos son el resultado de un análisis realizado a una cierta clase de problemas. Identificar aquellos problemas que son semejantes entre si y que pueden resolverse mediante el mismo mecanismo conduce al desarrollo de los algoritmos, por lo que su diseño está estrechamente relacionado con la búsqueda de formas para sistematizar el pensamiento. Las primeras contribuciones en este sentido fueron aportadas en la Antigua Grecia. Para que se desarrolle el método axiomático utilizado en las matemáticas actuales, cuyo origen puede fijarse en la construcción de la geometría como un sistema deductivo, es necesario el empleo de la lógica. Desde luego, su desarrollo no se realizó buscando cimentar las matemáticas de siglos posteriores, sino que atendía a una necesidad más inmediata: designar de manera objetiva al vencedor de un debate. Hacia el siglo V antes de nuestra era, en la Grecia Antigua, el nombre de sofista (portador de verdad) se asignaba a aquellos filósofos que eran maestros de literatura, ciencia y, en especial oratoria, que ponían más atención a los detalles formales de la poesía, que a sus contenidos extraliterarios. Para los sofistas, la retórica era más que el arte de hablar de una forma hermosa, también se debía hacer de manera correcta, a través de la composición de un discurso libre de ambigüedades y con un objetivo específico.

El método de enseñanza de los sofistas se valía de los debates, un conjunto de preguntas cerradas que dirigían al alumno a una contradicción, de manera que se le convencía de estar equivocado, por lo que eventualmente se desarrolló un sistema que permitía determinar quién de los participantes de un debate había logrado ganar la discusión. Los sofistas propusieron un conjunto de reglas que permiten razonar y discutir, distinguir lo verdadero de lo falso, una cuestión que resulta fundamental en toda actividad intelectual. Para tal fin, es necesario partir de un conjunto de axiomas, hechos o verdades que no pueden cuestionarse. Además, se requiere de un conjunto de reglas que permitan la transformación de los axiomas a otras expresiones equivalentes, que, al ser equivalentes a los axiomas, tampoco podrán negarse. Con ello, fue posible determinar si una proposición carecía de validez. La formulación de las mencionadas reglas de transformación, también llamadas *reglas de inferencia* requiere analizar la forma en la que los seres humanos llegan a conclusiones, es decir, en última instancia trata de proponer un algoritmo para el pensamiento. El uso de estos patrones de razonamiento no se limita a las matemáticas, en realidad forma parte de la vida cotidiana y es el origen de la disciplina conocida como *lógica matemática*.

Además de los algoritmos, la computación también busca desarrollar los medios para realizar cálculos, con mayor velocidad, sin errores y de manera más simple. A lo largo de la historia se han creado diferentes dispositivos con este propósito, pero el más antiguo parece ser el ábaco, cuya aparición en Oriente se estima en más de cinco milenios. Un ábaco es instrumento cuadrado, dentro del cual se han colocado una serie de varillas paralelas sobre las que es posible colocar una cierta cantidad de piezas móviles, que corren a lo largo de cada varilla y que se denominan *cuentas*. Las sucesivas filas de cuentas representan unidades, decenas, centenas, y en cada fila se fija un número concreto utilizando una cantidad equivalente de piezas. A pesar de la sencillez de su diseño, el ábaco permite a una persona entrenada realizar operaciones aritméticas elementales con una inusitada rapidez. La construcción de dispositivos similares en diferentes culturas, como el *nepohualtzintzin* de los mayas o el *schoty* de los rusos, muestra un interés que puede considerarse de carácter universal en la realización de máquinas que ayuden a calcular.

La idea de un cálculo mecánico, entendida como la realización de operaciones aritméticas más o menos complejas y repetitivas mediante el empleo de dispositivos con partes móviles, se preserva en el invento del clérigo británico William Oughtred en el siglo XVII, la llamada regla de cálculo. Aunque hoy son esencialmente reliquias de museo, estos dispositivos tan simples mostraron su valor a lo largo de más de tres siglos de uso, e incluso, se las empleo durante las misiones espaciales.

A pesar de que se considera que la primera calculadora fue obra del profesor alemán Wilhelm Schickard, construida en 1623, la máquina mecánica para calcular más famosa y con mayor repercusión histórica resulta aquella desarrollada por Blaise Pascal, llamada Pascalina. Se construyó en 1642, cuando Pascal sólo contaba con diecinueve años de edad, y consiste en una calculadora mecánica que opera mediante ruedas dentadas y pesas. Para sumar y restar, la Pascalina gira las ruedas, registrando así los valores, y utiliza las pesas para realizar la propagación del acarreo de una rueda a otra.

Los tres siglos siguientes a la aparición de la primera calculadora mecánica han visto la continua sucesión de ideas y máquinas. Cada nueva realización se ha ido apoyando, de un modo u otro, en las ya existentes. El ser humano es muy bueno imitando, pero realmente innova muy poco. Es la larga acumulación de variaciones lo que produce el progreso. Sin embargo, es muy posible que las aportaciones más trascendentes se deban a Charles Babbage, un matemático e ingeniero que intentó encontrar un método para realizar cálculos empleando una máquina. En aquella época, como ahora, se requería conocer el valor de algunas funciones matemáticas trascendentes, en cuyo caso se consultaban tablas elaboradas por personas que se dedicaban a ello. Sin embargo, las tablas eran propensas a errores, y eran estos últimos los que motivaron la visión de Babbage. Su proyecto inició en 1820, pretendiendo la construcción de una máquina con la capacidad de obtener los valores numéricos para funciones polinomiales, de manera automática, usando diferencias finitas[214].

El trabajo comenzó en 1820, con la creación del proyecto de una máquina calculadora capaz de obtener automáticamente valores numéricos de funciones polinómicas usando los cocientes de diferencias finitas. Las primeras piezas de dicha máquina se mostraron en junio de 1822. Sin embargo, aún cuando la construcción del modelo más avanzado de la máquina de diferencias no había concluido, Babbage comenzó el desarrollo de una nueva máquina, que hoy en día es considerada su mayor contribución a la computación. La denominaba máquina analítica y es sorprendente la cantidad de similitudes que guarda con las computadoras modernas. Utilizando tarjetas perforadas, debería ser capaz de contener datos que permitieran controlar una secuencia de operaciones y, como una aplicación aún mas sencilla pero igualmente importante, podía almacenar datos. En otras palabras, se trataba de la primera máquina programable concebida. Sin embargo, ninguna de sus máquinas llegó a funcionar plenamente. Sus aportaciones se consideran la base de la computación moderna. Con frecuencia se argumenta que fueron las limitaciones tecnológicas de la época las que le impidieron ver concluida su obra.

Pero, en el bicentenario de su nacimiento, el Museo de Ciencias de Londres construyó íntegramente una «Máquina de Diferencias No. 2», para conmemorar el 200 aniversario de su nacimiento. La construcción estaba basada en los planos de Babbage, utilizando exclusivamente técnicas y materiales disponibles a mediados del siglo XIX. La máquina funcionó perfectamente tras subsanar un pequeño número de errores evidentes encontrados en los planos dejados por Babbage[214].Si las técnicas y materiales de la época no fueron realmente limitantes, es muy probable que una de las causas que impidieron a Babbage finalizar sus máquinas haya sido su perfeccionismo. Por una parte, la contratación de un perfeccionista y lento artesano, que carecía de interés en el desarrollo de la máquina y solo buscaba obtener el dinero [137]. Por otra parte, Babbage mismo era conocido por un carácter perfeccionista [57]. Babbage trataba de construir la mejor de las máquinas que su cabeza podía concebir, lo que le obligaba a abandonar por completo la idea anterior previa, independientemente del grado de desarrollo que hubiera conseguido. Como evidencia, puede considerarse un comentario del matemático John Fletcher, sobre una visita que le hizo a Babbage:

> Una de las memorias mas tristes de mi vida es la visita al célebre matemático e inventor, el Señor Babbage. Aunque estaba muy avanzado en edad, su mente seguía tan vigorosa como siempre. Me llevó a recorrer sus talleres. En la primera sala vi las piezas de la Máquina Calculadora original, que muchos años antes se habían mostrado en estado incompleto e incluso habían tenido algún uso. Le pregunté sobre su estado actual. “No la he terminado porque mientras trabajaba en ella, se me ocurrió la idea de mi Máquina Analítica, que haría todo lo que era capaz de hacer la otra y mucho más. De hecho, la idea era tanto más simple que hubiera requerido más trabajo completar la máquina calculadora que diseñar y construir la otra en su totalidad, así que centré mi atención en la Máquina Analítica”. Después de unos minutos de charla pasamos al siguiente taller donde me mostró y explicó el funcionamiento de los elementos de la Máquina Analítica. Le pregunté si podía verla. "Nunca la he completado", dijo, "porque se me ocurrió la idea de hacer lo mismo con un método diferente y mucho más eficaz, y esto hizo inútil seguir las antiguas líneas". Luego pasamos a la tercera habitación. Había fragmentos dispersos de mecanismos, pero no vi rastro de ninguna máquina en funcionamiento. Me acerqué al tema con mucha cautela y recibí la temida respuesta: "Aún no está construido, pero estoy trabajando en ella y me tomará menos tiempo construirla entera de lo que supondría completar la Máquina Analítica desde el estado en el que la he dejado". Me despedí del anciano con el corazón apesadumbrado [108].

Construir una máquina de cálculo requiere de un gran trabajo técnico y los primeros intentos demuestran que no fue un desafío simple. Problemas como los materiales, tamaños y formas de cada pieza, así como la función de cada una de ellas dentro de un plan maestro requiere de mucha imaginación y experimentación. Pero es solo una parte de la historia. Cada mecanismo debe estar diseñado para cumplir con una tarea específica y el resultado de distintas tareas pequeñas debe ser la solución a un problema. La relación entre estas tareas pequeñas, el orden en el que deben realizarse y el objetivo que cumplen trata de imitar el proceso del pensamiento humano, de manera que la lógica resulta una herramienta y guía indispensable tanto para el diseño como para la operación de las máquinas de calcular o *computadoras*, como se les designa hoy en día.

Sin embargo, en la época de Babbage, la palabra computadora se empleaba para hablar de una persona y no a una máquina. Hacía referencia alguien dedicado a realizar tediosos cálculos manuales, tarea ejecutada en su mayoría por mujeres. Una evidencia de este hecho aparece en la revista Edinburgh Weekly Journal, de 1731, en donde se aconsejaba a las mujeres casadas jóvenes que se informaran sobre el ingreso de sus maridos y «fueran buenas computadoras, y se mantuvieran dentro del mismo» [29]. También era común encontrar avisos en los que se buscaban «computadoras» para trabajar en empresas y departamentos gubernamentales.

Una de las «computadoras humanas» mejor conocidas era la hija del poeta George Gordon Byron, Ada Lovelace. Ada estudiaba con el famoso lógico Augusto de Morgan. Una de las historias que dan cuenta de sus importantes habilidades fue registrada por la esposa de De Morgan. En aquella ocasión, Babbage estaba dando una demostración de la operación de su máquina de diferencias, en casa de la familia De Morgan:

> Recuerdo muy bien acompañarla a ver la maravillosa Máquina Analítica del Señor Babbage. Mientras otros visitantes miraban el funcionamiento de este hermoso instrumento con el tipo de expresión, y me atrevo a decir el tipo de sentimiento, que se dice que algunos salvajes mostraron al ver por primera vez un espejo o al oír un arma, si es que, de hecho, tenían una idea muy clara de su maravilla: la señorita Byron, joven como era, entendió su funcionamiento y vio la gran belleza de la invención[202].

Lovelace y Babbage trabajaron juntos. El inventor llegó a comentar que ella entendía y podía explicar mucho mejor que él las ventajas de su propia máquina. La gran ventaja del invento de Babbage respecto a otros intentos anteriores era su capacidad de realizar cálculos contando y podía emplear la lógica para definir sus operaciones. Ada Lovelace no solo fue capaz de comprender el funcionamiento, además pudo introducir ideas del todo novedosas que le permitieron programar la máquina, entre ellas el concepto de bucle, algo elemental en la programación actual. Lamentablemente, Lovelace falleció de cáncer, a los treinta y seis años, por lo que sus contribuciones se vieron limitadas, algo que seguramente también afectó el desarrollo de las máquinas de Babbage.

2.2. Álgebra booleana y compuertas lógicas

George Boole fue otra persona que también colaboro con De Morgan. Boole afirma que su mas importante aportación vino a él en forma de lo que llamó «una revelación», según la cual, todo razonamiento humano tendría una forma algebraica de escribirse. De ser así, todos los problemas lógicos se podrían reducir a ecuaciones. Siguiendo esta idea, se dedicó a definir las propiedades de la lógica necesarias para tratar variables binarias mediante el álgebra. En el libro «Una investigación de las leyes del pensamiento sobre las que se fundamentan las teorías matemáticas de la lógica y probabilidad»[35], publicado en 1854, aparecen los apartados más significativos de lo que hoy en día se conoce como álgebra booleana. Se trataba de un sistema simple que usa los elementos del álgebra elemental restringida a dos posibles valores: la Nada, representada por el número cero, y el Universo, al que identificaba con el uno. Dentro de este sistema, cada proposición tiene un valor de verdad. Si su valor de verdad no puede determinarse, no se le considera una proposición lógica.

2.2.1. Lógica binaria

Un sistema en el que las variables pueden adoptar únicamente alguno de dos posibles valores discretos, como cierto o falso, y en el que las operaciones entre variables representan operaciones lógicas se denomina *lógica binaria*. El nombre de los dos valores puede variar, pero la convención mas simple es 0 y 1. Esta representación tan simple proporciona un método sencillo para realizar el análisis de la operación lógica de cualquier sistema que pueda operar entre dos estados, como los tubos de vacío, los relevadores eléctricos, que son componentes de máquinas para calcular o cumplen esa misma función dentro de otras máquinas. Como en toda estructura algebraica, es necesario definir las operaciones del álgebra booleana. Si las letras x, y y z representan variables lógicas, se definen las siguientes operaciones lógicas básicas[180]:

1. *Operación AND*. El producto lógico está relacionado con la intersección de conjuntos ($\cap$) y la conjunción de proposiciones en lógica ($\wedge$). Se representa con un punto, o bien, yuxtaponiendo directamente a los valores sobre los que opera, prescindiendo del operador, como suele hacerse con el producto ordinario: $x \cdot y = z$ o $xy = z$. Ya que x, y y z son variables binarias, solo pueden tomar valores entre cero y uno, por lo que resultará que $z = 1$ si y sólo si $x = 1$ y $y = 1$; de lo contrario, $z = 0$, por lo que también se le denomina.
2. *Operación OR*. La suma lógica está relacionada con la unión ($\cup$) de conjuntos y con la disyunción ($\vee$) de proposiciones. Se representa con un signo más: $x + y = z$. Significa que $z = 1$ si $x = 1$ o si $y = 1$ o si $x = 1$ y $y = 1$. Si $x = O$ y $y = 0$, entonces $z = 0$.
3. *Operación NOT*. La inversión lógica corresponde a la operación de complementación de conjuntos(') y a la negación de una proposición en lógica($\neg$). Se representa con un apóstrofo: $x' = z$. Esta operación, puede representarse intuitivamente considerando que z es lo opuesto a x, por lo que si x toma como valor al 1, z será el 0 y viceversa.

Pese a que la idea original del álgebra booleana era el tratamiento de variables binarias, es posible definir formalmente un álgebra booleana con una cantidad arbitraria de elementos. Un conjunto B, con dos operadores binarios, + y $\cdot$, tales que cada uno posee las propiedades de cerradura, existencia de elemento identidad, conmutatividad, distributividad respecto al otro operador, existencia del inverso y, finalmente, la existencia de al menos dos elementos iguales en B es un álgebra booleana [131]. Así, si se desea determinar si cierto conjunto posee la estructura de álgebra booleana, es necesario especificar el conjunto B, es decir, conocer a sus elementos y las reglas de operación correspondientes a los operadores binarios. Con esta información podrá evaluarse si B, junto con los dos operadores, cumple los postulados. Esto tiene como consecuencia que será posible definir diferentes álgebras booleanas. La mas simple de todas posee únicamente dos elementos, es decir, $B = \{1, 0\}$, y los dos operadores binarios, + y $\cdot$ se definen de acuerdo con las siguientes tablas:

Como puede verse, se trata de las mismas reglas para las operaciones AND, OR y NOT, respectivamente. Es posible demostrar que el conjunto $B = \{1, 0\}$ satisface los postulados, con lo que el álgebra booleana queda definida de manera formal y, al mismo tiempo, se establece la equivalencia a la lógica binaria.

2.2.2. Funciones booleanas

Una función booleana $f : \{0, 1\}^n \rightarrow \{0, 1\}$ es una relación entre variables binarias. Como puede observarse, la entrada tiene n dimensiones, mientras que la salida es unidimensional. Para que la función booleana esté bien definida, debe construirse una lista en la que aparezcan todas las posibles combinaciones de ceros y

Tabla 2.1: Tablas de verdad para el producto lógico (·), la suma lógica (+) y la inversión (')

x	y	$x \cdot y$
0	0	0
0	1	0
1	0	0
1	1	1

x	y	$x + y$
0	0	0
0	1	1
1	0	1
1	1	1

x	x'
0	1
1	0

unos que puedan construirse con los n valores de entrada, asociadas al valor que le asigna f a cada caso. Esta representación recibe el nombre de *tabla de verdad*. Debido a que la cantidad de posibles entradas está determinada por el número de combinaciones de dos valores, el número de filas en la tabla será 2^n. Por otra parte, cada una de estas combinaciones corresponde a la expresión binaria de un número, con n entradas es posible construir los números de 0 hasta $2^n - 1$. Así, una función, como F, que puede definirse en términos de las tres operaciones lógicos, tiene también una tabla de verdad asociada, como es muestra en la Tabla 2.2. Para este caso, al tener tres entradas, existen ocho combinaciones posibles de ceros y unos ($2^3 = 8$)

Tabla 2.2: Tabla de verdad para $F = x + y'z$. Como puede notarse, F es igual a cero en los casos en que x, y y z son todos nulos.

Entrada x	Entrada y	Entrada z	Valor decimal	Salida F
0	0	0	0	0
0	0	1	1	1
0	1	0	2	0
0	1	1	3	0
1	0	0	4	1
1	0	1	5	1
1	1	0	6	1
1	1	1	7	1

La tabla de verdad es una forma exhaustiva de definir a la función, ya que incluye a cada entrada. Esto es posible debido a que n es un número natural. Es posible, por otra parte, encontrar una expresión algebraica que pueda contener la misma información. En el ejemplo anterior, la primera impresión podría ser que la salida fue asignada de manera completamente arbitraria, sin embargo, esta también puede encontrarse a partir de los valores de entrada si se efectúan las operaciones $x + y'z$. La representación de una función en forma de tabla de verdad es única [63]. Sin embargo, diferentes expresiones algebraicas pueden producir los mismos valores en la tabla de verdad. En tal caso, se dice que son expresiones equivalentes. En algunas ocasiones es posible manipular algebraicamente una expresión booleana para transformarla en otra equivalente, pero más simple. Simplificar no solo ayuda a reducir el espacio que una expresión puede ocupar, también permite una análisis más ágil y sencillo, entre otras ventajas.

Existe una representación más para las funciones booleanas, particularmente útil porque permite visualizarlas de forma gráfica, de manera muy cercana a lo que resultaría de una implementación real, que se denomina *diagrama de circuito* [180]. Dentro de estos diagramas, las variables de entrada se suelen colocar del lado izquierdo. La información codificada en estas variables sufrirá algunas transformaciones, producidas por las operaciones básicas AND OR y NOT. Cada operación lógica tiene asignado un símbolo (o bloque) que la representa en el diagrama. Las operaciones binarias requieren de dos valores de entrada, que estarán del lado derecho del símbolo que representa a la operación. Al tener varias operaciones relacionadas, es útil indicar esta relación en términos del flujo de información, la salida de una operación puede usarse como entrada en otra operación. Un ejemplo de diagrama puede observarse en la figura 2.1. Los dispositivos que realizan operaciones lógicas en una implementación real reciben el nombre de *compuertas*. Al terminar el recorrido por el circuito, la entrada se ha transformado en la variable binaria F, la salida del circuito, que se coloca del lado izquierdo.

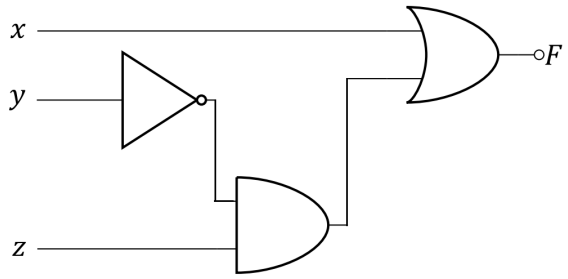


Figura 2.1: Diagrama de circuito para la función de ejemplo $F = x + y'z$. A la izquierda se encuentran los valores de entrada x, y y z. Puede observarse la forma en la que deben conectarse las compuertas lógicas para que la salida del circuito resulte el valor de F

Se considera que dos circuitos son equivalentes cuando, a partir de su operación, se extraen los mismos valores para las mismas entradas, en todas las posibles combinaciones binarias que pueden emplearse como variables de entrada. Aunque no se trata de una regla estricta, en general, se prefieren expresiones y circuitos lo más sencillo posible, para lo cual se recurre a la manipulación de la expresión a través del álgebra booleana o métodos gráficos. Simplificar no siempre es tan sencillo como sugiere la idea de reducir expresiones algebraicas, en muchas ocasiones se requiere incluso del empleo de programas de computadoras con programas especializados para poder realizar esta tarea [289].

2.2.3. Formas canónicas y estándar

La tarea de comprobar la equivalencia entre funciones booleanas aparece con frecuencia en el análisis de circuitos. Debido a ello, se han diseñado algunas estrategias para realizarla. Entre ellas, destaca el estudio de las llamadas *formas canónicas*[109], que son funciones lógicas expresadas de manera algebraica en las que se combinan las tres operaciones básicas. Para el complemento o negación, puede observarse que toda variable binaria puede aparecer dentro de una expresión booleana ya sea de manera ordinaria, llamada también *forma normal* (x), o bien, negada, la denominada *forma complementada* (x'). Para ejemplificar esto, puede considerarse el caso de dos variables x y y entre las que actúa el operador AND. Debido a que cada una de las variables aparecerá una sola vez, ya sea en la forma normal o en la complementada, se tendrán cuatro combinaciones posibles $x \cdot y$, $x' \cdot y$, $x \cdot y'$ o $x' \cdot y'$. Cada uno de estos productos recibe el nombre de *minitérmino*, también llamado *producto estándar*. Para el caso general en que la función booleana tiene una entrada de tamaño n, es decir, de la forma x_1,..,.x_n, un minitérmino se formará mediante el producto lógico de cada una de las n variables que forman la entrada, aunque algunas de ellas aparecerán en la forma complementada. El apóstrofo de complemento se colocará sobre la variable si el bit que corresponde al número binario es un cero, de acuerdo con:

$$\phi_i(x_i) = \begin{cases} x_i & \text{si } a_i = 1 \\ x_i' & \text{si } a_i = 0 \end{cases} \tag{2.1}$$

Cada minitérmino se designa con un símbolo de la forma m_j, donde el índice j es la expresión decimal correspondiente al número binario del minitérmino en cuestión.

Tabla 2.3: Minitérminos para tres variables binarias [180].

x	y	z	Términos	Designación
0	0	0	$x' \cdot y' \cdot z'$	m_0
0	0	1	$x' \cdot y' \cdot z$	m_1
0	1	0	$x' \cdot y \cdot z'$	m_2
0	1	1	$x' \cdot y \cdot z$	m_3
1	0	0	$x \cdot y' \cdot z'$	m_4
1	0	1	$x \cdot y' \cdot z$	m_5
1	1	0	$x \cdot y \cdot z'$	m_6
1	1	1	$x \cdot y \cdot z$	m_7

Cada maxitérmino es una suma lógica de tres variables, donde cada variable se complementa si el bit correspondiente del número binario es 1 y no se complementa si es 0

De manera análoga, es posible formar un término OR con n variables, en donde cada una de ellas puede estar en la forma normal o complementada, de manera que se obtienen 2^n combinaciones, que reciben el nombre de *sumas estándar* o *maxitérminos*. Las ocho combinaciones posibles para el caso de tres variables, junto con su designación simbólica. se muestran en la Tabla 2.4. Para obtener los 2^n maxitérminos que pueden formarse a partir de n entradas, se forma una suma lógica (OR) con las n variables.

La variable estará en la forma complementada si el bit correspondiente del número binario es uno y no se complementa si es cero [180]. Los maxitérminos y minitérminos son complemento uno del otro. La razón

Tabla 2.4: Maxitérminos para tres variables binarias [180]

x	y	z	Términos	Designación
0	0	0	$x+y+z$	M_0
0	0	1	$x+y+z'$	M_1
0	1	0	$x+y'+z$	M_2
0	1	1	$x+y'+z'$	M_3
1	0	0	$x'+y+z$	M_4
1	0	1	$x'+y+z'$	M_5
1	1	0	$x'+y'+z$	M_6
1	1	1	$x'+y'+z'$	M_7

principal para mencionar a los minitérminos aquí es el hecho de que toda función booleana se puede expresar como una suma de minitérminos [63]. Al igual que en el caso de un mal estudiante rellena las mismas letras del compañero de al lado y obtiene su misma nota sin necesidad de entender o leer las preguntas, es posible reducir todos los cálculos implicados por una función a observar los sitios en los que debe obtenerse un uno y dejar todos los demás en cero. La expresión algebraica para la función booleana se obtendrá a partir de la tabla de verdad utilizando miniterminos, realizando un OR sobre todas las combinaciones que producen un uno. De manera similar, toda función booleana también puede expresarse como el producto de maxitérminos [63]. Si se parte de la tabla de verdad de la función, todo lo que hará falta es formar un AND con todas las combinaciones de las variables que produzcan un 0. Las funciones que están escritas como la suma de minitérminos o producto de maxitérminos se dice que se están en su *forma canónica*[180]. Aunque es posible obtenerlas a partir de la tabla de verdad, es poco probable que sean las expresiones mas compactas debido a que, por definición, cada maxitérmino o minitérmino emplea en su expresión todas las variables, ya sea en la forma complementada o en la forma normal. Por ello, existen otras formas para expresar a las funciones booleanas. La *forma estándar* permite que los términos con los que se construye a la función contengan una cantidad arbitraria de literales. Al igual que en las formas canónicas, se tiene la posibilidad de formar la expresión a partir de la suma de productos, o a partir del producto de sumas[71]:

- **Suma de productos:** contiene términos AND (términos de producto) con una o más literales cada uno. La suma denota el OR de esos términos.
- **Producto de sumas**: contiene términos OR (términos de suma), que pueden tener cualquier cantidad de literales. El producto denota el AND de esos términos.

Evidentemente, existen funciones que no necesariamente estarán expresadas en alguna de las formas, canónica o estándar, pero es posible expresarla como suma de productos o producto de sumas mediante manipulación algebraica.

2.2.4. Compuertas lógicas digitales

Es posible construir 16 funciones boolenas distintas a partir de dos variables, usando todas las posibles combinaciones de ceros y unos (ver Tabla 4.15). Entre ellas se encuentra la suma y el producto, $x \cdot y$ y $x+y$.

Tabla 2.5: Funciones booleanas en dos variables. En realidad se trata de todas las combinaciones de ceros y unos que se pueden construir con cuatro bits [252]

x	y	f_0	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9	f_{10}	f_{11}	f_{12}	f_{13}	f_{14}	f_{15}
0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
0	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
1	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1

A cada una de estas funciones se le asigna un nombre y un operador binario, a pesar de que cada una de ellas se puede expresar en términos de los operadores básicos AND, OR y NOT, como puede verse en la Tabla 5.1. Estas 16 funciones mencionadas pueden clasificarse de la siguiente forma:

- Constantes: devuelven un valor constante. Son dos, la que siempre vale cero y la que siempre devuelve un uno.
- Unarias: se trata de operadores que actúan solo sobre una de las variables, independientemente de los valores que tenga la otra. Son cuatro, dos definen el complemento, negación o NOT, y dos la transferencia, también llamadas búfer.
- Binarias: se trata de operadores que dependen de dos variables. Son ocho operaciones diferentes, entre ellas las ya conocidas OR y AND, además de NAND, NOR, XOR exclusivo, equivalencia, implicación e inhibición. Estas dos últimas aparecen repetidas, por lo que se trata de diez en total.

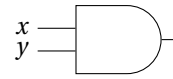
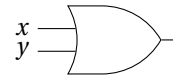
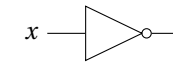
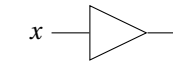
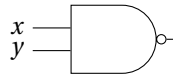
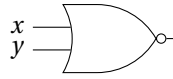
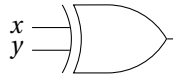
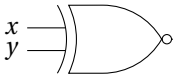
Tabla 2.6: Expresiones booleanas para las 16 funciones de dos variables[218]

Función	Nombre	Símbolo	Descripción
$f_0 = 0$	Nula		Constante binaria 0, siempre es cero
$f_1 = xy$	Conjunción	$x \cdot y$	Operador AND: x y y
$f_2 = xy'$	Inhibición de y	x/y	x, pero no y
$f_3 = x$	Transferencia de x		Búfer para x, siempre es x
$f_4 = x'y$	Inhibición de x	y/x	y, pero no x
$f_5 = y$	Transferencia de y		Búfer para y, siempre es y
$f_6 = xy' + x'y$	Disyunción exclusiva	$x \oplus y$	Operador XOR: x o y, no ambas
$f_7 = x + y$	Disyunción	$x + y$	Operador OR: x o y
$f_8 = (x + y)'$	Complemento de disyunción	$x \downarrow y$	Operador NOR
$f_9 = xy + x'y'$	Equivalencia	$(x \oplus y)'$	x es igual a y
$f_{10} = y'$	Complemento de y	y'	Negación de y
$f_{11} = x + y'$	Implicación de x	$x \subset y$	Si y, entonces x
$f_{12} = x'$	Complemento de x	x'	Negación de x
$f_{13} = x' + y$	Implicación de y	$x \supset y$	Si x, entonces y
$f_{14} = (xy)'$	Complemento de conjunción	$x \uparrow y$	Operador NAND
$f_{15} = 1$	Unidad		Constante binaria 1, siempre es uno

Estos operadores pueden emplearse para construir circuitos en implementaciones prácticas, pero su elección dependerá de las propiedades matemáticas que cumplan, la capacidad de que el operador pueda realizar, por si mismo o junto con un pequeño grupo de otros operadores, expresiones para todas las funciones booleanas, la viabilidad, considerando el costo de fabricarlo físicamente, el número de entradas que puede admitir[233]. Los operadores de implicacióne e inhibición son usados en lógica, pero no son conmutativas ni asociativas, por lo que carecen de utilidad práctica. De la misma forma, se descartan las funciones booleanas constantes, puesto que su valor es independiente de la entrada asignada, sólo pueden ser iguales a 1 o a 0. Es importante considerar que algunas de las funciones restantes se repiten, por lo que, en total, se tienen ocho compuertas estándar, sus nombres símbolos empleados tradicionalmente y expresiones algebraicas se pueden consultar en la Tabla 2.7. En cada caso las entradas se indican mediante las variables x y y, en tanto que la salida se indica mediante la variable binaria F.

Los símbolos y nombres mostrados en la tabla son de uso corriente en la literatura sobre electrónica. El hecho de que algunos de ellos guarden cierto parecido obedece a que tienen cierta relación, que se explica

Tabla 2.7: Compuertas lógicas digitales típicas [164]

Nombre	Símbolo	Función
AND		$F = xy$
OR		$F = x + y$
NOT		$F = x'$
BUFFER		$F = x$
NAND		$F = (xy)'$
NOR		$F = (x + y)'$
XOR		$F = x \oplus y$
XNOR		$F = (x \oplus y)'$

a continuación. Puede tomarse como punto de partida al operador $F = x$. Tras ser aplicado, la salida resulta igual a la variable de entrada.

Aunque una forma de verlo es considerar que se trata de la función es la identidad, a la que se suele describir en términos coloquiales como una función no hace nada al argumento, desde el punto de vista informático se debe considerar que la *transferencia* de información es una operación en si misma, la variable, x o y, se traslada a través de la compuerta sin sufrir modificaciones. La manera en que f_3 consigue esto es copiando la entrada de x al tiempo que ignora los valores de y. La función f_5 funciona de manera análoga, ignorando los valores correspondientes a x y replicando los de y. Puede parecer algo obvio y, sin embargo, debe considerarse que si una señal comienza con un determinado voltaje que le asigna un valor uno, cinco volts por ejemplo. Después de transmitirse a través de conductores con resistencia distinta de cero o después de activar varias otras puertas lógicas, el nivel de voltaje puede caer y acercarse peligrosamente al umbral que define el valor alto, con lo que se perdería información [84]. Para que las funciones f_3 y f_5 realicen la tarea de transmitir datos sin pérdidas, es necesario regenerar las señales lógicas almacenando el valor correspondiente de manera temporal en alguna parte, un espacio físico que recibe el nombre de *búfer*. Por extensión, es frecuente dar este mismo nombre a la operación. Se representa con un triangulo apuntando en la dirección en la que viaja la información. La operación de complemento (NOT) $F = x'$ se representa agregando un pequeño círculo, llamado burbuja, en la punta de este triángulo. Tanto f_{10} como f_{12} puede servir para este propósito, dependiendo de cual variable sea negada.

Las tablas de verdad correspondientes a transferencia y negación son complementarias. Esto puede emplearse para nombrar a otros operadores. A partir de la Tabla 5.1 puede observarse que la función f_{14} es el complemento de f_1, que está asociada al operador AND, por lo que se suele nombrar NOT AND, o de manera abreviada, NAND. De la misma forma, f_8 es el complemento de f_7, función asociada con el operador OR, por lo que recibe el nombre de NOT OR, abreviado NOR. El símbolo elegido para estos operadores, que son el complemento de otros, es agregar la burbuja de negación al símbolo correspondiente a la operación complementada. La operación de conjunción exclusiva está dada por f_6 devuelve 1 cuando una y solo una de las variables de entrada es uno, y su nombre abreviado es XOR. Debido a su parecido con OR, el símbolo de XOR es el de OR con una curva añadida del lado de las entradas. La función f_9 se denomina *equivalencia* porque devuelve como resultado un uno cuando ambas entradas tienen el mismo valor. Debido a que f_6 y f_9 son complementarias, se suele llamar NOR exclusivo a la función equivalencia, abreviado XNOR. Esto también motiva la representación con un símbolo de OR exclusivo seguido de una la burbuja en el lado de salida.

El párrafo anterior ha mostrado que las operaciones tienen nombres y símbolos semejantes porque es-

tán relacionadas entre si. El hecho de que no sean independientes hace posible definir a unos operadores binarios en términos de otros. El hecho de haber mencionado en primer lugar a los operadores AND, OR y NOT puede producir la impresión de que son mas fundamentales que los otros pero, en realidad, es posible partir del operador NOR y definir en términos de él a AND, OR y NOT [156]. La principal razón por la que se presenta de esta forma es porque los conceptos lógicos como conjunción (y), disyunción (o) y negación (no) son conocidos y se emplean comúnmente para expresar ideas lógicas aunque, de hecho, las compuertas NAND y NOR son comúnmente empleadas como bloques lógicos elementales,debido a que pueden construirse utilizando una cantidad mínima de componentes reduciendo la complejidad, costo y tamaño de los dispositivos resultantes [139].

Si bien, las compuertas discutidas hasta este momento tienen un bit o dos de entrada y siempre uno de salida, se pueden combinar en circuitos que admitan n de bits de entrada, para lo cual se requiere de compuertas que sean conmutativas y asociativas[180]. Dicho de manera más precisa, una compuerta calcula una función que asigna a los estados de entrada los estados de salida, pero, para entradas de longitud n, los circuitos tienen variables boolenanas $x_1, ..., x_n$ que pueden tomar, cada una, el valor 0 o 1. Los circuitos, por tanto, pueden describirse como una secuencia $G_1, ..., G_s$, de compuertas[291]. Cada compuerta G_i tiene dos entradas $E_{i,1}$ y $E_{i,2}$ que deben estar entre las compuertas anteriores $G_1, ..., G_{i-1}$ y la actual. La compuerta G_i aplica una operación binaria op_i a sus entradas. Si las entradas a una puerta G_i se realizan mediante las funciones $g_{i,1}$ y $g_{i,2}$, entonces la compuerta G, equivale a evaluar la función $g_i(a) = g_{i,1}(a) op_i g_{i,2}(a)$. Las variables de entrada y las constantes booleanas 0 y 1 también pueden considerarse funciones.

Es posible construir compuertas con m bits de salida. Un circuito realiza la función $f = (f_1, ..., f_m) : \{0,1\}^n \rightarrow \{0,1\}^m$ si cada función componente f_j se realiza mediante una entrada o una compuerta. En este tipo de arreglos es importante asignar y respetar el orden de los bits. Para indicar que un cierto bit se utiliza como entrada en dos o más compuertas, se suele hablar de una «compuerta» de copia, llamada FANOUT o COPY, que toma un bit y lo lleva a dos bits:

$$COPY : a \rightarrow (a, a) \tag{2.2}$$

En una implementación real, bastaría con tomar un conectar los diferentes puntos con un cable. También puede ser necesario intercambiar la posición de los bits, lo cual se indica mediante la compuerta de cruce, llamada CROSSOVER o SWAP, que intercambia los valores de dos bits:

$$SWAP : (a, b) \rightarrow (b, a) \tag{2.3}$$

Nuevamente, en una implementación real, cruzar físicamente los cables hace las veces de esta compuerta. Las representaciones de circuito para estas dos compuertas se muestran en la figura. La designación de estas compuertas es necesaria en la demostración de que un cierto conjunto de compuertas es suficiente para construir para cualquier función booleana. Las compuertas elementales descritas anteriormente se pueden unir para implementar cualquier cálculo complejo. Debe notarse que las compuertas elementales presentadas anteriormente no son todas independientes. Por ejemplo, AND, OR y NOT están relacionados por las identidades de De Morgan, y también es fácil comprobar que la compuerta XOR se puede construir mediante las compuertas AND, OR y NOT, de acuerdo con x XOR $y = (x + y)((x')(y'))$

2.2.5. Compuertas universales

Entre las consideraciones que deben hacerse para llevar a la práctica la realización de una computadora está el requisito de que su construcción se haga a partir de un número pequeño de compuertas. Si toda función booleana puede implementarse usando únicamente compuertas de un conjunto, el conjunto recibe el nombre de *conjunto funcional completo*[245] o *conjunto universal de compuertas para la computación clásica*[21].

Cualquier función $f : \{0,1\}^n \rightarrow \{0,1\}^m$ se puede construir a partir de las compuertas elementales AND, OR, NOT y FANOUT. Por tanto, se dice que éstas constituyen un conjunto universal de compuertas para la computación clásica. La prueba de este hecho[127] se basa en la consideración de que la función de m bits $f : \{0,1\}^n \rightarrow \{0,1\}^m$ es equivalente a m funciones booleanas de un bit, dadas por $f_i : \{0,1\}^n \rightarrow \{0,1\}, i = 1, 2, ..., m$, donde $f = (f_1, f_2, ..., f_m)$. Una forma de calcular estas funciones booleanas $f_i(a)$, $a = (a_n, a_{n-1}, ..., a_1)$ es a partir de los miniterminos, definidos previamente. Para hacerlo, se parte de la observación trivial de que $OR(x_1, OR(x_2, x_3)) = 0$ si y solo si $x_1 = x_2 = x_3 = 0$. Esto puede generalizarse fácilmente a cualquier cantidad de variables. Para abreviar, es conveniente usar la notación: $OR(x_1, OR(x_2, x_3)) = x_1 \vee x_2 \vee x_3$. De manera que, a partir de la función OR en dos variables, es posible construir una función como $x_1 \vee x_2 \vee ... \vee x_n$, que toma

valor 0 *si y solo si* todas las variables x_i son nulas. De manera similar, a partir la función $\wedge$ en dos variables, es posible construir la función $x_1 \wedge x_2 \wedge ... \wedge x_n$, que toma valor 1 *si y solo si* todas las variables son 1. Luego, se define la función M_a para cada $a = (a_n, ..., a_1)$

$$m_a(x_1, ..., x_n) = \phi_1(x_1) \wedge \phi_2(x_2) \wedge ... \wedge \phi_n(x_n) \tag{2.4}$$

donde

$$\phi_i(x_i) = \begin{cases} x_i & \text{si } a_i = 1 \\ \neg x_i & \text{si } a_i = 0 \end{cases} \tag{2.5}$$

Por lo que es claro que las funciones m_a pueden construirse utilizando únicamente $\wedge$ (compuertas AND) y $\neg$ (compuertas NOT). Además, $m_a(x_1, ..., x_n) = 1$ si y sólo si cada $\phi_i(x_i) = 1$ para cada i, lo cual ocurre si y solo si $x_i = a_i$. Es decir $m_a(x) = 1$ si y solo si $x = a$, es decir m_a es la *función característica*, Por lo tanto, es suficiente calcular los minitérminos y realizar las compuertas OR para obtener $f_i(a)$.

$$f_i(a) = m_{a_1} \vee m_{a_2} \vee ... \vee m_{a_k} \tag{2.6}$$

donde los $a_1, a_2, ..., a_k$ son los elementos para los cuales la función toma valor uno. Esta descomposición también requiere la implementación de compuertas FANOUT, debido a que se requieren k copias de la entrada a, ya que cada minitérmino debe actuar sobre ella.

Incluso es posible reducir el número de operaciones elementales. Resulta, por ejemplo, que NAND y FANOUT son un conjunto universal más pequeño[245]. La compuerta OR puede obtenerse de NOT y AND mediante las identidades de De Morgan: x' AND $y' = (x \text{ OR } y)' \rightarrow (x' \text{ AND } y')' = ((x \text{ OR } y)')' = x$ OR y También es posible obtener NOT de NAND y FANOUT (que se usa para copiar la entrada): $x \uparrow x = (x \text{ AND } x)' = x'$ OR $x' = x'$

Una forma de crear una puerta NAND es a través de resistores y transistores, como se muestra en la figura 2.2. En la lógica de resistencia transistor (RTL, por sus siglas en inglés), un bit se establece en 1 si el voltaje es positivo y en 0 si el voltaje es cero [221]. La corriente fluye a través de los transistores si y solo si ambas entradas tienen voltaje positivo ($a = b = 1$). En este caso, la salida tiene voltaje cero. Por el contrario si al menos una de las entradas tiene voltaje cero, no hay flujo de corriente y por lo tanto la salida tiene voltaje positivo, por lo que la salida corresponde a $a \uparrow b$ (a NAND b).

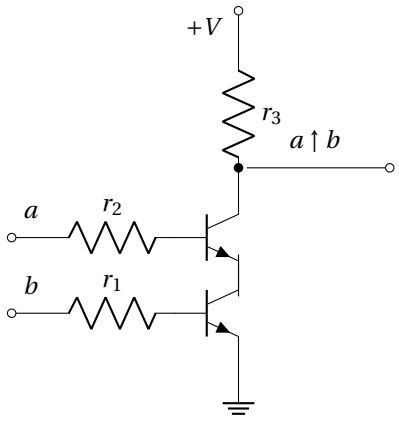


Figura 2.2: Compuerta NAND construida con transistores y resistores. El tipo de circuitos construidos mediante estos componentes se denomina *RTL*, por las iniciales de las palabras inglesas *Resistor, Transistor, Logic.*

Aunque esta implementación se considera de importancia histórica, hoy en día es obsoleta [104]. Primero, fue sustituida por la lógica de transistor-transistor (TTL) y más tarde por esquemas más complejos, como la puerta complementaria MOS, o CMOS, que se emplea en circuitos densamente integrados en los que el calor generado por un número igual de compuertas TTL o RTL no podría disiparse sin medidas costosas. Gracias al advenimiento de técnicas de fabricación modernas, es posible la construcción e interconexión de cientos de diodos, transistores, resistencias y similares en una sola oblea de silicio de unos pocos milímetros cuadrados de área llamados *circuitos integrados.*

En conclusión, para construir un circuito clásico que implemente una función booleana, se requiere de:

- Compuertas de varios tipos, NOT, AND, XOR
- Capacidad de replicar la entrada, una operación llamada FANOUT
- Capacidad de intercambiar dos bits, una operación llamada SWAP
- Un suministro de bits inicializados en estados particulares para forzar una salida específica sin importar la entrada. Esta clase de bit reciben el nombre de *bits auxiliares* o *bits ancilares*
- Un canal físico de comunicación entre las compuertas. Normalmente se refiere a cables.

Aunque introducida desde 1854, el álgebra de Boole tuvo que esperar hasta 1948 para encontrar un área de aplicación. Claude Shannon publicó en ese año su *Teoría matemática de la comunicación*[250], un artículo fundamental, que marca el nacimiento de la teoría de la información en términos de circuitos eléctricos que pueden conmutar entre dos estados, justo el tipo de lógica con la que Boole había estado trabajando. El trabajo de Shannon permitió llevar a la práctica el álgebra booleana, a través de un esquema electrónico. Con ello, se podía usar una pequeña cantidad de compuertas para poder calcular cualquier función booleana. Y dos años después, la primera computadora fue concebida, no como el producto terminado que se conoce y emplea en la actualidad, sino en forma de un esotérico artículo que trataba sobre lógica matemática.

2.3. La máquina de Turing

Un proceso computacional puede requerir de muchas instrucciones para realizarse. Estas instrucciones pueden estar agrupadas en alguna jerarquía, por lo que es necesario realizarlas en un cierto orden. Si alguna de ellas no se ha completado satisfactoriamente, el resto de las instrucciones no pueden realizarse. Esta clase de procesos introduce la noción de *etapa* o *paso*. Si un proceso computacional requiere un solo paso para ejecutarse, la lógica booleana basta para generar la respuesta correcta, con plenas garantías de exactitud lógica. Sin embargo, cuando un proceso computacional requiere múltiples pasos, se prefiere trabajar con un modelo nuevo. Los *autómatas*[11], son dispositivos que tienen la capacidad de identificar *su estado*, es decir, la actividad que están realizando. Al mantener estados durante un cierto tiempo, al autómata es capaz de tener memoria. Los pasos se representan mediante transiciones entre estados distintos.

2.3.1. El programa de Hilbert

El autómata que sirvió de precedente directo a la computación moderna surgió poco tiempo después de la célebre plática «Problemas Matemáticos», de David Hilbert, impartida el 8 agosto de 1900, en el marco del segundo Congreso Internacional de las Matemáticas, en París. En esa conferencia. Hilbert presentó una lista de diez problemas (en una publicación posterior se encontrarían 23) que, a su juicio, eran esenciales para el desarrollo de la matemática del siglo XX. Hilbert había trabajado en el desarrollo de la fundamentación axiomática tanto para la aritmética como para la geometría clásica, tras lo cual comenzó a preguntarse si era posible evitar la contradicción dentro de las matemáticas. Esto debía hacerse de tal forma que no solo se evitaran las contradicciones conocidas, sino que hubiera plenas garantías de que ninguna consecuencia lógica de los axiomas pudiera a contradecir a cualquier otra. La cuestión de la *no contradicción*, es decir, de la consistencia del sistema de axiomas, le llevó a proponer, bajo el apartado correspondiente a problemas fundamentales, el problema de «La compatibilidad de los axiomas aritméticos», que trataba sobre eliminar la posibilidad de contradicción de las matemáticas. Incluye la declaración:

> “Para probar que ellos [los axiomas para la aritmética] no son contradictorios, es decir, que un número finito de pasos lógicos basados en ellos nunca puede conducir a resultados contradictorios[52] ”

Al hablar de «axiomas para la aritmética», Hilbert hace referencia a los axiomas de los números reales. Como el mismo menciona, «Los axiomas de la aritmética son esencialmente nada más que las reglas conocidas de cálculo, con la adición del axioma de continuidad». Se trata de la búsqueda de una demostración de que no es posible demostrar una afirmación y, al mismo tiempo, lo opuesto, dentro del sistema axiomático de la aritmética y mediante razonamientos lógicos. A pesar de lo intuitiva que parezca la cuestión, la demostración que solicitó nunca llegó.

En 1930 Kurt Gödel probó un par de resultados que causaron mucha sorpresa: sus teoremas de incompletitud. El primero afirma que si se toman los axiomas de la aritmética de los números enteros no negativos, con las operaciones de multiplicación y suma, entonces hay un enunciado en el lenguaje formal que no puede probarse como verdadero y no puede probarse como falso por argumentos hechos en el lenguaje formal. Se habla de enunciado *indecidible* en términos del lenguaje formal. Sin embargo, el enunciado en sí es

tal que es verdadero como enunciado sobre los números enteros no negativos, fuera del contexto del sistema de axiomas. La existencia de tal enunciado, imposible de probar en el sistema formal pero verdadero como enunciado sobre los números enteros no negativos, es lo que le da el tiro de gracia a la propuesta de Hilbert. No se puede encontrar una demostración matemática del principio de no contradicción para la aritmética. Para probar el teorema, construyó la afirmación propuesta. Tomó la conocida paradoja del mentiroso, es decir, «Estoy mintiendo» y la convirtió en una declaración en el sistema formal de aritmética. El enunciado que construye Gödel tiene una clara interpretación, desde fuera, de «este enunciado no es demostrable». Resulta que las complejidades lógicas que surgen de la autorreferencia existen incluso en la aritmética. La idea es sencilla. Se hace una lista de lo que es posible probar y, usando una diagonalización, se construye un enunciado que diga: «Este enunciado no está en la lista de enunciados demostrables», lo que equivale a «Estoy mintiendo».

Básicamente la idea de Gödel es la siguiente[203]: toda declaración en el lenguaje formal demostrable o, incluso aquellas carente de todo sentido, en otras palabras, cualquier posible cadena finita de símbolos en el lenguaje formal para la aritmética, está compuesta de paréntesis, conectivas, signos de predicado, términos, cuantificadores y otros signos elementales.

- A cada uno de estos símbolos se le asigna un número entero no negativo, siguiendo ciertas reglas.
- Luego, los números asociados con los signos de la fórmula se utilizan como exponentes de los primeros diez números primos, para multiplicarlos después. El resultado obtenido es el *número de Gödel* que codifica la fórmula en cuestión. Al proceso se le llama *Numeración de Gödel*.
- En el siguiente paso, debe hacerse una lista de los números de código de todos los enunciados que se pueden demostrar a partir de los axiomas de la aritmética, que se puede ver como una función sucesión: $f(1)$ será el primer número en la lista, $f(2)$ el segundo, y así sucesivamente.
- A continuación, se crea una matriz con la que se generará, mediante la diagonalización de Cantor [1], una segunda lista. La primera columna de la matriz contendrá todas las declaraciones en el idioma en el que x aparece como una variable libre $(A_1(x), A_2(x), A_3(x), ...)$. Estos son enunciados que afirman algo acerca de x, verdadero o falso, por ejemplo, $x = x + 0$, $x = x + 2$. Podrían hablar sobre las diversas funciones de la aritmética, por ejemplo, $f(x) = 2$.
- Luego, en columnas paralelas se irá sustituyendo a x por 1, en la siguiente columna por 2, luego por 3. Equivale a una matriz de todos los ejemplos posibles de las declaraciones generales, por ejemplo, $1 = 1 + 0, 2 = 2 + 0$, ..., o, en el ejemplo falso, $1 = 1 + 2, 2 = 2 + 2$... La mayoría de estos ejemplos sería falso; algunos serían ciertos.

[1] El argumento diagonal de Cantor [53] fue utilizado como prueba de que los números irracionales son no numerables. El mecanismo con el que opera se puede explicar de manera simple con un ejemplo. Dada una lista de cinco «palabras» al azar, entendiendo palabra como un conjunto de cinco letras cualesquiera, como VAROS, ALNAR, SORNA, BURDR, AARRE es fácil afirmar que existe al menos una palabra de cinco letras que no está en la lista. Aunque esto parece evidente, una forma de comprobarlo es mediante la construcción de la palabra que sirva de contraejemplo. Esto se puede realizar escogiendo una letra cualquiera que sea distinta a la primera letra de la primera palabra. Aunque cualquiera distinta de V funciona, el mecanismo que se propone es que sea la letra sucesora, W, en este caso. Para la segunda letra, se toma la letra sucesora a la segunda letra de la segunda palabra, M, y se realiza lo mismo con todas las letras. La palabra resultante será distinta de todas las de la lista. En el caso de un número infinito de palabras, en donde cada palabra tiene un número infinito de letras, como:

- **E**NUNLUGARDELAMANCHADECUYONOMBRENOQUIERO...
- U**N**HIDALGODELOSDELANZAENASTILLEROADARGA...
- UN**A**OLLADEALGOMASVACAQUECARNEROSALPICON...

se puede construir una nueva palabra escogiendo la primer letra distinta de la E, la segunda letra distinta de la N, la tercera letra distinta de la A, con lo que se asegura estar cambiando las letras de la diagonal. No puede ser la primera de la lista (son distintas en la primer letra), ni la segunda (difieren en la segunda letra), ni la millonésima (diferirán en la millonésima letra), por lo que toda lista de palabras con infinitas letras omitirá algunas palabras infinitas. La manera en la que Cantor lo presentó consiste en suponer que existe una lista numerada con todos los números irracionales en el intervalo $[0,1]$. Como esta lista se encuentra numerada, los números irracionales estarán ordenados. Puede suponerse que el primero es $q_1 = d_1 \times 10^{-1} + d_2 \times 10^{-2} + d_3 \times 10^{-1} + ...$, el segundo es $q_2 = e_1 \times 10^{-1} + e_2 \times 10^{-2} + e_3 \times 10^{-1} + ...$, el tercero es el segundo es $q_3 = f_1 \times 10^{-1} + f_2 \times 10^{-2} + f_3 \times 10^{-1} + ...$ y así sucesivamente. Luego, al primer número se le agrega una unidad al primer decimal, al segundo número se le agrega una unidad al segundo decimal, y así sucesiva e indefinidamente. Con este mecanismo será posible construir un número que no se encuentra en la lista original, dado por $(d_1 + 1) \times 10^{-1} + (e_2 + 1) \times 10^{-2} + (f_3 + 1) \times 10^{-1} + ...$, porque será distinto al primero en al menos el primer decimal, será distinto del segundo en al menos el segundo decimal, del tercero en al menos el tercer decimal y así sucesivamente [195].

- Se presta atención a los elementos de la diagonal. Leyéndoles de forma descendente, se genera otra lista de declaraciones, algunas verdaderas y otras no, con referencia a la segunda lista, que asignaba números de código a las declaraciones demostrables en aritmética:

  1. «El número de código de la primera declaración en la diagonal no es en la lista maestra de declaraciones demostrables.»
  2. «El número de código del segundo enunciado de la diagonal no está en la lista maestra de enunciados comprobables.»
  3. «El código del tercero...»
  4. «El código del cuarto... »

La clave es que esta nueva lista de declaraciones se construye de manera regular desde la diagonal por lo que es posible escribir una fórmula general utilizando una variable libre, x, es decir, una oración en el lenguaje formal que habla de x. Esa fórmula es: «El número de código de la declaración x en la diagonal no está en la lista maestra de declaraciones comprobables». Captura todas las declaraciones en nuestra nueva lista como instancias. Pero este enunciado general es en sí mismo un enunciado de una variable libre, x, y por lo tanto debe aparecer en nuestra lista como $A_n(x)$ para algún n. Además, Gödel puede encontrar este número, por lo que la prueba es totalmente constructiva.

La lista de todas las declaraciones con x como variable libre que hace Gödel no es falsa, nada se omite. La declaración que se genera con la ayuda de la diagonal está en la lista. Si se evalúa esta afirmación enésima, sustituyendo n por x. Es, según la definición dada, el enunciado «El número de código del enésimo enunciado en la diagonal no está en la lista maestra de enunciados comprobables». Dado que en sí mismo es la afirmación *enésima*, lo que realmente está afirmando es: «No soy demostrable». Es completamente una afirmación sobre funciones bien definidas en la aritmética de los números enteros. Si se pudiera probar en el sistema formal, se contradiría afirmando que no era demostrable. Si fuera posible probar su negación, la negación se contradiría a sí misma. Se concluye que el sistema es inconsistente.

Dentro de su conferencia, Hilbert insistió en que era necesario demostrar cada afirmación y toda resolución propuesta a un problema matemático. En caso de que un problema careciera de solución, entonces debería proporcionarse la demostración de su inexistencia. Como puede notarse, la afirmación de que todo problema debería tener una solución o la demostración de que ella no existía, es, en sí misma, una afirmación matemática y, de acuerdo con su propio esquema, Hilbert estaba obligado a presentar una demostración de ella, algo que no hizo en ese preciso momento. De hecho, durante todo el siglo, los lógicos han trabajado esta cuestión.

2.3.2. Un matemático mecánico

Una forma equivalente de enunciar el problema de Hilbert es en términos de la posibilidad de construir un procedimiento mecánico (matemático) universal mediante el cual pueda probarse si cualquier conjetura matemática es verdadera o falsa. Debido a que Gödel estableció la existencia de conjeturas matemáticas indecidibles, la cuestión inicial se transforma en la búsqueda del diseño de un procedimiento mecánico (matemático) mediante el cual se pueda probar la decidibilidad o indecidibilidad de cualquier conjetura matemática? (es decir, si la conjetura matemática puede, en principio, probarse o no). Y es esta pregunta la que Turing intenta responder, después de escucharla durante un curso de conferencias a las que asistió en la Universidad de Cambridge.

Es muy probable que Hilbert se refiriera a un algoritmo cuando hablaba sobre un procedimiento mecánico. Sin embargo, Turing lo tomo de forma literal y trató de modelar el proceso de pensamiento de un matemático. Para hacerlo, reflexionó sobre aquello que pasa por la mente de un ser humano mientras realiza una tarea matemática básica.

> “La idea detrás de las computadoras digitales puede explicarse diciendo que estas máquinas están destinadas a realizar cualquier operación que pueda realizar una computadora humana. Se supone que la computadora humana sigue reglas fijas; no tiene autoridad para desviarse de ellas en ningún detalle. Podemos suponer que estas reglas se proporcionan en un libro, que se modifica cada vez que se le asigna un nuevo trabajo. También tiene un suministro ilimitado de papel en el que hace sus cálculos[283].”

Los cálculos mentales consisten en la transformación de una cierta cantidad de números de entrada a través de una serie de estados intermedios que progresan de manera secuencial y, de acuerdo con un conjunto de reglas fijo, hasta que se llega a la solución. En algunas ocasiones hará falta usar lápiz y papel para registrar los estados intermedios. Al tener un enfoque matemático, se requieren definiciones más rígidas para los estados mentales que aquellas proporcionadas por la filosofía o la metafísica. Por ello, Turing tuvo que trabajar en definir los conceptos de la forma mas clara posible.

Los argumentos de Turing pueden explicarse a través de un modelo bastante vago sobre el proceso que se realiza para llegar escribir una demostración [220], en el que se considera observar al matemático durante su trabajo. Al escribir, es frecuente encontrarse con letras y símbolos, que generalmente están alineados. Al terminarse el espacio en el papel, se salta a la línea siguiente. Se completa una línea de cálculo tras otra hasta terminar la página. Una vez que termina una página, escribe en otra página y continúa hasta que se completa la demostración. Pero, si en lugar de tener papel en dos dimensiones, las líneas se pudieran unir unas a otras, será posible reemplazarle por una larga tira continua de papel, a la que se llama cinta. La longitud no puede fijarse por anticipado, ya que, cuando un matemático comienza la prueba un teorema, o resuelve un problema desconocido, no tendrá idea de cuantas hojas podría necesitar. Sería terrible que quedara a mitad de un trabajo o que, al descubrir «una demostración verdaderamente maravillosa, el margen sea demasiado estrecho para contenerla»[255], por lo que la cinta debería tener una longitud infinita.

Un matemático a menudo avanza y retrocede en los pasos de cálculo. Para replicar esta actividad del matemático, la máquina de tener poseer una extremidad que pueda llevar a cabo la tarea de «lectura/escritura», avanzando y retrocediendo a lo largo de la cinta. Aquí se muestra que la cinta, en lugar del papel, es una gran idea, ya que simplifica el movimiento de la extremidad (o cabezal) de lectura y escritura, limitando su movimiento hacia atrás y hacia adelante. Será suficiente permitirle que se mueva desde su posición actual a cualquier otra posición en la cinta.

Para escribir la demostración, un matemático emplea un alfabeto finito, compuesto de símbolos matemáticos, números, letras, signos de agrupación y otros. Esto se puede simular en una máquina mediante un alfabeto más simple, que esté compuesto únicamente de un espacio en blanco, representado por un asterisco (*), y dos símbolos distintos, que serán representados por 0 y 1. Una cadena de ceros y unos se distinguirá de otra utilizando el espacio. Los símbolos que el matemático utilice en su demostración pueden codificarse asignando a cada uno de ellos un patrón específico de estos dígitos. En particular, los números podrían expresarse en representación binaria, en lugar de la decimal.

La cinta infinita, el cabezal de lectura capaz de moverse y el alfabeto son elementos necesarios para realizar la tarea de cómputo, pero no suficientes. Al escribir una demostración se requiere conocer un conjunto de reglas lógicas, axiomas e, incluso, puede ser de utilidad recordar algunas conclusiones obtenidas con anterioridad. El cabezal de lectura escritura, o algún componente adicional conectado a él, deberá equiparse con dicho conocimiento. Para simplificar su operación, puede sugerirse que la cinta se divida en una secuencia de celdas idénticas bien identificadas, permitiendo a la máquina escribir solo un símbolo (es decir, 0 o 1) o dejar un espacio en blanco (que aquí se representa con un asterisco) dentro de cada celda. Esto es suficiente porque la máquina está usando un alfabeto que tiene solo dos símbolos. Dependiendo del conjunto de instrucciones presentes en el cabezal, cambiará el símbolo o lo dejará sin cambios y luego la cabeza se moverá, avanzando o retrocediendo. La cinta funciona para organizar la información y recuperarla en caso de que sea necesario, por lo que recibe también el nombre de memoria.

Con todo ello se tiene la capacidad de simular a un matemático mediante un procedimiento mecánico abstracto (una máquina abstracta), es decir, tiene la capacidad de realizar todas las tareas que pueda requerir un matemático. Por ejemplo, la evaluación de funciones que motivo el desarrollo de los dispositivos mecánicos mencionados, implica que puede ser reducida a un conjunto de tareas de cálculo mecánicas. La cuestión de escribir la demostración de teoremas no se antoja tan simple, pero se debe considerar realizable siempre que un matemático humano puede hacerlo, por el método que sea, y de forma que pueda escribirse en el papel. Así, cualquier conjetura matemática que pueda probarse mediante cualquier procedimiento también puede probarse en la máquina abstracta. Pero esto lleva a una conclusión sorprendente. Si una conjetura matemática no puede probarse en la máquina, entonces no puede probarse por ningún otro medio. Esto dio respuesta al problema de Hilbert. Solo queda pendiente la cuestión de si la máquina puede hacerlo de manera eficiente. En su honor, esta máquina abstracta recibe el nombre de máquina de Turing.

2.3.3. El prototipo de una computadora moderna

La máquina propuesta por Turing es conceptualmente sencilla. Para realizar las actividades propias de un matemático humano, será necesario que cuente con los siguientes elementos: cinta, unidad de control y

cabezal de lectura/escritura. Además es necesario considerar que, durante su funcionamiento, la máquina se encontrará en una cierta condición de operación, llamada *estado*, definida de acuerdo con la actividad que la máquina este realizando en ese momento. La cantidad de estados que la máquina puede tener es finita. El proceso realizado por la máquina está comprendido ente dos estados especiales: comienza en el llamado *estado inicial* y concluye al alcanzar el *estado de detención*.

La *memoria* será proporcionada por una cinta de papel infinitamente larga, dividida en varias secciones mas pequeñas, denominadas *casillas*, en las que es posible registrar un símbolo b_i. El conjunto de todos los símbolos que la máquina es capaz de identificar y registrar se denomina *alfabeto de la máquina*. El cabezal debe ser capaz de escribir símbolos en la cinta de papel o leerlos, permitiendo descifrar su significado. Finalmente debe contar con un *programa*, un conjunto de reglas que le indique a la máquina lo que debe hacer, por ejemplo, el que se muestra en la Tabla 2.8. Para que la máquina realice alguna tarea, debe conocer su propio estado y el contenido de la celda actual, información que le permitirá interpretar las órdenes indicadas en un programa. Si se tiene un conjunto de reglas apropiado, la máquina de Turing será capaz de computar todo número susceptible de ser calculado, escribiéndolo como combinación de ceros y unos.

Al realizar un cálculo, la máquina de Turing este mecanismo sigue una secuencia de acciones o pasos que serán ejecutados por la unidad de control. Cada uno de ellos requiere completar un número pequeño de tareas:

- **Lectura**. Consiste en identificar el símbolo escrito en la casilla actual.
- **Escritura**. Consiste en registrar un símbolo dentro de la casilla, no necesariamente distinto al leído. Escribir el mismo símbolo equivale a no realizar cambios.
- **Desplazamiento**. A partir del símbolo y del estado de la máquina, el cabezal se desplazará una casilla, ya sea hacia la izquierda o hacia la derecha.
- **Transición**. De acuerdo con las instrucciones correspondientes al símbolo leído y al estado anterior se abandona el estado actual e, inmediatamente, se adopta uno nuevo.

En la figura 2.3 puede encontrarse un representación esquemática para una máquina de Turing.

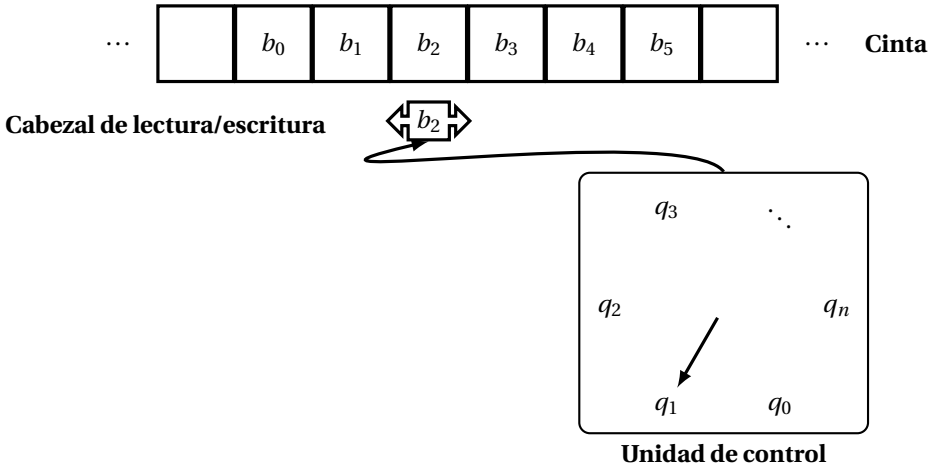


Figura 2.3: Esquema básico de una máquina de Turing. Está conformada una cinta infinita, un cabezal con la capacidad de identificar y registrar símbolos sobre la cinta y algo llamado *unidad de control*, un elemento capaz de identificar estados, gobernar el movimiento del cabezal y tomar decisiones. La cinta está representa con una tira horizontal, que está dividida en varias partes mas pequeñas, denominadas *casillas*, en las que es posible registrar un símbolo del alfabeto, aquí representados por b_i. El cabezal se representa mediante un cuadrado con dos flechas, que indican que puede desplazarse hacia la izquierda o la derecha. Este se coloca en la posición debajo de la celda que se está leyendo actualmente. El resultado de la medición se replica en el centro del cuadrado. Tras la lectura del símbolo en la casilla actual, se envía la información a la unidad de control, que interpretará los símbolos. Dependiendo del estado q_j en el que se encuentre la máquina, procederá a seguir las instrucciones indicadas por el programa.

Para visualizar la forma en la que opera, puede considerarse el siguiente ejemplo. La cinta de la máquina mostrada en 2.4 contiene el valor 5 (101_2). Se desea que la máquina de Turing incremente en una unidad ese valor, lo que en términos de la operación de una máquina significa modificar los valores de ceros y unos de tal forma que el patrón de bits resultante represente al siguiente entero, que en este caso corresponderá al 6 (110_2). Para hacerlo, se propone la ejecución del programa de la Tabla 2.8.

Tabla 2.8: Programa para aumentar en 1 el valor actual en una máquina de Turing [48]

Estado	**Lectura**	**Registrar**	**Desplazar a**	**Cambiar a**
INI	*	*	←	SUMA
SUMA	0	1	→	RET
SUMA	1	0	←	LLEVA
SUMA	*	*	→	PARA
LLEVA	0	1	→	RET
LLEVA	1	0	←	LLEVA
LLEVA	*	1	←	DESB
DESB	(Ignorado)	*	→	RET
RET	0	0	→	RET
RET	1	1	→	RET
RET	*	*	Sin movimiento	PARA

Un asterisco (*) sirve para señalar el final, en el extremo derecho, de una cadena binaria, la posición inicial de la máquina se ajustará precisamente en la celda que lo contiene. En todo momento, la máquina se encontrará en alguno de los estados siguientes: inicial (INI), sumar (SUMA), acarreo (LLEVA), desbordamiento (DESB), volver (RET) y detención (PARA). Los nombres de estos estados tienen relación con las acciones que les corresponden. Como puede verse, tras la lectura del contenido de la casilla y conociendo el estado actual, se realizan tres acciones: Se escribe uno de tres posibles símbolos, indicado en la columna «Escribir»; se desplaza al cabezal en alguna de las dos direcciones, según lo indique la columna «Mover a» y, finalmente, la máquina efectúa una transición hacia un nuevo estado, establecido en la columna «Cambiar a». La máquina siempre inicia a operar en el estado INI, como puede verse en 2.4.

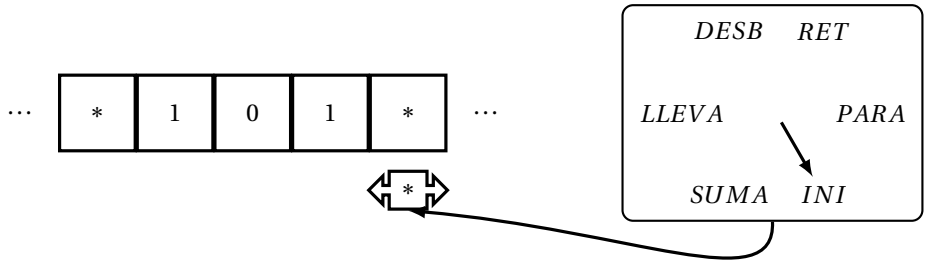


Figura 2.4: Estado inicial de la máquina de Turing del ejemplo. En la cinta puede observarse el número 5, escrito en sistema binario. Pude observarse la posición inicial del cabezal en la casilla del extremo derecho. La lectura indica *. Ademas del estado inicial (INI), la unidad de control indica los estados sumar (SUMA), acarreo (LLEVA) desbordamiento (Overflow, DESB), volver (RET) y detención (PARA). De acuerdo con el programa contenido en la Tabla 2.8, la combinación del estado INI con la lectura de * indica que debe escribirse un * en esa casilla, desplazar el cabezal hacia la izquierda y pasar al estado SUMA.

Cuando la máquina se encuentra en el estado INI y el contenido de la casilla actual es *, la tabla indica que se debe reescribir un *, desplazar el cabezal una casilla a la izquierda y pasar al estado SUMA. El contenido original de la cinta no se ha modificado, la máquina únicamente ha cambiado de posición y de estado, ahora está lista para una nueva lectura y realizar la siguiente instrucción del programa. La posición de la máquina será la mostrada en 2.5.

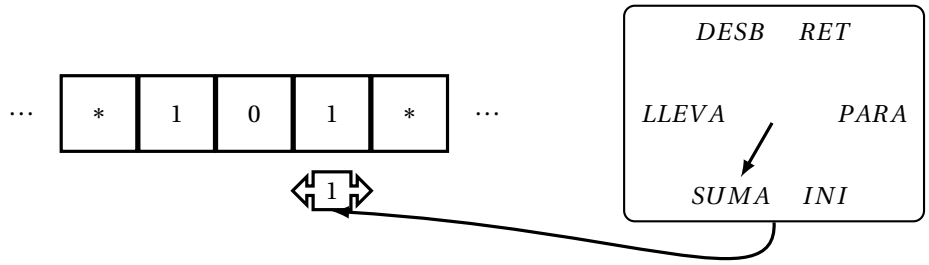


Figura 2.5: La máquina de Turing tras escribir *, desplazar el cabezal una casilla a la izquierda y pasar al estado SUMA. La combinación del símbolo 1 con el estado SUMA indican que la máquina debe cambiar al 1 por un 0, y después mover al cabezal hacia la izquierda y entrar en el estado LLEVA.

En el siguiente paso, al encontrarse en el estado SUMA y con la lectura de la casilla actual en 1, el programa indica que se debe sustituir el 1 por un 0, y luego mover el cabezal un espacio hacia la izquierda, tras lo cual entrará en el estado LLEVA, con lo que la máquina tendrá la configuración (posición del cabezal, estado de la máquina y contenido de la memoria) mostrada en 2.6

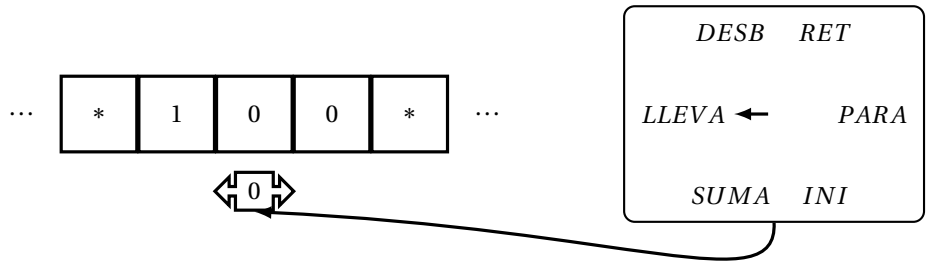


Figura 2.6: Máquina de Turing después de haber cambiado al 1 de la primer casilla de la derecha por un 0, haber desplazado al cabezal una casilla hacia la izquierda, y alcanzar el estado LLEVA.

La siguiente etapa, en la que se combina el estado LLEVA con la lectura de un cero en la casilla actual, ordena la sustitución del 0 por un 1 para desplazar posteriormente al cabezal una casilla a la derecha. Tras realizar esto, la máquina entra en el estado RET, conduciendo a la configuración siguiente (fig. 2.7):

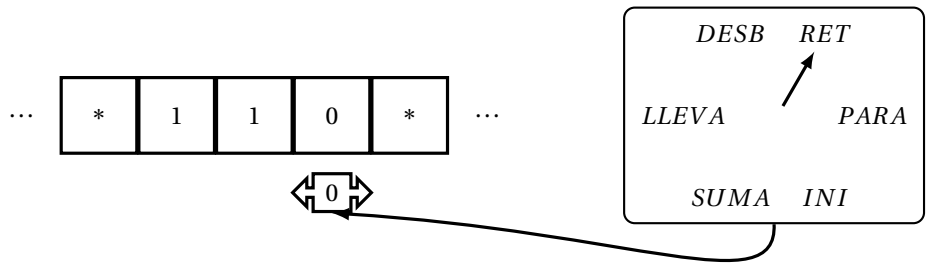


Figura 2.7: Maquina de Turing después de sustituir el 0 por un 1 en la segunda casilla, mover el cabezal una casilla hacia la derecha y alcanzar el estado RET.

A partir de ahora es necesario volver a la posición inicial. Sin embargo, la operación de la máquina requiere recorrer una casilla a la vez, por lo que será necesario volver a pasar por la primer casilla, sin afectar su contenido esta vez. Para realizarlo, la tabla indica que se sustituye el 0 de la casilla actual por otro 0, y luego se podrá mover el cabezal una casilla hacia la derecha, luego podrá permanecer en el estado RET, lo que lleva a la configuración mostrada en la figura 2.8.

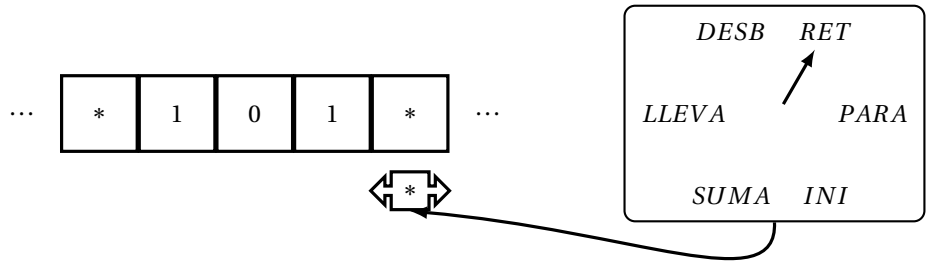


Figura 2.8: Máquina de Turing en estado RET llega a una casilla marcada con *

Finalmente, al colocarse en la posición inicial, debe reescribir el asterisco de esa casilla con otro asterisco. Tras realizar esto, puede entrar en el estado PARA. Por lo tanto, la máquina se detendrá en la configuración mostrada en 2.9. Como puede verse, el resultado es el número seis (110_2), como se esperaba.

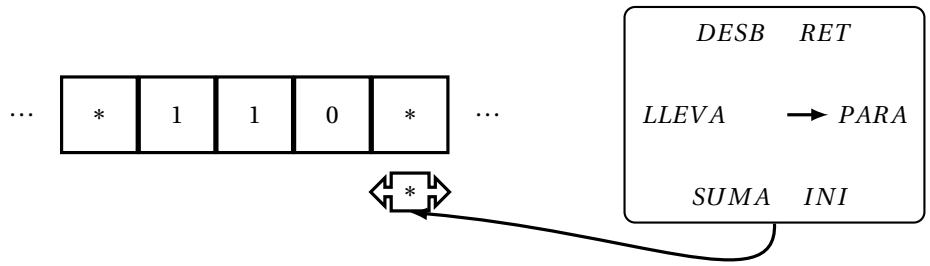


Figura 2.9: La máquina ha alcanzado el estado de detención

El programa seguido por esta máquina de Turing puede parecer excesivamente simple. Sin embargo, si en lugar de darle como entrada un cinco, se le asigna un entero no negativo arbitrario n, puede emplearse para calcular la función que le asigna un valor de salida igual a su sucesor, $n+1$. En general, una función numérica de k argumentos es *computable según Turing* si puede demostrarse la existencia de una máquina de Turing que, siguiendo un procedimiento similar al mencionado en el ejemplo, pueda evaluar a la función (una definición mas rigurosa puede encontrarse en [36]).

Pese a su sencillez, las máquinas de Turing son capaces de resolver problemas muy complejos. Se pueden utilizar para simular cualquier operación realizada en una computadora moderna, con la única diferencia en el tiempo de ejecución. A primera vista, puede parecer algo extraño que un dispositivo tan simple sea la base de la computación moderna. Existen, por supuesto, diferencias entre la máquina abstracta y una implementación real. Una de ellas es la cantidad de símbolos que pueden almacenarse en cada locación de almacenamiento. Mientras una computadora ordinaria puede almacenar 32 o 64 símbolos, la máquina de Turing únicamente puede guardar un símbolo. Esta diferencia no es esencial, pero hace a las computadoras reales más eficientes. Ambos tipos de máquinas pueden cambiar estos símbolos en un solo movimiento y cambiar de estado. Las computadoras reales cuentan con *acceso aleatorio* a la memoria, lo que significa, por ejemplo, que pueden leer el contenido de la ubicación de memoria 100 y luego, al ejecutar un movimiento, leer la ubicación de memoria 1000. La máquina de Turing puede simular eso fácilmente, pero debe hacer 900 movimientos para mover su cabeza 900 celdas a la derecha. Una vez más, la diferencia está en la eficiencia más que en la potencia informática fundamental. La máquina de Turing es tan buena como cualquier otra computadora[128].

2.3.4. Tesis de Church-Turing

Si una función puede calcularse mediante una máquina de Turing, un humano podría, en principio, llevar a cabo ese mismo cálculo y obtener el valor correspondiente. La idea de que una función esté al alcance de la mente humana se pretende introducir mediante la expresión *efectivamente computable.* Respecto de esta expresión, una discusión interesante puede encontrarse en [188] :

> Los matemáticos utilizan este término para seleccionar un determinado conjunto de funciones, pero no es un término matemático definido con precisión. Se aplica a una función cuando existe un método eficaz para obtener el valor de cualquiera de sus argumentos. Un método de este tipo consiste en una lista de instrucciones que podría seguir un ser humano trabajando sólo con lápiz y papel y que, en principio y si se siguen sin errores, proporcionaría el valor correcto para cualquiera de sus argumentos. El método debe ser completamente explícito en cada paso y no debe requerir ingenio ni información distinta a la contenida en sus pasos a seguir. Debe ser, en otras palabras, el tipo de método que podría implementarse mediante un dispositivo automatizado incapaz de pensar de forma independiente. El método también debe contener un número finito de pasos, cada uno de los cuales debe tardar un tiempo finito en realizarse. Sin embargo, incluso con estas restricciones, existen métodos que prácticamente ningún ser humano podría realizar. Por ejemplo, existen métodos que requieren que se escriban más dígitos, en cualquier sistema numérico, que átomos en el universo. Y hay métodos que, a pesar de ser finitos, requerirían que cualquier humano real trabajara lo más rápido que pudiera hasta que se complete la muerte por calor del universo. Pero éstas son limitaciones prácticas de espacio y tiempo, y se ignoran cuando se habla de computación efectiva. Para que una función sea efectivamente computable, todo lo que se requiere es que el método, dado suficiente espacio y tiempo, e ignorando cualquier otra dificultad práctica con su implementación, produzca el valor correcto para cualquiera de sus argumentos. A menudo se dice que efectivamente computable es un término intuitivo o informal y, a veces, se dice que es vago. Lo que sí es cierto es que no se trata de un término matemático definido con precisión. Esto se debe a que se define en términos de expresiones (por ejemplo, que no requieren ingenio para seguirlas) que en sí mismas no son términos matemáticos definidos con precisión.

El tipo de cálculos que puede efectuar una máquina de Turing se puede comprender de manera bastante intuitiva, por lo que toda función que sea computable según Turing resultará ser efectivamente computable. La *tesis de Turing* es que, a la inversa, cualquier función efectivamente computable es computable según Turing, de modo que si existe un algoritmo para calcular una función, entonces el cálculo puede ser realizado por una máquina de Turing[67]. Esta expresión, formulada en 1936 de manera independiente por Alan Turing y Alonzo Church, es ampliamente aceptada, pero no puede demostrarse matemáticamente. Al afirmar que el conjunto de las funciones efectivamente computables (la clave está en la noción intuitiva de computabilidad) y el conjunto de las funciones computables según Turing (donde la máquina de Turing es un objeto matemático bien definido) coinciden, no es posible agotar todas las posibles definiciones de computabilidad, por lo que no es un enunciado matemático y no se puede probar formalmente, resultando en una tesis filosófica más que matemática [55].

2.3.5. Otros modelos de la máquina de Turing

En particular, la tesis establece que no importa qué tan poderoso sea un dispositivo informático dado, hay problemas que esta máquina no puede resolver, fijando un límite que dicta lo que puede y lo que no puede calcularse mediante cualquier dispositivo informático imaginable, estableciendo a la máquina de Turing como un punto de referencia el que se puede comparar cualquier otro sistema de computación. Cuando algún sistema tiene la capacidad de calcular cualquier función computable según Turing, entonces se considera que su potencia es la máxima posible. Si un problema no puede resolverse con una máquina de Turing, ninguna otra máquina de cálculo podrá resolverlo. Es posible imaginar que algunas mejoras a la máquina de Turing podrían incrementar la cantidad de problemas que pueden resolvese con ellas, por ejemplo, empleando un mayor alfabeto, ampliando el espacio de trabajo a una rejilla rectangular, de manera que el cabezal pudiera desplazarse tanto arriba como abajo, además de a la izquierda ao a la derecha. La tesis de Turing implica que ninguna de estas propuestas puede ampliar la clase de funciones computables, porque todas aquellas funciones efectivamente computables en cualquier manera serán computables según Turing, lo cual significa que la máquina más rudimentaria sigue haciendo el cálculo, quizá con una diferencia en la velocidad.

La máquina de Turing no es el único modelo de cálculo conocido o explorado, pero otros modelos, como la máquina con acceso aleatorio[142], el Juego de la Vida de Conway[237], los autómatas celulares[171] e incluso cualquier lenguaje de programación[176], se puede computar en una máquina de Turing, por lo que

resultan equivalentes a esta última. Hay que añadir que la simplicidad de las máquinas de Turing permite que sean sencillas de analizar matemáticamente. Algunas modificaciones, no obstante que que no amplian la clase de funciones computables, permiten una mejor comprensión de su operación, o las hacen más adecuadas para un cierto tipo de problemas.

La máquina de Turing más básica es la llamada *máquina de Turing determinista*. Utiliza un conjunto fijo de reglas para que determinan sus acciones. El análisis se realiza considerando que la máquina requiere de una cantidad de tiempo y de memoria para para realizar los cálculos, es decir tiempo y espacio, que se suponen ilimitados. Para una máquina determinista, cada estado conduce a un siguiente estado único, por lo que el siguiente estado es una función del estado actual. Tal máquina se caracteriza por un espacio de estados equipado con una función de siguiente estado. Existen problemas computacionales que pueden abordarse de una manera mas sencilla si los recursos asignados a la máquina se modifican, por lo que pueden proponerse algunas construcciones alternativas.

Una *máquina de Turing multicinta* o de cadena múltiple tiene varias cintas, cada una con su propio cabezal de lectura/escritura, pero todos ellos conectados a un único dispositivo de control. En un momento dado, la máquina se encuentra en un estado único. Los cabezales concluyen la lectura de los símbolos escaneados en un solo paso. Dependiendo del estado actual y de estos símbolos, la máquina reescribe algunas celdas o mueve algunos cabezales hacia la izquierda o hacia la derecha mientras cambia su estado. Una máquina multicinta con una sola cinta es una máquina de Turing ordinaria.

Una *máquina de Turing no determinista* es una aquella que puede tener múltiples acciones futuras posibles desde un estado dado. Es natural describir una máquina discreta no determinista mediante una relación de próximo estado. Cada estado tiene una colección de próximos estados. Dada una colección de estados en los que podría estar la máquina, el conjunto de estados en los que podría estar a continuación es la unión de los conjuntos para cada uno de los estados de la colección. El valor final devuelto por la máquina no determinista es el valor devuelto por la primera máquina determinista que puede terminar el algoritmo. Por supuesto, en general, esta es una colección de valores. Una forma de visualizar esta característica es pensando en que la máquina se bifurca en diferentes caminos computacionales posibles en cada paso. Si alguna de estas ramificaciones resuelve el problema, entonces se dice que la máquina ha resuelto el problema. La realización de una máquina de este tipo está lejos de toda posibilidad, es solo una máquina abstracta de interés académico, la forma de trabajo en ramas captura muchos de los modelos matemáticos estudiados, por lo que el tiempo no determinista resulra ser muy importante para el análisis de problemas computacionales.

Una *máquina de Turing probabilística* es una máquina de Turing que, si bien es determinista, posee un suministro adicional de bits aleatorios que se emplean en cada paso del cálculo, por lo que en cada etapa es posible tener dos resultados, uno con probabilidad p y el otro con probabilidad $1-p$. Todos los cálculos en la misma entrada requieren el mismo número de pasos y el cálculo finaliza con rechazo o aceptación. Todos los cálculos posibles de una máquina de Turing probabilística se pueden describir mediante un árbol binario completo cuyos bordes se dirigen desde la raíz hasta las hojas. Cada cálculo es un camino desde la raíz hasta una hoja, que representa la decisión final. Una variable aleatoria, por ejemplo, el lanzamiento de una moneda, caracterizada por el parámetro p, elige uno de las dos posibles salidas de un nodo para que sea el siguiente paso en la ruta computacional. Usualmente, la probabilidad es un medio. Una máquina de Turing probabilística es capaz de realizar una elección binaria aleatoria en cada paso, y las reglas de transición de estado se modifican para tener en cuenta estos bits aleatorios. Muchas veces el empleo de decisiones basadas en probabilidad resulta de ayuda para que un algoritmo pueda encontrar la solución de una manera mas eficiente. Los algoritmos que utilizan bits aleatorios reciben el nombre de *algoritmos aleatorios*.

2.3.6. Modelo de circuito de computación

La máquina de Turing permitió presentar de una manera clara, bien definida e intuitiva un modelo matemático para el cómputo, años antes de la existencia de los dispositivos que se conocen hoy en día. Es la base de las teorías sobre computabilidad y complejidad computacional, una herramienta de razonamiento conceptualmente simple y elegante que, sin embargo, resulta difícil de asimilar como un sistema de programación. En particular porque tiene una mayor relación con el hardware o el lenguaje de máquina que con un lenguaje de alto nivel. Por esta razón existen otros modelos de computación. Son equivalentes entre sí, pero abordan la operación de la computadora desde enfoques diferentes, lo que puede resultar conveniente para la comprensión de la forma en que estas máquinas trabajan.

Uno de estos enfoques se basa en el conocimiento de como opera una computadora actual. Una computadora clásica es un dispositivo digital construido con el propósito de ejecutar una secuencia de instrucciones sobre una cantidad de datos determinada. Dependiendo del conocimiento y necesidades de la per-

sona que emplea la computadora (el usuario), esta puede modificar el programa o los datos, hecho que le confiere a las computadoras digitales su característica flexibilidad, que le permite realizar tareas de procesamiento de datos que pertenecen a un amplio rango de aplicaciones. Las partes principales de una computadora son la *unidad de memoria*, en la que se almacenan datos de entrada, de salida, las instrucciones para procesar esos datos y los pasos intermedios que realizará mientras sigue las instrucciones; la *unidad central de procesamiento*, dedicada a la realización de operaciones aritméticas y lógicas para el procesamiento de los datos de acuerdo con las intrucciones conetenidas en el programa; y los *dispositivos de entrada y salida*, que permiten a los usuarios comunicarse e interactuar con la computadora [180].

Una computadora digital es sólo un ejemplo de un conjunto mas amplio de dispositivos, llamados *sistemas digitales*, caracterizados por su capacidad para manipular elementos discretos de información. El término digital, proviene del hecho de que los elementos discretos de información deben representarse en términos de ceros y unos. El soporte físico correspondiente a estos ceros y unos se denomina *señal*. Las señales más comunes son eléctricas, como las diferencias de potencial y las corrientes eléctricas. Mediante un par de valores que puedan distinguirse claramente, se puede codificar información. Para hacerlo, se considera que la unidad fundamental de información clásica tiene dos posibles valores (es binaria), cero o uno. A esta unidad se le llama *bit* (de *binary digit*, en inglés).

En cualquier sistema digital, los datos digitales se representan, almacenan y transmiten como un grupo de bits binarios llamados códigos binarios. Si el grupo tiene n elementos, el código binario tiene 2^n combinaciones distintas de unos y ceros y cada una deelas puede usarse para representar a un elemento de un conjunto dado(no necesariamente numérico). Por ejemplo, dos bits son suficientes para codificar un conjunto de cuatro elementos, asignando a cada elemento una de las combinaciones posibles: 00, 01, 10, 11. Así, cinco bits bastarían para codificar a las 32 entidades federativas que conforman a los Estados Unidos Mexicanos asignando un bit a cada una. También es posible representar números. La manera mas simple consiste en expresar a un entero arbitrario N en base 2. El número resultante estará compuesto por varios dígitos, cada uno de los cuales será un cero o un uno:

$$N = \sum_{k=0}^{n-1} 2^k a_k \tag{2.7}$$

Por ejemplo, el número 49 puede representarse en base dos como 110001_2, que una computadora puede representar como una serie de voltajes, como puede verse en la figura 2.10.

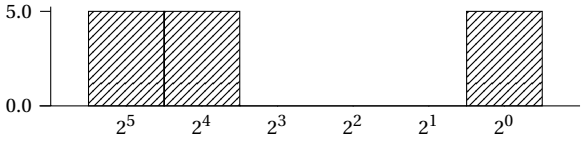


Figura 2.10: El número 49 puede representarse en notación binaria como 110001, cada uno está asociado con un voltaje *alto*, mientras que el cero corresponde a un voltaje *bajo*. Por ejemplo, el 1 puede estar representado por una diferencia de potencial de 5 V, mientras que el 0 por 0 V.

Para tratar la información, una computadora clásica realiza cálculos a través de varios componentes electrónicos, denominados *circuitos integrados*. Estos, a su vez, se construyen con *compuertas lógicas*, es decir, dispositivos electrónicos que pueden realizar operaciones lógicas, asignando a los estados de entrada los estados de salida correspondientes. Con diferentes formas de comunicar estas compuertas se puede conseguir que realicen funciones distintas. Esta conexión, a nivel físico, se hace por medio de cables.

Un *circuito booleano* es una colección de compuertas y entradas interconectadas entre sí a través de cables de tal manera que no se formen ciclos. Una de las compuertas se designa como *compuerta de salida* y produce la salida de todo el circuito. Cuando se asignan variables booleanas $x_1, x_2, ...x_n$ a las entradas, una compuerta de salida calcula una función booleana $f(x_1, x_2, ...x_n) : \{0,1\}^n \rightarrow \{0,1\}$. El *tamaño de un circuito* es el número de compuertas que lo componen, mientras que la *profundidad del circuito* es el tamaño de la trayectoria mas larga desde una entrada hasta la salida.

También es posible definir a un circuito booleano con m compuertas de salida. Tal circuito calcula una función booleana $f(x_1, x_2, ...x_n) : \{0,1\}^n \rightarrow \{0,1\}^m$. Por ejemplo, se puede construir un circuito sencillo, que sirva para realizar una suma, incluyendo acarreo, denominado *sumador completo*, como se muestra en el siguiente diagrama.

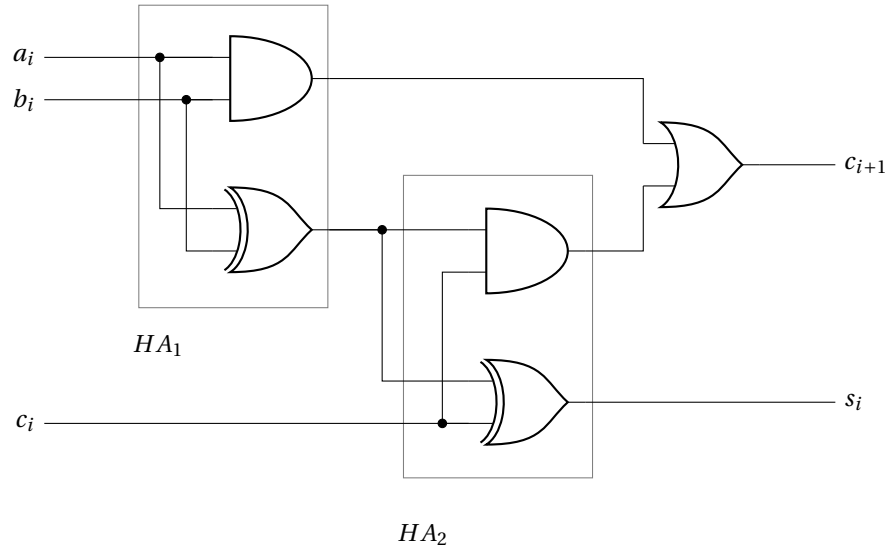


Figura 2.11: Un circuido llamado *sumador completo (de un bit)* es un circuito capaz de efectuar la adición de tres bits a_i, b_i (los sumandos) y c_i (el acarreo procedente de la etapa anterior menos significativa), proporcionando dos salidas, la suma $s_i = a_i \oplus b_i \oplus c_i$ y la cifra acarreada, c_{i+1}. Es posible construir un sumador completo a partir de dos circuitos, llamados *semisumadores*, HA_1 y HA_2 [160].

Este circuito realiza una tarea computacional específica, para lo cual requiere de algunas compuertas lógicas, conectadas de una forma especial. Cada compuerta se encarga de una pequeña parte de la tarea total. Muchas compuertas pueden unirse para formar una máquina de cálculo completa. De manera que compuertas y conexiones pueden emplearse para construir un modelo de computación clásica. La cantidad de compuertas requeridas para construir un circuito capaz de calcular $f : \{0,1\}^n \rightarrow \{0,1\}^m$ para cualquier número entero positivo n y m es finita, los conjuntos de compuertas que pueden realizar esto reciben el nombre de *bibliotecas* o *conjuntos universales de compuertas para el cómputo clásico* y existen varios de ellos, por ejemplo {NAND} (la compuerta NAND puede conectarse de diferentes maneras, consiguiendo realizar las operaciones lógicas de otras compuertas), {NOR}, {Toffoli} entre otras. De manera que es posible construir circuitos que puedan calcular cualquier función y, en consecuencia, también podrá calcularse todo lo que sea computable a usando una máquina de computación construida con circuitos. Este modelo recibe el nombre de *modelo de circuito de la computación.*

Tabla 2.9: Tabla de verdad para un sumador completo [180].

a_i	b_i	c_i	s_i	c_{i+1}
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

En términos de poder de cómputo, el modelo de circuito de cómputo es *equivalente* al modelo de máquina de Turing [112]. Esta declaración requiere algunas consideraciones especiales, comenzando por el hecho de que el modelo de circuito se encarga de la implementación práctica y por tanto no puede tratar con sistemas infinitos, en tanto que el modelo de la máquina de Turing supone recursos ilimitados, como la cinta infinita. Dado un algoritmo para entradas de longitud arbitraria también se tiene un algoritmo para cualquier longitud particular n, la máquina de Turing maneja entradas de varias longitudes que se colocan en la cinta cuando la máquina de Turing comienza a funcionar. El caso de los circuitos es distinto, porque un circuito en particular trata entradas de cierta longitud fija. Una forma de compensar esta diferencia es introducir una *familia de circuitos,* en lugar de un solo circuito[185]. Una familia de circuitos es un conjunto infinito

de circuitos $\{C_0, C_1, C_2, ...\}$, donde cada C_n tiene n variables booletanas de entrada, es decir, tiene una entrada x con n bits de longitud, su salida se denota como $C_n(x)$. Una familia de circuitos calcula una función $f : \{0,1\}^n \rightarrow \{0,1\}$ si $C_n(x) = f(x)$ para cualquier entrada $x \in \{0,1\}^n$. Mientras que un circuito es un objeto finito y concreto, una familia de circuitos puede calcular una función con dominio infinito. Pero, para hacer esto, es necesario contar con una forma eficiente de construir el circuito que corresponda al tamaño de entrada específico [294] .

Tanto el algoritmo como el circuito tienen una descripción finita, pero no es el caso de la familia de circuitos. Si bien, el modelo de circuito puede expresar algoritmos en términos de compuertas lógicas, para construir el circuito correspondiente a un algoritmo específico, es necesario tener un protocolo de diseño que indique el tipo de compuertas que se necesitan, la manera en que deben conectarse y cómo localizar los resultados de un cálculo. La necesidad de este protocolo conduce a una conexión mas profunda entre el modelo computacional del circuito y el de la máquina de Turing, debido a que se hace necesario tomar en cuenta la capacidad para distinguir entre funciones computables y no computables, de manera que se pueda evitar la incorporación de una función no computable en el mismo protocolo, así como contar con mecanismos que impidan que un algoritmo no muestre claramente la cantidad de memoria y tiempo que necesitará para su ejecución, es decir que impidan al algoritmo ocultar su *complejidad* (una explicación mas detallada de este concepto, se verá en la siguiente sección).

Considérese una máquina de computación M que calcula una función f a partir de una entrada x de tamaño n para producir el resultado deseado $f(x)$. Si se desea ocultar la cantidad de recursos requeridos por una máquina para ejecutar el algoritmo, debe cambiarse la entrada y presentarla a otra máquina M'. Esta segunda máquina debe tener la capacidad de producir el mismo resultado en un tiempo que sea proporcional a un polinomio en n. La nueva entrada consta entonces de dos partes: el protocolo que determina qué función calcula la primer máquina M y la entrada original x para la función deseada. Para que esta segunda máquina opere, se requiere un algoritmo para que una máquina de Turing pueda generar una descripción del protocolo. Una familia de circuitos es *uniforme* y se denota como $\{C_n\}$ si existe un algoritmo para que una máquina de Turing genere una descripción del protocolo para el diseño del circuito, dado el tamaño de la entrada. Para precisar esta noción, debe considerarse que el cálculo de la función podría depender de la entrada, por lo que sería posible que haya circuitos muy pequeños C_n para f_n que sean muy difíciles de calcular y circuitos más grandes C'_n para f_n que sean mucho más fáciles de calcular. La familia de circuitos $\{C_n\}$ será uniforme si $C_n(x)$ puede calcularse a partir de n en un espacio cuyo tamaño sea proporcional a $\log(n)$ [291]. Si no existe tal algoritmo para una máquina de Turing, entonces la familia de circuitos se llama *no uniforme*. La equivalencia del circuito y los modelos computacionales de Turing está restringida a familias de circuitos uniformes.

2.4. Complejidad computacional

Una vez concretada la definición del modelo computacional, comienzan a diseñarse los dispositivos que se encargarán de realizar las tareas del algoritmo. Las implementaciones reales se verán limitadas por los materiales, medios y técnicas de las que se disponga en el momento de su creación. En los años cincuenta se construyeron las primeras computadoras de propósito general. La memoria de estos dispositivos les permitía realizar operaciones y registrar los resultados, pero la principal característica era que las instrucciones para operar, el programa, podía almacenarse ahí mismo. Para lograrlo, John von Neumann, Arthur W. Burks y Hermann H. Goldstine trabajaron en una codificación numérica para las instrucciones, basada en la numeración de Gödel. En palabras de los autores:

> “Es evidente que la máquina debe ser capaz de almacenar de alguna manera no sólo la información digital necesaria en un cálculo determinado, como valores límite, tablas de funciones (como la ecuación de estado de un fluido) y también los resultados intermedios del cálculo (que puede ser necesario durante períodos de tiempo variables), sino también las instrucciones que gobiernan la rutina real que se realizará con los datos numéricos.”

> En una máquina para fines especiales, estas instrucciones son parte integral del dispositivo y constituyen parte de su estructura de diseño. Para una máquina multiuso debe ser posible ordenar al dispositivo que realice cualquier cálculo que pueda formularse en términos numéricos. Por tanto, debe haber algún órgano capaz de almacenar estas órdenes de programa. Además, debe existir una unidad que pueda comprender estas instrucciones y ordenar su ejecución.[51]

Desde entonces, la organización, diseño e implementación de las máquinas de propósito general están fundamentados en lo que se denomina arquitectura de von Neumann. En esta arquitectura las instrucciones no se diferencian de los datos, ambos son números, todo depende del uso que se le asigne.

Las primeras computadoras contaban con una potencia bastante limitada, debido a que los procesadores eran demasiado lentos y la memoria escasa, por lo que estos recursos tenían que usarse de la manera más eficiente posible. También era necesario examinar el tipo de problemas que podían resolverse con estas máquinas. Se ha demostrado que varias tareas interesantes son intrínsecamente *no computables*, lo que significa que ninguna computadora puede resolverlas sin entrar en ciclos infinitos, es decir, sin detenerse nunca, para ciertos valores de entrada.

Para evitar este tipo de escollos, se desarrollaron herramientas que permitían identificar la cantidad mínima de recursos computacionales necesarios para la implementación de un algoritmo, como la memoria, el tiempo y la energía. Este estudio constituyo eventualmente un campo de conocimiento que hoy recibe el nombre de *complejidad computacional* y una de sus tareas principales es *determinar los recursos mínimos necesarios para resolver un problema.* Dado que un mismo problema puede resolverse por más de un método, es de esperarse que exista más de un algoritmo para resolver el problema y es necesario compararlos de alguna forma. La *eficiencia* de un algoritmo está relacionada tanto con el tiempo empleado para su proceso en la computadora, como con la memoria necesaria para su ejecución. Cuanto menor sea el tiempo de ejecución o la cantidad de memoria requerida, mayor será la eficiencia alcanzada por el algoritmo.

2.4.1. Eficiencia y complejidad

La eficiencia de un algoritmo se cuantifica estudiando cómo escala el número de operaciones básicas que realiza conforme aumenta el tamaño de la entrada [10]. La cantidad total de memoria utilizada al ejecutar un algoritmo mide su *complejidad espacial.* Una definición mas precisa es la siguiente. Para una máquina de Turing determinista M que alcanza el estado de parada ante cualquier valor de entrada, se define la complejidad espacial de M como una función $f(n) : N \to N$, donde $f(n)$ es la cantidad máxima de celdas que M visita para procesar una entrada de longitud n.

La *complejidad temporal* se refiere al número de operaciones que realiza el algoritmo para completar su tarea, suponiendo que cada operación tiene la misma duración. Se puede analizar el peor escenario, tomando el tiempo mas largo de todas las entradas como medida, o bien, un escenario promedio donde la duración base se toma del promedio de todas las entradas. Luego, si M es una máquina de Turing determinista que alcanza el estado de detención ante cualquier valor de entradas, la complejidad temporal o *tiempo de ejecución* de M es la función $f(n) : N \to N$, donde $f(n)$ indica la cantidad máxima de pasos que la máquina M usa para una entrada de longitud n. También se dice que M *corre* en un tiempo $f(n)$[256].

Las funciones de complejidad pueden ser difíciles de escribir y de tratar, por lo que usualmente sus valores son estimados. Una buena aproximación para medir la eficiencia es conocer el orden de magnitud de la complejidad conforme la longitud de la entrada crece, es decir, analizar el *comportamiento asintótico de la función.* Con este propósito se considera únicamente el término de mayor orden en la expresión considerada, y sin tomar en cuenta el coeficiente, ya que el mayor orden domina a los demás términos para entradas muy grandes. Por ejemplo, en la función $f(n) = 6n^3 - 2n^2 + 20n + 45$ se tienen cuatro términos, donde el de mayor orden es $6n^3$. Despreciando el coeficiente 6, se dice que f se comporta asintóticamente *dominada* por n^3. Para formalizar esta noción, se introduce la siguiente la notación de Landau, también llamada «O mayúscula»($\mathcal{O}$):

> Sean f y g dos funciones. Se dice que $f(x) = \mathcal{O}(g(x))$ conforme $x \to \infty$ si y solo si existe una constante positiva c y un valor x_0 tales que
>
> $$\big|f(x)\big| \leq c \cdot g(x) \forall x \geq x_0$$

En un punto $x = a$, se tiene $f(x) = \mathcal{O}(g(x))$ conforme $x \to \infty$ si y solo sí, existen c y δ constantes positivas tales que $\left|f(x)\right| \leq c \cdot g(x)$ cuando $0 < |x - a| < \delta$. Si el contexto está libre de ambigüedades, se suele escribir $f = \mathcal{O}(g)$ y se dice que $g(x)$ es una cota superior asintótica para $f(x)$[20].

Esta notación significa que

- Para x tendiendo a infinito, la magnitud de $f(x)$ es aproximadamente (menor que una constante multiplicada por) la magnitud de $g(x)$ cuando x es lo suficientemente grande.
- Para x próximo a a, la magnitud de $f(a)$ es aproximadamente (menor que una constante multiplicada por) la magnitud de $g(x)$ cuando x está lo suficientemente cerca de a.

Es claro que las expresiones del tipo $f = \mathcal{O}(g)$ no pueden proporcionar el tiempo exacto que le llevará realizar un determinado cálculo, pero sirven para clasificar el desempeño relativo entre algoritmos. Para apreciar la utilidad de esta notación, puede considerarse un ejemplo simple. Partiendo del hecho de que el producto de números naturales es una forma abreviada de adición, se puede intuir que es más fácil sumar dos números cualesquiera que multiplicarlos. Esta idea se refuerza si se considera un par de algoritmos básicos. Por un lado, la suma de dos números enteros, que requiere un número de pasos que crece linealmente con el número total n de bits en los números de entrada, es decir, el tiempo necesario para ejecutar el algoritmo es $t_a = \alpha n$. Por otra parte, la multiplicación, en donde el número de pasos necesarios para calcular el producto de estos dos números es proporcional al cuadrado de n: $t_m = \beta n^2$.

La complejidad computacional pretende «medir la complejidad de un problema dado en términos de su algoritmo de solución *más compacto*» [42], lo que equivale a expresarla «en términos de la complejidad de sus soluciones» [48], de tal forma que un problema se considerara *simple* cuando se le conozca una solución sencilla y, en caso de no tener ninguna, será complejo. De acuerdo con este enfoque, la complejidad de la adición es $\mathcal{O}(n)$ mientras que la complejidad del producto es $\mathcal{O}(n^2)$. Por supuesto, hará falta demostrar que cada algoritmo es el más compacto posible, pero esta primera comparación parece sugerir que la multiplicación es más compleja que la suma. Sin embargo, tal conclusión estaría basada en algoritmos particulares para calcular sumas y multiplicaciones: los aprendidos en la escuela primaria. Cabe preguntarse si al utilizar distintos algoritmos se obtendrían conclusiones diferentes.

La suma de dos números no puede realizarse en menos pasos que n, porque al menos deben leerse los dos números de entrada de n dígitos, la complejidad de la adición no puede reducirse. El caso del producto es diferente. En 1962 Anatoly Karatsuba [143] encontró un algoritmo de multiplicación más rápido, que reduce la complejidad del producto a $\mathcal{O}(n^{\log_2 3} \approx n^{1.58})$[297]. Poco después, en 1971, Schönhage y Strassen [248] publicaron un algoritmo más rápido para la multiplicación, basado en la transformación rápida de Fourier, que requiere $\mathcal{O}(n\log(n)\log(\log(n)))$ pasos para multiplicar dos números de n dígitos en una máquina de Turing. En 2007, Fürer mejoró aún más este algoritmo [102], obteniendo un tiempo de ejecución de $n \cdot \log(n) \cdot 2^{O(\log\star(n))}$, donde $\log\star(n)$ es el número de veces que se debe iterar el logaritmo binario para llevar n por debajo de 2; por ejemplo, $\log\star(65536) = 4$ y $\log\star(10^{10000}) < 5$. Dado que $\log\star(n)$ es una cantidad prácticamente constante, parece probable que la verdadera complejidad de la multiplicación sea $\mathcal{O}(n \cdot \log(n))$ [199]. El último algoritmo publicado, del 2014, requiere $n \cdot \log(n) \cdot 8^{\log\star(n)}$ operaciones [121]. Sin embargo, no se puede excluir posibilidad de que existan mejores algoritmos para calcular la multiplicación todavía por descubrir.

El ejemplo anterior muestra que a veces, las tareas computacionales tienen algoritmos no intuitivos que resultan mas eficientes que los algoritmos usados durante miles de años. Además, muestra la manera en la que la complejidad permite comparar algoritmos, y cómo es posible asignar una complejidad a los problemas, en términos del mejor algoritmo conocido para resolverlos. Sin embargo, la última línea invita a una reflexión. La complejidad de un problema no será conocida a menos que pueda demostrarse que el algoritmo conocido es el mejor. Esto significaría que debe asegurarse que no existe algoritmo por descubrir que sea más eficiente, matemáticamente significa que debe demostrarse que todo algoritmo posible es menos eficiente que el algoritmo conocido, aunque esto rara vez se consigue [10].

2.4.2. Problemas tratables y no tratables

La interpretación asintótica del esfuerzo computacional de un algoritmo proporciona una herramienta que permite agrupar algoritmos en clases que se caracterizan por el mismo comportamiento de crecimiento en el tiempo, lo que se traduce en un una clasificación en términos de la resolubilidad algorítmica práctica. El concepto de eficiencia intuitivamente trata de capturar la idea de que un problema pueda resolverse usando

las computadoras disponibles en la actualidad. Un algoritmo se dice que es *eficiente*, si la cantidad de recursos necesarios para su ejecución, en el peor de los casos, está acotada por una expresión polinómica al tamaño de la entrada. Aunque la cota polinomial para representar un cálculo eficiente parece arbitraria, esta elección se ha justificado con el tiempo por los siguientes motivos:

- Los polinomios tipifican funciones de crecimiento lento, al punto que la definición para una *función de crecimiento lento* habla de una «función f localmente integrable, con $f : \mathbb{R} \to \mathbb{C}$ tal que $f(x) = \mathcal{O}(\$^{\backslash})$, para alguna n conforme $|x| \to \infty$» [115]. Por otra parte, las funciones exponenciales crecen más rápido que cualquier polinomio dado.

- La cerradura de polinomios bajo suma, multiplicación y composición preserva la noción de eficiencia bajo algunas prácticas consideradas naturales en programación, como usar dos programas en secuencia, o usar uno como subrutina de otro. Por ejemplo, si la salida de un algoritmo de tiempo polinomial alimenta a la entrada de otro, el algoritmo compuesto es polinomial. Si un algoritmo de tiempo polinomial realiza una cantidad constante de llamadas a subrutinas de tiempo polinomial, el tiempo de ejecución del algoritmo compuesto es polinomial [68].

- Esta elección elimina la necesidad de describir el modelo computacional con precisión. La experiencia con muchos modelos diferentes de computación ha llevado a la creencia generalizada de que «cualquier algoritmo que se ejecute en tiempo polinomial en una máquina puede implementarse en tiempo polinomial en cualquier otra máquina» [146].

- Desde un punto de vista práctico, un tiempo de ejecución de , n^2 es mucho más deseable que n^{100} y, por supuesto, el tiempo lineal es aún mejor. De hecho, incluso el coeficiente constante del tiempo de ejecución del polinomio puede ser crucial para la viabilidad de un algoritmo en la vida real. Curiosamente muy pocos algoritmos conocidos de tiempo polinomial para problemas naturales tienen exponentes superiores a 3 o 4 (aunque en el momento del descubrimiento, el exponente inicial puede haber sido 30 o 40), mientras que muchos problemas naturales importantes que resisten tanto a los algoritmos eficientes no pueden resolverse en la actualidad mejor que en tiempo exponencial, lo que no resulta práctico, incluso para datos de entrada pequeños. Reducir la complejidad de tales problemas de exponencial a (cualquier) polinomio resultaría una gran mejora [293].

Si n denota el número de bits necesarios para especificar la entrada, es posible dividir los problemas solucionables en dos categorías, considerando la resolubilidad algorítmica práctica: los tratables y los intratables.

Los problemas que pueden resolverse usando recursos que están acotados por un polinomio en n, es decir, aquellos cuya complejidad es $\mathcal{O}(f(n))$ con $f(n)$ un polinomio o acotada por un polinomio. Se dice que se pueden resolver de manera eficiente. También se dice que son fáciles, tratables o factibles. La suma y la multiplicación pertenecen a esta clase.

Los problemas que requieren recursos que son *superpolinómicos* (es decir, que crecen más rápido que cualquier polinomio en n). Estos se consideran difíciles, intratables o inviables, lo que no significa que sean imposibles de resolver, sino que es posible resolverlos siempre y cuando se espere suficiente tiempo. Sin embargo, el tiempo de cálculo puede llegar a ser tan grande como la edad del universo conocido, por lo que en la práctica no hay forma de conocer el resultado. Aunque no está probado, se considera que el problema de encontrar los factores primos de un número entero pertenece a esta clase, lo que significa que el mejor algoritmo conocido para resolver este problema es superpolinómico en el tamaño de entrada n. No se ha demostrado que no pueda existir un algoritmo polinomial.

La frontera entre problemas tratables e intratables puede cambiar con el tiempo. Los avances en la velocidad de las computadoras han hecho que lo que se consideraba en los límites de lo intratable hace veinte años hoy resulte tratable. Pero, los algoritmos superpolinómicos solo permiten aumentos modestos en el tamaño de las instancias con el aumento de la potencia de la computadora. Por el contrario, los algoritmos cuyo tiempo de ejecución se ve acotado por una función polinomial en el tamaño de la entrada se beneficiarán plenamente de la mejora del rendimiento informático.

2.4.3. Clases de complejidad

La idea de separar a los problemas en dos categorías diferentes, dependiendo si es posible resolverlos o no utilizando los recursos computacionales actuales introduce la noción de *clase de complejidad* como una colección de algoritmos que pueden procesarse empleando los mismos recursos. En realidad, las clases de complejidad no se componen de problemas, sino de *lenguajes* [236].

En este contexto, un lenguaje, o problema de decisión, se refiere a funciones booleanas, cuya salida es un solo bit. De manera mas precisa, la función f se identifica con el subconjunto $L_f = \{x : f(x) = 1\}$ de $\{0,1\}^n$ y se llama a tales conjuntos lenguajes o problemas de decisión[10]. Se identifica al problema computacional de calcular f (es decir, dado x obtener $f(x)$) con el problema de decidir el lenguaje L_f (es decir, dado x, decidir si x es L_f). En otras palabras, los lenguajes son conjuntos de cadenas en algún alfabeto fijo como 0,1. Cualquier problema computacional de interés se puede transformar en un lenguaje correspondiente de una manera que capture su complejidad. La definición de clase de complejidad requiere especificar varios parámetros:

- El modelo subyacente de computación. Aunque se trata de una formalidad, ya que resulta ser de escasa importancia. «Dos modelos computacionales cualesquiera que satisfagan ciertos requisitos razonables pueden simularse entre sí y, por lo tanto, son equivalentes en potencia»[256].
- Un modo de cálculo. Se refiere a la forma en la que una máquina acepta su entrada. Los dos modos más importantes son el modo determinista y el modo no determinista.
- Un recurso. Es algo costoso que la máquina consume y cuya disponibilidad es limitada. Los recursos básicos son tiempo y espacio.
- Una función de acotamiento. Es una función f asignación de enteros no negativos a enteros no negativos que sirve para indicar la separación entre clases de complejidad.

La clase de complejidad se define entonces como «el conjunto de todos los lenguajes decididos por alguna máquina M de Turing de cadena múltiple que opera en el modo apropiado, y tal que, para cualquier entrada x, M gasta como máximo $f(|x|)$ unidades del recurso especificado» [217].

La noción de *clase de complejidad temporal* fue introducida en la sección anterior al hablar sobre los problemas tratables y los intratables. Los problemas tratables son aquellos que pueden resolverse en tiempo polinomial $\mathcal{O}(n^k)$ para alguna k en los enteros y n el tamaño de la entrada. Por lo general esta clase de problemas pueden resolverse fácilmente con una máquina de Turing. A esta clase de complejidad se le denomina **P**. Otra clase histórica es **NP**, que significa *tiempo polinomial no determinista*. Contiene problemas que pueden resolverse en tiempo polinomial mediante un algoritmo no determinista. Es posible mostrar que la complejidad de los cálculos no deterministas es equivalente a la complejidad de la verificación determinista de una prueba dada[129], es decir, los problemas **NP** se pueden definir de manera equivalente como aquellos que admiten una salida cuya validez se puede verificar en tiempo polinomial. Para un problema **NP** la única fuente de intratabilidad es el tamaño exponencial típico del espacio de búsqueda. Algunos problemas conocidos que caen en esta categoría son los siguientes:

- Problema de la mochila o embalaje ($\mathcal{O}(2n)$). Se tienen n productos con valor c_i y peso p_i que deben ser embalados en una mochila (caja, maleta, bolsa, un camión o cualquier otro contenedor) y se tiene limite máximo en el peso a cargar P. El problema consiste en encontrar el embalaje óptimo de los productos que cumpla las restricciones de peso y maximice el valor total de los productos embalados. Se debe entonces decidir que producto se embala ($x_i = 1$) o que producto no se embala ($x_i = 0$) resultando de allí la condición binaria de la variable x_i [241].
- Problema del agente viajero ($\mathcal{O}(n!)$). Una persona debe visitar n clientes, sin repetir cualquier camino entre dos pares de clientes, saliendo y terminando su recorrido en el mismo lugar, a un costo mínimo. Es decir, se trata de minimizar la función $\sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}x_{ij}$, con $i = 1,...,n$; $j = 1,...,n$, donde x_{ij}, que indica que hay un camino desde la ubicación del cliente i hasta la ubicación del cliente j y, por otra parte, c_{ij}, que es el costo de ir desde el lugar en donde se encuentra el cliente i hasta donde se encuentra el cliente j [271].
- Torre de Hanoi ($\mathcal{O}(2^n–1)$). Este problema fue propuesto alrededor de 1883 por el matemático François Édouard Anatole Lucas. El rompecabezas se puede resumir de la siguiente manera: hay tres agujas y una torre de n discos en la primera, con el más pequeño arriba y el más grande abajo. El objetivo del rompecabezas es mover toda la torre desde la primera aguja a la segunda, moviendo sólo un disco cada vez y teniendo cuidado de no poner un disco más grande encima de uno más pequeño. La forma mas simple de resolver este rompecabezas es mover una torre de altura $n-1$ a la tercera aguja, mover el último anillo a la segunda aguja y finalmente mover la torre de la tercera aguja a la segunda. Se puede demostrar que cuando se comienza con una torre de altura n, entonces hay que realizar $2^n - 1$ movimientos para resolver el rompecabezas. Por ejemplo, cuando se tiene una torre de 64 discos, entonces es necesario realizar 18,446,744,073,709,551,615 movimientos para resolver el problema [267].

- Ciclo hamiltoniano ($\mathcal{O}(n!)$). Encontrar un ciclo hamiltoniano en un grafo no dirigido es un problema que se ha estudiado durante más de cien años. Formalmente, «un ciclo hamiltoniano de un grafo no dirigido $G = (V, E)$ es un ciclo simple que contiene cada vértice en V»[34], es decir, un ciclo que pasa por cada vértice. Introducidos por Kirkman en 1855, los ciclos hamiltonianos llevan el nombre de William Hamilton, quien describió un juego en la gráfica del dodecaedro, en el que un jugador especificaría una ruta de cinco vértices y el otro tendría que extenderla a un ciclo. El juego se comercializó como *El Dodecaedro del Viajero*, una versión de madera, en la que los vértices llevaban el nombre de 20 ciudades importantes. No tuvo éxito comercial [68].

Todos estos problemas pertenecen a **NP**. Sin embargo, existen problemas que no entran en esta clasificación. Dado que siempre se puede agregar un conjunto de instrucciones no deterministas a cualquier algoritmo determinista sin afectar su rendimiento, debe ser claro que $\mathbf{P} \subset \mathbf{NP}$. Además, cada problema de **NP** puede estar en **P**. Hoy en día, sigue siendo un problema abierto fundamental de las matemáticas y la informática si existen problemas en **NP** que no estén en **P**. Se conjetura, aunque no está probado, que $\mathbf{P} \neq \mathbf{NP}$. Si este fuera el caso, habría problemas difíciles de resolver, pero cuya solución podría comprobarse fácilmente. Los esfuerzos por comprobar si las clases **NP** y **P** coinciden han llevado al descubrimiento de un tipo de problemas dentro de la clase **NP** a los que se denomina **NPC** (**NP** *completos*), los mas difíciles en el sentido de que si existen problemas de **NP** que no pertenece a **P**, serían justo esos[69]. La propiedad que los caracteriza es que, si algún problema en **NPC** puede resolverse en tiempo polinomial, entonces todo problema en **NP** tiene un algoritmo de tiempo polinomial. Esto significa que si se desarrolla un algoritmo determinista que pudiera resolver cualquiera de los problemas **NPC** en tiempo polinomial, ese mismo podría ampliarse para tratar cualquier otro problema **NP**, también en tiempo polinomial, lo que a su vez implicaría que **NP** y **P** coinciden [140].

Las clases de complejidad van más allá de **NP**. La clase **EXP** contiene, de manera informal, todos los problemas que se pueden resolver mediante algoritmos de tiempo exponencial. Mediante una extensión cuantitativa directa de la prueba de diagonalización que establece que el problema de detención es indecidible, se puede demostrar que hay problemas en **EXP** que no están en **P**. **EXP** en sí mismo es un subconjunto propio de los lenguajes decidibles. Y se sabe que **EXP** contiene todo **NP** [125].

Un ejemplo de problema que está fuera de la clase **NP** está relacionado con el *Problema de la Detención*. Dada una máquina de Turing M a la que se le ha asignado una determinada entrada x, ¿se detendrá eventualmente? La pregunta es bastante natural. La máquina podría terminar en el estado de detención tras un tiempo finito o entrar en un ciclo infinito que le impida alcanzarlo. Turing demostró que no existe algoritmo capaz de resolver este problema. Las principales investigaciones sobre computabilidad y computabilidad eficiente están ligadas a tres formas básicas de este problema [94]:

1. *Problema de la Detención.* Dada una máquina de Turing M como entrada, si M se inicia con una cinta vacía, ¿se detendrá alguna vez?
2. *Problema de la Detención en tiempo polinomial para las máquinas de Turing no deterministas.* Dada una máquina de Turing no determinista M, ¿es posible que M alcance un estado de detención en n pasos, donde n es la longitud de la descripción de M?
3. *Problema de la Detención en el paso k para las máquinas de Turing no deterministas.* Dada una máquina de Turing M no determinista y un entero positivo k. (El número de transiciones que se pueden realizar en cualquier paso del cálculo es ilimitado y el tamaño del alfabeto tampoco tiene restricciones), ¿es posible que M alcance un estado de parada en k pasos como máximo?

La primera forma del problema de la detención es útil para estudiar la pregunta: «¿Existe *algún* algoritmo para *este* problema?» La segunda forma ha demostrado ser útil durante casi 30 años para abordar la cuestión: «¿Existe un algoritmo para *este* problema... *parecido* a aquellos para el ordenamiento y la multiplicación de matrices?». Esta versión más simple del problema de detención, pregunta si una máquina de Turing determinista M se detendrá en un máximo de n pasos, para alguna entrada dada x (escrita en notación binaria). Tal tarea solo puede resolverse en tiempo exponencial, ya que una simulación requiere un tiempo $\mathcal{O}(n)$, mientras que la entrada $x = n$ se codifica utilizando $\mathcal{O}(\ln(n))$ bits.

Las clases de complejidad también se ocupan de otros recursos además del tiempo, sobre todo el espacio. En analogía con **P**, **PSPACE** es la clase de complejidad de todos los idiomas que puede reconocer una computadora utilizando una cantidad de memoria (por ejemplo, el número de cuadrados de cinta de la máquina

de Turing) que está limitada por un polinomio en el tamaño de la entrada. La memoria es un recurso que es más poderoso y robusto que el tiempo (obviamente, puede calcular más cosas con 1 000 000 de palabras de memoria y tiempo ilimitado, que con 1 000 000 de instrucciones y memoria ilimitada). Por ejemplo, **PSPACE** contiene tanto **P** como **NP** (pero está contenido en **EXP**). Además, otra señal de la solidez del espacio es que el no determinismo no hace una gran diferencia en el dominio del espacio, y el no determinismo

2.5. Energía e Información

Se ha discutido una disciplina dedicada al tratamiento de la información empleando dispositivos electrónicos, cuyo funcionamiento se basa en la lógica. Pero, en última instancia, estos dispositivos en los que se procesa, almacena y transmite información, están sujetos a las leyes de la física y, en particular, a las leyes de la termodinámica. Ya que todo cálculo realizado con una computadora en el mundo real involucra de manera inevitable alguna clase de estructura física y procesos que realizan los pasos computacionales simbólicos, las leyes de la física pueden implicar límites a la computación. La pregunta ¿la información es física? ha despertado un profundo interés y un ávido debate académico, siendo Landauer uno de los mayores defensores de que la respuesta es afirmativa [163].

En un artículo influyente[161], Landauer usó argumentos termodinámicos para concluir que las operaciones lógicamente irreversibles, como el borrado, conducen necesariamente a la generación de calor en el proceso de cómputo, siendo así una fuente fundamental de disipación de energía. Este artículo generó muchos estudios posteriores y creó una escuela de pensamiento que vincula íntimamente la termodinámica y la computación, aunque también es el origen de una intensa controversia.

Para entender el argumento de Landauer, es necesario precisar lo que se entiende por información. Una respuesta intuitiva es considerar aquello que aún no se sabe [174]. Leer un anuncio en el periódico con un hecho que es de dominio público, como que el agua moja, tiene un valor de información bajo para todo aquel que haya experimentado alguna vez la sensación de humedad. Por el contrario, encontrarse con una predicción sobre el lugar y momento exactos en el que el enemigo podría atacar representa un valor elevado de información. En términos rigurosos, la cantidad de información se cuantifica mediante la denominada entropía de información H, introducida por Shannon en 1948, a mayor cantidad de información le corresponde una mayor entropía [250].

Ya que la información puede codificarse en bits, el dispositivo mas sencillo que permite almacenar información sera un sistema que tenga dos estados distintos. En caso de que se conozca el estado actual del sistema, es decir, si se sabe que el sistema se encuentra en cierto estado con probabilidad uno, una medición en el sistema no revelara ninguna información nueva, por lo que la entropía de Shannon sera nula. Pero, si no se conoce el estado de antemano, por lo que los dos estados pueden ocuparse con la misma probabilidad, la medición del sistema proporcionará información novedosa, por lo que en este caso la entropía de Shannon será igual a $ln(2)$, valor que corresponde a la menor cantidad de información posible, es decir, a un bit.

La observación empírica de que algunos procesos naturales ocurren espontáneamente en una sola dirección llevo a Clausius a la formulación de la segunda ley de la termodinámica. Para caracterizar la irreversibilidad de los procesos macroscópicos definió la entropía termodinámica S, una cantidad que, al contrario de lo que pasa con la energía, no se conserva, sino que se incrementa en sistemas aislados. Esta asimetría impone una restricción a los fenómenos físicos que puede suceder en la realidad y, de manera similar, establece limitaciones a las tareas de procesamiento de información. Se plantea la posibilidad de obtener trabajo útil a partir de lo que se conoce del sistema y, en un nivel mas fundamental, la posibilidad de que la entropía termodinámica y la entropía de Shannon estén relacionadas de alguna forma[219].

2.5.1. El demonio de Maxwell y el motor de Szilard

Las discusiones sobre el vínculo entre información y energía tienen una larga historia, que se remonta al *demonio de Maxwell*, una situación hipotética propuesta en el siglo XIX como un desafío a la segunda ley de la termodinámica. La existencia de tal situación sugiere que, al hacer una medición y luego usar el resultado de esa medición para realizar un conjunto de procesos reversibles, la entropía de un sistema puede reducirse sin hacer trabajo neto. Cerca del final de su libro sobre teoría del calor, Maxwell escribió en una sección llamada «Limitación de la segunda ley de la termodinámica» en la que puede leerse:

> Antes de concluir, deseo dirigir la atención a un aspecto de la teoría molecular que merece consideración. Uno de los hechos mejor establecidos en termodinámica es que es imposible producir alguna desigualdad de temperatura o de presión sin gasto de trabajo dentro de un sistema encerrado en una envoltura que no permite cambio de volumen ni paso de calor, y en el que tanto la temperatura como la presión son iguales en todas partes. Esta es la segunda ley de la termodinámica, y es indudable que es cierta en la medida en que podamos tratar con cuerpos solo en masa, y no tengamos el poder de percibir o manipular las moléculas separadas que los componen. Pero si concebimos un ser cuyas facultades sean tan aguzadas que tenga la capacidad de seguir cada molécula en su curso, tal ser, cuyos atributos son todavía tan esencialmente finitos como los nuestros, sería capaz de hacer lo que actualmente nos es imposible. Porque hemos visto que las moléculas en un recipiente lleno de aire a temperatura uniforme se mueven con velocidades que no son uniformes, aunque la velocidad media de cualquier gran número de ellas, arbitrariamente seleccionada, es casi exactamente uniforme. Supongamos ahora que tal recipiente está dividido en dos porciones, A y B, por una división en la que hay un pequeño orificio, y que un ser, que puede ver las moléculas individuales, abre y cierra este orificio, de modo que permita que sólo las moléculas más rápidas pasen de A a B, y que sólo las más lentas pasen de B a A. Así, sin gasto de trabajo, elevará la temperatura[189].

De acuerdo con esto, la energía se encuentra distribuida de forma equitativa en los sistemas macroscópicos que se encuentran en equilibrio, pero un ser humano o una máquina, puede crear regiones de temperatura o presión distintas a costa de un trabajo externo. Así, mientras se opere macroscópicamente será imposible influir en las moléculas individuales que constituyen la materia. Por ello, imagina un «ser» (que posteriormente sería bautizado como «el demonio inteligente de Maxwell» por Thomson[169]) con la capacidad de seguir a las moléculas individualmente y con una inteligencia y entrenamiento similar al humano, de tal forma que puede observarlas y controlar a las moléculas mediante un acto simple que no influye en sus energías.

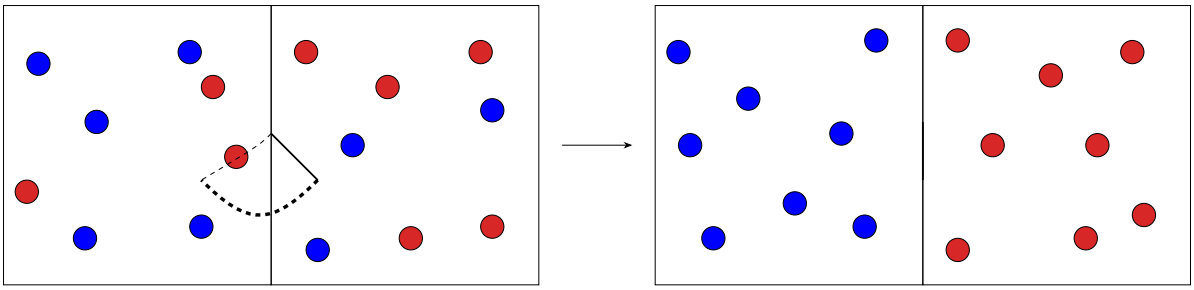


Figura 2.12: La acción del demonio de Maxwell consiste en observar a las partículas para seleccionar cuáles deben pasar a la cámara de la derecha y cuáles terminarán en la de la izquierda. Si esta selección permite distinguir partículas con velocidades altas de aquellas con velocidades bajas, al final del proceso se tendrán dos regiones, una a temperatura alta y otra a temperatura baja. En este proceso, no se ha suministrado energía a las partículas, puesto que la única acción del demonio ha sido abrir o cerrar la puerta.

Maxwell tenía una idea clara de la distribución de velocidades para las moléculas de un gas, por lo que sabía que no son uniformes. Divide al gas con una pared rígida que posee un pequeño orificio en el centro. El ser inteligente observa las moléculas individualmente y abre o cierra el orificio de manera que permita que las moléculas mas rápidas se queden en una de las mitades y las mas lentas en la otra, de tal forma que el número de moléculas en cada cámara se mantenga constante. Ya que el ser no las impulsa, no intercambia energía con las moléculas, únicamente se limita a abrir y cerrar una pequeña puerta para generar la diferencia de temperatura en ambas cámaras. Maxwell concluyó que la existencia de tal ser supondría una violación a la segunda ley de la termodinámica. Pero, destaca aún mas la observación, hecho por el mismo Maxwello de que, incluso sin el ser inteligente, se tendría una probabilidad distinta de cero de que ocurrieran transferencias moleculares anisotrópicas similares a las que tendrían lugar si el demonio estuviera en su puesto, lo cual conduciría a la misma diferencia de temperatura, al menos durante un tiempo breve. Únicamente habría que dejar abierta la pequeña puerta. Las diferentes velocidades darían como resultado más moléculas

de alta velocidad en la cámara B y más moléculas de menor velocidad en la cámara A, al menos durante un intervalo breve de tiempo.

> Este es solo uno de los casos en los que las conclusiones que hemos extraído de nuestra experiencia de cuerpos que consisten en un número inmenso de moléculas pueden no ser aplicables a las observaciones y experimentos más delicados que podemos suponer realizados por alguien que puede percibir y manejar las moléculas individuales que nosotros solo trabajamos en grandes masas [189].

El punto principal de Maxwell es claro, la segunda ley de la termodinámica es una ley estadística, no absoluta[168]. El ser inteligente imaginario trabaja en el mundo microscópico, que difiere considerablemente del mundo macroscópico. Sin embargo, existe una forma alternativa de enfrentar la contradicción introducida por el planteamiento de Maxwell. Leo Szilard sugirió un motor simplificado de una partícula en 1929[268] (la traducción al inglés fue publicada varios años después[269]). Su sugerencia parte de un cilindro vertical y hueco, cerrado por ambos extremos y que puede separarse en dos secciones V_1 y V_2 insertando una partición horizontal que puede moverse hacia arriba o hacia abajo, por lo que puede actuar como pistón. El cilindro está inmerso en un baño térmico de temperatura constante T, por lo que todo gas presente en el cilindro se expandirá isotérmicamente conforme el pistón se desplace. En particular, el gas debe estar compuesto por una sola molécula. Mientras el pistón no sea colocado, la única molécula podrá recorrer todo el cilindro gracias a su movimiento térmico. Además, sugiere a una persona que en un cierto momento introduce el pistón en el cilindro. De alguna manera, esta persona tiene la capacidad de observar si la molécula está atrapada en la parte inferior o superior del volumen. De encontrarse en el superior, el pistón se movería lentamente hacia abajo, hasta que llegue al fondo del cilindro, permaneciendo por encima del pistón en todo momento. Sin embargo, ya no está limitado a la parte superior del cilindro sino que rebota muchas veces contra el pistón que ya se está moviendo en la parte inferior del cilindro. De esta manera, la molécula realiza una cierta cantidad de trabajo sobre el pistón, que corresponderá a la expansión isotérmica de un gas ideal (de una sola molécula) desde el volumen V_1 al volumen V_1+V_2. Cuando el pistón llegue al final del recipiente, la molécula será libre de moverse nuevamente en el volumen total, tras lo cual se retira el pistón. La persona colocará el pistón hacia arriba o hacia abajo dependiendo del lugar en el que se observe a la molécula. Este movimiento puede ser debido a un mecanismo que transmite la fuerza del pistón al peso, de manera que el peso se desplace siempre hacia arriba, incrementando de manera constante su energía potencial.

En este dispositivo, o ligeras modificaciones en las que el cilindro es colocado horizontalmente mientras la división es vertical[153], la partícula tiene una probabilidad de un medio de encontrarse en alguna de las dos cámaras. Al mirar dentro del contenedor, la persona, o el demonio, adquiere información sobre el estado real del sistema, aprendiendo lo que no sabía antes. Si la molécula se encuentra en la cámara derecha, el peso se une al lado derecho del pistón, que luego se libera de su posición anterior. Durante la expansión del gas, el pistón se empuja hacia la izquierda y el peso se tira hacia arriba, realizando un trabajo contra la gravedad De manera similar, si la molécula se observa en la cámara izquierda, el pistón se unida al lado correspondiente. La segunda ley de la termodinámica limita la cantidad máxima de trabajo que puede producir el motor Szilard a $k_B T \ln(2)$, con k_B la constante de Boltzmann y T la temperatura del gas. El 2 dentro del logaritmo proviene de que el sistema de Szilard tiene dos cámaras. De esta forma $k_B T \ln(2)$ corresponde a la cantidad máxima de energía que se puede obtener de un bit de información. Históricamente es la primera declaración de una relación entre información y energía, implicando que las entropías de información y termodinámica son iguales hasta un factor multiplicativo k_B, introducido por razones dimensionales, ya que la entropía de Shannon H es adimensional, es decir $S = k_B H$.

La equivalencia entre S, una cantidad de calor intercambiada de manera reversible con un depósito, y H, que caracteríza el contenido de información de un mensaje enviado a través de un canal de información no deja de parecer extraña. Edwin Jaynes nota, citando al trabajo de Shannon, que:

> Además, muchas personas han considerado que el desarrollo de la teoría de la información es de gran importancia para la mecánica estadística, aunque la forma exacta en que debería aplicarse ha permanecido oscura. A este respecto es fundamental señalar lo siguiente. El mero hecho de que la misma expresión matemática $-\sum p_i \ln p_i$ aparezca tanto en la mecánica estadística como en la teoría de la información no establece por sí mismo ninguna conexión entre estos campos. Esto sólo puede hacerse encontrando nuevos puntos de vista desde los cuales la entropía termodinámica y la entropía de la teoría de los iones de la información aparezcan como el mismo concepto[135].

Tras lo cual sugiere una reinterpretación de la mecánica estadística, de forma que la teoría de la información pueda aplicarse al problema de justificar la mecánica estadística. Ya que la formulación de Gibbs encuentra algunas dificultades que no pueden entenderse en términos de mecánica clásica, fue necesario agregar restricciones adicionales que no estaban contempladas en el marco de la mecánica clásica y fue solo gracias al desarrollo posterior de la mecánica cuántica que las suposiciones originalmente arbitrarias pudieron verse como consecuencias necesarias de las leyes físicas. En su opinión, es posible que la mecánica estadística ya no dependa de hipótesis físicas, sino que puede convertirse simplemente en un ejemplo de inferencia estadística. Para demostrar este punto, predice las propiedades termodinámicas de equilibrio mediante un tratamiento elemental que implica sólo las probabilidades asignadas a los estados estacionarios. Su enfoque puede ilustrarse considerando el problema de determinar el trabajo máximo W_{max} que se puede extraer de un motor térmico, uno de los problemas clásicos de la termodinámica. De acuerdo con la segunda ley, los procesos que ocurren a temperatura constante T tienen $W_{max} = -\Delta F = -(\Delta E - T\Delta S)$, donde F es energía libre y E es energía. En otras palabras, no toda la energía termodinámica se puede convertir en trabajo, de manera similar a lo que ocurre con la energía mecánica; existe una parte no utilizable dada por $T\Delta S$. Ese resultado puede interpretarse desde el punto de vista de la información:

- Cualquier sistema termodinámico puede describirse microscópica o macroscópicamente.
- El microestado, caracterizado por las posiciones y velocidades de todas las partículas constituyentes de un sistema, contiene información completa sobre ese sistema.
- Pero los sistemas típicos contienen tantas partículas, del orden de 10^{24}, por lo que el microestado no es accesible experimentalmente.
- Por el contrario, el macroestado, definido por parámetros macroscópicos como la temperatura, la presión y el volumen, es medible pero solo contiene información parcial.

Jaynes reconoció que la entropía esencialmente cuantifica el grado de ignorancia sobre el estado del sistema, es decir, la cantidad de información de microestado que se pierde cuando se examina el sistema macroscópicamente, por lo que la energía se puede convertir completamente en trabajo solo si el microestado, que incluye la información completa sobre el sistema, se conoce. La observación de Jaynes proporcionó una base teórica firme para la noción de Szilard de una equivalencia entre la termodinámica y las entropías de la información. También mostró que la equivalencia debería, en principio, mantenerse para cualquier sistema arbitrario en equilibrio. El hecho de que la entropía termodinámica solo pueda aumentar en un sistema aislado podría entenderse como que el contenido de información del sistema solo puede disminuir espontáneamente. La ganancia o aumento de información, es decir, su adquisición a través de la observación directa de las moléculas, se complementa con la eliminación o borrado de información, con lo que la relación entre la entropía de la información y la termodinámica puede establecerse. Es en este punto donde aparece Landauer.

2.5.2. Principio de Landauer y la propuesta de Bennett

Partiendo de un sistema de dos estados que inicialmente almacena un bit de información, esto es, ambos estados se tienen misma probabilidad de encontrar ocupados. De acuerdo con Landauer, representan un conjunto de bits de los que no se conoce el estado inicial individual. Es posible borrar el contenido de información, mediante una operación que devuelve a todos los bits a un estado particular, por ejemplo, al cero, tras lo cual, todos los bits tendrán el mismo valor. Desde el punto de vista de la información, al borrar se conoce el estado del sistema, por lo que se ocupará uno solo de los estados con una probabilidad igual a uno. El número de estados cubiertos por el conjunto, en el sentido de que exista posibilidad de que estén ocupados, se ha reducido a la mitad. Al conocerse el estado no hay información nueva que adquirir y, por tanto, la situación corresponde a una nula entropía de Shannon, es decir, la entropía del sistema ha disminuido.

Sin embargo, en un sistema cerrado, como el que representa una computadora con su propia batería, la entropía debe incrementarse, no disminuir, por tanto, esta entropía debe aparecer en otra parte, en forma de calor, suministrando la misma cantidad por bit borrado a los alrededores. Por ello, Landauer propone que la aplicación de la segunda ley de la termodinámica a esta situación debería implicar que el borrado de información es un proceso necesariamente disipativo: borrar un bit de información produce de al menos $k_B T \ln(2)$ de calor al entorno[161]. Esta propuesta ha llegado a conocerse como la *tesis de disipasión de Landauer* o, con intención de hacerla parecer más fundamental el *principio de Landauer*.

Según esta propuesta, existe una diferencia fundamental entre el proceso de escribir y borrar información. Se considera como dispositivo a un sistema que puede estar en cualquiera de dos estados posibles, por ejemplo, una caja que contiene a una molécula que puede ubicarse en la sección izquierda o en la derecha. Un dispositivo que escribe, está realizando una tarea que consiste en copiar información desde un dispositivo A a otro B. Así, cada vez que se encuentra a una molécula en A del lado izquierdo (o derecho), se asigna la posición izquierda (o derecha) a la molécula correspondiente en B. Este mapeo uno a uno se puede realizar en principio sin disipar calor, si se considera que, a nivel intuitivo, no hay nueva información y, por tanto no hay cambio en la entropía. Un punto de vista que pretende mayor rigor consiste en considerar que, de acuerdo con la mecánica estadística, se conserva el volumen en el espacio de fase. Por el contrario, borrar información es una transformación de dos a uno: los estados izquierdo y derecho se asignan a un solo estado, por ejemplo, el derecho, independientemente de su estado inicial, perdiendo información en el proceso, por lo que no se conservará el volumen en el espacio de fase y, por lo tanto, será disipativo.

Al considerar la posible relación entre energía e información, resulta natural preguntarse si el principio de Landauer tiene implicaciones que pudieran ayudar a resolver la paradoja del demonio de Maxwell. Mucho antes de la publicación del artículo de Landauer ya existía una considerable cantidad de intentos para «exorcizar al demonio» [82]. Uno de ellos argumentaba que en un recinto cerrado, a temperatura constante, la radiación correspondería a la de un *cuerpo negro*, y por tanto el demonio sería incapaz observar las moléculas. Por lo tanto, no podría operar la trampilla y esto le impediría violar el segundo principio de la termodinámica [47]. Debido a que para poder ordenar las moléculas se requiere verlas, es necesario utilizar algún tipo de fuente luminosa, por lo que el precio energético de la medición compensará el cambio en la entropía. Cada vez que una molécula es observada, será necesario disipar, como mínimo, la energía de un fotón y esta debe superar cierta energía dada por la temperatura del sistema formado por las partículas del gas, la caja y el demonio.

Sin embargo, Charles Bennett anunciaría la posibilidad de realizar mediciones de manera reversible, y en cambio utilizó el principio de borrado de Landauer para proponer una solución a la paradoja del demonio de Maxwell. Para ello, señala que el demonio tiene que almacenar la información que adquiere sobre las moléculas de gas en una *memoria* [25]. Las mediciones que el demonio de Maxwell requiere, pueden realizarse de manera reversible, con la condición de que el dispositivo que emplee para realizarlas se encuentre en un estado específico antes de realizar cada medición, de tal forma que en una nueva medición no se sobreescriba un registro previo. Por tanto, el acto esencial irreversible no es la medición en si, sino la restauración del aparato de medición a un estado de preparación antes de efectuar la próxima medición. Después de un ciclo de producción de energía a partir de la recopilación de información, la memoria del aparato debe restablecerse a su estado inicial para permitir una nueva iteración y, por lo tanto, su contenido de información debe borrarse. El olvido del estado lógico anterior implica un mapeo de muchos a uno del estado físico del demonio que no puede lograrse sin un aumento de entropía en otra parte. De acuerdo con la tesis de Landauer, el proceso de borrado disipará una cantidad de energía que siempre es mayor que la cantidad de energía producida por el demonio durante un ciclo. En consecuencia, el demonio tiene que pagar un precio energético para clasificar las moléculas y hacer que fluya calor desde la cámara más fría a la cámara caliente, en concordancia con la segunda ley de la termodinámica.

2.5.3. Controversia y consecuencias de la tesis de Landauer

De acuerdo con el último párrafo de la sección anterior, el demonio puede realizar la medición sin gastar energía, pero esta información luego se almacena en alguna memoria, por ejemplo, en la mente del demonio. Si hay disipación de energía, ocurre en el momento en el que el demonio borra la memoria. Esta interpretación tiene un serio problema: ¿qué pasa si el demonio elige aclarar su mente un año después de que se realicen las operaciones en las moléculas de gas? La energía debería gastarse cuando las moléculas de gas se mueven, y no en algún momento posterior arbitrario. Este es sólo un ejemplo de los argumentos que se han presentado en contra de la tesis de Landauer [228]. En general, el empleo de la entropía en el contexto de la disipación de energía en computación ha suscitado una importante discusión[229],[227],[162], [148],[211], [158]. Aunque

la lista de artículos que discuten el tema es mucho mas amplia (puede consultarse, por ejemplo [170]), los argumentos que sostienen que el Principio de Landauer es en realidad falso o sin sentido pueden agruparse en tres clases[27]:

> 1. Es falso o sin sentido porque no hay conexión entre las cantidades termodinámicas como el calor y el trabajo y las propiedades matemáticas como la reversibilidad lógica, de modo que comparar las dos es equivalente a comparar peras con manzanas.
> 2. Es falso (o más precisamente su recíproco) porque todas las operaciones de procesamiento de datos, sean lógicamente reversibles o no, requieren la disipación de al menos $k_B T \ln(2)$ de energía, y de hecho, generalmente mucho más, para ser realizadas por cualquier aparato físico real.
> 3. Es falso porque, en principio, incluso las operaciones lógicamente irreversibles pueden realizarse sin un aumento de entropía en el entorno.

Parte de la controversia se puede atribuir a «fallas en el debate, como la definición imprecisa o incompleta de información y otras cantidades clave; una especificación imprecisa o incompleta de las operaciones, procesos, límites del sistema y escenarios físicos involucrados; confianza excesiva en sistemas modelo simples y generalización excesiva de los resultados obtenidos de los análisis de estos sistemas; y también a una fusión de límites inviolables (posiblemente inalcanzables) con límites prácticamente alcanzables, pero las raíces del debate se encuentran en la preocupación sobre la metodología e interpretación, incluyendo la adecuada aplicación de la termodinámica, las distinciones o posibles conexiones entre la entropía física y entropía de la información, y a las diferencias o relaciones entre reversibilidad física y reversibilidad lógica» [8]. Este último punto es importante porque la totalidad de la bibliografía consultada supone que la computación cuántica debe cumplir con el requisito de ser reversible *a todos los niveles*, es decir, lógica y físicamente hablando, pero apenas dedica alguna mención a la equivalencia entre ambas clases de reversibilidad, dando por hecho que la tesis de Landauer realiza esta conexión. Con el objeto de aclarar la distinción entre reversibilidad lógica y física, se presentan por separado dos relaciones y una conclusión que suelen confundirse[251]:

- (A) Tesis de disipación de Landauer: El cálculo lógicamente irreversible es disipativo. Aumenta la entropía en $k_B T \ln(2)$ por bit de información perdida, independientemente de la tecnología mediante la cual se lleve a cabo el cálculo. Esta disipación es consecuencia de la irreversibilidad lógica del cálculo en cuestión. Por lo tanto, el cálculo lógicamente reversible no está sujeto a la disipación de este origen. A esta disipación uno obviamente debe agregar la disipación de otros orígenes, ya que no implica que sea posible un cálculo absolutamente sin disipación.
- (B) Un teorema sobre la reversibilidad lógica de los cálculos: Todos los cálculos se pueden llevar a cabo de una manera lógicamente reversible. A cada algoritmo lógicamente irreversible corresponde uno lógicamente reversible que tiene la misma salida más una salida que permite invertir el cálculo. Al reemplazar un algoritmo lógicamente irreversible por su correspondiente reversible, se realiza el cálculo original de forma reversible [24] [99]. Pero se trata de un teorema *lógico*. La única manera de dotarlo de una implicación física es asociándolo con la tesis de Landauer, si esta se rechaza, no tiene nada que ver con la disipación de energía.
- (C) La tesis de la computación reversible: A y B juntos implican que todos los cálculos pueden (idealmente) llevarse a cabo sin disipación, es decir, sin el tipo de disipación asociada con la irreversibilidad lógica. La opinión prevaleciente actualmente es la última: se acepta C, la tesis de la computación reversible, como subyacente tanto en A como en B.

Una consecuencia importante del entusiasmo y preocupación por el significado de la tesis de Landauer es la pretensión de desarrollar un esquema de computación «sin disipación», es decir, computación (físicamente) reversible. Como en última instancia debe evitarse cualquier pérdida de información, es la base detrás de la creación de la computación (lógicamente) reversible. La idea suena plausible debido a que las leyes fundamentales de la física, las ecuaciones de Newton en la mecánica clásica, son reversibles. Por lo tanto, debería existir algún proceso físico reversible subyacente que permita implementar compuertas irreversibles. Con esta idea en mente, Bennett demostró que un autómata informático típico de propósito gen-

eral (por ejemplo, una máquina de Turing) lógicamente irreversible puede hacerse lógicamente reversible en cada paso, conservando su simplicidad y su capacidad para realizar cálculos generales[24]. Al contrario de lo que creía Landauer, quién sostenía la posibilidad de evitar borrar o sobrescribir bits manteniendo todos los resultados intermedios en un enorme registro (que, sin embargo, solo pospondría la generación de entropía a una etapa posterior, cuando la computadora se reinicie para comenzar un nuevo cálculo), Bennett demostró que era posible restablecer este registro al estado inicial antes de finalizar el proceso computacional. Por lo tanto, la única entropía inevitable el gasto sería proporcional a la cantidad total de entrada y salida, no a la longitud de los cálculos intermedios.

Aunque el argumento central de Landauer parece sostenido en primeros principios, está lejos de ser trivial y es materia de discusión actual [290], por lo que, de ser falso, pondría en duda la necesidad de uno de los ingredientes esenciales del cómputo cuántico: la computación reversible. Desde luego, una mayor discusión de las implicaciones de la tesis de Landauer a la computación cuántica debe posponerse hasta introducir los conceptos básicos de mecánica y computación cuánticas; por ahora puede decirse que el motivo por el que se menciona en este punto es que que la computación reversible se encuentra profundamente arraigada en el desarrollo de la computación cuántica.

Finalmente, Earman y Norton [83], argumentan a través de un dilema simple que los intentos de resolver la paradoja de Maxwell, ya sea que siguiendo a Szilard en la búsqueda de un costo de entropía compensatorio en la adquisición de información o a Landauer buscando ese costo en el borrado de la memoria son ineficientes. En la medida en que el demonio sea un sistema termodinámico ya gobernado por la Segunda Ley, no se necesitan más suposiciones sobre información y entropía para salvar la Segunda Ley. En la medida en que el demonio no sea un sistema de este tipo, ninguna suposición sobre el costo de entropía de la adquisición y el procesamiento de la información puede salvar a la Segunda Ley del demonio.

2.5.4. Computación reversible

En primer lugar, conviene mencionar la distinción entre dos tipos de reversibilidad lógica introducida por Asher Peres [222], a través de un ejemplo simple. Considérese un generador de números aleatorios típico definido por la relación de recurrencia:

$$x_{n+1} \equiv a x_n \pmod{b} \tag{2.8}$$

donde los enteros a y b son primos relativos. Claramente, esta expresión muestra cómo obtener x_{n+1} a partir de x_n, pero también muestra cómo obtener x_n de x_{n+1}. La relación define una secuencia finita de enteros pseudoaleatorios, en donde el período no puede ser mayor que b. Sin embargo, es considerablemente más difícil obtener x_n de x_{n+1} que x_{n+1} de x_n, puesto que, para hacer, se necesita realizar muchas más operaciones. Por lo tanto, el proceso descrito puede llamarse *globalmente reversible*. De acuerdo con Peres, la reversibilidad global no es suficiente para obtener una computadora útil porque se debe conocer la solución de todos los problemas que la computadora puede resolver para escribir su hamiltoniano explícitamente. Por otro lado, la *reversibilidad local* significa que si un paso hacia adelante involucra solo unos pocos bits, entonces un paso hacia atrás también involucrará solo unos pocos bits. Desde el punto de vista de la física esto significa que el hamiltoniano involucra solo interacciones entre pequeños grupos de partículas. En adelante, al hablar de reversibilidad o irreversibilidad, se estará haciendo referencia a la versión local del término.

Si se desea obtener funciones reversibles a partir de las irreversibles, puede observarse que toda función irreversible $f : \{0,1\}^m \to \{0,1\}^n$ puede incrustarse dentro de una función reversible, a partir de la función $\tilde{f} : \{0,1\}^{m+n} \to \{0,1\}^{m+n}$ tal que $\tilde{f}(x,y) = \big(x, [y + f(x)](mod 2^n)\big)$ donde x representa m bits, mientras que y y $f(x)$ representan n bits. Como f lleva entradas distintas a salidas distintas, resultará una función invertible de $(m+n)$ bits. Dado que se ha demostrado que un circuito booleano construido a partir de compuertas irreversibles AND OR y NOT puede calcular cualquier función $f : \{0,1\}^m \to \{0,1\}^n$, al usar esas mismas compuertas se puede construir cualquier circuito reversible. Además, ya que cualquier máquina de Turing también puede simularse mediante una máquina de Turing reversible, todos los circuitos booleanos pueden simularse utilizando solo compuertas reversibles. De hecho:

- La compuerta NOT ya es reversible, por lo que es posible usarla directamente en computación reversible.
- La compuerta AND es irreversible, pero es posible emular su acción a partir de una compuerta reversible.
- Puesto que $x_1 \vee x_2 = \neg(\neg x_1 \wedge \neg x_2)$, se puede reemplazar todas las compuertas OR en un circuito booleano por compuertas AND y NOT.

- La compuerta FANOUT, que representa múltiples cables que salen de una sola compuerta, se emula mediante una puerta reversible.

De existir las compuertas reversibles con capacidad para emular el comportamiento de AND y FANOUT, será posible encontrar compuertas reversibles universales para la computación. Mas aún, dado que NAND y FANOUT son el conjunto universal más pequeño de compuertas clásicas, todo lo que hará falta es probar que las compuertas universales NAND y FANOUT pueden construirse a partir de compuertas reversibles.

Para evitar la pérdida de información, una función reversible debe convertir n bits de entrada en n bits de salida. Las funciones reversibles son permutaciones de las entradas posibles de 2^n, en total hay $(2^n)!$ de ellas. Para $n = 1$ existen 2 compuertas reversibles de un solo bit: la compuerta identidad y la compuerta NOT. En el caso de compuertas de dos bits, una de las mas importantes es la llamada NOT-controlado (CNOT) o XOR reversible, que está definida en la Tabla 2.11.

Tabla 2.10: Tabla de verdad para *XOR* reversible o *NOT* controlado

a	b	a'	b'
0	0	0	0
0	1	0	1
1	0	1	1
1	1	1	0

Su forma de operar puede entenderse a partir de la observación del comportamiento del segundo bit (b) . Como puede verse, su valor cambia, o se invierte, si y solo si el primer bit (a) se establece en uno. En caso de que el primer bit sea cero, se quedará igual. Por esta razón, se dice que el primer bit actúa como control, mientras que el segundo es el objetivo. Puede verse que su valor no cambia en la salida: $a' = a$. El segundo bit recibe el nombre de objetivo. La inversión condicionada puede escribirse como $b' = a \oplus b$. Por lo tanto, en la salida, el segundo bit proporciona el XOR de las entradas a y b. Puede observarse que la puerta CNOT es reversible, ya que a partir de los valores de la salida (a', b') puede determinarse cuales fueron los valores de entrada(a, b). Desde el punto de vista de la operación XOR, el bit a' es innecesario, pero se vuelve útil para identificar la entrada a partir de la salida. Los bits de salida que solo se utilizan para mantener la reversibilidad se denominan *salidas basura*. También hay entradas que se fijan a alguna constante (0 o 1) que se utilizan en circuitos reversibles para almacenar valores intermedios durante el cálculo, a las que se denomina *bits ancilla* [222]. Por ejemplo, si se fija el bit de objetivo en 0 ($b = 0$), puede observarse a partir de la tabla de verdad que el valor del bit de control a se copia en b', es decir, $(a, 0) \rightarrow (a, a)$, por lo que la compuerta CNOT puede usarse como FANOUT mediante un bit ancilla.

Tabla 2.11: Tabla de verdad para *XOR* reversible o *NOT* controlado, con bit ancilla b fijado en 0

a	b	a'	b'
0	0	0	0
1	0	1	1

Las compuertas reversibles de dos bits no son suficientes para construir un conjunto universal de compuertas para la computación clásica [28]. Para ver que esto es así, puede verse que solo hay $4! = 24$ compuertas reversibles de dos bits, y todas ellas son lineales, es decir, todas pueden expresarse como $T(x) = Ax + b$, donde $b \in \{0,1\}^2$ y A es una matriz invertible sobre el campo binario por lo que cualquier función compuesta por ellas también será lineal. Sin embargo, existen puertas reversibles no lineales, como la puerta de Toffoli, que no pueden construirse a partir de ellas. La compuerta NAND reside dentro de una compuerta de tres bits. Esta es llamada NOT controlada-controlada (CCNOT), o Toffoli, y está definida en la Tabla 2.12. Esta puerta actúa de la siguiente manera: los dos bits de control no cambian ($a' = a$ y $b' = b$) mientras que el bit objetivo se invierte si y solo si los dos bits de control se establecen en 1, es decir, $c' = c \oplus ab$. Para probar que la puerta Toffoli es universal, basta construir las compuertas NAND y FANOUT. De hecho, si haciendo $a = 1$, la puerta Toffoli actúa sobre los otros dos bits como CNOT. La puerta FANOUT se puede construir a partir del CNOT. Para construir la puerta NAND, se establece $c = 1$, de modo que $c' = 0$ si y solo si $a = 1$ y $b = 1$, es decir, $c' = 1 \oplus ab = aNANDb$

Otra compuerta reversible universal es la compuerta de Fredkin o de INTERCAMBIO controlado, cuya tabla de verdad se muestra en la Tabla 2.13 Esta compuerta intercambia los bits de entrada b y c si y solo si el

Tabla 2.12: Tabla de verdad para una compuerta Toffoli o $CCNOT$

a	b	c	a'	b'	c'
0	0	0	0	0	0
0	0	1	0	0	1
0	1	0	0	1	0
0	1	1	0	1	1
1	0	0	1	0	0
1	0	1	1	0	1
1	1	0	1	1	1
1	1	1	1	1	0

bit de control a se establece en 1.

Tabla 2.13: Tabla de verdad para una compuerta Fredkin o de INTERCAMBIO

a	b	c	a'	b'	c'
0	0	0	0	0	0
0	0	1	0	0	1
0	1	0	0	1	0
0	1	1	0	1	1
1	0	0	1	0	0
1	0	1	1	1	0
1	1	0	1	0	1
1	1	1	1	1	1

Las compuertas irreversibles, como AND y OR, pueden integrarse en puertas reversibles. Sin embargo, el precio a pagar es la introducción de bits adicionales y, en la salida, esto produce bits *basura*, que no se reutilizan durante el cálculo. Estos bits extra son necesarios para almacenar la información que permitiría revertir las operaciones. Por ejemplo, si se establece $c = 1$ en la entrada de la puerta Toffoli, se obtendrá $c' = aNANDb$ más dos bits de basura ($a' = a$ y $b' = b$).

2.5.5. Álgebra lineal y el modelo de circuito

Dada una descripción de un circuito y una especificación de algunos valores de bits de entrada, si se pidiera la predicción del valor a la salida del circuito, la tarea que típicamente se realiza es rastrear el circuito de izquierda a derecha, actualizando los valores de los bits almacenados en cada uno de los cables después de cada compuerta. En otras palabras, se está siguiendo el *estado* de los bits de los cables a medida que avanzan por el circuito, lo que significa caracterizar a un fenómeno físico por un conjunto de variables. En el caso de una computador, cuando se habla de su estado en determinado punto, se hace referencia al estado de los bits de los cables en ese punto del circuito.

El estado asociado con un punto dado en un circuito determinista (no probabilístico) se puede especificar enumerando los valores de los bits en cada uno de los cables del circuito. El *estado* de cualquier cable en particular en un punto dado de un circuito es simplemente el valor del bit en ese cable (0 o 1). Sin embargo, para un circuito probabilístico esta simple descripción no es suficiente. Un solo bit que está en el estado 0 con probabilidad p_o y en el estado 1 con probabilidad p_1. Esta información puede condensarse mediante un vector bidimensional de probabilidades.

$$\begin{pmatrix} p_0 \\ p_1 \end{pmatrix} \tag{2.9}$$

Claramente, esta descripción puede extenderse a circuitos deterministas. Un cable en un circuito determinista cuyo estado es 0 podría especificarse mediante las probabilidades $p_0 = 1$ y $p_1 = 0$, con el vector correspondiente $\begin{pmatrix} 1 \\ 0 \end{pmatrix}$, mientras que si el estado es 1 podría especificarse mediante las probabilidades $p_0 = 0$ y $p_1 = 1$, con el vector correspondiente $\begin{pmatrix} 0 \\ 1 \end{pmatrix}$. Ya que se ha representado el estado de los cables, un conjunto de cables también tendría una representación vectorial. En consecuencia, las compuertas lógicas deberán

representarse mediante operadores que actúen sobre los vectores de estado de manera apropiada. Los operadores se describen convenientemente mediante matrices. Se puede considerar como ejemplo a la compuerta lógica NOT. Se define al operador (matriz) de tal forma que se comporte en los vectores de estado de manera consistente con el comportamiento de la puerta NOT. Si se sabe que un cable está en el estado 0 ($p_0 = 1$), la compuerta NOT le asignará el estado 1 ($p_1 = 1$) y viceversa. En términos de las representaciones vectoriales de estos estados:

$$NOT\begin{pmatrix}1\\0\end{pmatrix} = \begin{pmatrix}0\\1\end{pmatrix} \quad NOT\begin{pmatrix}0\\1\end{pmatrix} = \begin{pmatrix}1\\0\end{pmatrix} \tag{2.10}$$

Lo que indica que la representación matricial estará dada por :

$$NOT = \begin{bmatrix}0 & 1\\1 & 0\end{bmatrix} \tag{2.11}$$

Para *aplicar* la puerta a un cable en un estado dado, se multiplica al vector de estado correspondiente a la izquierda por la representación matricial de la puerta:

$$NOT\begin{pmatrix}p_0\\p_1\end{pmatrix} = \begin{bmatrix}0 & 1\\1 & 0\end{bmatrix}\begin{pmatrix}p_0\\p_1\end{pmatrix} \tag{2.12}$$

Para el caso en el que desee describir el estado asociado con un punto dado en un circuito probabilístico que tiene dos cables, suponiendo que el estado del primer cable en el punto dado es 0 con probabilidad p_0 y 1 con probabilidad p_1 y el estado del segundo cable en el punto dado es 0 con probabilidad q_0 y 1 con probabilidad q_1. Las cuatro posibilidades para el estado combinado de ambos cables en el punto dado son $\{00, 01, 10, 11\}$ (donde la cadena binaria ij indica que el primer cable está en el estado i y el segundo cable en el estado j). Las probabilidades asociadas con cada uno de los estados anteriores se obtienen multiplicando las probabilidades correspondientes a cada uno de los estados anteriores:

$$prob(ij) = p_i q_j \tag{2.13}$$

Lo que significa que el estado combinado de ambos cables se puede describir mediante el vector de probabilidades de 4 dimensiones:

$$\begin{pmatrix}p_0q_0\\p_0q_1\\p_1q_0\\p_1q_1\end{pmatrix} \tag{2.14}$$

Que también puede verse como el producto tensorial de los vectores bidimensionales para los estados del primer y segundo alambre por separado:

$$\begin{pmatrix}p_0q_0\\p_0q_1\\p_1q_0\\p_1q_1\end{pmatrix} = \begin{pmatrix}p_0\\p_1\end{pmatrix} \otimes \begin{pmatrix}q_0\\q_1\end{pmatrix} \tag{2.15}$$

Los productos tensoriales surgen naturalmente al considerar sistemas probabilísticos compuestos por dos o más subsistemas. Se vuelve necesario, además, tener una representación para las compuertas que actúan sobre más de un cable. Por ejemplo, CNOT. La compuerta CNOT se puede representar mediante la matriz

$$CNOT = \begin{bmatrix}1 & 0 & 0 & 0\\0 & 1 & 0 & 0\\0 & 0 & 0 & 1\\0 & 0 & 1 & 0\end{bmatrix} \tag{2.16}$$

Para un par de cables tales que el primer cable está en el estado 1 y el segundo en el estado 0. Esto significa que el vector de 4 dimensiones que describe el estado combinado del par de cables es

$$\begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} \tag{2.17}$$

Si se aplica la compuerta CNOT a este par de cables, con el primer cable como bit de control y el segundo como bit de destino, de acuerdo con la descripción de la puerta CNOT, se espera que el resultado sea que el bit de control (primer cable) permanezca en el estado 1 y el bit de destino (segundo cable) cambie al estado 1. Es decir, se espera que el vector de estado resultante sea

$$\begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} \tag{2.18}$$

algo que es sencillo de verificar a partir del producto

$$CNOT \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} \tag{2.19}$$

A pesar de que no se trata de un enfoque típico en cómputo clásico, la descripción de circuitos en términos de álgebra lineal es particularmente útil en computación cuántica.

2.6. Limitaciones de la computación clásica

Las computadoras clásicas ocupan un lugar importante en la sociedad moderna. El notable desarrollo tecnológico de principios de siglo es imposible de imaginar sin el empleo de las computadoras para resolver problemas. La evolución de las ideas que llevaron hasta su realización ha avanzado lentamente y no ha estado exenta de tropiezos. La descripción dada aquí no ha pretendido ser exhaustiva y mucho hay que decir sobre mecanismos utilizados en la antigüedad para realizar cálculos, como el mecanismo de Anticitera, o bien, de las computadoras analógicas que se usaron durante las grandes guerras del siglo XX. Pero lo que se ha mencionado ha sido con el propósito de explicar los principios elementales sobre los que se construye una computadora y entender los límites. En cada etapa de la computación, el soporte físico sobre el que operan las máquinas limita su operación. No únicamente en las discutidas cuestiones sobre lo que no puede hacer, si no que tiene una influencia en la precisión que puede entregar. Mas allá de las fronteras físicas, se planteó también la cuestión de aquellos cálculos que pueden realizarse efectivamente. Las ideas de complejidad ayudan a entender que hay circunstancias en las que el algoritmo para encontrar la solución de algún problema puede conocerse en principio, pero su ejecución podría llevar mas tiempo que el disponible. El descubrimiento de nuevos algoritmos podría cambiar eso y, sin embargo, no hay garantías de que dichos algoritmos puedan existir.

A pesar de los numerosos problemas resueltos por la computación y los avances tecnológicos que han transformado radicalmente nuestro mundo desde la introducción de las computadoras personales en la década de los 80, las computadoras se enfrenan a un serio límite. Todas las computadoras actuales, ya sean dispositivos móviles, televisores inteligentes, equipos de oficina o máquinas de última generación utilizadas en centros de investigación, se basan en el mismo principio fundamental: el modelo de la máquina de Turing. Como se mencionó, Turing estableció que cualquier tarea computacional puede ser realizada mediante una secuencia de operaciones elementales. A pesar de los avances tecnológicos y los componentes modernos, las computadoras siguen siguiendo este enfoque secuencial. En otras palabras, no son más sofisticadas que la Máquina Analítica de Babbage, diseñada por Charles Babbage en el siglo XIX. Por lo tanto, aunque las computadoras modernas son increíblemente poderosas y versátiles, su funcionamiento seguirá limitado por este principio fundamental.

Por ejemplo, si se desea encontrar una contraseña que consiste en una combinación de cuatro dígitos, se debe probar todas las posibles combinaciones de manera secuencial, sin atajos. Un enfoque lógico es comenzar con la combinación más baja posible 0000 y trabajar de forma iterativa a través de cada combinación posible. La computadora resuelve el problema probando todos los valores posibles uno tras otro.

Entonces, si la combinación es 4268, la computadora habrá probado 4269 combinaciones. En promedio, requerirá 5000 combinaciones y, en el peor de los casos, la combinación podría ser 9999, lo que le habría valido diez mil intentos. Estos números pueden parecer insignificantes frente a las computadoras modernas. Pero, si en lugar de una combinación de 4 dígitos se tuviera que descifrar una clave con un millón de combinaciones alfanuméricas posibles, tomaría mucho más tiempo obtener la combinación correcta. Los protocolos de seguridad actuales permiten cifrar información confidencial mediante el empleo de dos claves, una pública que codifica el mensaje y otra privada, que permite recuperar el contenido. La selección de ambas claves no es un mero accidente, el mecanismo con el que se encuentra en lo mas profundo de la Teoría de los Números y es un problema verdaderamente que aún no tiene solución. Descomponer a un número en un producto de primos puede parecer sencillo y de hecho, cualquier computadora puede realizar la tarea, pero resulta más que tedioso, para números verdaderamente grandes (mas de cuarenta cifras) llevar un tiempo mucho mas grande que la edad del universo, por lo que en la práctica se considera irrealizable. En general, no es posible resolver problemas grandes fuera de la clase **P** utilizando computadoras.

Simular sistemas físicos o químicos complejos resulta también ser un problema formidable para las computadoras actuales. En las últimas tres décadas, las simulaciones atómicas basadas en la solución de la ecuación básica de la mecánica cuántica han desempeñado un papel cada vez más importante en la predicción de las propiedades de materiales funcionales. Especialmente en el caso de materiales complejos y heterogéneos, la gran mayoría de las simulaciones de primeros principios se realizan utilizando la teoría funcional de la densidad, que en principio es exacta pero en la práctica requiere aproximaciones para permitir los cálculos, obteniendo bastante éxito en predecir numerosas propiedades de sólidos, líquidos y moléculas, y en proporcionar interpretaciones clave para una variedad de resultados experimentales. Sin embargo, resulta inadecuada para tratar los llamados *estados electrónicos fuertemente correlacionados*, aquellos que no pueden describirse mediante teorías estáticas de campo medio. Para tratarlos, se han desarrollado varios métodos teóricos y computacionales, incluida la teoría dinámica del campo medio y la Monte-Carlo cuántica5, además, los métodos *ab initio* de química cuántica, tradicionalmente desarrollados para moléculas, también se han aplicado recientemente a problemas de estado sólido. Desafortunadamente, estos enfoques son exigentes desde el punto de vista computacional y todavía resulta difícil aplicarlos a materiales complejos que contienen defectos e interfaces, incluso utilizando arquitecturas informáticas de alto rendimiento. En las computadoras clásicas, las moléculas con mucho menos de cien electrones fuertemente correlacionados ya están fuera del alcance de métodos ab initio sistemáticamente mejorables que podrían lograr la precisión requerida.

Buena parte del éxito de la integración de las computadoras a la vida cotidiana se debe a un continuo proceso de reducción en el tamaño de los componentes, algo que se denomina miniaturización. Las computadoras modernas están construidas a partir de microprocesadores y circuitos integrados, pero, en última instancia, estos componentes son, a su vez, diseñados a partir de transistores, en particular los llamados MOSFET. Los dispositivos semiconductores de procesamiento de información se encuentran entre las estructuras más sofisticadas, complejas y de alto rendimiento jamás construidos. Se ha observado, desde 1965, que los transistores reducen su tamaño continuamente, son, de manera aproximada, 2 veces más pequeños cada dos años, un fenómeno que se ha denominado Ley de Moore [201]. Se trata de una ley empírica que, a la que los fabricantes han tratado de adherirse, lo que ha conducido a una computación más eficaz y más barata y con ella, mejoras significativas en la productividad y la calidad de vida en general a través de la proliferación de computadoras, dispositivos de comunicaciones y otros productos electrónicos industriales y de consumo.

Sin embargo, esta tendencia no puede continuar indefinidamente. Por una parte, es obvio considerar que existe un límite fundamental que impide realizar reducciones que estén por debajo de la escala atómica. Pero, incluso mucho antes de llegar a ese punto, se presentan desafíos técnicos serios relacionados con la miniaturización. Para muestra de esto, puede considerarse, en primer lugar, que el costo de producción se ha incrementado de manera exponencial [31], a una tasa mayor que el incremento de la capacidad de las computadoras, algo que se ha denominado *segunda ley de Moore*. Aunque el tamaño del transistor todavía sigue la ley de Moore, los beneficios observados en los productos realizados han caído por debajo de las expectativas anteriores. Una de las razones principales es que la alta velocidad de la ley de Moore ha llevado a un uso menos cuidadoso de la memoria y del número de microinstrucciones de la máquina necesarias para llevar a cabo una función básica. Estos dos fenómenos combinados pueden disminuir el incentivo a las empresas que insisten en mantener el desarrollo conforme a lo esperado por la primera ley de Moore, llevándola a su fin, porque la promesa de una tecnología que apenas resultará ser marginalmente mejor que la actual no podría justificar las enormes inversiones que representan su investigación, desarrollo y manufactura.

También es necesario considerar que todo cómputo realizado dentro de un sistema físico está sujeto, en última instancia, a las leyes de la física y, por lo tanto, serán estas últimas las que dicten el límite final a la capacidad de toda computadora. Entre ellos:

- **La velocidad de la luz es la máxima en el universo**. Ninguna señal podrá propagarse por encima de este valor. Aunque la velocidad de la luz es alta, el límite que esto impone a la velocidad de cómputo se alcanzará muy pronto. Con una velocidad de procesamiento de 100 GHz, la luz puede recorrer una distancia máxima de 1.5 mm, en el vacío, y regresar durante un ciclo. De hecho, la velocidad de propagación de la señal será menor y también se necesitará tiempo para recopilar información de la memoria. Esto impone como requisito a toda arquitectura informática de alta velocidad la necesidad de que los componentes se encuentren cercanos (localizados), para minimizar la latencia en comunicaciones y tener un acceso a la memoria correcto.

- **Principio de Landauer**. Todas las computadoras comerciales de hoy operan de forma irreversible. Toda energía que se use para realizar cálculos cambiando el estado físico del medio, termina transformándose en calor, lo que limita la potencia de procesamiento. Considerando la tesis de Landauer como válida, la energía mínima necesaria para borrar información es $kT\ln 2$ y, por lo tanto, la cantidad mínima de calor generada por una operación a temperatura ambiente es 3×10^{-21} J [161]. Pero, además, cambiar el estado lógico de un circuito CMOS requiere transferir una cierta cantidad de carga (del orden de 3×10^{-18}J en 2015, [170]), que regresa al terminal de tierra y la energía de carga se convierte en calor. Si los cálculos fueran completamente reversibles, la energía, incluida la energía termodinámica y de carga, podría recuperarse.

- **Rapidez y energía de conmutación**. En muchos otros aspectos, la tecnología CMOS ya está relativamente cerca de los límites físicos. La temperatura de funcionamiento limita la velocidad máxima alcanzable de actualización para lógica binaria tradicional, es decir, la velocidad máxima del reloj o frecuencia de funcionamiento. Este límite surge del hecho de que la energía mínima requerida para un interruptor binario ($kT\ln 2$) debe ser mayor que la incertidumbre energética $\Delta E = hf/2$. Por lo tanto, a temperatura ambiente, no será posible obtener velocidades de reloj superiores a 8.7 THz [244].

- **Disipación de energía**. La disipación de energía en circuitos con un gran número de dispositivos es proporcional a la cantidad de recursos, considerando tanto dispositivos como interconexiones, la frecuencia operativa, la probabilidad de que un dispositivo esté activo durante un ciclo de reloj y a la disipación de energía promedio por ciclo de reloj por recurso. Con los materiales utilizados actualmente, la emisión de $P = NfpE$ es de aproximadamente 100 W/cm^2. Por ejemplo, con $N = 10^{12}$ recursos en un chip de 1 cm^2 y estableciendo pE en 50 veces el límite termodinámico de temperatura ambiente kT, la frecuencia operativa máxima sería $f = 2$ GHz [209].

El conocimiento de estas limitaciones, como la necesidad de impulsar las capacidades de la computación al punto de poder resolver problemas prácticos tan complejos como el de la fijación de nitrógeno han motivado el desarrollo de un nuevo paradigma de la computación, uno que vaya mas allá de la física clásica, donde existen fenómenos que podrían mejorar la manera en que se realizan cálculo: la Mecánica Cuántica.

3

Mecánica cuántica

Las propiedades especiales de los objetos cuánticos son la superposición, la interferencia y el entrelazamiento [265]. La superposición se refiere a partículas que existen en todos los estados posibles simultáneamente. La interferencia es la situación en la que la intervención del ruido del entorno afecta al objeto cuántico, y también la posibilidad de que las funciones de onda de las partículas puedan reforzarse o disminuirse entre sí. El entrelazamiento significa que grupos de partículas están conectados y pueden interactuar de tal forma que es imposible describir de manera el estado cuántico de cada partícula de manera independiente del estado de las demás, aun y cuando exista una gran distancia entre ellas. Esta clase de propiedades son contrarias a toda intuición, por lo que su existencia solo se ha corroborado tras la preparación cuidadosa de experimentos sumamente ingeniosos.

3.1. Introducción

Al final del siglo XIX permeaba una sensación de satisfacción dentro de la comunidad científica. Todo cuanto había que conocer sobre la física parecía cubierto. Una y otra vez, mediante dispositivos cada vez más sensibles, las leyes de Newton se habían podido comprobar. El éxito de la dinámica se transformó en el punto de partida de todo intento por comprender la naturaleza de forma consistente y profunda, a tal punto que se empleó como modelo para el desarrollo de otras ciencias. Por otra parte, el trabajo de Maxwell había logrado unir dentro de un marco robusto y elegante a fenómenos como la electricidad y el magnetismo. Como muestra de su alcance, el punto culminante lo alcanzan los experimentos de Hertz en los que se investigan y observan las ondas electromagnéticas predichas por la teoría de Maxwell. Finalmente, una gran variedad de fenómenos podía explicarse de manera unificada y satisfactoria a través de las leyes de la termodinámica y la teoría cinética.

Sin embargo, el comienzo del siglo XX estuvo marcado por una profunda transformación que parecía involucrar a todas las actividades humanas, desde la economía hasta las artes y la física no quedaría exenta. De manera aparentemente repentina, varios experimentos arrojaron resultados que no podían explicarse en términos de las bien establecidas teorías físicas. Así, aunque las propiedades de las ondas electromagnéticas de Maxwell y Hertz se entendían bien, los experimentos para estudiar las propiedades del medio en que se transmitían estas ondas resultaron infructuosos. Ni los experimentos realizados para analizar las ondas electromagnéticas producidas por objetos calientes ni aquellos que trataban sobre la emisión de protones en superficies iluminadas pudieron explicarse a través de la termodinámica y el electromagnetismo. Es posible que este par de experimentos parezcan poco significativos, sobre todo si se les compara con muchos experimentos exitosos y bien comprendidos del siglo XIX. Sin embargo, su efecto sería tanto profundo como duradero, y no se limitaría a la física, sino que influiría en toda la ciencia en general, en la estructura política del mundo, e incluso, en la manera en la que el ser humano se percibe. En el corto período de tiempo que significan dos décadas al compararlas con los trescientos años que habían pasado desde la publicación de los trabajos de Newton, los resultados de estos experimentos llevarían hasta la teoría especial de la relatividad

y la teoría cuántica. Y poco después se vería el desarrollo de la física atómica, la nuclear y relativa al estado sólido, cuyas aplicaciones han impactado profundamente en la vida cotidiana.

Durante las tres primeras décadas del siglo XX ocurrió un profundo cambio en la comprensión de los fenómenos que tienen lugar en el reino microscópico, que no tiene comparación en toda la historia de las ciencias naturales. Mientras que por un lado se evidenciaron severas fallas y limitaciones en la validez de la física clásica, por otro lado se desarrollo la teoría alternativa, una teoría que pudo remplazar a las teorías anteriores, con un rango de aplicabilidad mucho mas amplio. La base de esta teoría alternativa radicaba en la suposición de que existen cantidades fundamentales e imposibles de dividir. Esta idea fue presentada el 14 de diciembre de 1900, durante una reunión de la Sociedad Alemana de Física, con la lectura de «La teoría de la ley de distribución de energías del espectro normal», presentada por Max Planck. Durante los próximos 27 años, se desarrollaría la mecánica cuántica moderna, para hacer frente a la un conjunto de hechos que no encontraron explicación en el marco de la mecánica clásica. Es una teoría general, aplicable a todo, desde partículas subatómicas hasta cúmulos de galaxias. Pero el interés se centra naturalmente en aquellos fenómenos que llevaron a su descubrimiento. A continuación, se presentan algunos de los experimentos que revelan la naturaleza cuántica del universo y cuyas propiedades se aprovechan en el diseño de componentes y algoritmos en cómputo cuántico.

3.2. Experimento de Stern Gerlach

Durante las últimas décadas del siglo XIX y las primeras del siglo XX, muchos científicos estaban trabajando, sin saberlo aún, en investigaciones que ahora son consideradas antecedentes para el desarrollo de la mecánica cuántica. Se perseguían objetivos distintos, por lo que, aparentemente, no había conexión entre ellas. Esta es la razón por la que el modelo atómico de Niels Bohr, presentado en 1913 [207], se considera una muestra de su genialidad. Tuvo la capacidad para integrar una amplia gama de conocimientos de diferentes áreas de la física y pudo formular la primera hipótesis concreta (y audaz), sobre la estructura de la materia. De acuerdo con su modelo atómico, los electrones se encontraban girando alrededor de un núcleo denso en órbitas *cuantizadas*, es decir, que tenían niveles de energía específicos. Con esta idea sentó las bases para comprender la estabilidad de la materia y la formación de enlaces químicos

En el modelo Bohr se utilizaron ciertas condiciones, llamadas «cuánticas», para seleccionar ciertas órbitas circulares «permitidas» en las que se encontraban los electrones dentro de un átomo. El electrón solo tenía un grado de libertad, por lo que un «número cuántico» resultaba suficiente para describir el movimiento del electrón. A este se le llamó *número cuántico principal*, n. Arnold Sommerfeld perfeccionó este modelo al darse cuenta de que una condición más generalizada, aplicada a las variables de acción, sería consistente con la condición de Bohr y también permitiría órbitas elípticas, lo que agregaría un grado de libertad adicional y un nuevo numero cuántico[261]. Esta condición equivalía a la cuantificación del espacio de fase ocupado por una órbita. Según la teoría de Sommerfeld, el electrón se mueve en órbitas elípticas, que están cuantificadas en cuanto a su magnitud y forma. Posteriormente agregaría una contribución crucial, según la cual los electrones no solo tendrían un momento angular cuantificado, sino que la orientación de sus órbitas está restringida a ángulos específicos. De acuerdo con esto, el ángulo que forma el plano en el que se encuentra la órbita del electrón no puede tomar valores arbitrarios, sino que esta inclinación está cuantizada, o, de manera equivalente, afirmar que la componente z del momento angular está cuantizada. A esto se le denominaba *Richtungsquantelung*, cuya traducción significa «cuantización del espacio», refiriéndose con ello a la dirección. La cuantización del espacio proporcionó una explicación para el efecto Stark, el efecto Zeeman ordinario y la estructura fina del espectro del hidrógeno pero no era capaz de explicar el efecto Zeeman anómalo, por lo que no podía superar al modelo clásico propuesto por Zeeman y Lorentz. Si esta cuantización era algo más que un artificio matemático, un gas de átomos hidrogenoides dentro de un campo magnético debería presentar birrefringencia óptica porque el electrón orbitaría en un plano perpendicular a la dirección del campo, algo que no se había podido observar experimentalmente[101].

Otto Stern y Walther Gerlach partieron de la idea de que la cuantización del espacio era una realidad física. Estimulados por el deseo de observar una propiedad implícita en la cuantización del espacio que no había sido reportada, se dedicaron a diseñar y realizar un experimento que les haría pasar a la historia. Stern recordó que la cuestión de la birrefringencia se planteó en un seminario [101]. En ese momento trabajaba bajo la dirección de Max Born a quien le comentó su idea, pero se encontró con una dura respuesta:

> Me tomó bastante tiempo antes de tomarme esta idea en serio. Siempre pensé que la cuantización [del espacio] era una especie de expresión simbólica de algo que no se entiende. Pero tomar esto literalmente como lo hizo Stern, esta era su propia idea ... Traté de persuadir a Stern de que no tenía sentido, pero luego me dijo que valía la pena intentarlo. [39]

Pese a la negativa de Born, Stern pudo encontrar apoyo en Gerlach, quien hasta ese momento no tenía conocimiento sobre la cuestión de la cuantización espacial, pero tenía experiencia con experimentos de haces atómicos debido a su interés en las propiedades de los átomos en los sólidos magnéticos. En particular, Gerlach quería comprobar si los átomos presentaban momentos magnéticos, para lo cuál estaba considerando un experimento en el que un haz de átomos de bismuto pudiera recorrer una región bajo la influencia de un campo magnético que no fuera homogéneo. En caso de que el átomo tuviera un momento magnético dipolar, experimentaría un par de fuerza, debido al campo magnético y, por lo tanto, tendría que rotar. Pero si el campo magnético no es uniforme, la fuerza en un extremo de un dipolo será mayor que el par en el extremo opuesto, lo que conduciría a una fuerza neta sobre el átomo, que se desviaría mientras viaja a través del campo magnético no homogéneo. Finalmente, el tamaño de la desviación revelaría la magnitud del momento magnético del átomo. Stern se dio cuenta de que tal experimento sería una excelente manera de probar la cuantización del espacio. Si los átomos tienen momentos magnéticos que pueden apuntar en cualquier dirección (como sugeriría la física clásica), entonces el haz de átomos se ensancharía continuamente a medida que pasa a través del campo magnético no homogéneo. Sin embargo, si los momentos magnéticos del átomo están cuantizados en el espacio, apuntando en direcciones opuestas (arriba y abajo) a lo largo del campo no homogéneo, entonces el haz de átomos se dividiría en dos. Como resultado, los átomos hacia arriba y hacia abajo se desviarían en direcciones opuestas, proporcionando una clara evidencia de la cuantización del espacio. La observación de birrefringencia constituiría una demostración de la cuantificación del espacio, un fenómeno fuera del ámbito de la física clásica que implicaría la necesidad de reemplazar a la física clásica con la física cuántica.

3.2.1. Descripción

La importancia de este experimento radica en su capacidad de demostrar la insuficiencia de la mecánica clásica para describir los fenómenos físicos que ocurren a escala microscópica. Las conclusiones a las que conducen sus resultados obligan a abandonar la descripción clásica de la naturaleza, y obligan a pensar en términos de la mecánica cuántica. El fenómeno observado demuestra un comportamiento que es común a los sistemas cuánticos, donde las predicciones clásicas quedan sin efecto.

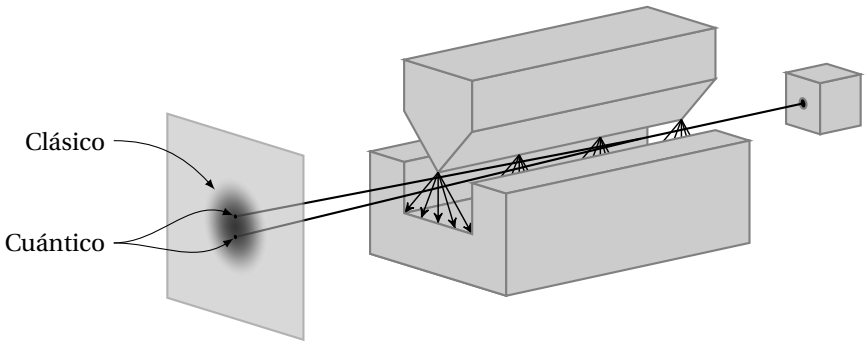


Figura 3.1: Experimento de Stern-Gerlach. En la configuración experimental para medir el momento magnético intrínseco de los electrones, un haz de átomos de plata expulsados de un horno en la dirección y se envía a través de un imán de Stern-Gerlach orientado en la dirección z. El campo magnético no homogéneo creado por el imán de Stern-Gerlach desvía los átomos a lo largo de la dirección z de acuerdo con la componente z de sus momentos magnéticos, que es causada por el momento magnético intrínseco (espín) de sus electrones de valencia únicos. Contrariamente a la expectativa clásica, que sugiere una distribución continua, el espín del electrón y, por tanto, la distribución en la pantalla se cuantifica en dos valores discretos.

En el experimento de Stern-Gerlach, se hierve plata en un horno. La razón para usar este metal específicamente es que sus átomos tienen un comportamiento semejante al de los átomos de hidrógeno, que poseen solo un electrón, con la ventaja de que el bajo punto de fusión de la plata facilita la creación de un haz de átomos, y el hecho de que presenta una respuesta fuerte a los campos magnéticos tiene una fuerte respuesta

a los campos magnéticos[93]; además de que puede detectarse en placas fotográficas. Los átomos de plata escapan por una abertura hacia una región evacuada, desde donde viajan en línea recta, de manera horizontal a través de rendijas para colimarles. Se considera que este recorrido se hace a lo largo de la dirección Y. Después, se hace que el haz de átomos viaje por un campo magnético orientado en la dirección Z, que será no homogéneo. Al finalizar la interacción, los átomos impactan en una placa metálica en donde serán detectados.

La electrodinámica clásica predice que los átomos deberían sufrir una desviación al pasar por la región del campo magnético, proporcional al gradiente del campo en la dirección Z y a la componente z del momento magnético de los átomos. Para un gradiente de campo constante, la desviación del átomo permitirá determinar la componente del momento magnético en la dirección del campo (μ). Al salir del horno, cada átomo tendrá un momento magnético que puede estar dirigido en cualquier dirección, por lo que se esperan desviaciones de diferentes magnitudes. Al final, se esperaría que la componente z de m tome todos los posibles valores entre $+\mu$ y $-\mu$, y por lo tanto, los átomos acabarían distribuyéndose de manera uniforme sobre la placa fotográfica, lo que resultaría en la producción de una sola figura. Sin embargo, el experimento reveló dos haces claramente distinguibles. A la misma distancia del centro, se encontraron dos figuras de igual intensidad pero en sentidos opuestos, lo que reveló que los átomos estaban sufriendo únicamente dos posibles desviaciones después pasar a través del campo magnético no homogéneo, es decir, la componente z del momento angular del electrón tiene únicamente dos valores posibles. El valor obtenido es múltiplo de la unidad fundamental de momento angular.

3.2.2. Acoplando dispositivos de Stern-Gerlach

Mas allá de la primera implicación del experimento, se trata de un dispositivo de utilidad, que se construye de la siguiente manera. Dado un dispositivo de Stern-Gerlach cuyo campo magnético está orientado en la dirección vertical, es decir en la dirección Z dSG(Z). Los átomos de plata corren sobre el eje Y, como en el experimento original y serán dispersados sobre en arriba (+Z) y abajo (-Z). El haz se separa en dos componentes, que corresponden a los estados de espían arriba y espín abajo de los electrones. Estas componentes se identifican mediante $|+\rangle_z$ y $|-\rangle_z$. Si se utiliza una barrera, se podrá evitar que los los átomos con componente Z de espín negativa continúen, absorbiéndoles, mientras que aquellos de componente positiva continuarán su camino si interferencias. Este dispositivo, en lo sucesivo denominado dSG(+Z), actuará como filtro, permitiendo la selección de átomos cuya componente de espín en la dirección Z sea positiva. Esto se comprueba porque, en caso de colocar un dSG(Z) después del primer dSG(+Z), solo se observa un haz correspondiente a la componente $|+\rangle_z$.

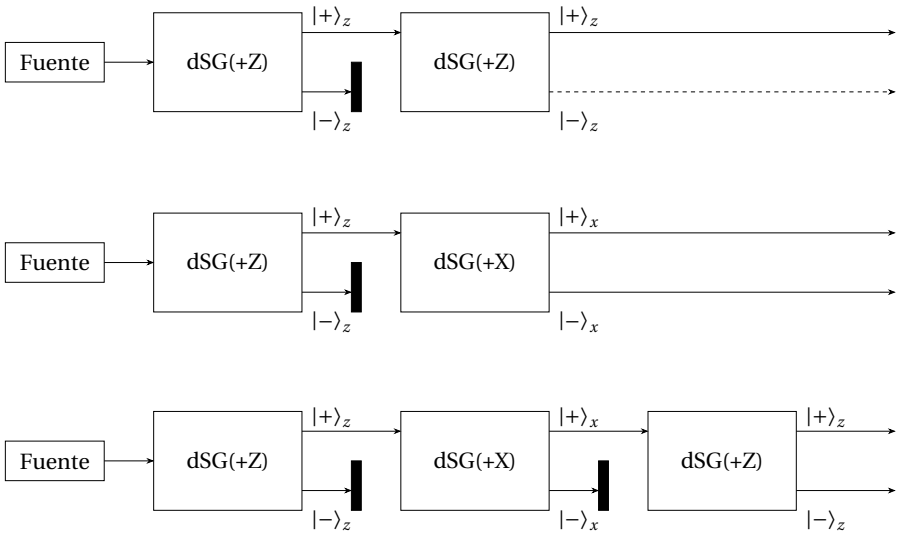


Figura 3.2: Experimentos secuenciales de Stern-Gerlach. El haz atómico atraviesa dos o más aparatos SG en secuencia. El primer arreglo es relativamente sencillo. Se somete al haz procedente del horno a la disposición que se muestra en la figura superior, dSG(+Z) representa un aparato con un campo magnético no homogéneo en la dirección Z. Posteriormente, se bloquea la componente $|-\rangle_z$: que sale del primer aparato y se deja que la componente restante, $|+\rangle_z$, se someta a otro aparato. Esta vez solo sale un componente del haz del segundo aparato, $|+\rangle_z$. La disposición que se muestra en la figura central muestra al primer aparato dSG(+Z), igual que antes, pero el segundo tiene un campo magnético no homogéneo en la dirección X. El haz $|+\rangle_z$ que ingresa al segundo aparato (dSG(+X)) ahora se divide en dos componentes, un componente $|+\rangle_x$ y una componente $|-\rangle_x$, con intensidades iguales.

Si los átomos que salieron del dispositivo se hacen pasar a través de otro dSG orientado en una dirección X (dSG(X)), es decir, perpendicular al primer dispositivo, se observará que dos haces de igual intensidad emergen de este aparato, que corresponden a $\mu_x = m$ y $\mu_x = -m$, donde μ_x denota la proyección del momento magnético de los átomos en el eje X. Estas componentes se identifican con los símbolos $|+\rangle_x$ y $|-\rangle_x$. De la misma forma que se hizo antes, es posible bloquear el paso de los átomos correspondientes a la componente X de espín negativa, formando un dSG(+X). Los átomos supervivientes tendrán a la componente X de espín positiva, algo que puede comprobarse si se les hace pasar nuevamente por otro dSG(X).

Esto puede parecer poco interesante, de no ser por un extraño y alarmante detalle. Después de hacer que un conjunto de átomos pase por un dSG (+Z) y un dSG(+X), se hace pasar por un nuevo dSG (Z). Al haber «seleccionado» a los átomos con componentes +Z y +X, no se esperaría que un último dSG(Z) separara el haz en dos partes. Sorprendentemente, la mitad de los átomos salen con componente +Z, mientras que la otra mitad saldrá con componente -Z. Al volver a filtrar el haz de átomos, para seleccionar aquellos que tuvieran un valor positivo para la componente X de espín, se ha perdido toda información referente a la seleción anterior, para la componente Z. Esto indica que no se puede seleccionar conjuntos de átomos de plata que tengan bien definidas, al mismo tiempo, dos componentes distintas de espín o, dicho de otra forma, las componentes de espín en dos direcciones distintas son observables incompatibles.

3.3. Experimento de doble rendija de Young

3.3.1. La naturaleza de la luz

Las características distintivas de la mecánica cuántica están bien representadas por otro experimento, conocido como el *Experimento de la doble rendija* y ligado al nombre Young. Como se sabe, la naturaleza de la luz ha protagonizado un amplio y profundo debate, que permaneció durante siglos. Las primeras propuestas, en principio antagónicas, pretendían explicar a la luz como un haz de partículas o como una onda. La concepción corpuscular fue adoptada y defendida por Isaac Newton, ya que creía que la presencia de una sombra bien definida detrás de un objeto no podía explicarse si la luz fuera una onda. La luz no es una onda, como el sonido, que se puede escuchar incluso detrás de los objetos que lo producen, o a la vuelta de la esquina. A pesar de estos argumentos, que apelan a la experiencia cotidiana, la luz presenta interferencia, una característica que la teoría corpuscular no podría explicar. Los experimentos que los demuestran fueron presentados en el siglo XIX, tras lo cual, otras propiedades que al principio se creyeron exclusivas de las partículas, pudieron ser explicadas en términos de ondas. Thomas Young formuló el principio de superposición: si dos ondas, emitidas por una misma fuente, se proyectan sobre una pantalla, la amplitud de la onda resultante será la suma algebráica de las amplitudes de cada onda incidente. Esta propiedad da origen a las franjas de interferencia que pueden apreciarse en un experimento de doble rendija. La luz es emitida por la fuente S, pasa por dos rendijas, O_1 y O_2, e ilumina una pantalla. Las franjas de interferencia típicas se producen en la pantalla. El punto principal es que la *intensidad* $I(x)$ de la luz en la pantalla (el módulo al cuadrado de la amplitud de la luz) no es igual la suma algebraica de las intensidades $I_1(x)$ e $I_2(x)$ producidas por las dos rendijas por separado, es decir, cuando se cierra la rendija O_2 o O_1, respectivamente.

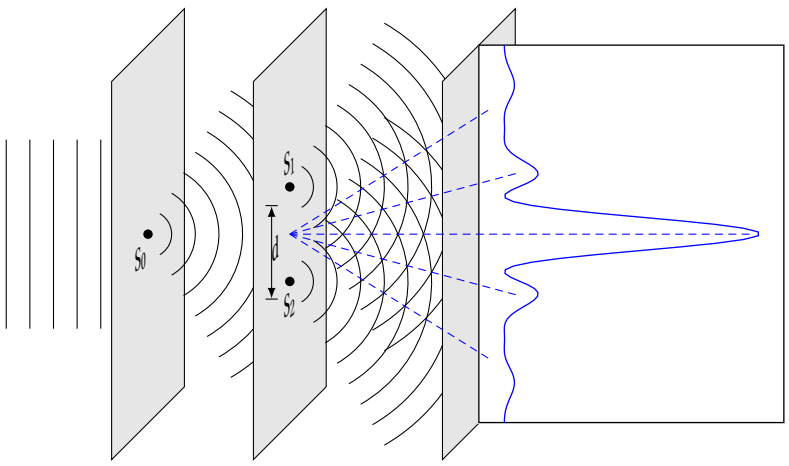


Figura 3.3: Diagrama del experimento de interferencia de doble rendija de Young. Una fuente S_0 emite luz, que puede pasar a través de dos rendijas S_1 y S_2, antes de incidir en una pantalla. La intensidad $I(x)$ de la luz en la pantalla cuando ambas rendijas están abiertas es diferente de la suma algebraica de las intensidades $I_1(x)$ y $I_2(x)$.

En la segunda mitad del siglo XIX, tras el monumental trabajo realizado por Maxwell, quedó claro que la luz es una onda electromagnética. Esta teoría predecía la existencia de ondas que se propagan con una velocidad $c = 2.998 \times 10^8$m/s en el vacío, que coincide con la velocidad medida para la luz. A su vez, está relacionada con ciertas constantes eléctricas y magnéticas. Sin embargo, como ya se adelantó en la introducción a este capítulo, la distribución de energía correspondiente a la radiación de un cuerpo negro no podía explicarse con la teoría electromagnética. Este problema llevó a Planck en 1900 a introducir la hipótesis de que la luz se emite o se absorbe solo en múltiplos enteros de una unidad básica de energía $E = h\nu$, donde h representa a una constante, la *constante de Planck*, y ν es la frecuencia de la luz. Einstein, al explicar el efecto fotoeléctrico, recurría nuevamente a la teoría de corpuscular, al explicar el comportamiento de la luz en términos de partículas, ahora llamadas fotones. Cada fotón tendría una energía $h\nu$. Del estudio de la interacción luz-materia, se concluye que tales partículas tienen un momento $p = h\nu/c$. Tanto la teoría de Maxwell, que describe la luz como una onda electromagnética, como la teoría de Einstein, que describe la luz como un haz de partículas elementales, recibieron una amplia confirmación experimental y esto había resucitado al antiguo debate sobre la naturaleza de la luz.

3.3.2. Motivación del experimento

El experimento de Young de doble rendija toma relevancia debido a que solo es posible obtener una descripción completa del la luz si se aceptan tanto los aspectos de onda como de partícula de manera simultanea.

La relevancia crucial del experimento de la doble rendija de Young radica en el hecho de que solo se puede obtener una descripción completa aceptando simultáneamente los aspectos de onda y partícula de la luz. La intensidad luminosa $I(x)$ en un punto x de la pantalla es proporcional al módulo al cuadrado del campo eléctrico $E(x)$ en el mismo punto. Si se denota por $E_1(x)$ y $E_2(x)$ al campo eléctrico producido en la posición x por los rayos que pasan a través de las rendijas O_1 y O_2, respectivamente. Las intensidades de luz correspondientes están dadas por $I_1(x) = a|E_1(x)|^2$ que se observa cuando se cierra la segunda rendija, y $I_2(x) = a|E_2(x)|^2$, obtenido cuando se cierra la primera rendija. Si se abren ambas rendijas, las intensidades de campo se suman algebráicamente, por lo que $E(x) = E_1(x)+E_2(x)$ y la intensidad de la luz resultante estará dada por $I(x) = a|E(x)|^2 = |E_1(x) + E_2(x)|^2$, de donde es posible observar que $I(x) = I_1(x) + I_2(x)$. Esto está de acuerdo con las predicciones de la teoría ondulatoria de la luz. Sin embargo, en el caso extremo en que la intensidad de la luz se reduce lo suficiente como para emitir fotones individuales, sucederá que cada fotón producirá un impacto localizado en algún punto de la pantalla. Si se expone la placa fotográfica durante un tiempo lo suficientemente pequeño como para que únicamente algunos pocos fotones golpean la pantalla, se observan algunos puntos de impacto localizados, pero no un patrón de interferencia. Este resultado requiere de una interpretación corpuscular de la luz porque, si se tratara como onda, se esperaría que la intensidad de las franjas de interferencia disminuyeran pero sin desaparecer, algo muy distinto a lo observado.

Sin embargo, si el tiempo de exposición es lo suficientemente largo como para que la placa fotográfica pueda capturar muchos fotones, las franjas de interferencia aparecerán. La intensidad de la luz recogida en cualquier punto dado es proporcional a la densidad de impactos de fotones en ese punto. Por lo tanto, los fotones, a medida que llegan, acumulan las franjas de interferencia y estas franjas pueden explicarse mediante una interpretación de ondas en lugar de una interpretación de partículas de la luz. En resumen, no es posible explicar la totalidad de los resultados experimentales usando solo las predicciones de la teoría de partículas o solo las de la teoría de ondas. Tanto la naturaleza corpuscular como la ondulatoria de la luz están presentes.

Aquí es importante señalar que, colocando un contador de fotones cerca de una de las dos rendijas, es posible determinar por qué rendija pasa cada fotón, pero al hacerlo se destruyen las franjas de interferencia. El punto crucial es que es físicamente imposible observar el patrón de interferencia y determinar a través de qué rendija pasa cada fotón. Estos resultados nos obligan a revisar algunos de los conceptos fundamentales de la física clásica. El hecho de que se observen franjas de interferencia si y sólo si no se sabe por qué rendija pasa cada fotón obliga a abandonar el concepto de trayectoria. De hecho, cada fotón producido por la fuente golpea la pantalla en una posición diferente. No es posible predecir esta posición, solo se tiene la probabilidad $p(x)$ que un fotón incide sobre la pantalla en el punto x. Esta probabilidad es proporcional a la intensidad $I(x)$, es decir, a $|E(x)|^2$. Aunque todos los fotones se emiten en las mismas condiciones, no es posible conocer de antemano dónde incidirá cada fotón en la pantalla. Por lo tanto, se debe rechazar el concepto clásico de que, una vez conocidas las condiciones iniciales y las fuerzas externas que actúan sobre una partícula, se puede seguir, al menos en principio, la evolución en el tiempo de las coordenadas de la partícula.

También se debe abandonar el concepto de que las descripciones de partículas y ondas son mutuamente excluyentes, ya que ambos aspectos son necesarios para explicar los resultados experimentales. Por

lo tanto,conduce inevitablemente al concepto de *dualidad onda-partícula*. Es importante destacar que un número abrumador de experimentos demuestran que tal dualidad no se limita a la descripción de los fenómenos ópticos. De hecho, las partículas materiales también pueden exhibir propiedades ondulatorias. La comprobación experimental de este punto es una versión del experimento de doble rendija de Young aplicado a la interferencia de electrones individuales.

3.3.3. Distintas versiones del experimento de interferencia

La mayoría de las discusiones sobre los experimentos de doble rendija con partículas se refieren a la cita de Feynman:

> “Elegimos examinar un fenómeno que es imposible, absolutamente imposible, de explicar de forma clásica, y que tiene en sí mismo el corazón de la mecánica cuántica. En realidad, contiene el único misterio... Deberíamos decir de inmediato que no debe intentar configurar este experimento. Este experimento nunca se ha hecho de esta manera. El problema es que el aparato tendría que fabricarse a una escala increíblemente pequeña para mostrar los efectos que nos interesan. Estamos haciendo un «experimento mental», que hemos elegido porque es fácil de pensar. Sabemos los resultados que se obtendrían porque hay muchos experimentos que se han hecho, en los que se ha elegido la escala y las proporciones para mostrar los efectos que escribiremos [92].”

A pesar del considerable detalle con el que Feynman describe el experimento de electrones, y otros experimentos de doble rendija con agua (para representar el comportamiento ondulatorio) y balas (que representan el comportamiento corpuscular), llega a sorprender que no destaque la formación de un patrón de interferencia en el caso en que un solo electrón se encuentre en el aparato. Tampoco menciona que en el primer experimento de doble rendija con electrones se había llevado a cabo en 1961, quizá porque fue el mismo año en el que comenzó a dictar sus conferencias y no estaba al tanto (aunque estas se publicarían hasta 1963). El nombre asociado al experimento de doble rendija se debe a que Thomas Young llevó a cabo su experimento original de doble rendija con luz en algún momento de la primera década del siglo XIX y demostró que las ondas de luz de las dos rendijas interferían para producir un patrón de franjas característico en una pantalla. Así que, si la luz está realmente compuesta de partículas, como sugieren la radiación del cuerpo negro y el efecto fotoeléctrico, hay que dar una explicación del experimento de la doble rendija basada en partículas. J.J. Thomson sugirió en 1907 que los patrones de luz observados en el experimento de la doble rendija podrían ser el resultado de diferentes fotones que de alguna manera interfieren entre sí. Thomson sugirió así que si la intensidad de la luz se redujera lo suficiente, los fotones de la luz se separarían ampliamente y el patrón de interferencia podría desaparecer. En 1909, Geoffrey Ingram Taylor se propuso probar esta sugerencia para lo cual realizó un experimento en el que se reducía la intensidad de la luz se reducía drásticamente, hasta el punto de que una de las imágenes tardaba tres meses en formarse. Aún en este caso, que «equivalente a una vela encendida a una distancia ligeramente superior a una milla», pudo observar que se formaban las franjas de interferencia[274]. Dado que los resultados de Taylor sugieren que la interferencia persiste incluso cuando los fotones están muy separados, puede considerarse que los fotones no interfieren entre sí, lo que conduce a la famosa afirmación de Dirac «Cada fotón interfiere sólo consigo mismo» [78]. Para decirlo de otra manera, dado que no hay interferencia cuando solo hay una rendija, los resultados de Taylor sugieren que cada fotón individual pasa a través de ambas rendijas.

En 1927, Clinton Davisson y Lester Germer diseñaron y construyeron un aparato de vacío, con la intención de confirmar el comportamiento ondulatorio de los electrones (que debería observarse, de acuerdo con la hipotesis de De Broglie) mediante la difracción de electrones. A través de la medición de la energía de los electrones dispersados desde una superficie de metal pudieron observar la difracción en haces de electrones dentro de un cristal de níquel, con lo que demostraron, primera vez en la historia, las propiedades ondulatorias de las partículas [72]. George Thompson hizo lo mismo con películas delgadas de celuloide y otros materiales poco después [277]. Davisson y Thomson compartieron el premio Nobel de 1937 por el «descubrimiento de los fenómenos de interferencia que surgen cuando los cristales se exponen a haces electrónicos» [224], pero ninguno realizó un experimento de doble rendija con electrones. A principios de la década de 1950, Ladislaus Laszlo Marton demostró la interferencia de electrones, pero para hacerlo empleó un interferometro de Mach-Zehnder en lugar de realizar el experimento con doble rendija, aduciendo lo siguiente:

> Es bastante sencillo concebir un interferómetro que funcione con haces de electrones basándose en el experimento óptico de luz equivalente de Young. En principio, se podría emplear el método de la doble rendija, pero cálculos simples, basados en analogías ópticas ligeras, indican que las dimensiones y complejidades de tal instrumento son bastante indeseables. Debido a la longitud de onda corta, el tamaño de la fuente, el tamaño de las rendijas y su separación, así como la separación de las franjas, se vuelven tan pequeños que puede ser necesario un gran esfuerzo experimental para abordarlos. En vista de estas dificultades, vale la pena explorar las posibilidades de una interfaz de haz amplio [183].

Cuatro años despúes, Gottfried Möllenstedt y Heinrich Düker utilizaron un *biprisma* de electrones para dividir un haz de electrones en dos componentes y observar la interferencia entre ellos [75]. Sería hasta 1961 que Claus Jönsson realizara *por primera vez un experimento real de doble rendija con electrones* mostrando la interferencia hasta con cinco rendijas [141]. La cuestión de si un solo electrón puede interferir consigo mismo no sería comprobada hasta quince años mas tarde, cuando Pier Giorgio Merli, Giulio Pozzi y Gian-Franco Missiroli llevaron a cabo experimentos de interferencia de doble rendija con electrones individuales [190]. Este experimento se narra en una película llamada Interferencia electrónica, que recibió el premio en la categoría de física en el Festival Internacional de Cinematografía Científica de Bruselas en 1976 [242]. No obstante, es más conocido el experimento de Hitachi de 1989, en el que Akira Tonomura y sus compañeros usaron una fuente de electrones muy debil y un biprisma, con el que pudieron observar la acumulación del patrón de franjas, producido [279].

3.4. Postulados de la mecánica cuántica

3.4.1. Motivación

Toda teoría física está basada en un conjunto de leyes, que se consideran *fundamentales*, mismas que se formulan dentro de un marco matemático que les da rigor, estructura y herramientas para tratar problemas. Debe contar, además, con un conjunto de reglas que permita establecer correspondencia entre los objetos físicos y los conceptos matemáticos empleados para representarlos. Estas reglas de correspondencia suelen usarse en primer lugar para expresar un problema físico en términos matemáticos. Y una vez que se formula la versión matemática del problema, será posible resolverlo mediante el uso de técnicas exclusivamente matemáticas, que no requerirán ninguna interpretación física adicional. De obtenerse una solución formal, esta puede llevarse de nueva cuenta al régimen físico a través de las reglas de correspondencia.

En ocasiones, esta correspondencia entre los entes físicos y matemáticas es muy obvia y no requerirá de mayor consideración para comprenderse. Un ejemplo, muy simple, lo constituye la asociación entre la posición de una partícula, que es un concepto físico, con un número real, el concepto matemático, usado para hablar de su magnitud. Estrictamente hablando, la noción de un número real en matemáticas puras no es trivial, sin embargo, la regla de correspondencia antes enunciada pude ser entendida intuitivamente por la mayoría de las personas sin riesgo de confusión. Pero, al contrario de lo que ocurre con la mecánica clásica, el formalismo matemático de la mecánica cuántica resulta mucho más abstracto y está desprovisto de todo apoyo a la intuición. En particular, los postulados estándar de teoría cuántica se expresan en términos matemáticos abstractos que involucran *espacios de Hilbert* y *operadores hermitianos* que actúan sobre ellos, para los que parece no haber un significado físico claro, al menos a primera vista.

Así, mientras que en otras teorías físicas, como la relatividad especial o la termodinámica, el formalismo puede derivarse de postulados que tienen un significado físico directo, a menudo expresado en términos de la posibilidad o imposibilidad de realizar ciertas tareas, la descripción cuántica resulta extraña porque el mundo no parece estar formado por operadores hermitianos y vectores de estado de dimensión infinita, por lo que resulta necesario prestar atención cuidadosa y explícita a las reglas de correspondencia que relacionan el formalismo matemático abstracto con la realidad observable. Si bien, la teoría cuántica proporciona la base sobre la cual se asienta la mayoría de las teorías físicas y la comprensión de la naturaleza, este papel preponderante contrasta con una comprensión limitada de la teoría cuántica en sí misma y la falta de consenso entre los físicos sobre lo que dice esta teoría acerca de cómo funciona la naturaleza.

La primera formulación matemática consistente de la mecánica cuántica fue la llamada *mecánica matricial*[134], que fue propuesta conjuntamente por Werner Heisenberg, Max Born y Pascual Jordan . En una serie de artículos [37, 38, 123] proponen una teoría sistemática de la mecánica cuántica basada en matrices. Esta teoría describe las propiedades físicas de las partículas como matrices, en donde los observables evolu-

cionan con el tiempo. De manera independiente, y con un enfoque distinto, en 1926, Erwin Schrôdinger [246] formuló una ecuación diferencial parcial que describe la evolución temporal del estado de los sistemas cuánticos cerrados, llevando a la formulación matemática del concepto físico de dualidad onda partícula, según el cual las partículas subatómicas poseen propiedades de ondas electromagnéticas y partículas clásicas al mismo tiempo. La función de onda resultante determina al sistema cuántico en cada posición espacial y temporal.

En la formulación moderna, ambas propuestas se conocen como *representaciones*. La representación de Heisenberg y la representación de Schrödinger parten, originalmente, de marcos matemáticos distintos, pero, esencia, difieren solo en si la evolución se aplica al estado o al observable, respectibamente. En 1930 y 1932 respectivamente, Paul Dirac y John von Neumann demostraron que la mecánica de matrices y la ecuación de Schrödinger eran dos enfoques diferentes de la misma teoría. Propusieron los llamados axiomas de Dirac-von Neumann proporcionando una descripción teórica de la mecánica cuántica en términos de operadores en un espacio de Hilbert [78][287].

A pesar del enorme éxito de la teoría cuántica, su significado e interpretación son, todavía, objeto de debate [186]. Diferentes libros de texto abordan distintos enfoques y emplean postulados que pueden diferir en cantidad, orden e incluso interpretación, pero, esencialmente, tienen el mismo contenido. Una forma de abordar la presentación de los postulados puede encontrarse en [62], en donde se usa como apoyo el hecho de que la relación entre conceptos físicos y matemáticos parece evidente en mecánica clásica. La descripción clásica de un sistema físico se puede resumir de la siguiente manera:

1. El estado del sistema en un tiempo fijo t_0 se define especificando N coordenadas generalizadas $q_i(t_0)$ y sus N momentos conjugados $p_i(t_0)$
2. El valor, en un momento dado, de las diversas cantidades físicas está completamente determinado cuando se conoce el estado del sistema en ese momento: conociendo el estado del sistema, uno puede predecir con certeza el resultado de cualquier medición realizada en cualquier tiempo t_0
3. La evolución temporal del estado del sistema viene dada por las ecuaciones de Hamilton-Jacobi. Al tratarse de ecuaciones diferenciales de primer orden, su solución $q_i(t), p_i(t)$ es única si el valor de estas funciones en un momento dado es fijo, $q_i(t_0), p_i(t_0)$. El estado del sistema se conoce para siempre si se conoce su estado inicial.[62]

Si bien diferentes obras y autores emplean una cantidad variable de postulados como base para su estudio de la mecánica cuántica[264], todos pretenden responder, de una forma u otra, las siguientes cuestiones (que corresponden a los tres puntos enumerados anteriormente para la descripción clásica):

- ¿Cómo se describe matemáticamente el estado de un sistema cuantitativo en un momento dado?
- Dado este estado, ¿cómo predecir los resultados de la medición de varias cantidades físicas?
- ¿Cómo se puede encontrar el estado del sistema en un tiempo arbitrario t cuando se conoce el estado en el tiempo t_0?

Estas preguntas pueden responderse a través de los siguientes postulados[62]:

> **Primer Postulado:** En un tiempo fijo t_0 el estado de un sistema físico aislado se define especificando un *ket* $|\psi(t_0)\rangle$ perteneciente al espacio de estados E[62].

El conjunto de estados E resulta ser un espacio vectorial y un *ket* será, por tanto, un vector en dicho espacio. Por tanto, este postulado implica que el principio de superposición es válido en esta descripción, po lo que una combinación lineal de vectores de estado será un vector de estado.

> **Segundo Postulado:** Toda cantidad física medible A se describe mediante un operador A que actúa en E; este operador es un observable [62].

La mecánica cuántica posee un descripción que se distingue fundamentalmente de la clásica: un estado se representa mediante un vector mientras que para una cantidad física se usa un operador.

Tercer Postulado: El único resultado posible de la medida de una cantidad física A es uno de los valores propios del correspondiente observable A [62].

Una medida de A siempre da un valor real, ya que A es por definición Hermitiano. Si el espectro de A es discreto, los resultados que se pueden obtener midiendo A están cuantizados.

Dado un sistema cuyo estado se caracteriza, en un momento dado, por el ket $|\psi\rangle$, que se supone normalizado a 1:

$$\langle\psi|\psi\rangle = 1 \tag{3.1}$$

Si se desea predecir el resultado de la medida, en este momento, de una cantidad física A asociada al observable A. Esta predicción es de tipo probabilístico. El siguiente postulado, que cubre casos diferentes, indica las reglas que permiten calcular la probabilidad de obtener cualquier valor propio dado de A. Si el espectro de A es completamente discreto, y todos los valores propios a_n de A son no degenerados, hay asociado con cada uno de ellos un vector propio único $|u_n\rangle$ (dentro de un factor constante)

$$A|u_n\rangle = a_n|u_n\rangle \tag{3.2}$$

Como A es un observable, el conjunto de $|u_n\rangle$, normalizado, constituye una base en E, y el vector de estado $|\psi\rangle$ se puede escribir:

$$|\psi\rangle = \sum_n c_n|u_n\rangle \tag{3.3}$$

Cuarto Postulado (caso de un espectro discreto no degenerado): Cuando la cantidad física A se mide en un sistema en el estado normalizado $|\psi\rangle$, la probabilidad $p(a_n)$ de obtener el valor propio no degenerado a_n del correspondiente observable A es:

$$p(a_n) = |\langle u_n|\psi\rangle|^2 \tag{3.4}$$

donde $|u_n\rangle$ es el vector propio normalizado de A asociado con el valor propio a_n [62].

Ahora bien, si algunos de los valores propios a_n son degenerados, a ellos les corresponden varios vectores propios ortonormales:

$$A\left|u_n^i\right\rangle = a_n\left|u_n^i\right\rangle, i = 1,2,\dots g_n \tag{3.5}$$

$|\psi\rangle$, todavía se puede expandir sobre la base ortonormal $\{|u_n^i\rangle\}$:

$$|\psi\rangle = \sum_n \sum_{i=1}^{g_n} c_n^i\left|u_n^i\right\rangle \tag{3.6}$$

lo que lleva al siguiente postulado.

Cuarto Postulado (caso de un espectro discreto): Cuando la cantidad física A se mide en un sistema en el estado normalizado $|\psi\rangle$, la probabilidad $p(a_n)$ de obtener el valor propio a_n del correspondiente observable A es:

$$p(a_n) = \sum_{i=1}^{g_n} |\left\langle u_n^i\middle|\psi\right\rangle|^2 \tag{3.7}$$

donde g_n es el grado de degeneración de a_n y $\{u_n^i\}(i = 1,2,g_n)$ es un conjunto ortonormal de vectores que forma una base en el subespacio propio E asociado con el valor propio a_n de A [62].

Cuarto Postulado (caso de un espectro continuo no degenerado): Cuando la magnitud física A se mide sobre un sistema en estado normalizado $|\psi\rangle$, la probabilidad $dp(a_n)$ de obtener un resultado comprendido entre a y $a+da$ está dada por

$$dp(a_n) = |\langle v_a|\psi\rangle|^2 da \tag{3.8}$$

donde $|v_a\rangle$ es el vector propio correspondiente al valor propio a del observable A asociado a A [62].

Quinto Postulado: Si la medida de la cantidad física A en el sistema en el estado$|\psi\rangle$ da como resultado a_n, el estado del sistema inmediatamente después de la medida es la proyección normalizada, $\frac{P_n|\psi\rangle}{\sqrt{\langle\psi|P_n|\psi\rangle}}$, de $|\psi\rangle$ sobre el subespacio propio asociado con a_n. El estado del sistema inmediatamente después de la medición resulta siempre un vector propio de A con valor propio a_n [62] .

Sexto Postulado: La evolución temporal del vector de estado $|\psi(t)\rangle$ se rige por la ecuación de Schrödinger:

$$i\hbar\frac{d}{dt}|\psi(t)\rangle = H(t)|\psi(t)\rangle \tag{3.9}$$

donde $H(t)$ es el observable asociado a la energía total del sistema. H se denomina operador hamiltoniano del sistema, ya que se obtiene a partir del hamiltoniano clásico [62].

Finalmente, se debe mencionar el interesante trabajo por disminuir la cantidad de postulados, mostrando que algunos de ellos son redundantes y pierden, por tanto, su calidad de postulados[54] y otros esfuerzos más por deducir los postulados de la mecánica cuántica de teorías más generales. En este trabajo no se pretende introducir un nuevo conjunto de axiomas o reducir algún sistema ya existente, sino presentar aquellos que resultan necesarios para la posterior discusión de cómputo cuántico. Nuevamente, se parte de algunas preguntas que servirán de guía para enunciar los postulados

- ¿Qué es un sistema cuántico?
- ¿Cómo se puede representar formalmente el estado del sistema en cualquier momento?
- ¿Cuáles son los equivalentes de las variables y medidas clásicas?

La información relevante contenida en los postulados anteriores puede condensarse en tres [21], que se presentan a continuación.

3.4.2. Evolución dinámica

Postulado I (PI) : Cada sistema físico S está asociado con un espacio de Hilbert $\mathscr{H}_S$ en el campo complejo. Un estado dado del sistema se describe completamente mediante un vector unitario $|\psi\rangle$, que se denomina vector de estado, o función de onda, en el espacio de Hilbert $\mathscr{H}_S$[21].

La evolución temporal del vector de estado $|\psi\rangle$ de un sistema cuántico cerrado se rige por la ecuación de Schrödinger

$$i\hbar\frac{d}{dt}|\psi(t)\rangle = H(t)|\psi(t)\rangle \tag{3.10}$$

donde $H(t)$ es un operador autoadjunto conocido como *hamiltoniano* del sistema y $\hbar = h/2\pi$. Se trata de una ecuación diferencial lineal y de primer orden en el tiempo, por lo que conocer el estado inicial $|\psi(t_0)\rangle$, permite determinar de manera única y total cualquier estado $|\psi(t)\rangle$ para toda t mediante su solución. Además, al tratarse de una ecuación lineal, el principio de superposición implica que si se tiene un par de soluciones linealmente independiente $|\psi_1(t)\rangle$ y $|\psi_2(t)\rangle$ entonces toda combinación lineal de ellas $\alpha|\psi_1(t)\rangle + \beta|\psi_2(t)\rangle$, con α y β complejos, también será una solución. En el caso de un hamiltoniano independiente del tiempo $H(t_0)$, la solución es

$$|\psi(t)\rangle = \exp\left[-i/\hbar H(t_0)(t-t_0)\right]|\psi(t_0)\rangle \tag{3.11}$$

La generalización de este resultado, para el caso en el que el hamiltoniano depende del tiempo, requiere de un operador, llamado operador de evolución temporal $U(t_0,t)$, definido de manera que lleve del estado $|\psi(t_0)\rangle$ al estado $|\psi(t)\rangle$:

$$|\psi(t)\rangle = U(t,t_0)|\psi(t_0)\rangle \tag{3.12}$$

Dado que la ecuación de Schrödinger dependiente del tiempo preserva la norma de los estados, es necesario que el operador $U(t_0,t)$ sea unitario.

Si $t > t_0$, puede considerarse que el hamiltoniano $H(t)$ es constante a trozos, es decir $H(t) = H(t_p)$ para $t_p < t < t_{p+1}$, con t_p la cantidad de veces que el hamiltoniano cambia repentinamente, es decir, el número de

trozos. Si se conoce el ket al tiempo inicial t_0, es posible calcular el ket en al tiempo $t \in (t_n, t_{n+1})$, mediante la aplicación repetida de $\left|\psi(t)\right\rangle = \exp\left[-i/\hbar H_0(t-t_0)\right]\left|\psi(t_0)\right\rangle$

$$\left|\psi(t)\right\rangle = e^{-i/\hbar H(t_n)(t-t_n)}\left|\psi(t_n)\right\rangle \tag{3.13}$$
$$= e^{-i/\hbar H(t_n)(t-t_n)}e^{-i/\hbar H(t_{n-1})(t_n-t_{n-1})}\left|\psi(t_{n-1})\right\rangle \tag{3.14}$$
$$= e^{-i/\hbar H(t_n)(t-t_n)}e^{-i/\hbar H(t_{n-1})(t_n-t_{n-1})}...e^{-i/\hbar H(t_0)(t_-t_0)}\left|\psi(t_0)\right\rangle \tag{3.15}$$

El operador que actúa sobre $\left|\psi(t_0)\right\rangle$ es unitario ya que es el producto de operadores unitarios. Es importante observar el orden de los operadores: el exponencial calculado con el hamiltoniano $H(t_p)$ está a la izquierda de todos los exponenciales calculados con $H(t_q)$ si $t_p > t_q$, es decir, si t_p es posterior a t_q. Para asegurar este orden, es necesario introducir el operador de ordenamiento cronológico u operador de ordenamiento temporal $\mathscr{T}$. A pesar de su nombre, $\mathscr{T}$ no es un operador en el sentido habitual, sino más bien una regla que establece cómo reorganizar los productos de los operadores. Se define el producto ordenado cronológicamente de m hamiltonianos en los tiempos $t_m > ... > t_1$ como

$$\mathscr{T}[H(t_{p_{(m)}})H(t_{p_{(m-1)}})...H(t_{p_{(1)}})] = H(t_m)H(t_{m-1})...H(t_1) \tag{3.16}$$

para toda las permutaciones P de $(1, ..., m-1, m)$. La acción de $\mathscr{T}$ es arreglar los factores, tomando cualquier producto de operadores y cambiando el orden, de modo que cada operador tenga solo operadores posteriores a la izquierda y operadores anteriores a la derecha, es decir, para cualesquiera operadores dependientes del tiempo $A(t_1)$ y $B(t_2)$ se tiene:

$$\mathscr{T}[A(t_1)B(t_2)] = \begin{cases} A(t_1)B(t_2) & \text{si } t_1 > t_2 \\ B(t_2)A(t_1) & \text{si } t_1 < t_2 \end{cases} \tag{3.17}$$

Si los intervalos de tiempo están igualmente espaciados, es decir, si $t_p = t_0 + p\Delta_t$, y mediante la fórmula de Baker-Campbell-Hausdorff para dos operadores que conmutan, es posible demostrar [262] que la solución a la ecuación de Schrödinger con $H(t)$ se puede escribir como

$$\left|\psi(t)\right\rangle = \mathscr{T}\, exp\Big[-i/\hbar\int_{t_0}^{t} d\tau H(\tau)\Big] \tag{3.18}$$

3.4.3. Resultado de una medición

En mecánica clásica, los observables son una cantidad o propiedad del estado del sistema, que puede ser medida (observada), mediante una secuencia de acciones u operaciones físicas, por ejemplo, leer los valores indicados en un equipo para medición: dichos observables se representan por funciones $f(x, p)$ que dependen de la posición y el momento (espacio fase). En cambio, en mecánica cuántica, los observables requieren más estructura, la definición de observable cuántico está incluida en segundo postulado de la mecánica cuántica.

Postulado II (PII): Cualquier observable físico A está asociado con un operador autoadjunto A en el espacio de Hilbert $\mathscr{H}_S$. El posible resultado de una medición del observable A es uno de los valores propios del operador A. Escribiendo la ecuación de valores propios,

$$A\left|i\right\rangle = a_i\left|i\right\rangle \tag{3.19}$$

donde $\left|i\right\rangle$ es una base ortonormal de vectores propios del operador A, y expandiendo el vector de estado $\left|\psi(t)\right\rangle$ sobre esta base:

$$\left|\psi(t)\right\rangle = \sum_i c_i(t)\left|i\right\rangle \tag{3.20}$$

entonces la probabilidad de que una medida del observable A en el tiempo t resulte en el resultado a_i está dada por

$$p_i = \left|\left\langle i\middle|\psi(t)\right\rangle\right|^2 = |c_i(t)|^2 \tag{3.21}$$

[21].

Esta regla básica sobre cómo se puede extraer información de un estado cuántico fue enunciada por primera vez por Born. Proporciona el vínculo entre las amplitudes y los números que realmente pueden

obtenerse tras realizar una medición. En aras de la simplicidad, se establece el PII para el caso en el que los valores propios de A no son degenerados. Los observables son el análogo cuántico de las variables dinámicas en la mecánica clásica. como la posición, el momento lineal y angular. En cambio, otras características de un sistema, como la masa o la carga eléctrica, no pertenecen a la clase de observables, sino que se consideran como parámetros del hamiltoniano del sistema.

Ya que los valores propios de un operador autoadjunto son reales (al igual que los posibles resultados de una medición) y sus vectores propios forman un conjunto ortonormal y completo en el espacio de Hilbert $\mathscr{H}_S$ asociado con el sistema, los operadores autoadjuntos están asociados con observables físicos. Como $\big|\psi(t)\big\rangle$ tiene norma unitaria, resulta

$$\Sigma_i p_i(t) = \Sigma_i |c_i(t)|^2 = 1 \tag{3.22}$$

Por lo que las probabilidades están normalizadas, es decir, la probabilidad total de obtener un resultado a partir de la medida del observable A es igual a 1. Es precisamente por esta razón que PI requiere que $\big|\psi(t)\big\rangle$ tenga norma unitaria.

En el caso particular en que el vector de estado $\big|\psi(t_0)\big\rangle$ en un momento dado t_0 coincide con un vector propio del operador A con valor propio a_i

$$\big|\psi(t_0)\big\rangle = |i\rangle \tag{3.23}$$

entonces una medida del observable A en el tiempo t_0 da, con probabilidad uno, el resultado a_i Los vectores propios del operador A también se denominan estados propios de A.

Suponiendo que $\big|\psi_1(t)\big\rangle$, y $\big|\psi_2(t)\big\rangle$ son dos vectores propios normalizados distintos del operador A, con valores propios a_1 y a_2, respectivamente. El principio de superposición dice que el estado

$$\big|\psi\big\rangle = c_1\big|\psi_1\big\rangle + c_2\big|\psi_2\big\rangle \tag{3.24}$$

con números complejos c_1 y c_2, también es un estado permitido del sistema, siempre que $|c_1|^2 + |c_2|^2 = 1$, de modo que $\big|\psi\big\rangle$tenga norma unitaria. Por tanto, si el sistema está descrito por el vector de estado $\big|\psi\big\rangle$ y se realiza una medida del observable A, se obtendrían los resultados a_1 o a_1 con probabilidad $|c_1|^2$ y $|c_2|^2$, respectivamente.

Caso conservativo

Si el Hamiltoniano H del sistema no tiene dependencia explícita del tiempo, significa que se trata de un sistema conservativo y, de acuerdo con la mecánica clásica, su energía será una constante de movimiento. En el caso cuántico, la solución a la ecuación de Schrödinger se escribe en términos de los valores y vectores propios del operador hamiltoniano H. La ecuación de valores propios para H

$$H|n\rangle = E_n|n\rangle \tag{3.25}$$

(caso de espectro no degenerado $E_n \neq E_m$ para $n \neq m$). Dado que H no depende del tiempo, los valores propios E_n y los vectores propios $|n\rangle$ tampoco dependerán de él. La solución $\big|\psi(t)\big\rangle$ de la ecuación de Schrödinger se puede expandir sobre la base de las funciones propias del operador H como sigue:

$$\big|\psi(t)\big\rangle = \sum_n c_n(t)|n\rangle \tag{3.26}$$

con $c_n(t) = \big\langle n\big|\psi(t)\big\rangle$.

La solución $\big|\psi(t)\big\rangle$ está determinada únicamente por la condición inicial $\big|\psi(t_0)\big\rangle$, donde es posible tomar $t_0 = 0$. La condición inicial $\big|\psi(t_0)\big\rangle$ está determinada si los coeficientes $c_n(0) = \big\langle n\big|\psi(0)\big\rangle$ están fijos. Si se usa esta expansión en la ecuación de Schrödinger, se obtiene:

$$i\hbar\frac{d}{dt}c_n = E_n c_n \tag{3.27}$$

cuya solución es

$$c_n(t) = c_n(0)e^{-i(E_n/\hbar)t} \tag{3.28}$$

por lo tanto, el vector de estado $\big|\psi(t)\big\rangle$ estará dado por

$$\big|\psi(t)\big\rangle = \sum_n c_0 e^{-(iE_n/\hbar)t}|n\rangle \tag{3.29}$$

En el caso especial en que $|\psi(0)\rangle$ coincide con un vector propio $|n\rangle$ del operador hamiltoniano H, debido a la ortogonalidad entre los diferentes vectores propios, la solución a la ecuación de Schrödinger se reduce a

$$|\psi(t)\rangle = e^{-(iE_n/\hbar)t}|n\rangle \tag{3.30}$$

Por lo tanto, los vectores de estado $|\psi(0)\rangle$ y $|\psi(t)\rangle$ solo difieren en un factor de fase global sin importancia física (la probabilidad no se ve afectada por el factor de fase). Por esta razón, los estados propios de un hamiltoniano H independiente del tiempo se denominan estados estacionarios: si un sistema se describe mediante dicho estado, sus propiedades físicas no cambian con el tiempo.

3.4.4. Estado después de la medición

Como siguiente punto, se debe analizar el efecto del proceso de medición sobre un estado del sistema. Al medir un observable A se obtiene como resultado a_n, un valor propio no degenerado del operador autoadjunto A. Si la medición no destruye al sistema y sigue inmediatamente una nueva medición del observable A, se obtiene nuevamente el resultado a_n con probabilidad uno. Este resultado experimental puede explicarse si se admite que la función de onda del sistema, en el instante inmediato anterior a la primera medición se encontraba en el estado $|\psi\rangle$, inmediatamente después de la medición colapsa en el estado propio $|n\rangle$ de A asociado con el valor propio a_n. Si se trata de un caso degenerado, es posible expandir al estado $|\psi\rangle$ antes de la medición de la siguiente manera:

$$|\psi\rangle = \sum_n \sum_{s=1}^{g_n} c_{n_s}|n_s\rangle \tag{3.31}$$

donde g_n mide el orden de degeneración del valor propio a_n, es decir, la dimensión del subespacio generado por los vectores propios de A con el mismo valor propio a_n. Después de una medición que da como resultado a_n, el estado del sistema pertenece a este subespacio y está dado por

$$|\psi\rangle = \frac{1}{\sqrt{\sum_{s=1}^{g_n}|c_n|^2}} \sum_{s=1}^{g_n} c_{n_s}|n_s\rangle \tag{3.32}$$

Este estado es la proyección normalizada de $|\psi\rangle$ sobre el subespacio correspondiente al valor propio a_n generado por los vectores propios de A con valor propio a_n. Con estas consideraciones, es posible introducir el siguiente postulado:

Postulado III (PIII): Si un sistema está descrito por el vector de onda $|\psi\rangle$ y se mide un observable A, obteniendo a_n, el estado inmediatamente después de la medición es:

$$|\psi\rangle = \frac{P_n|\psi\rangle}{\sqrt{\langle\psi|P_n|\psi\rangle}} \tag{3.33}$$

donde , P_n es el operador de proyección sobre el subespacio correspondiente a a_n [21].

Si el vector de onda $|\psi\rangle$ viene dado por $\sum_n \sum_{s=1}^{g_n} c_{n_s}|n_s\rangle$ entonces el operador de proyección o proyector P_n representa

$$P_n = \sum_{s=1}^{g_n} |n_s\rangle\langle n_s| \tag{3.34}$$

Dado que los vectores propios de A constituyen una base ortonormal para el espacio de Hilbert $\mathscr{H}_S$ asociado con el sistema, los proyectores P_n satisfacen la relación de completitud,

$$\sum_n P_n = I \tag{3.35}$$

y la condición de ortogonalidad

$$P_n P_m = \delta_{mn} P_m \tag{3.36}$$

En el caso sin degeneración $g_n = 1$, la función de onda del sistema después de la medición colapsa en el estado $\frac{1}{|c_n|}c_n|n\rangle$ y por lo tanto en el estado propio$|n\rangle$ correspondiente al valor propio a_n, despreciando un

factor de fase global sin significado físico. Si el sistema está descrito por el vector de onda, entonces, al medir el observable A, la probabilidad de obtener cualquier resultado dado p_n está dada por

$$p_n = \langle\psi| P_n |\psi\rangle \tag{3.37}$$

Para p_n se recupera PII, en el caso no degenerado ($g_n = 1$).

$$\langle A\rangle = \sum_n a_n p_n \tag{3.38}$$

Si ahora se obtiene el valor promedio del observable A, se obtiene

$$\langle A\rangle = \sum_n a_n \langle\psi| P_n |\psi\rangle = \langle\psi| \left(\sum_n a_n P_n\right) |\psi\rangle = \langle\psi| A |\psi\rangle \tag{3.39}$$

donde se emplea la descomposición espectral $A = \sum_n a_n P_n$. La desviación estándar ΔA asociada con las observaciones de A viene dada por

$$\Delta A = \sqrt{\langle \langle A - \langle A\rangle\rangle^2 \rangle} = \sqrt{\langle A^2\rangle - \langle A\rangle^2} \tag{3.40}$$

Por lo tanto, si se realiza una gran cantidad de experimentos en los que se prepara el estado $|\psi\rangle$ y se mide el observable A, se obtendrían resultados con valor medio $\langle A\rangle$ y desviación estándar ΔA.

3.5. Principio de incertidumbre de Heisenberg

Aunque Heisenberg publicó la la primera formulación matemática coherente de la teoría cuántica, la teoría alternativa se Schrodinger se considera la teoría más intuitiva y era favorecida por muchos físicos de esa época. En 1927, Heisenberg publicó un artículo [124] para demostrar que las predicciones de la mecánica matricial que conducen a las relaciones de incertidumbre de Heisenberg, no deben considerarse contraintuitivas, sino que deben considerarse integradas en cada medición. En el artículo, Heisenberg presentó un experimento mental para medir la posición de un electrón con un microscopio que utiliza rayos gamma de alta energía para iluminación. Reduciendo la longitud de onda de los rayos gamma y aumentando el diámetro del objetivo del microscopio, se puede medir la posición del electrón con la precisión deseada. Suponiendo una óptica limitada por difracción, el valor de la incertidumbre asociada a la medición de la posición será del orden de $\Delta x \sim \lambda/(2\sin(\theta))$. Sin embargo, un rayo gamma dispersa el electrón cuya posición se está midiendo. En el ángulo sólido subtendido por el objetivo del microscopio en la posición de este electrón, la conservación de la energía y del momento requiere que el electrón retroceda, lo que produce una incertidumbre en el momento del electrón dispersado, ya que el rayo gamma puede dispersarse en cualquier ángulo dentro del cono de aceptación del objetivo.

La incertidumbre en la componente x del momento es del orden de $\Delta p_x \sim (2h/\lambda)\sin(\theta)$. Por lo tanto, el experimento mental del microscopio de Heisenberg conduce a un producto de incertidumbres $\Delta x \Delta p_x \sim h$, para dos variables mecánico-cuánticas conjugadas, establece que la precisión con la que dos de esas variables pueden medirse simultáneamente está sujeta a la restricción de que el producto de las incertidumbres en las dos mediciones sea al menos de orden h, es decir, conforme se aumenta la precisión de la medición para la velocidad de la partícula, disminuye la precisión para la posición, es decir, aumenta la incertidumbre en la posición, y viceversa.

Este experimento mental contiene la noción de que la relación de incertidumbre es el resultado de una perturbación del electrón durante el proceso de medición, lo que puede llevar a suponer que sin esta perturbación el electrón podría tener una posición y un impulso bien definidos. Esto entra en conflicto con la comprensión actual de la mecánica cuántica. El principio de incertidumbre se aplica a todos los objetos cuánticos y no debe verse únicamente como el resultado de que no sea posible realizar una medición precisa, la limitación no debe a imperfecciones de los instrumentos o a errores humanos. Es una característica intrínseca y se expresa mediante las relaciones de incertidumbre posición-momento de Heisenberg:

$$\Delta x \Delta P_x \geq \hbar/2 \qquad \Delta y \Delta P_y \geq \hbar/2 \qquad \Delta z \Delta P_z \geq \hbar/2 \tag{3.41}$$

donde Δx y ΔP_x son las incertidumbres en la posición y el momento de la partícula en el eje X (y respectivamente para cada eje). Unos dos años después de que apareciera el artículo sobre la incertidumbre de

Heisenberg, H. P. Robertson presentó lo que hoy es la forma general más conocida de las relaciones de incertidumbre [240]. En esta formulación general, la posición y el impulso no juegan un papel privilegiado. Para dos operadores hermitianos cualesquiera A y B, Robertson encontró que

$$\Delta A \Delta B \geq \frac{1}{2} |\langle \psi | [A,B] | \psi \rangle| \tag{3.42}$$

Una derivación de esta forma general puede encontrarse en [295]. Debido a que $\Delta A \Delta B$ está tan estrechamente relacionado con el valor esperado $|\langle \psi | [A,B] | \psi \rangle|$, se espera que los productos de incertidumbre y los conmutadores compartan características importantes. Los conmutadores generalmente dan como resultado nuevos operadores, por lo que los cálculos que involucran conmutadores generalmente dependen del estado. Por lo tanto, los productos de incertidumbre también dependen generalmente del estado. Claramente, $\Delta x \Delta p_x > \hbar/2$ es un caso excepcional, al igual que $[x, p_x] = i\hbar$ para conmutadores.

El principio de Heisenberg establece la existencia de un límite a la precisión de la medida simultanea de un par de observables que no conmutan. Dado sistema preparado en un estado propio de A, con un cierto valor propio a_i bien definido, al medir el observable A se obtendrá siempre el resultado a_i. Sin embargo, una medición sobre B ocasionará que el vector de onda del sistema colapse en un estado propio de B, que no será un estado propio de A (excepto si A y B conmutan). Luego, al medir A de nuevo, se obtendrán resultados distintos, con las probabilidades indicadas por PII. En la mecánica cuántica, el proceso de medición perturba el sistema: si el observable A se mide con cierta precisión ΔA, el observable B se perturba en alguna cantidad ΔB y $\Delta A \Delta B$ satisface la desigualdad de Heisenberg antes mencionada. Dados dos observables A y B que no conmutan, es imposible medir al mismo tiempo A y B con un grado arbitrario de precisión: aumentar la precisión en A implica que la precisión en B disminuye y viceversa. La medición de un observable necesariamente perturba al otro. No existe un fenómeno similar en la mecánica clásica.

Un ejemplo de sistema que sirve para preparar y medir estados cuánticos es el dispositivo empleado en el experimento de Stern-Gerlach. Suponiendo que un haz de átomos de espín 1/2 entra en un aparato de Stern-Gerlach, si el aparato se orienta a lo largo del eje Z, se obtiene como salida alguno de los estados $|+\rangle_z$ o $|-\rangle_z$. Estos estados son los vectores propios del operador de Pauli σ_z, con valores propios +1 y -1. Si se bloquea el estado $|-\rangle_z$, entonces únicamente se obtendrá $|+\rangle_z$, en este caso, al actuar como *polarizador*, se le está usando para preparar un estado. De manera similar, si el aparato se orienta a lo largo del eje X, las dos componentes $|+\rangle_x$ y $|-\rangle_x$ salen del aparato. Si se bloquea la componente $|-\rangle_x$, entonces se ha preparado el estado $|+\rangle_x$. También puede usarse como *analizador*. Si un haz de átomos entra en el aparato orientado, por ejemplo, a lo largo de Z, se puede el valor de σ_z. Si el estado entrante se describe, por ejemplo, por $|\psi\rangle = |+\rangle_x = \frac{1}{\sqrt{2}}(|+\rangle_z + |-\rangle_z)$, entonces el sistema se encontrará en una superposición de los dos estados propios de σ_z, es decir, $|+\rangle_z$ o $|-\rangle_z$, y de acuerdo con PII, la medida de σ_z dará los valores propios +1 o −1 de σ_z con probabilidades iguales $p_+ = p_- = 1/2$.

En este breve capítulo se han presentado algunos experimentos que han conducido al desarrollo de la mecánica cuántica y su formulación en términos de postulados, que puede variar de un libro a otro, pero manteniendo la intención de reducir al mínimo las suposiciones básicas y tratando de ajustarse a los hechos observados. No es la intención desde este trabajo constituir un tratado de mecánica cuántica, pero es necesario presentar algunos de los conceptos, fenómenos y notación que se emplean de manera continua al hablar de cómputo cuántico.

4

Computación cuántica

El hecho de que los hitos en computación cuántica sigan apareciendo, en algunos casos, como titulares de prensa, pueden producir la impresión de que la computación cuántica es un asunto novedoso. Sin embargo, las ideas y principios sobre las que se sostiene llevan bastante tiempo existiendo. Y si bien, algunos nombres son perfectamente conocidos dentro de este campo, debe admitirse que no ha sido una idea que pueda atribuirse a una sola persona. Los momentos estelares en la historia de la humanidad en los que genios solitarios conciben ideas revolucionarias para resolver problemas cuya solución no era obvia son más bien excepciones a una regla, según la cual el desarrollo de la ciencia es producto de intercambio y desarrollo continuos de ideas que impulsan la curiosidad humana, vista como un colectivo, en nuevas direcciones.

4.1. Antecedentes

4.1.1. Primeras contribuciones

Mirando con atención la historia de la ciencia, es posible apreciar que, antes del nacimiento de un nuevo campo de investigación, generalmente hay un intervalo de tiempo en que muchas ideas similares están en el aire. En este periodo, los científicos comienzan a apreciar que hay algo nuevo en formación sin poder identificarlo plenamente. Y, de manera súbita, alguien logra articular un nuevo concepto, impulsando al resto a trabajar en esa nueva dirección. A partir de ese momento, pueden pasar años o incluso décadas hasta que se comprendan todas las implicaciones. Tal es el caso de la computación cuántica.

Como se ha mencionado con anterioridad, en la década de 1960, Landauer propuso relación entre termodinámica e información, argumentando que el hecho de restablecer un bit de información en una memoria de una computadora va inevitablemente acompañado de la generación de calor. A pesar de la controversia, esto sirvió como punto de partida para la investigación de un posible vínculo entre la física y la teoría de la información. Para 1980, la relación entre la física y la computación se hizo aún mas evidente, al punto que comenzaron a verse involucrados algunos conceptos de la mecánica cuántica. Dentro de la introducción a su libro *Computable e Incomputable* [178], el matemático soviético Yuri Manin introdujo la noción de un *autómata cuántico* que tenía la capacidad de emplear superposición y entrelazamiento para la solución de un problema que no podría resolverse con sistemas mecánicos clásicos.

“En la física posnewtoniana, la principal expresión de la idea de determinismo es el principio según el cual el desarrollo de un sistema físico aislado en el espacio-tiempo está determinado por ecuaciones diferenciales («leyes de la naturaleza») y el límite (condiciones iniciales). Este principio también se acepta en la ideología cuántica: el aspecto probabilístico de la teoría cuántica es esencial para describir interacciones, en particular, con un dispositivo de medición, pero no para la teoría de un sistema aislado.

El proceso computacional puede considerarse como otro modelo de la idea de determinismo. Es en gran medida paralela a la primera: «ley» corresponde a la estructura del dispositivo informático, las condiciones iniciales corresponden a los programas. La división del proceso computacional en pasos elementales, incluyendo, en particular, los pequeños cambios más simples en los contenidos de la memoria (como borrar o insertar un carácter en la cinta de una máquina de Turing), puede compararse con la idea de diferenciación. En procedimientos como la solución de la ecuación del calor por el método de la cuadrícula, realizamos una imitación bastante sencilla del determinismo continuo con la ayuda del discreto, pero, en términos generales, la comparación de estos dos modelos está lejos de ser trivial.

La biología molecular proporciona patrones de comportamiento de los sistemas naturales (no hechos por el hombre), que nos vemos obligados a describir en términos cercanos a los aceptados en la teoría de los autómatas discretos. La Figura 2 [aquí se omite] muestra un diagrama de la síntesis de proteínas sobre el ARN mensajero: es muy similar a la imagen de una máquina de Turing copiando información de una cinta a otra.

Los sistemas continuos clásicos controlados por ecuaciones diferenciales pueden imitar autómatas discretos solo con una estructura extremadamente compleja de su espacio de fase: una gran cantidad de regiones de estabilidad separadas por barreras de baja energía. Entrar en el programa hace un sofisticado sistema de pasajes en estas barreras, predeterminando el movimiento de la trayectoria de fase a lo largo de este laberinto. Como sistema físico, la calculadora debe ser muy inestable, porque un error de un signo en el programa, en términos generales, conduce a una trayectoria completamente diferente. Pero el proceso de cálculo en sí debe ser incomparablemente estable, es decir, los errores espontáneos (trayectoria que atraviesa una barrera que debe cerrarse como resultado de las fluctuaciones) deben tener una probabilidad muy pequeña. Es bien sabido que estos requisitos (combinados con la lentitud de operación y el crecimiento exponencial de la energía disipada con una complejidad creciente) creó una barrera para el desarrollo de computadoras mecánicas. Mientras tanto, a menudo tratamos de describir la acción de los «autómatas genéticos» precisamente en esos términos mecánicos. Una de las paradojas más famosas a las que conduce tal descripción es la imagen hipotética del despliegue de la doble hélice en el proceso de replicación. En esta imagen, la doble hélice del cromosoma bacteriano se retuerce unas 300000 vueltas. Dado que su duplicación en circunstancias favorables tarda 20 minutos, según el modelo mecánico de la replicación, cuando la hélice se despliega, parte del cromosoma debe girar a una velocidad de al menos 125 revoluciones por segundo. Paralelamente, debe ocurrir una red compleja de transformaciones bioquímicas inconfundibles.

Quizás, para avanzar en la comprensión de este tipo de fenómenos, necesitamos una teoría matemática de los autómatas cuánticos. Tal teoría nos proporcionaría modelos matemáticos de procesos deterministas con propiedades bastante inusuales. Una razón de esto es que el espacio de estado cuántico tiene una capacidad mucho mayor que el clásico: para un sistema clásico con N estados, su versión cuántica que permite la superposición acomoda c^N estados. Cuando unimos dos sistemas clásicos, su número de estados N_1 y N_2 se multiplican, y en el caso cuántico obtenemos un crecimiento exponencial $c^{N_1 N_2}$.”

> Estas estimaciones crudas muestran que el comportamiento cuántico del sistema podría ser mucho más complejo que su simulación clásica. En particular, dado que no existe una descomposición única de un sistema cuántico en sus partes constituyentes, un estado del autómata cuántico puede considerarse de muchas maneras como un estado de varios autómatas clásicos virtuales. Compare el siguiente comentario instructivo al final de [citando a R. P. Poplavskii en *Thermodynamical models of information processing[226]*]:
>
> > 'El cálculo de la mecánica cuántica de una molécula de metano requiere 10^{42} puntos de cuadrícula. Suponiendo que en cada punto tengamos que realizar solo 10 operaciones elementales, y que el cálculo se realice a la temperatura extremadamente baja $T = 3.10^{-3}K$, todavía tendríamos que usar toda la energía producida en la Tierra durante el último siglo.'
>
> La primera dificultad que debemos superar es la elección del equilibrio correcto entre los principios matemáticos y físicos. El autómata cuántico tiene que ser abstracto: su modelo matemático debe apelar únicamente a los principios generales de la física cuántica, sin prescribir una implementación física. Entonces el modelo de evolución es la rotación unitaria en un espacio de Hilbert de dimensión finita, y la descomposición del sistema en sus partes virtuales corresponde a la descomposición del producto tensorial del espacio de estados. En algún lugar de esta imagen debemos acomodar la interacción, que se describe mediante matrices de densidad y probabilidades [178].

El problema en cuestión pertenece al área de biología molecular, y se trata de describir la síntesis de proteínas sobre el ARN mensajero, algo que resulta muy similar a una máquina de Turing copiando información de una cinta a otra, por lo que. aparentemente pueden describirse en términos de la teoría de autómatas discretos. Sin embargo, argumenta que la descripción de sistemas clásicos por medio de ecuaciones diferenciales resulta ser una estructura extremadamente compleja. El espacio fase tendría una gran cantidad de regiones de estabilidad separadas por barreras de baja energía. Si se pretende simular estos sistemas, será necesario un sofisticado proceso de pasajes a través de estas barreras. Un sistema mecánico creado para simular este comportamiento sería inestable porque un error provocaría una trayectoria diferente. Y sin embargo, se requiere estabilidad en el cálculo, con una mínima probabilidad de errores. Añadiendo a ello la lentitud de los sistemas mecánicos y el crecimiento exponencial de la energía disipada con la complejidad, se entiende porque las computadoras mecánicas no son útiles para simular, por lo que le sorprende todo intento por describir a los autómatas genéticos en términos puramente mecánicos, cuando le parece clara la necesidad de incluir fenómenos cuánticos en la descripción. La principal ventaja que observa es que el tamaño del espacio que describe a los estados, en el caso cuántico crece de manera exponencial. Finalmente indica que su modelo matemático debe apelar únicamente a los principios generales de la física cuántica, sin prescribir una implementación física. Entonces, el modelo de evolución es la rotación unitaria en un espacio de Hilbert de dimensión finita, y la descomposición del sistema en sus partes virtuales corresponde a la descomposición del producto tensorial del espacio de estados.

También en 1980, Paul Benioff discutió la «construcción de un modelo de computación microscópico gobernado por las leyes de la mecánica cuántica, representado por máquinas de Turing» [22]. Entre las razones que establece para motivar la construcción de un modelo cuántico para la computación se encuentra, en primer lugar, la posibilidad de «extender el éxito obtenido al desarrollar modelos cuánticos simples de sistemas complejos a procesos aún más complicados» [22]. Menciona también que si se desea avanzar hacia una descripción mecánica cuántica de los seres inteligentes (siempre y cuando tal cosa sea posible), «se necesita primero obtener esa descripción para las computadoras y el proceso de cómputo» [22]. Para ello, parte de la consideración de que el proceso de cálculo consiste en una secuencia de procedimientos de decisión elementales en los que la operación realizada en el paso $(n+1)$ depende de los resultados obtenidos de la operación realizada en el paso n. De acuerdo con esto, construir un modelo cuántico del proceso de cómputo significa que es posible construir un modelo cuántico (al menos simple) del procedimiento de decisión. Además, plantea la pregunta con respecto a la posibilidad de construir un modelo del proceso de cálculo que no solo es cuántico, sino que es un modelo hamiltoniano. El éxito en esto significaría que debería ser posible modelar a la computadora, dentro del marco de la mecánica cuántica, como un sistema conservativo cerrado en evolución. Sin embargo, presenta una lista de razones por las que se pensaba que se trataba de una tarea imposible:

- Todas las computadoras construidas hasta ese momento (y al día de hoy) resultan ser sistemas disipativos abiertos que requieren una entrada de energía para funcionar. Los sistemas complejos organizados se consideran sistemas disipativos abiertos que requieren un aporte de energía para mantener su complejidad.

- Ya que los pasos de cálculo a menudo deben borrar información, son tanto lógicamente como físicamente irreversibles. Como resultado se requiere una disipación de energía del orden de kT por bit borrado (está hablando del Principio de Landauer). Sin embargo, menciona que una computadora siempre puede hacerse lógicamente reversible, haciéndole crear y guardar un historial de su cálculo. Además, la existencia de computadoras lógicamente reversibles (como las máquinas de Turing) que son equivalentes a las habituales y que borran sus propios historiales de manera reversible (la propuesta de Bennett).

- Los pasos de un proceso de cálculo son procedimientos de decisión. Dado que la computadora actúa sobre la base de la información leída o medida en poco tiempo, el principio de incertidumbre establece un límite a la rapidez con la que una computadora puede procesar la información.

Después de todo, cualquier modelo de este tipo evoluciona como un sistema aislado con una energía total constante. Por tanto, considera que la construcción de modelos cuánticos del proceso de cálculo que sean completamente satisfactorios en todos los aspectos es una gran empresa. Para avanzar en ella, realiza varias aproximaciones simplificadoras. La primera es limitarse a construir modelos del proceso de cálculo representado por las máquinas de Turing que, aunque tienen la desventaja de que ser muy lentas (se necesitan muchos pasos para realizar operaciones simples) resultan ser muy poderosas; cualquier computadora digital puede ser reemplazada por una máquina de Turing de igual o mayor poder de cómputo.

Para simplificar el análisis matemático (dejando intacta la naturaleza cuántica esencial del proceso) se usa el modelo de Coleman [126] para la parte de energía cinética del hamiltoniano. En palabras del autor (la cursiva no forma parte del original):

> “En este modelo, el operador de energía cinética $\nabla^2/2m$ se reemplaza por $v \cdot V$, donde v es una velocidad fija. Esto tiene como consecuencia que todos los sistemas se mueven con *velocidad constante*, la energía es *proporcional al momento* y *no hay propagación de paquetes de ondas*. [...] En particular, el único uso que se hace del modelo de Coleman es encender y apagar, una y otra vez, tres partes separadas de la interacción hamiltoniana que corresponden a tres tipos de pasos del modelo. Esto se logra haciendo que una secuencia de partículas pase por un sistema fijo e interactúe con él por medio de un potencial de corto alcance. Sin embargo, toda la complejidad del proceso reside en la interacción hamiltoniana, que es independiente de la aproximación del modelo de Coleman. [22]”

Otra aproximación que hace consiste en restringir al modelo para contener *un número finito de grados de libertad*. Como se ha mencionado, la máquina de Turing tiene la capacidad de trabajar con una cinta cuya extensión no está limitada. Los modelos habituales de máquinas de Turing contienen una o más cintas de longitud infinita y describen correctamente todos los procesos de cálculo de un número finito o infinito de pasos, pero para evitar posible problemas matemáticos relacionados con la mecánica cuántica de sistemas con infinitos grados de libertad, se construyen modelos de máquinas de Turing con cintas de longitud finita y que describen correctamente un número finito de pasos de un proceso de cálculo. Debido a que las computadoras reales son sistemas que cuentan con un número limitado (no infinito) de grados de libertad, no se trata realmente de una restricción, en realidad es una descripción adecuada para una implementación real. Las cintas infinitas solo serían necesarias en el límite de procesos que requieran un número infinito de pasos.

Finalmente, se indica que los modelos del proceso de computación construidos *crean y guardan una cinta histórica*. Aunque se ha demostrado la existencia de computadoras lógicamente reversibles (como las máquinas de Turing) que son equivalentes a las usuales y que borran sus propias cintas históricas, se supone, que estas máquinas reversibles pueden modelarse mediante sistemas termodinámicamente reversibles que disipan arbitrariamente poca energía por cada paso siempre que se operen con suficiente lentitud.

Con las consideraciones anteriores, Benioff logra demostrar que «para cada número N y máquina de Turing Q existe un hamiltoniano H_N^Q y una clase de estados iniciales apropiados tales que si $\Psi_N^Q(0)$ es tal estado inicial, entonces $\Psi_N^Q(t) = \exp\left(-iH_N^Q t\right)\Psi_N^Q(0)$ describe correctamente en los tiempos $t_3, t_6, ..., t_{3N}$ los

estados del modelo que corresponden a la finalización del primer, segundo,..., N-ésimo paso de cálculo de Q»[22]. Después de describir una máquina de Turing trabajando bajo los principios de la mecánica cuántica, en trabajos posteriores presentaría la teoría de la información cuántica, aplicando los fundamentos de la física, las matemáticas y la lógica. Se considera que su mayor logró consistió en su demostración de que es posible crear un modelo de computo cuántico reversible [23].

Luego, en mayo de 1981, un congreso sobre la «Física de la Computación», organizada por el MIT e IBM, reunió a físicos e informáticos. Entre los participantes se encontraban algunos científicos de renombre como Freeman Dyson, John Archibald Wheeler o Richard Feynman, Landauer y Benioff, ademas de otras personas cuyos nombres aparecen hoy en día en casi cualquier libro sobre computación cuántica: Charles Bennett, Tommaso Toffoli, Edward Fredkin. El contenido de las disertaciones se publicó al año siguiente en el *International Journal of Theoretical Physics* [1]. Es difícil decir hasta qué punto estos documentos fueron influenciados por las discusiones en la reunión o si las ideas presentadas habían sido articuladas por científicos individuales de antemano. La mayoría de los participantes hicieron referencia a los artículos de otros, excepto Feynman, quien no citó a nadie[1] y solo transcribió su discurso de apertura, con una ligera edición [89].

El título de la charla de Feynman fue «Simulación de la física con computadoras», pero, como explicó rápidamente, su propósito no era explorar cómo las computadoras pueden modelar la física de manera aproximada en cuestiones como el pronóstico del tiempo, modelado aerodinámico y predicción de las órbitas de planetas y cometas, porque no había nada particularmente nuevo en ellas. En cambio, Feynman estaba interesado en la posibilidad de que hubiera *simulaciones exactas*, tales como que una computadora pudiera hacer exactamente lo mismo que la naturaleza, por lo que propuso la idea de usar computadoras cuánticas para simular sistemas cuánticos que son demasiado difíciles de simular usando computadoras digitales clásicas convencionales. El discurso (*keynote*) de Feynman es considerado por muchos como el punto de lanzamiento de la computación cuántica como campo de estudio. Feynman tomó las ideas que estaban en el aire (la computación es un proceso físico, tal vez incluso cuántico) y luego las invirtió preguntando cómo calcular (simular) la física. En primer lugar, establece claramente su objetivo, que es simular sistemas cuánticos, utilizando recursos que se adapten bien al tamaño del sistema:

> “La regla de simulación que me gustaría tener es que la cantidad de elementos de computadora requeridos para simular un sistema físico grande sea solo proporcional al volumen de espacio-tiempo del sistema físico [89].”

Para la fecha en que este evento tuvo lugar, la computación (que a partir de este discurso pasaría a nombrarse *clásica*) estaba en pleno desarrollo. Su avance podía dar a los asistentes algún motivo para albergar la esperanza de que las computadoras pudieran, en algún momento, alcanzar la capacidad para tratar con ese problema. Por ello, Feynman proporcionó un argumento con el que demostró que las computadoras digitales no eran, ni pueden llegar a ser nunca, adecuadas para ejecutar esa labor:

> “Ahora voy explícitamente a la cuestión de cómo podemos simular con una computadora, un autómata universal o algo así, los efectos de la mecánica cuántica. (La formulación habitual es que la mecánica cuántica tiene algún tipo de ecuación diferencial para una función ψ). Si tiene una sola partícula, ψ es una función de x y t, y esta ecuación diferencial podría simularse como hice con mi ecuación probabilística. Eso estaría bien y uno ha visto gente hacer pequeñas computadoras que simulan la ecuación de Schröedinger para una sola partícula. Pero la descripción completa de la mecánica cuántica para un sistema grande con R partículas viene dada por la función $\psi(x_1, x_2...x_R, t)$, que llamamos amplitud de encontrar a las partículas $x_1, x_2...x_R$ y, debido a que tiene demasiadas variables, *no se puede simular* con una computadora normal con un número de elementos proporcional a R o proporcional a N [que se refiere a la cantidad de puntos en el espacio, cada uno de los cuales se encuentra en un determinado estado s_i]. Tuvimos los mismos problemas con la probabilidad en la física clásica. Y, por tanto, el problema es, ¿cómo podemos simular la mecánica cuántica? [89]”

A continuación, propone la construcción de una nuevo tipo de computadora, una cuya operación esté

[1] El editor tuvo que agregar una nota en la versión impresa para hacer referencia a los otros trabajos que aparecieron en la misma revista.

basada en elementos cuánticos que obedezcan las leyes de la mecánica cuántica, algo totalmente distinto a lo que hasta ese momento se consideraba como computadora, podría estar a la altura del trabajo:

> ¿Puedes hacerlo con un nuevo tipo de computadora, una computadora cuántica? Ahora resulta, por lo que sé, que puedes simular esto con un sistema cuántico, con elementos de computadora cuántica. No es una máquina de Turing, sino una máquina de otro tipo [89].

Feynman especifica dos condiciones que debe cumplir la computadora: sus componentes deben estar *interconectados localmente*, es decir, la información sólo debe fluir entre componentes adyacentes, similar a un autómata celular; y el número de sus componentes sólo debería aumentar en proporción al volumen del espacio a simular físicamente. Lo mismo se aplica al tiempo de cálculo necesario: también debería aumentar en proporción al lapso de tiempo simulado. Por eso no queremos tener un ordenador de tamaño ilimitado, en cuyo interior largos cables conecten elementos remotos y que tarde arbitrariamente en realizar sus simulaciones. Las máquinas de tal tipo no existían aún. Pero, en la concurrencia contaba con físicos e ingenieros, todos ellos dedicados a la computación, justo las personas que tenían conocimientos y recursos para investigar si la cuestión tenía posibilidades de desarrollarse. Era natural que pretendiera encender los ánimos del público para llevarlos a explorar las posibilidades y capacidades de este nuevo modelo de computación. Lo hace a través del siguiente desafío:

> Lo presento como otro problema interesante: resolver las clases de diferentes tipos de sistemas mecánicos cuánticos que son realmente intersimulables, que son equivalentes, como se ha hecho en el caso de las computadoras clásicas [89].

Alrededor de la mitad de la charla se dedica a elaborar el argumento de que las computadoras digitales serán inadecuadas para simular sistemas cuánticos de manera eficiente. Enfatiza que la teoría cuántica no admitirá una descripción de variables ocultas locales, y sigue una discusión interesante sobre las desigualdades de Bell y la evidencia experimental de que muestra su violación, aunque todo ello sin la mínima mención a Bell.

> Si considera que la computadora es del tipo clásico que he descrito hasta ahora (no del tipo cuántico descrito en la última sección) y no hay cambios en ninguna ley, y no hay trucos, la respuesta es ciertamente, «¡No!» Esto se llama el problema de la variable oculta: es imposible representar los resultados de la mecánica cuántica con un dispositivo universal clásico [89].

Parece bastante probable que la parte más conocida de todo el discurso, y del evento en general, sea la conmovedora conclusión, cuyo célebre coloquialismo sobrevivió a la edición:

> La naturaleza no es clásica, *maldita sea*, y si quieres hacer una simulación de la naturaleza, será mejor que la hagas mecánica cuántica, y por Dios, es un problema maravilloso porque no parece tan fácil [89].

Feynman estaba pidiendo un tipo de máquina para calcular totalmente nueva, a nivel fundamental y fue capaz de visualizar lo que resulta su aplicación más natural: realizar una «simulación de la naturaleza». A pesar de ello, en el momento en que la conferencia se llevaba a cabo, ninguno de los presentes, incluyendo a Feynman, tenían una idea clara de como construir una computadora de ese tipo. Realizar esta visión resulto ser, en efecto, «un problema maravilloso» y a cuarenta años de aquél discurso, sigue vigente que «no parece tan fácil». Feynman mostró que, si bien la mecánica cuántica no parece ser imitable por una computadora clásica local, podría ser abordada por «computadoras cuánticas: simuladores cuánticos universales». Aunque Manin había tenido una intuición similar al notar que «el comportamiento cuántico del sistema podría ser mucho más complejo que su simulación clásica», no la llevó mas lejos. Si bien es difícil asignar un solo momento en el tiempo como punto de partida de la computación cuántica, es posible tomar al número de 1982 del *International Journal of Theoretical Physics* como la cristalización de la idea de una computadora cuántica y el nacimiento de una nueva disciplina, dedicada a investigar el poder computacional y las posibilidades de

las computadoras basadas en la mecánica cuántica: la computación cuántica.

Al principio, hubo pocas personas que de verdad creyeran en la viabilidad práctica de construir una computadora cuántica. Se llegó a considerar a la computación cuántica como un asunto hipotético, de interés exclusivamente académico. Sin embargo, en 1985, David Deutsch publicó un artículo que cambió esta percepción . En él, describió una computadora cuántica universal capaz de simular el comportamiento de cualquier otra computadora cuántica, según el principio de Church-Turing ampliado. Además, Deutsch fue el primero en formular un algoritmo cuántico y en proponer que las computadoras cuánticas podrían ejecutarlo, junto con otros del mismo tipo.

Finalmente, en 1994, el matemático Peter Shor desarrolló un algoritmo revolucionario, con la capacidad de descomponer números enteros grandes en sus factores constituyentes. Aunque puede parecer un problema que pertenece exclusivamente al reino de lo abstracto, en realidad tiene implicaciones significativas en la seguridad de la información y la criptografía. Si el algoritmo de Shor se implementa de manera práctica, podría romper sistemas criptográficos, como RSA, el más extendido de los actuales, cuya base de funcionamiento es la dificultad de factorizar un número entero en primos. Aunque los detalles del funcionamiento del algoritmo deberán aguardar, a nivel elemental puede comentarse que el algoritmo de Shor opera en dos etapas principales: primero, se emplea una transformada cuántica de Fourier para encontrar el período de una función modular, y posteriosmente, se utiliza este resultado para deducir los factores primos del número en cuestión. La parte crítica del algoritmo es la transformada cuántica de Fourier, que puede realizar operaciones en paralelo sobre todos los posibles valores de período de manera simultanea, aprovechando la superposición cuántica, algo que no puede realizarse en una computadora clásica. Esto le permite al algoritmo de Shor encontrar el período más rápidamente que cualquier algoritmo clásico conocido, demostrando así que las computadoras cuánticas pueden resolver problemas de interés económico y social, lo que en definitiva a motivado a muchas empresas a invertir en su desarrollo en los últimos años.

4.2. La computadora cuántica

Existe una relación entre lo que las máquinas de calcular pueden hacer y las leyes físicas que obedecen al hacerlo. Una computadora no podrá, por ejemplo, ejecutar un algoritmo que le requiera ir en contra de la Segunda ley de la Termodinámica. En palabras de David Dutsch:

> “La teoría de la computación se ha estudiado tradicionalmente casi en su totalidad en abstracto, como un tema de matemáticas puras. Aquí se pierde el punto. Las computadoras son objetos físicos y los cálculos son procesos físicos. Lo que las computadoras pueden o no calcular está determinado únicamente por los fundamentos de la física, y no por las matemáticas puras [74].”

Si la física fuera distinta, la capacidad de las computadoras podría ser otra. Se mencionó con anterioridad que el cálculo automático se desarrollo en diferentes etapas. Las primeras máquinas, con componentes móviles como cuentas o engranajes puede considerarse como etapa *mecánica*, mientras la computación actual, en sus diferentes generaciones (válvulas de vacío, transistores, circuitos integrados y microprocesadores) corresponde a una etapa *electrónica*, esto en cuanto al tipo de tecnología con el que se construyen. Sin embargo, tanto la máquina analítica como el Frontier o un teléfono celular son computadoras clásicas porque siguen las leyes de la mecánica clásica. En cambio, una computadora cuántica es una máquina que trabaja usando las propiedades de sistemas cuánticos para realizar cálculos. Entre estas propiedades se encuentran la superposición, el entrelazamiento y la interferencia. A grandes rasgos los algoritmos seguidos por estas máquinas crean la superposición de todas las posibilidades a explorar, incluyendo las que son solución al problema (y por tanto se les considera convenientes), como aquellas que no lo resuelven (las inconvenientes). Después crea entrelazamiento entre estas posibilidades con sus respectivos resultados para, finalmente, conseguir que las posibilidades inconvenientes interfieran entre sí para que solo sobrevivan las soluciones. Esta última fase, de aniquilar opciones desfavorables, resulta ser la mas complicada y delicada de todo el proceso. Es una especie de coreografía matemática complicada, que solo se sabe llevar a cabo para algunos problemas específicos.

4.2.1. Unidad básica de información cuántica

En computación clásica, el *bit* es la unidad fundamental de información. Es una cantidad que puede tomar uno de dos valores, el cero o el uno. Por ello, la información se representa físicamente a través de sistemas

que pueden adoptar uno de dos estados distintos, uno de los cuales se identifica como *apagado* mientras el otro será *encendido*, similar al funcionamiento de un foco. Una sucesión de ceros y unos puede emplearse para representar información de diversas formas. En su uso más básico, permite expresar valores numéricos mediante el sistema binario. Sin embargo, también es posible codificar caracteres de un alfabeto, signos de puntuación, imágenes, sonidos y otros tipos de datos. Este principio constituye la base de la informática moderna, donde cada tipo de información se traduce en combinaciones binarias que pueden ser procesadas y almacenadas eficientemente

Los bits son símbolos que pueden agruparse para codificar información, pero, además de almacenarla, se puede manipular mediante operaciones lógicas. Para realizar esto, se requiere de sistemas físicos, llamados *compuertas*, que admitan y devuelvan bits. La salida dependerá de la combinación de valores recibida. Estas compuertas pueden agruparse en estructuras más complejas y ordenadas, denominadas *circuitos*, con el objetivo de realizar cálculos específicos o ejecutar instrucciones, es decir, ejecutar algoritmos, para resolver problemas y realizar operaciones de manera eficiente. Estos elementos conforman el denominado *modelo de circuito para la computación*. En el, una computadora clásica se describe como un registro finito de n bits. Este modelo también puede usarse para describir a una computadora cuántica, aunque con algunas diferencias, empezando por las unidades básicas de información

El comportamiento cuántico escapa a la experiencia natural humana y, por tanto, a toda intuición. Los fenómenos que tienen lugar a escala cuántica le confieren a las computadoras que operen bajo su esquema capacidades que ninguna computadora clásica podría tener. Esto tiene importantes implicaciones a nivel fundamental. La computación clásica y la cuántica diferirán hasta en la unidad básica de información. En teoría de información clásica, un bit es una cantidad de información básica e indivisible, empleada como unidad. Toma como valores *SI* o *NO*, *VERDADERO* o *FALSO*, y de manera más simple, 0 o 1.

Si se desea codificar información mediante un sistema cuántico, este deberá tener a capacidad de tomar alguno de dos estados discretos al realizar una medición sobre él. Uno de ellos se representará con un vector unitario apuntando hacia arriba, y se identificara con el *cero cuántico*, mientras que el otro apunta hacia abajo y se representa con un el *uno cuántico*, de manera análoga al caso clásico, constituyendo los *estados base*. Sin embargo, un sistema se establece en un estado definido solo una vez que se realiza una medición sobre él, a menos que se prepare en un estado específico. De manera que, si no se prepara de una determinada forma o sin una medición, los sistemas se encuentran en un estado que puede representarse mediante un vector apuntando en cualquier otra dirección, es decir, se encontrará en una superposición de ambos estados.

Esta es la idea detrás de la unidad elemental de información cuática, denominada *bit cuántico*, abreviada mediante la grafía «*qubit*» [247], o *Qbit*[191], aún más sucinta[2], mientras que el procesamiento de información cuántica se realiza mediante la acción de un conjunto de compuertas, llamadas *compuertas cuánticas*. Las propiedades del *Qbit* son heredadas del entorno cuántico. Para precisar la noción mencionada con anterioridad, un *Qbit* se define como un sistema cuántico de dos niveles. Tales sistemas se describen mediante un espacio de Hilbert complejo bidimensional, con dos estados de base ortonormales que se denotan como

$$|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \qquad |1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \tag{4.1}$$

lo que podría representar, por ejemplo, espín hacia arriba y espín hacia abajo para una partícula de espín $\frac{1}{2}$, o el estado fundamental y el estado excitado para el átomo de dos niveles.

En mecánica cuántica, el estado de un sistema cuántico S se representa mediante un vector normalizado $|\psi\rangle$ en el espacio de Hilbert. Por lo tanto, si $|\psi_1\rangle$ y $|\psi_2\rangle$ son dos estados cuánticos ortogonales, entonces cualquier superposición coherente de los dos estados,

$$|\psi\rangle = c_1 |\psi_1\rangle + c_2 |\psi_2\rangle \tag{4.2}$$

donde c_1 y c_2 son dos números complejos que satisfacen $|c_1|^2 + |c_2|^2 = 1$, también es un estado cuántico. Esta propiedad de los vectores en un espacio de Hilbert se denomina principio de superposición de estados cuánticos en la mecánica cuántica y es una característica fundamental que le distingue de la mecánica clásica, debido a que implica la posibilidad de encontrar a un Qbit en cualquier combinación lineal de los estados base, es decir

$$|\psi\rangle = \alpha |0\rangle + \beta |1\rangle = \begin{pmatrix} \alpha \\ \beta \end{pmatrix} \tag{4.3}$$

[2]La forma indicada de acuerdo con la ortografía del español es «cúbit», recomendada, pero sin reconocimiento oficial de la RAE. A la fecha de la publicación de este trabajo, continua sin aparecer en el diccionario.

donde $|\alpha|^2 + |\beta|^2 = 1$. Esta combinación lineal de los estados de la base recibe el nombre de estado puro y los coeficientes se denominan amplitudes de probabilidad, debido a que, si se realiza una medida sobre el Qbit, como estado cuántico, tendrá una probabilidad $|\alpha|^2$ de encontrarse en el estado $|0\rangle$y una probabilidad $|\beta|^2$ de encontrarse en el estado $|1\rangle$. Mientras que un bit clásico puede encontrarse en el estado 0 o en el estado 1, el hecho de poder escribir al bit cuántico como $\left|\psi\right\rangle = \alpha|0\rangle + \beta|1\rangle$, implica que puede existir el estado $|0\rangle$ y $|1\rangle$ hasta ser observado. Una forma alternativa de escribir $\left|\psi\right\rangle$ es [296]:

$$\left|\psi\right\rangle = e^{i\gamma}\left[\cos\left(\frac{\theta}{2}\right)|0\rangle + e^{i\phi}\sin\left(\frac{\theta}{2}\right)|1\rangle\right] \tag{4.4}$$

con $\alpha = e^{i\gamma}\cos\left(\frac{\theta}{2}\right)$ y $\beta = e^{i(\gamma+\phi)}\sin\left(\frac{\theta}{2}\right)$, las expresiones polares correspondientes a las amplitudes complejas. La libertad para elegir γ y ϕ viene de observar que la parte correspondiente a la fase en la expresión polar de un número complejo tiene norma unitaria, es decir $|e^{i\delta}| = |\cos\delta + i\sin\delta| = 1$, por lo que un Qbit $\left|\psi\right\rangle' = e^{i\delta}\left|\psi\right\rangle = \alpha e^{i\delta}|0\rangle + \beta e^{i\delta}|1\rangle$ tendrá una probabilidad $|\alpha e^{i\delta}|^2 = |\alpha|^2$ de encontrarse en el estado $|0\rangle$ y una probabilidad $|\beta e^{i\delta}|^2 = |\beta|^2$ de encontrarse en el estado $|1\rangle$, es decir, las mismas que probabilidades que tenía el Qbit original.

Esto demuestra que no hay consecuencia física alguna sobre el sistema al multiplicar por una fase global, ya que no afecta las predicciones físicas. La estructura matemática dotada de este comportamiento recibe el nombre de *rayo*, y está definido como una clase de equivalencia entre vectores dada por $\left|\psi\right\rangle' \sim \left|\psi\right\rangle \iff \left|\psi\right\rangle' = e^{i\delta}\left|\psi\right\rangle$, por lo que un estado cuántico resulta ser, en realidad, un rayo en el espacio proyectivo de Hilbert, *no un vector*. Esta distinción se omite en casi todos los libros de texto, algo que posiblemente se deba a que la ecuación de Schrödinger actúa sobre vectores en el espacio de Hilbert. El uso familiar, pero impreciso, de «vector de estado» en lugar de «rayo» resulta por ello una costumbre casi imposible de evitar[30], algo que no tampoco se pretende en el presente trabajo.

Por lo anterior, dado un vector de estado $\left|\psi\right\rangle = \alpha|0\rangle + \beta|1\rangle$, es posible elegir un ángulo ϕ apropiado para que $\gamma = 0$, de tal forma que el coeficiente α resulte real y positivo (exceptuando para el estado base $|1\rangle$, en el que $\alpha = 0$ y $\beta = 1$). Luego el estado genérico de un Qbit puede escribirse [296]:

$$\left|\psi\right\rangle = \cos\left(\frac{\theta}{2}\right)|0\rangle + e^{i\phi}\sin\left(\frac{\theta}{2}\right)|1\rangle = \begin{pmatrix}\cos\left(\frac{\theta}{2}\right)\\ e^{i\phi}\sin\left(\frac{\theta}{2}\right)\end{pmatrix} \tag{4.5}$$

con ($0 < \theta < \pi$ y $0 < \phi < 2\pi$). Debido a la condición de normalización, el estado del Qbit se puede representar mediante un punto en una esfera de radio unitario, denominada *esfera de Bloch*, que es útil para trabajar con Qbits, ya que proporciona una imagen geométrica tanto del Qbit como de las transformaciones que se pueden realizar en el estado de un Qbit. Esta esfera se puede incrustar en un espacio tridimensional de coordenadas cartesianas ($x = \cos(\phi)\sin(\theta), y = \sin(\phi)\sin(\theta), z = \cos(\theta)$). Por lo tanto, el estado $\left|\psi\right\rangle = \cos\left(\frac{\theta}{2}\right)|0\rangle + e^{i\phi}\sin\left(\frac{\theta}{2}\right)|1\rangle$ puede escribirse como

$$\left|\psi\right\rangle = \begin{pmatrix}\sqrt{\frac{1+z}{2}}\\ \frac{x+iy}{\sqrt{2(1+z)}}\end{pmatrix} \tag{4.6}$$

Un vector cuyas componentes (x, y, z) señalan un punto en la esfera de Bloch recibe el nombre de *vector de Bloch* y debe satisfacer la condición de normalización $x^2 + y^2 + z^2 = 1$. Además, también puede definirse en términos de los ángulos θ y ϕ, como en 4.1

Cualquier sistema cuántico de dos niveles se puede usar en la práctica como un Qbit siempre que sea posible manipularlo, en los siguientes términos:

- Preparación. Se debe contar con la capacidad para preparar en algún estado bien definido, por ejemplo, el estado $|0\rangle$, llamado *estado fiduciario* del Qbit.
- Transformación. Cualquier estado del Qbit puede transformarse en otro estado arbitrario. Tales transformaciones se realizan mediante transformaciones unitarias. Estas transformaciones son las compuertas cuánticas, cuyo efecto será realizar una rotación en el Qbit.
- Medición. El estado del Qbit puede medirse en la base computacional $\{|0\rangle, |1\rangle\}$, es decir, se puede medir la polarización del Qbit a lo largo del eje Z. Para tal fin, será de utilidad el empleo del operador de Pauli Z.

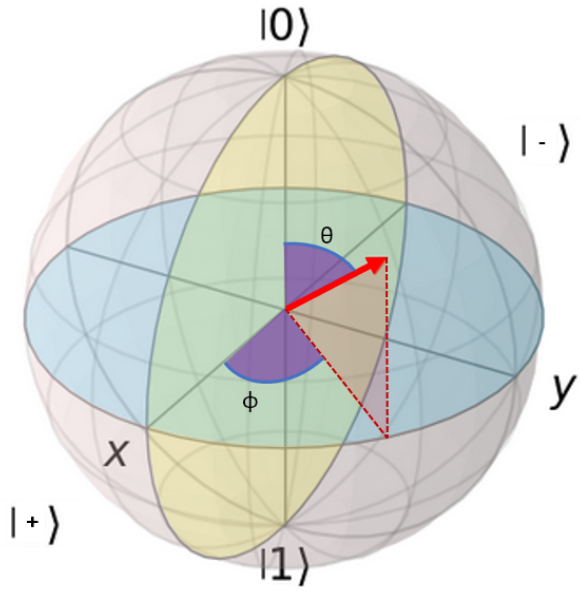


Figura 4.1: La esfera de Bloch es una esfera de radio unitario. Los vectores radiales representan los posibles estados de un Qbit. Los vectores de la base ortogonal (o base computacional), corresponden al eje Z. El Qbit 0 está en el estado $|0\rangle$, mientras que el Qbit 1 se encuentra en el estado $|1\rangle$. Debido a la condición de normalización, es posible sustituir los parámetros complejos α y β, por parámetros reales θ y ϕ de tal forma que todo estado de un Qbit pueda representarse mediante el estado descrito por la ecuación 4.5

Los requisitos mencionados pueden cumplirse actualmente en experimentos de laboratorio. Las implementaciones físicas de un Qbit pueden obtenerse a partir de sistemas de dos estados, por ejemplo:

- Una partícula de espín $\frac{1}{2}$ (aquí solo importan los estados internos en lugar de la función de onda espacial). El espín $+\frac{1}{2}$ representa$|0\rangle$, en tanto que $-\frac{1}{2}$ representaría al $|1\rangle$. Ambos estados pueden alcanzarse usando un campo magnético sin disipar calor. En el aparato de Stern-Gerlach alineado con el eje Z, se obtiene $|+\rangle_z$ que se identificará con $|0\rangle$ mientras que $|-\rangle_z$ se representa con $|1\rangle$
- Un átomo de dos niveles (donde todos los estados excitados superiores se ignoran si nunca entran en la dinámica de interés) en donde $|0\rangle$ corresponde al estado fundamental atómico y $|1\rangle$ al primer estado excitado.
- Un par de Cooper haciendo un túnel entre dos islas superconductoras ($|0\rangle$ si el par está en una isla y $|1\rangle$ si está en la otra isla). La evolución unitaria controlada del estado de un Qbit es implementada entonces mediante campos magnéticos o láser y aparatos de medida eficientes ahora se ha desarrollado.

La superposición es un concepto fundamental de la mecánica cuántica y una de las principales diferencias entre la computación cuántica y la clásica. Mientras que en la computación clásica un bit solo puede representar un único valor en un instante determinado, en la computación cuántica, un *Qbit* puede existir en una superposición de ambos estados simultáneamente. Gracias a la superposición, las computadoras cuánticas pueden realizar cálculos sobre múltiples configuraciones de entrada en paralelo. Cuando varios *Qbits* conforman una computadora cuántica, el sistema puede adoptar una superposición de una gran cantidad de estados posibles. Esto permite que ciertos tipos de cálculos, se realicen de manera más rápida que en una computadora clásica. En un sistema clásico, cada conjunto de datos de entrada requiere una ejecución separada del algoritmo. En cambio, una computadora cuántica puede procesar múltiples entradas en una sola ejecución, aprovechando el paralelismo cuántico, que es una de las bases fundamentales de la computación cuántica. Sin embargo, la superposición es un fenómeno que puede alterarse por factores externos como el ruido o la medición, lo que dificulta la estabilidad del cómputo cuántico. Por esta razón, la computación cuántica requiere hardware especializado y algoritmos diseñados para preservar estos delicados estados cuánticos.

4.2.2. Medición

Para describir el estado de un solo bit clásico, hace falta tan solo un bit de información, por lo que es suficiente indicar si el estado del bit es 0 o 1. En el caso cuántico, para describir el estado correspondiente a un Qbit

arbitrario $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ con un grado alto de precisión, se requiere de un número alto de bits, ya que deben especificarse dos números complejos, con la única restricción dada por la condición de normalización. Ya que los Qbits, además de un conjunto de estados mucho más rico que los bits clásicos también pueden trabajar con un conjunto de transformaciones más amplio que sus contrapartes clásicas, puede entenderse que la computadora cuántica es mucho más poderosa que la clásica.

Sin embrago, debe considerarse que, en el caso clásico, si se desea averiguar cuál es el estado de un conjunto de bits, lo único que hace falta es mirar (medir) cada uno de ellos. No existe problema en conocer el estado de cada bit en el caso clásico y, por tanto, es posible averiguar el resultado de cualquier cálculo hecho a partir de operaciones con estos bits. Además, el proceso de lectura no altera al estado de los bits clásicos. Puede leerse el contenido de los bits clásicos en cualquier etapa de un cálculo sin que ello altere las etapas posteriores.

El caso de los Qbits es bastante diferente. Si se tienen n Qbits en una superposición de los estados base computacionales $|\psi\rangle = \sum_i c_i |i\rangle$, no hay forma de extraer de estos Qbits la gran cantidad de información que está contenida en las amplitudes c_i. Debido a que la lectura de los valores de las amplitudes es imposible, tampoco se podrá conocer su estado. La única forma de extraer información de los n Qbits es mediante la realización de una medición en la base computacional clásica, acto que consiste en aplicar una prueba determinada a cada Qbit, devolviendo en cada caso un 0 o un 1, que corresponden a $\sigma_z = +1$ y $\sigma_z = -1$, respectivamente. El patrón de unos y ceros que se produzca debido a la prueba no estará determinado, en general, por el estado de los Qbits, ya que la mecánica cuántica indique solo es posible determinar la probabilidad de los resultados. Por ejemplo, si se desea obtener un resultado en particular, como $|010110\rangle$ si se trata de un sistema de 6 Qbits, la probabilidad de obtenerlo estará determinada por el cuadrado de la magnitud de la amplitud del estado $|010110\rangle$ en la expansión del estado de los Qbits en los 2^6 estados de la base computacional. En general, si el estado de n Qbits es $\sum_i c_i |i\rangle$ la probabilidad de que el patrón de ceros y unos resultantes de las mediciones de todos los Qbits sea igual a la expansión binaria correspondiente al entero i es $p_i = |\langle i|\psi\rangle|^2 = |c_i|^2$. El cuadrado de la magnitud de la amplitud es igual a la probabilidad de los resultados de las mediciones.

El estado para un Qbit arbitrario $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ puede medirse, en principio, siempre que se tenga una gran cantidad de Qbits preparados de manera idéntica. Es posible medir las coordenadas x, y y z correspondientes a un Qbit en la esfera de Bloch, mediante la proyección respecto al eje del que se trate, tras lo cual se podrá encontrar α y β, hasta un factor de fase global. Los operadores de Pauli, en la base computacional, están dados por:

$$\sigma_x = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \qquad \sigma_y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix} \qquad \sigma_z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \tag{4.7}$$

Al aplicar las matrices de Pauli sobre un estado arbitrario $|\psi\rangle$, se se obtiene:

$$\begin{aligned} \sigma_x|\psi\rangle = e^{i\phi}\sin\left(\frac{\theta}{2}\right)|0\rangle + \cos\left(\frac{\theta}{2}\right)|1\rangle \qquad \sigma_y|\psi\rangle = -ie^{i\phi}\sin\left(\frac{\theta}{2}\right)|0\rangle + \cos\left(\frac{\theta}{2}\right)|1\rangle \\ \sigma_z|\psi\rangle = \cos\left(\frac{\theta}{2}\right)|0\rangle - e^{i\phi}\sin\left(\frac{\theta}{2}\right)|1\rangle \end{aligned} \tag{4.8}$$

Que pueden emplearse para obtener los siguientes valores esperados para $|\psi\rangle$:

$$|\langle\psi|\sigma_x|\psi\rangle| = x \qquad |\langle\psi|\sigma_y|\psi\rangle| = y \qquad |\langle\psi|\sigma_z|\psi\rangle| = z \tag{4.9}$$

Las coordenadas (x, y, z) se pueden obtener con precisión arbitraria por medio de medidas proyectivas estándar sobre la base computacional, es decir, midiendo σ_z. En tal caso se obtendrá 0 con probabilidad $p_0 = |\langle 0|\psi\rangle|^2 = \cos^2(\theta/2)$ y 1 con probabilidad $p_1 = |\langle 1|\psi\rangle|^2 = \sin^2(\theta/2)$, por lo que $z = \cos(\theta) = \cos^2(\theta/2) - \sin^2(\theta/2) = p_0 - p_1$, lo cual muestra que la coordenada z está dada por la diferencia de probabilidades de obtener resultados 0 o 1 a partir de una medida de σ_z. Si se posee un gran número N de sistemas idénticamente preparados en el estado $|\psi\rangle$, se podrá estimar la coordenada z como $(N_0/N) - (N_1/N)$ donde N_0 y N_1 cuentan el número de medidas con resultados 0 y 1. Las otras coordenadas se pueden obtener utilizando la posibilidad de operar una transformación unitaria en el Qbit para rotar lo ejes. La transformación

$$U_1 = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ -1 & 1 \end{bmatrix} \tag{4.10}$$

corresponde a una rotación de la esfera de Bloch en el sentido de las manecillas del reloj en un ángulo $\frac{\pi}{2}$ sobre el eje Y, de manera que el eje X se transforma en el eje Z, con lo que se obtiene el estado $|\psi\rangle^{(1)} = U_1|\psi\rangle$. Una

medida proyectiva en la base computacional dará 0 con probabilidad 0 con probabilidad $p_0^{(1)} = |\langle 0|\psi^{(1)}\rangle|^2$ o 1 con probabilidad $p_1^{(2)} = |\langle 1|\psi^{(1)}\rangle|^2$, que puede usarse para obtener

$$p_0^{(1)} - p_1^{(1)} = \cos\phi \sin(\theta) = x \tag{4.11}$$

con lo que la coordenada x se puede calcular midiendo σ_z. De manera semejante, puede usarse la transformación

$$U_2 = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & i \\ i & 1 \end{bmatrix} \tag{4.12}$$

que indica rotar a la esfera de Bloch a través de un ángulo $\frac{\pi}{2}$ sobre el eje X en el sentido de las agujas del reloj , para obtener el estado $|\psi\rangle^{(2)} = U_2|\psi\rangle$. Una medida proyectiva en la base computacional dará 0 con probabilidad $p_0^{(2)} = |\langle 0|\psi^{(2)}\rangle|^2$ o 1 con probabilidad $p_1^{(2)} = |\langle 1|\psi^{(2)}\rangle|^2$ y finalmente se obtiene

$$p_0^{(2)} - p_1^{(2)} = \sin(\phi)\sin(\theta) = y \tag{4.13}$$

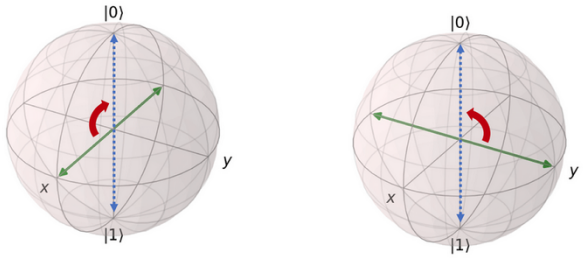


Figura 4.2: La coordenada z está dada por la diferencia de probabilidades de obtener resultados 0 o 1 a partir de una medida de σ_z. Las otras coordenadas se pueden obtener utilizando la posibilidad de operar una transformación unitaria en el Qbit para rotar lo ejes. La transformación U_1 corresponde a una rotación de la esfera de Bloch en el sentido de las manecillas del reloj en un ángulo $\frac{\pi}{2}$ sobre el eje Y, de manera que el eje X se transforma en el eje Z. De manera semejante, puede usarse la transformación U_2 indica rotar a la esfera de Bloch a través de un ángulo $\frac{\pi}{2}$ sobre el eje X en el sentido de las agujas del reloj.

4.2.3. Comparación entre bits clásicos y cuánticos

A diferencia del bit clásico, que solo se puede establecer en 0 o 1, el Qbit reside en un espacio vectorial, parametrizado por las variables continuas α y β (o bien, θ y ϕ), de manera que se permite un continuo de estados posibles. Aunque esto parece indicar que un solo Qbit podría usarse para almacenar una cantidad infinita de información, se debe considerar que la extracción de esta información requiere de la realización de una medición. De acuerdo con los postulados de la mecánica cuántica, la medición del estado de polarización de un Qbit a lo largo de cualquier eje devolverá un único bit de información.

Para comprender mejor el principio de superposición cuántica y cómo un bit cuántico resulta ser diferente de uno clásico en términos de distribución de probabilidad, es necesario entender la consecuencia de una medición cuántica. La medición de un observable está relacionada con la aplicación de un operador hermitiano M al sistema S. El resultado de la medición será alguno de los valores propios del operador, cuya descomposición espectral está dada por

$$M = \sum_i c_i |\phi_i\rangle\langle\phi_i| \tag{4.14}$$

donde cada c_i es un valor propio de M y $|\phi_j\rangle$ es el vector propio correspondiente. Ya que M es hermitiano, siempre se puede elegir$\{|\phi_j\rangle\}$ como base ortonormal del espacio de Hilbert, es decir, $\langle\phi_i|\phi_j\rangle = \delta_{ij}$ y $\sum_i |\phi_i\rangle\langle\phi_i| = I$. La probabilidad de obtener el valor c_i es entonces $p_i = \langle\psi|\phi_i\rangle\langle\phi_i|\psi\rangle$, por lo que $\sum_i p_i = 1$. Es decir, cuando se trata de una medición, un estado cuántico se asocia con una distribución de probabilidad clásica, y las correlaciones de la teoría de la probabilidad se aplican al dominio cuántico.

El caso de un Qbit con estado $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ puede servir de ejemplo. Si se mide un operador cuyos vectores propios son $|0\rangle$ y $|1\rangle$, con valores propios 1,-1, respectivamente, es decir, $|0\rangle\langle 0| + |1\rangle\langle 1|$, que resulta ser el operador de Pauli σ_z, cuya forma matricial en la base $\{|0\rangle, |1\rangle\}$, es

$$Z = \sigma_z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \tag{4.15}$$

Al medir Z, las probabilidades $p_0 = \alpha^2 = p$ de obtener $|0\rangle$ y $p_1 = \beta^2 = 1 - p$ de obtener $|1\rangle$.

Para ver la diferencia real entre bits clásicos y cuánticos, sea W un bit con probabilidades de $p(W=0) = |\alpha|^2$, $p(W=1) = 1—|\alpha|^2$. Si, por ejemplo, $\alpha = \beta = \frac{1}{\sqrt{2}}$ por lo que $p(W=0) = p(W=1) = \frac{1}{2}$. Para un estado de Qbit correspondiente $|\psi\rangle = a|0\rangle + b|1\rangle$, al medir el operador de Pauli se obtendrá $|0\rangle$ o $|1\rangle$, con probabilidad $\frac{1}{2}$ en cada caso. En este sentido, el estado del Qbit $|\psi\rangle$ se comporta de manera similar al bit W. Sin embargo, hay algo más que se puede hacer con el Qbit. Los operadores de Pauli, en su forma matricial, pueden escribirse como:

$$X = \sigma_x = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \qquad Y = \sigma_y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix} \tag{4.16}$$

Ya que $|\psi\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$es un vector propio de σ_x con valor propio 1, al medir σ_x, se obtiene un valor definido de 1. Sin embargo, al medir σ_y, nuevamente se obtiene cada valor propio de σ_y con una probabilidad de un medio para cada uno. El ejemplo muestra que la distribución de probabilidad correspondiente a un estado cuántico está asociada con la medición escogida. Si el estado corresponde a un vector propio del operador, la medición tendrá un valor sin incertidumbre, en caso de que no sea un vector propio, existirá cierta incertidumbre. Solo puede asignarse un valor de incertidumbre a un estado cuántico puro si se especifica la medición.

4.3. El modelo de circuito de la computación cuántica

4.3.1. Definición de computadora cuántica

En el modelo de circuito, una computadora clásica puede describirse como un registro finito de n bits. Todo cálculo comienza preparando bits individuales en un determinado valor inicial. Después, para realizar las operaciones aritméticas necesarias, se requiere de una secuencia de operaciones elementales, que se realizan por medio un conjunto de compuertas digitales, como como NOT o AND, que pueden operar sobre los bits individuales o en parejas de bits. Al final, deben leerse los bits de salida para conocer los resultados de la operación. Independientemente de la complejidad de las operaciones, estas pueden reducirse a una secuencia de operaciones en compuertas digitales.

Una computadora cuántica funciona de manera similar. Se describirá como un a colección finita de n Qbits, un registro cuántico de tamaño n. Para resolver un problema, se requiere de un circuito cuántico que implemente un algoritmo específico. En primer lugar, al igual que en el caso clásico, el registro debe inicializarse en un valor determinado. Posteriormente, se le debe transformar, a través de un conjunto de compuertas, que en este caso serán cuánticas y que deben cumplir con algunos requisitos que no se exigían a sus contrapartes clásicas. Una computadora cuántica se verá como una red cuántica (o una familia de redes cuánticas) compuesta por compuertas lógicas cuánticas; cada compuerta realizando una operación unitaria elemental en uno, dos o más Qbits. Cada Qbit representará una unidad elemental de información, correspondientes a los valores de bit clásicos 0 e 1.

Debido a que la computadora cuántica es un sistema cuántico de n cuerpos, la evolución temporal de la función de onda que le corresponde está regida por la ecuación de Schrödinger y, en consecuencia, los postulados de la mecánica cuántica indican que su evolución está descrita por un operador unitario, que tiene una representación matricial. Luego, la evolución temporal de una función de onda de n Qbits se describe mediante una matriz unitaria de $2^n \times 2^n$ que siempre puede descomponerse como un producto de operaciones unitarias que actúan solamente sobre uno o dos Qbits. Estas operaciones son las compuertas cuánticas elementales que constituyen la base del modelo de circuito de la computación cuántica.

Mientras que el estado de una computadora clásica de n bits se describe en notación binaria mediante un número entero $i \in [0, 2^n - 1]$ de la forma $i = i_{n-1}2^{n-1} + ... + i_1 2 + i_0$, con $i_n \in [0,1]$ dígitos binarios, el estado de una computadora cuántica de n Qbits está dado por

$$|\psi\rangle = \sum_{i=0}^{2^n-1} c_i |i\rangle = \sum_{i_{n-1}=0}^{1} \cdots \sum_{i_1=0}^{1} \sum_{i_0=0}^{1} c_{i_{n-1},...,i_1,i_0} |i_{n-1}\rangle \otimes ... |i_1\rangle \otimes |i_0\rangle \tag{4.17}$$

con los números complejos c, restringidos por la condición de normalización $\sum_{i=0}^{2^n-1} |c_i|^2 = 1$.

De manera que el estado de una computadora cuántica de n Qbits es una función de onda que reside en un espacio de Hilbert de 2^n dimensiones, construido como el producto tensorial de n espacios de Hilbert de 2 dimensiones, uno para cada Qbit. Teniendo en cuenta la condición de normalización y el hecho de que el estado de cualquier sistema cuántico sólo está definido hasta una fase global sin significado físico, el estado de la computadora cuántica está determinado por $2(2^n - 1)$ parámetros reales. Por ejemplo, si $n = 2$

$$\left|\psi\right\rangle = \sum_{i=0}^{2^2-1} c_i \left|i\right\rangle = \sum_{i_1=0}^{1}\sum_{i_0=0}^{1} c_{i_1,i_0} \left|i_1\right\rangle \otimes \left|i_0\right\rangle \tag{4.18}$$

Con frecuencia se empleará la conveniente notación $|i_1\rangle \otimes |i_0\rangle = |i_1\rangle\,|i_0\rangle = |i_1 i_0\rangle$, con lo que el estado de una computadora cuántica de n Qbits puede escribirse de manera mas compacta como

$$\left|\psi\right\rangle = \sum_{i_{n-1},\ldots,i_1,i_0=0}^{1} c_{i_{n-1},\ldots,i_1,i_0} \left|i_{n-1}\ldots i_1 i_0\right\rangle \tag{4.19}$$

En esta expresión, puede notarse el principio de superposición. Comparando el caso clásico, en donde n bits pueden almacenarse en un único entero i, en un registro cuántico de n Qbits, se puede preparar al estado correspondiente $|i\rangle$ de la base computacional, pero también en superposición. La cantidad de estados que pueden crearse usando la superposición es 2^n, mostrando un crecimiento exponencial con el número de Qbits.

Al realizar un cálculo en una computadora clásica, diferentes entradas de datos requieren ejecuciones separadas. En una computadora cuántica es posible realizar un cálculo para muchas entradas en una sola corrida, paralelismo que es la base de la computación cuántica. El principio de superposición, sin embargo, no es una característica exclusiva de los sistemas cuánticos y puede encontrarse en casos clásicos, como las ondas. Un ejemplo es la ecuación de onda para una cuerda vibrante con extremos fijos. Las soluciones satisfacen el principio de superposición, por lo que el estado mas general para una cuerda es una combinación lineal de estas soluciones, de la forma

$$\left|\phi\right\rangle = \sum_{i=0}^{2^n-1} c_i \left|\phi_i\right\rangle \tag{4.20}$$

Por lo que la capacidad extraordinaria de la computación cuántica no puede deberse exclusivamente al principio de superposición. Un análisis de los recursos necesarios para representar la superposición en física clásica muestra que para representar la superposición de 2^n niveles, estos deben pertenecer al mismo sistema. Desde luego, en física clásica no existe entrelazamiento, por lo que los estados clásicos de sistemas separados nunca pueden superponerse, por lo que hará falta una cantidad de niveles que crecerá de manera exponencial con n. Si D es la separación de energía típica entre dos niveles consecutivos, la cantidad de energía requerida para este cálculo está dada por $D2^n$. Por lo tanto, la cantidad de recursos físicos necesarios para el cálculo crece exponencialmente con n^2. En contraste, debido al entrelazamiento, en la física cuántica una superposición general de niveles de 2^n puede representarse mediante n Qbits. Así, la cantidad de recursos físicos (energía) crece solo linealmente con n.

4.3.2. Compuertas de un solo Qbit

Las transformaciones unitarias son análogas a las transformaciones ortogonales para vectores reales en las que el producto escalar permanece invariante. Para llevarlas a cabo, se requiere de un operador unitario actuando sobre un vector de estado. Desde el punto de vista de la información cuántica, los operadores cuánticos desempeñan un papel básico en el procesamiento de la información, como componentes básicos de los circuitos cuánticos y en la codificación de las comunicaciones [208]. Si bien, se considera que la evolución cuántica es unitaria, en estricto sentido haría falta considerar los efectos de decoherencia no unitaria debidos al acoplamiento no deseado de la computadora cuántica con su entorno. En primera aproximación, estos pueden ser ignorados. Una serie de operaciones cuánticas ordenadas puede realizar tareas informáticas, desarrollar sistemas físicos y simular fenómenos cuánticos.

Hay varias razones por las que los operadores unitarios se investigan desde el nacimiento de la computación cuántica[307]. La principal es que las evoluciones temporales son naturalmente unitarias en la mecánica cuántica convencional, toda vez que la evolución temporal del estado de un sistema cuántico está descrita por la relación $\left|\psi(t)\right\rangle = U(t,t_0)\left|\psi(t_0)\right\rangle$, que preserva la normalización del ket asociado para algún operador unitario U. Otra razón puede deberse al éxito del algoritmo de búsqueda acelerado por raíz cuadrada de Grover lo que refuerza el punto de vista de que el núcleo de un algoritmo es diseñar la evolución unitaria paso a paso del sistema. Es suficiente aplicar una serie de operaciones unitarias para simular una variedad de sistemas hermitianos y fenómenos relacionados, lo que lleva a los científicos a prestar más atención a los productos de los operadores unitarios.

Una compuerta cuántica es simplemente un operador que actúa sobre Qbits, transformandolos. Esta evolución se caracteriza por operadores unitarios, que tiene una representación matricial. A una compuerta

que actúa sobre n Qbits le corresponde a una matriz unitaria de $2^n \times 2^n$. Si H es una matriz Hermitiana ($H^\dagger = H$), la exponencial $U = exp(iH)$ es unitaria y reciprocamente, cada matriz unitaria U es la exponencial de iH para alguna matriz hermitiana H, por lo que se trata de matrices invertibles. Los estados cuánticos sobre los que actúan las puertas son vectores complejos de dimensión 2^n. A continuación se describen algunas compuertas de un Qbit

Compuerta identidad
El ejemplo mas sencillo de una transformación que puede aplicarse a un solo Qbit se tiene a la compuerta identidad I, que, en realidad, no realiza ninguna operación. Deja los estados base $|0\rangle$ y $|1\rangle$ sin modificar. En forma matricial, se representa por la matriz identidad.

$$I = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{4.21}$$

La acción de la compuerta de identidad sobre los estados base se puede observar al realizar el producto:

$$I|0\rangle = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix} = |0\rangle \qquad I|1\rangle = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix} = |1\rangle \tag{4.22}$$

De manera que la tabla de verdad resulta

Tabla 4.1: Tabla de verdad para I

Entrada	*Salida*
$\|0\rangle$	$\|0\rangle$
$\|1\rangle$	$\|1\rangle$

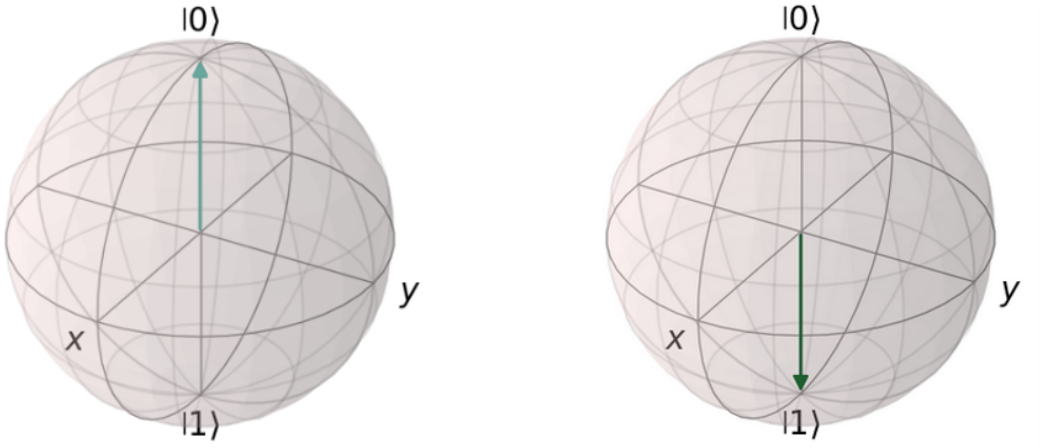


Figura 4.3: La compuerta I, al actuar sobre los estados base $|0\rangle$ y $|1\rangle$ no produce cambio alguno

Compuerta X de Pauli
También se le llama *Bit-Flip* (inversión de bit), se representa mediante el cuadro con una "X" dentro. La compuerta X de Pauli aplica un pulso de rotación $\frac{\pi}{2}$ a lo largo del eje X. La acción de esta compuerta es voltear el Qbit del estado $|0\rangle$ al $|1\rangle$ y viceversa. La forma matricial de esta compuerta equivale a la matriz X de Pauli

$$X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \tag{4.23}$$

La acción de esta compuerta sobre los estados base se puede observar al realizar el producto:

$$X|0\rangle = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix} = |1\rangle \qquad X|1\rangle = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix} = |0\rangle \tag{4.24}$$

De manera que la tabla de verdad resulta

Tabla 4.2: Tabla de verdad para X

Entrada	*Salida*
$\|0\rangle$	$\|1\rangle$
$\|1\rangle$	$\|0\rangle$

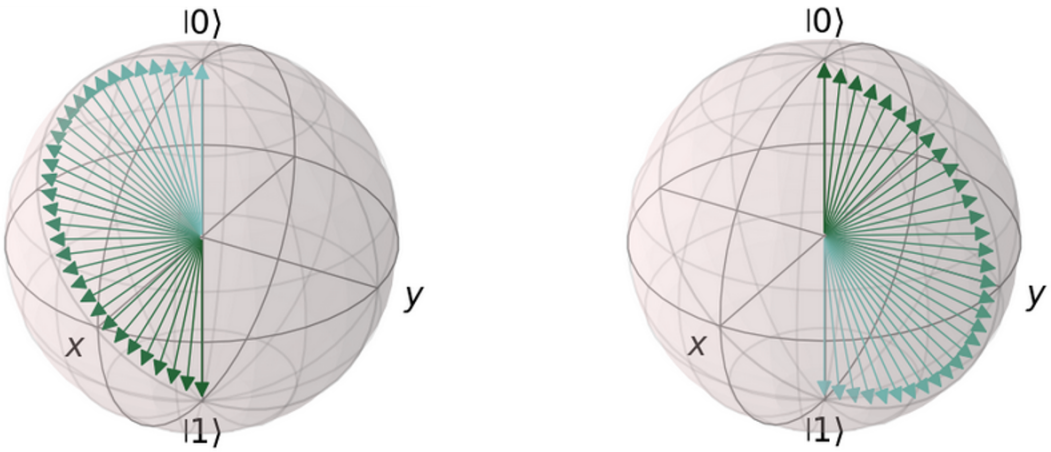


Figura 4.4: Compuerta X actuando sobre los vectores base $|0\rangle$ y $|1\rangle$, puede se puede notar que actúa intercambiando los estados. En este y en los diagramas subsiguientes el estado inicial se indica con color aguamarina y el final en verde oscuro. El resto de flechas, con degrado entre ambos tonos, son una representación pictórica para destacar el eje de rotación

La compuerta X actúa como la compuerta NOT, negando correctamente los estados de de la base computacional. Puede dar la impresión de que aplicar X a un Qbit arbitrario $|\psi\rangle = \cos\left(\frac{\theta}{2}\right)|0\rangle + e^{i\phi}\sin\left(\frac{\theta}{2}\right)|1\rangle$ resultará en su antípoda $|\psi^{\perp}\rangle$, el vector que se encuentra diametralmente opuesto en la esfera. Geométricamente, el punto antípoda se obtiene al prolongar una línea recta desde el estado original a través del origen para cortar la superficie de la esfera de Bloch en el lado opuesto. Puede demostrarse que [181] :

$$|\psi^{\perp}\rangle = \cos\left(\frac{\pi-\theta}{2}\right)|0\rangle + e^{i(\phi+\pi)}\sin\left(\frac{\pi-\theta}{2}\right)|1\rangle = \sin\left(\frac{\theta}{2}\right)|0\rangle - e^{i\phi}\cos\left(\frac{\theta}{2}\right)|1\rangle \tag{4.25}$$

mientras que la evaluación directa de X

$$X|\psi\rangle = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{pmatrix} \cos\left(\frac{\theta}{2}\right) \\ e^{i\phi}\sin\left(\frac{\theta}{2}\right) \end{pmatrix} = \begin{pmatrix} e^{i\phi}\sin\left(\frac{\theta}{2}\right) \\ \cos\left(\frac{\theta}{2}\right) \end{pmatrix} = e^{i\phi}\sin\left(\frac{\theta}{2}\right)|0\rangle + cos\left(\frac{\theta}{2}\right)|1\rangle \tag{4.26}$$

por lo que X no es una compuerta NOT universal para Qbits.

Compuerta Y de Pauli
La compuerta Y de Pauli aplica una de rotación de π radianes a lo largo del eje Y. La forma matricial de esta compuerta equivale a la matriz Y de Pauli

$$Y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix} \tag{4.27}$$

La acción de esta compuerta sobre los estados base se puede observar al realizar el producto:

$$Y|0\rangle = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ i \end{pmatrix} = i|1\rangle \qquad Y|1\rangle = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} -i \\ 0 \end{pmatrix} = -i|0\rangle \tag{4.28}$$

De manera que la tabla de verdad resulta

Tabla 4.3: Tabla de verdad para Y

Entrada	*Salida*
$\|0\rangle$	$i\|1\rangle$
$\|1\rangle$	$-i\|0\rangle$

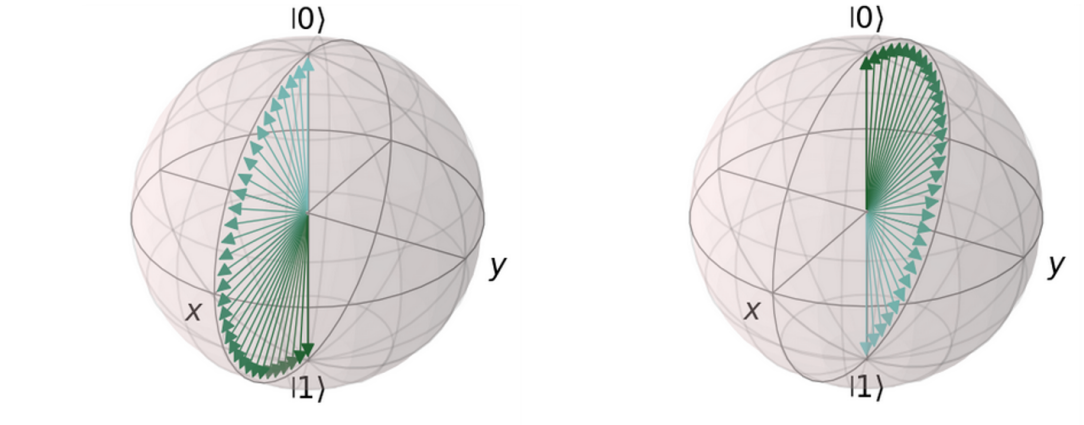


Figura 4.5: Computerta Y

Compuerta Z de Pauli

La compuerta Z de Pauli aplica una de rotación de π radianes a lo largo del eje Z. La forma matricial de esta compuerta equivale a la matriz Z de Pauli

$$Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \tag{4.29}$$

La acción de esta compuerta sobre los estados base se puede observar al realizar el producto:

$$Z|0\rangle = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix} = |0\rangle \qquad Z|1\rangle = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ -1 \end{pmatrix} = -|1\rangle \tag{4.30}$$

Se observa que se produce un cambio de fase en π al estado $|1\rangle$ mientras deja el estado $|0\rangle$ sin cambios. De manera que la tabla de verdad resulta

Tabla 4.4: Tabla de verdad para Z

Entrada	*Salida*
$\|0\rangle$	$\|0\rangle$
$\|1\rangle$	$-\|1\rangle$

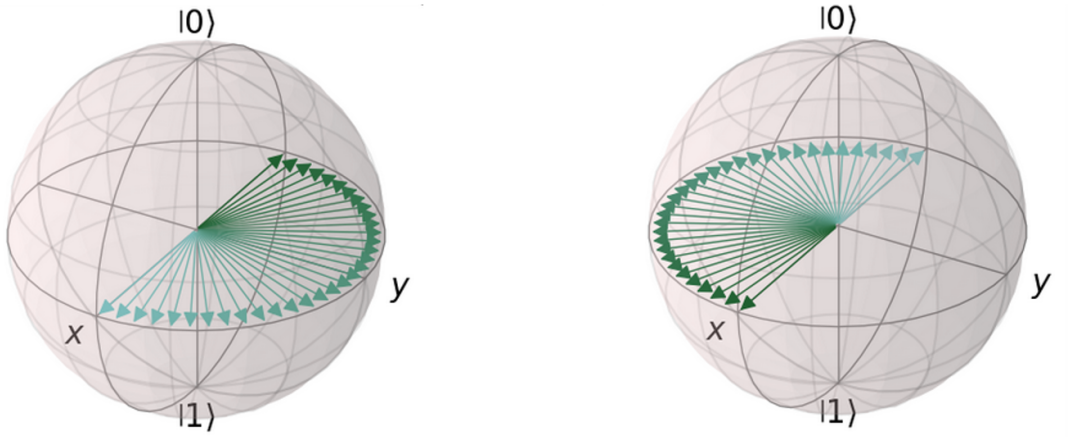


Figura 4.6: Compuerta Z, Los vectores base $|0\rangle$ y $|1\rangle$ permanecerán invariantes ante esa rotación, por lo que su acción se ilustra sobre $|+\rangle$ y $|-\rangle$

Compuerta S
La compuerta S, también llamada $Z90$ o compuerta de fase aplica una de rotación de $\frac{\pi}{2}$ radianes a lo largo del eje Z. La forma matricial de esta compuerta es

$$S = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\,\pi/2} \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix} \tag{4.31}$$

La acción de esta compuerta sobre los estados base se puede observar al realizar el producto:

$$S|0\rangle = \begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix} = |0\rangle \qquad S|1\rangle = \begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ i \end{pmatrix} = i\,|1\rangle \tag{4.32}$$

De manera que la tabla de verdad resulta

Tabla 4.5: Tabla de verdad para S

Entrada	*Salida*
$\|0\rangle$	$\|0\rangle$
$\|1\rangle$	$i\,\|1\rangle$

Compuerta $S^\dagger$
La compuerta $S^\dagger$, es la transpuesta conjugada de la compuerta S. Aplica una de rotación de $-\frac{\pi}{2}$ radianes a lo largo del eje Z. La forma matricial de esta compuerta es

$$S^\dagger = \begin{bmatrix} 1 & 0 \\ 0 & e^{-i\pi/2} \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & -i \end{bmatrix} \tag{4.33}$$

La acción de esta compuerta sobre los estados base se puede observar al realizar el producto:

$$S^\dagger|0\rangle = \begin{bmatrix} 1 & 0 \\ 0 & -i \end{bmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix} = |0\rangle \qquad S^\dagger|1\rangle = \begin{bmatrix} 1 & 0 \\ 0 & -i \end{bmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ i \end{pmatrix} = i\,|1\rangle \tag{4.34}$$

De manera que la tabla de verdad resulta

Tabla 4.6: Tabla de verdad para $S^\dagger$

Entrada	*Salida*
$\|0\rangle$	$\|0\rangle$
$\|1\rangle$	$-i\,\|1\rangle$

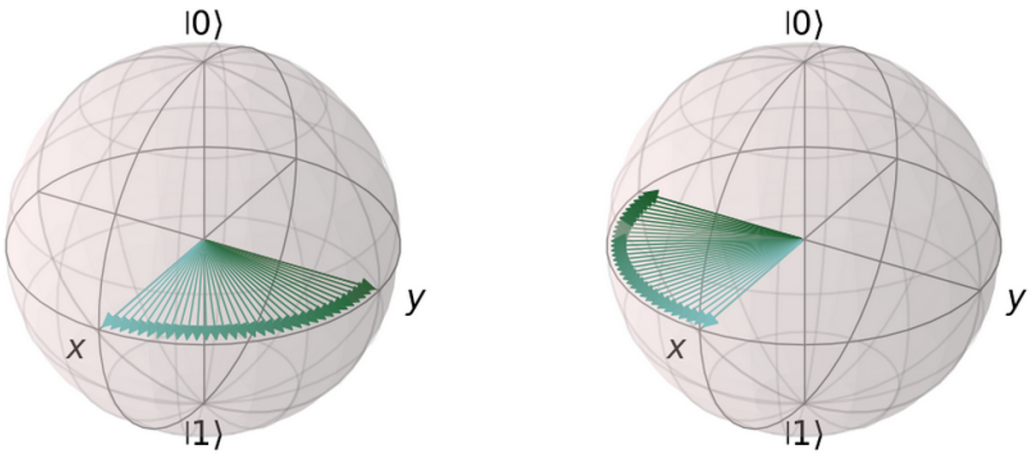


Figura 4.7: Compuerta S (izquierda) y $S^\dagger$ (derecha)

Compuerta R_X
La compuerta R_X gira el Qbit a lo largo del eje X en θ radianes. La forma matricial de esta compuerta es la siguiente:

$$R_X(\theta) = \begin{bmatrix} \cos(\theta/2) & -i\sin(\theta/2) \\ -i\sin(\theta/2) & \cos(\theta/2) \end{bmatrix} \tag{4.35}$$

Se puede escribir en términos de la matriz de Pauli X y la identidad

$$R_X(\theta) = \cos(\theta/2)\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} - i\sin(\theta/2)\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} = \cos(\theta/2)I - i\sin(\theta/2)X \tag{4.36}$$

Compuerta R_Y
La compuerta R_Y gira el Qbit a lo largo del eje Y en θ radianes. La forma matricial de esta compuerta es la siguiente:

$$R_Y(\theta) = \begin{bmatrix} \cos(\theta/2) & -\sin(\theta/2) \\ \sin(\theta/2) & \cos(\theta/2) \end{bmatrix} \tag{4.37}$$

Se puede escribir en términos de la matriz de Pauli Y y la identidad

$$R_Y(\theta) = \cos(\theta/2)\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} - i\sin(\theta/2)\begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix} = \cos(\theta/2)I - i\sin(\theta/2)Y \tag{4.38}$$

Compuerta R_Z
También llamada compuerta de corrimiento de fase. La compuerta R_Z gira el Qbit a lo largo del eje Z en ϕ radianes. La forma matricial de esta compuerta es la siguiente:

$$R_Z(\phi) = \begin{bmatrix} e^{-i\phi/2} & 0 \\ 0 & e^{i\phi/2} \end{bmatrix} \tag{4.39}$$

Se puede escribir en términos de la matriz de Pauli Y y la identidad

$$R_Z(\phi) = \cos(\phi/2)\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} - i\sin(\phi/2)\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} = \cos(\phi/2)I - i\sin(\phi/2)Z \tag{4.40}$$

Una factorización simple en esta matriz puede conducir a una forma particularmente útil

$$R_Z(\phi) = \begin{bmatrix} e^{-i\phi/2} & 0 \\ 0 & e^{i\phi/2} \end{bmatrix} = e^{-i\phi/2}\begin{bmatrix} 1 & 0 \\ 0 & e^{i\phi} \end{bmatrix} \tag{4.41}$$

recordando que el factor de fase global $e^{-i\phi/2}$ carece de importancia física, se puede considerar que la siguiente matriz realiza la misma operación

$$R_Z(\phi) = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\phi} \end{bmatrix} \tag{4.42}$$

Que describe una una rotación en un ángulo ϕ' para un vector de estado arbitrario $|\psi\rangle = \cos\left(\frac{\theta}{2}\right)|0\rangle + e^{i\phi}\sin\left(\frac{\theta}{2}\right)|1\rangle$, por lo que también se le denomina *compuerta desplazamiento de fase.* Al actuar sobre la base, esta compuerta transforma $|0\rangle$ en $|0\rangle$ y $|1\rangle$ en $e^{i\phi}|1\rangle$. Dado que las fases globales no tienen significado físico, los estados de la base computacional, $|0\rangle$ y $|1\rangle$, permanecen sin cambios. Sin embargo, al actuar sobre un estado arbitrario, se obtiene una diferencia de fase relaviva que si puede medirse

$$R_Z(\phi')|\psi\rangle = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\phi'} \end{bmatrix} \begin{pmatrix} \cos\left(\frac{\theta}{2}\right) \\ e^{i\phi}\sin\left(\frac{\theta}{2}\right) \end{pmatrix} = \cos\left(\frac{\theta}{2}\right)|0\rangle + e^{i(\phi+\phi')}\sin\left(\frac{\theta}{2}\right)|1\rangle \tag{4.43}$$

Las rotaciones en la esfera de Bloch no se ajustan a lo que la intuición o la experiencia cotidiana. En particular, por lo general, una rotación de 2π de un objeto sólido sobre cualquier eje, restaura ese objeto a su orientación inicial. Pero esto no es lo que sucede para rotaciones en la esfera de Bloch. Al rotar un estado cuántico a través de 2π en la esfera de Bloch, no se le devuelve al estado inicial, debido a la aparición de un factor de fase. Por ejemplo. si se rota un estado puro arbitrario en 2π se obtiene :

$$R_Z(2\pi)|\psi\rangle = \begin{bmatrix} e^{-i\pi} & 0 \\ 0 & e^{i\pi} \end{bmatrix} \begin{pmatrix} \cos\left(\frac{\theta}{2}\right) \\ e^{i\phi}\sin\left(\frac{\theta}{2}\right) \end{pmatrix} = \begin{bmatrix} -1 & 0 \\ 0 & -1 \end{bmatrix} \begin{pmatrix} \cos\left(\frac{\theta}{2}\right) \\ e^{i\phi}\sin\left(\frac{\theta}{2}\right) \end{pmatrix} = -\left[\cos\left(\frac{\theta}{2}\right)|0\rangle + e^{i\phi}\sin\left(\frac{\theta}{2}\right)|1\rangle\right] = -|\psi\rangle \tag{4.44}$$

Las compuertas que rotan los estados alrededor de los ejes X, Y y Z son de especial importancia, debido a que toda transformación de un Qbit puede descomponerse como una secuencias ellas.

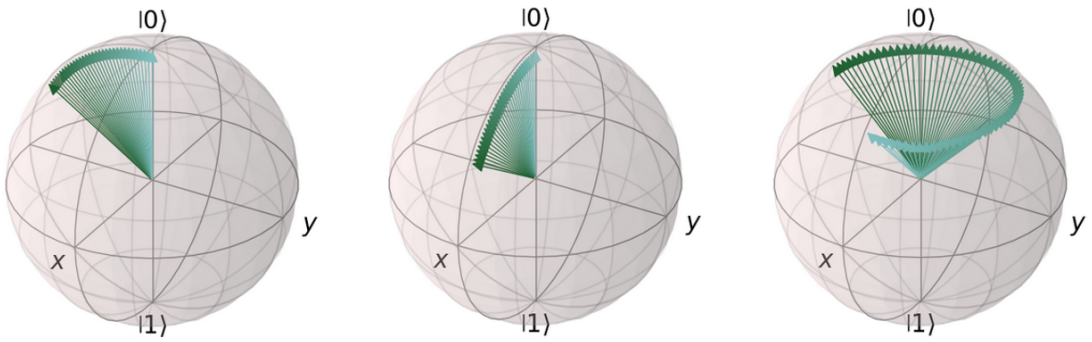


Figura 4.8: De izquierda a derecha, compuertas R_X, R_Y, R_Z.

Compuerta Hadamard

Por simplicidad, con frecuencia se le llama compuerta H y es una de las compuertas más empleada en cómputo cuántico. Toma su nombre a partir de la transformada de Walsh-Hadamard, una transformada ortogonal, similar a la transformada de Fourier, solo que, en lugar de trabajar con funciones sinusoidales, hace corresponder a una serie numérica otra serie formada por funciones de Walsh[4], por lo que puede usarse para descomponer a un vector en una superposición de funciones de Walsh. Las funciones de Walsh solo toman valores entre -1 y +1, por lo que resultan ser las más adecuadas para transformaciones de series discretas de números, mientras que la transformada de Fourier es óptima para señales continuas, como puede apreciarse en la figura 4.9.

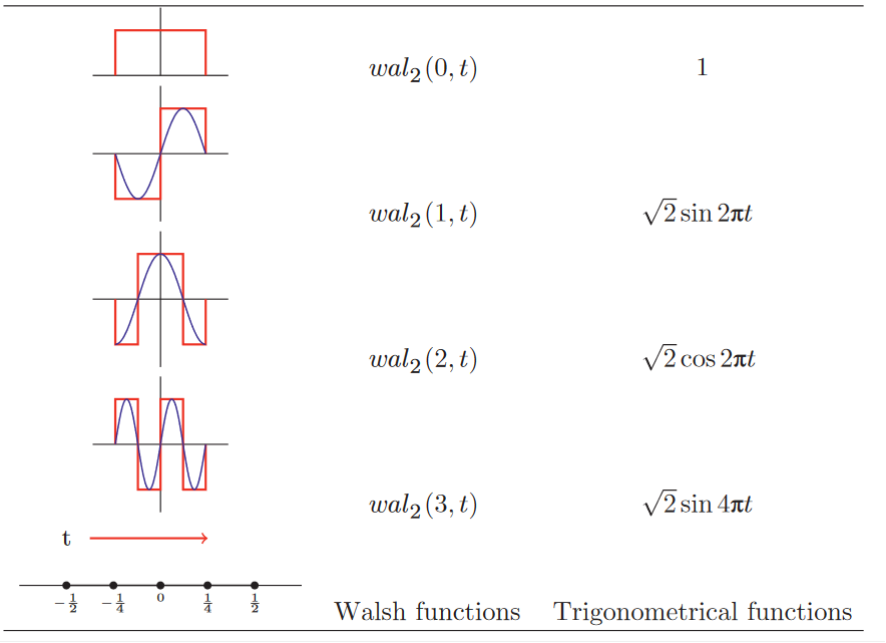


Figura 4.9: Funciones trigonométricas y funciones de Walsh[249]

El muestreo uniforme de las funciones de Walsh de cualquier orden N da como resultado matrices de Walsh-Hadamard del orden correspondiente $H_w(N)$[85]. Para el caso $N = 1$ se define:

$$H_w(1) = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \tag{4.45}$$

Muchos algoritmos cuánticos utilizan la transformada de Walsh-Hadamard como paso inicial, debido a que asigna n Qbits inicializados en $|0\rangle$ a una superposición de los 2^n estados ortogonales en la base computacional $|0\rangle, |1\rangle$ con igual peso [132]. En la caso de un solo Qbit, la compuerta H se define como

$$H = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \tag{4.46}$$

Geométricamente, la acción de la compuerta Hadamard se puede interpretar como una rotación del estado del Qbit π radianes alrededor de un eje diagonal al plano XZ. Dado que la información del Qbit a lo largo del eje X se gira hacia el eje Z, la compuerta H brinda una forma de medir el Qbit en base X. Esta compuerta convierte la base computacional $\{|0\rangle, |1\rangle\}$ en la nueva base $\{|+\rangle, |-\rangle\}$, llamada *base polar*, cuyos estados son una superposición de los estados de la base computacional.

Para observar la forma en que la compuerta Hadamard realiza sobre los estados base, se puede realizar el producto:

$$H|0\rangle = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 1 \end{pmatrix} = \frac{1}{\sqrt{2}} (|0\rangle + |1\rangle) = |+\rangle \tag{4.47}$$

$$H|1\rangle = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ -1 \end{pmatrix} = \frac{1}{\sqrt{2}} (|0\rangle - |1\rangle) = |-\rangle \tag{4.48}$$

De manera que la tabla de verdad resulta

Tabla 4.7: Tabla de verdad para H

Entrada	*Salida*
$\|0\rangle$	$\|+\rangle$
$\|1\rangle$	$\|-\rangle$

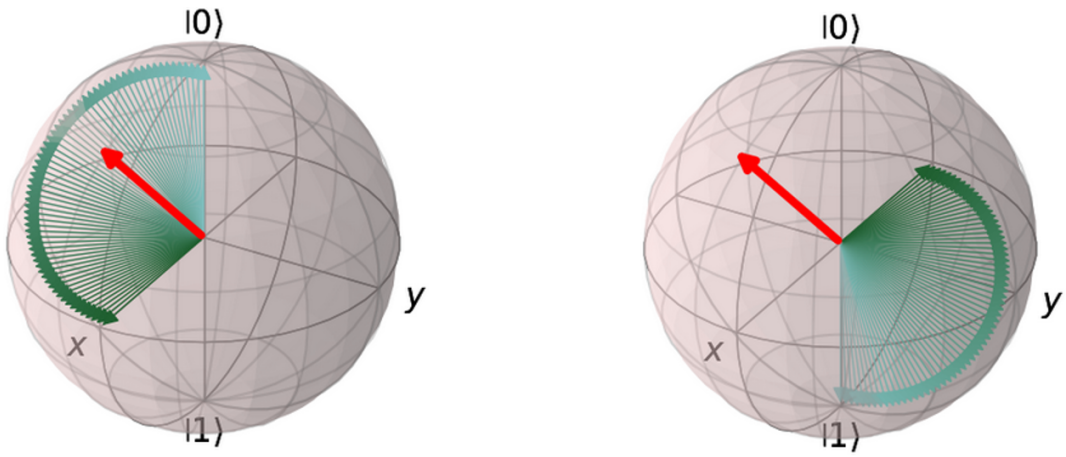


Figura 4.10: La compuerta H rota el estado del Qbit por π radianes alrededor de un eje diagonal, sobre el plano XZ, aquí indicado con la flecha roja.

La aplicación de la compuerta H puede verse de manera equivalente como una rotación del Qbit $\pi/2$ radianes a lo largo del eje Y seguida de una rotación de $\pi/2$ radianes a lo largo del eje X, como puede verse en la figura 4.11. Aunque aparentemente bastaría con la acción de la primera rotación sobre el eje Y, porque una rotación sobre el eje X aparentemente no produce resultado, hay una sutileza que la imagen por si misma no puede revelar. Al comparar la rotación en $\pi/2$ radianes sobre el eje Y:

$$R_Y(\pi/2) = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & -1 \\ 1 & 1 \end{pmatrix}. \tag{4.49}$$

con la definición dela compuerta H dada previamente, podrá verse que hay una diferencia en la posición del -1 en la segunda columna. La aplicación de la compuerta X devuelve al -1 en $R_Y(\pi/2)$ al lugar correcto para obtener la compuerta de Hadamard. La aplicación de H sobre el estado base $H|0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$ entrega el mismo resultado que $R_Y(\pi/2)|0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$, pero aplicar $R_Y(\pi/2)|1\rangle = \frac{1}{\sqrt{2}}(-|0\rangle + |1\rangle)$ es diferente de $H|1\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$, la aplicación de X después de R_Y cambia los estados $|0\rangle$ y $|1\rangle$, haciendo que el eje Z gire hacia el eje X y viceversa.

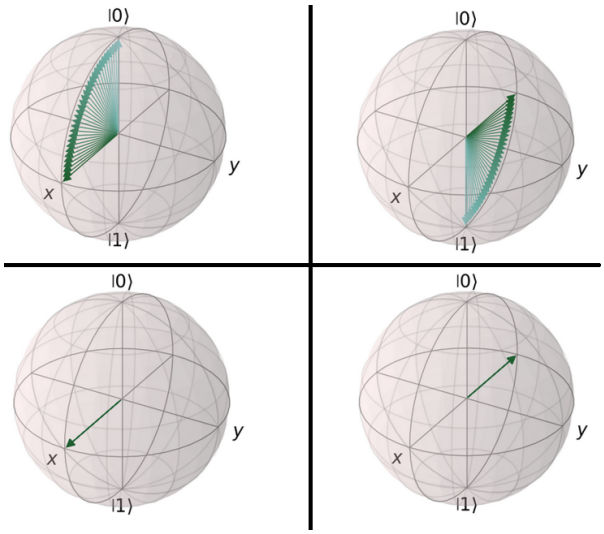


Figura 4.11: La compuerta H puede verse como una rotación del Qbit $\pi/2$ radianes a lo largo del eje Y (imágenes superiores, a la izquierda aplicada sobre $|0\rangle$ y a la derecha sobre$|1\rangle$) seguida de una rotación de $\pi/2$ radianes a lo largo del eje X (imágenes inferiores).

Como un factor de fase global no afecta las predicciones, es posible representar a esta compuerta de una manera alternativa, notando que

$$H|0\rangle = |+\rangle = \frac{1}{\sqrt{2}}\left[\begin{pmatrix}1\\0\end{pmatrix} + (\cos(0) + i\sin(0))\begin{pmatrix}0\\1\end{pmatrix}\right] = \frac{1}{\sqrt{2}}\left[\begin{pmatrix}1\\0\end{pmatrix} + e^{i\pi(0)}\begin{pmatrix}0\\1\end{pmatrix}\right] \tag{4.50}$$

$$H|1\rangle = |-\rangle = \frac{1}{\sqrt{2}}\left[\begin{pmatrix}1\\0\end{pmatrix} + (\cos(\pi) + i\sin(\pi))\begin{pmatrix}0\\1\end{pmatrix}\right] = \frac{1}{\sqrt{2}}\left[\begin{pmatrix}1\\0\end{pmatrix} + e^{i\pi(1)}\begin{pmatrix}0\\1\end{pmatrix}\right] \tag{4.51}$$

Se puede escribir de manera compacta

$$H|x\rangle = \frac{1}{\sqrt{2}}\left[\begin{pmatrix}1\\0\end{pmatrix} + e^{\pi(x)}\begin{pmatrix}0\\1\end{pmatrix}\right] = \frac{1}{\sqrt{2}}\Big(|0\rangle + (-1)^x|1\rangle\Big) \tag{4.52}$$

Una forma mas conveniente resulta

$$H|x\rangle = \frac{1}{\sqrt{2}}\left[(-1)^0|0\rangle + (-1)^x|1\rangle\right] = \frac{1}{\sqrt{2}}\left[(-1)^{x(0)}|0\rangle + (-1)^{x(1)}|1\rangle\right] = \frac{1}{\sqrt{2}}\Sigma_{y=0}(-1)^{xy}\big|y\big\rangle \tag{4.53}$$

$$H|x\rangle = \frac{1}{\sqrt{2}}\Sigma_{y=0}(-1)^{xy}\big|y\big\rangle \tag{4.54}$$

Si se aplica una Hadamard después de otra se obtiene

$$\begin{aligned} H\Big[H|x\rangle\Big] = H\left[\frac{1}{\sqrt{2}}\big(|0\rangle + (-1)^x|1\rangle\big)\right] &= \frac{1}{\sqrt{2}}\Big[H|0\rangle + (-1)^x H|1\rangle\Big] = \frac{1}{\sqrt{2}}\Big[|+\rangle + (-1)^x|-\rangle\Big] \\ &= \frac{1}{\sqrt{2}}\left[\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) + (-1)^x\frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)\right] \\ &= \frac{1}{2}\Big[\big(|0\rangle + |1\rangle\big) + (-1)^x\big(|0\rangle - |1\rangle\big)\Big] \end{aligned} \tag{4.55}$$

Así, para $|x\rangle = |0\rangle$:

$$H\Big[H|0\rangle\Big] = \frac{1}{2}\Big[\big(|0\rangle + |1\rangle\big) + (-1)^0\big(|0\rangle - |1\rangle\big)\Big] = \frac{1}{2}\Big[|0\rangle + |1\rangle + |0\rangle - |1\rangle\Big] = |0\rangle \tag{4.56}$$

Mientras que $|x\rangle = |1\rangle$:

$$H\Big[H|1\rangle\Big] = \frac{1}{2}\Big[\big(|0\rangle + |1\rangle\big) + (-1)^1\big(|0\rangle - |1\rangle\big)\Big] = \frac{1}{2}\Big[|0\rangle + |1\rangle - |0\rangle + |1\rangle\Big] = |1\rangle \tag{4.57}$$

Por lo que resulta que la compuerta Hadamard es su propia inversa. Como puede verse, al aplicar la compuerta Hadamard sobre los vectores base $|0\rangle$ y $|1\rangle$, se obtienen las superposiciones $|+\rangle$ y $|-\rangle$, respectivamente, mientras que una segunda aplicación devolverá los vectores base originales. Es importante destacar que, mientras $|0\rangle$ y $|1\rangle$ son los vectores propios de σ_x, $|+\rangle$ y $|-\rangle$ son los vectores propios de σ_x y cada miembro de una base puede representarse como una superposición de los vectores de la otra.

Existen algunas otras compuertas lógicas de cuánticas de un solo Qbit (como puede consultarse en[144]), pero resulta mas conveniente mencionar el tipo más general de compuerta cuántica para un solo Qbit. Ya que un estado puro de un solo Qbit está representado por un punto en la superficie de la esfera de Bloch, el efecto de una sola compuerta de un Qbit que actúa en este estado es mapearlo en algún otro punto de la esfera de Bloch. Aunque se mencionó que es posible descomponer cualquier compuerta cuántica unitaria de 1 Qbit como una secuencia de rotaciones alrededor de los ejes X, Y y Z, resulta que cualquier operación unitaria en un solo Qbit se puede construir usando solo compuertas Hadamard y de corrimiento de fase. Así, un estado genérico $\big|\psi\big\rangle = \cos\Big(\frac{\theta}{2}\Big)|0\rangle + e^{i\phi}\sin\Big(\frac{\theta}{2}\Big)|1\rangle$ puede alcanzarse (salvo un factor de fase) desde $|0\rangle$ mediante la aplicación de la siguiente serie de transformaciones

$$R_Z(\pi/2 + \phi)HR_z(\theta)|0\rangle = e^{i\theta/2}\Big(\cos(\theta/2)|0\rangle + e^{i\phi}\sin(\theta/2)|1\rangle\Big) \tag{4.58}$$

4.3.3. Diagramas de circuito

De manera similar al caso clásico, los circuitos cuánticos tienen una representación gráfica. Los diagramas contienen información acerca de la acción de una secuencia de compuertas que actúan sobre los Qbits. Del lado izquierdo se coloca el estado inicial de los Qbits, mientras que en el extremo derecho se encuentra el estado final. En la parte central se colocan las compuertas. El diagrama mas simple posible muestra a un Qbit inicialmente en el estado $|\psi\rangle$, recibe la acción de una compuerta U de 1 Qbit, que realiza una operación unitaria sobre el Qbit, con el resultado de este adquiere el nuevo estado $U|\psi\rangle$.

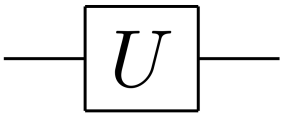


Figura 4.12: El diagrama de circuito cuántico mas simple es aquél que solo tiene un Qbit, representado por un solo cable, una compuerta de un Qbit U. El lado izquierdo representa la entrada, el derecho la salida

La línea que entra y sale de la caja U se vuelve útil cuando se comienzan a mostrar secuencias de compuertas, por analogía con los diagramas clásicos se denomina cable. Cada circuito cuántico corresponde a un operador unitario particular. Las operaciones unitarias también implican que los circuitos cuánticos tienen el mismo número de entradas que de salidas. Además, los circuitos cuánticos son acíclicos, lo que significa que no hay circuitos de retroalimentación o circuitos con ciclos.

Un circuito cuántico consta de tres etapas:

1. Preparar una cierta cantidad de Qbits con el estado inicial bien definido que se denomina el estado fiduciario de la computadora, por ejemplo, el estado $|0...00\rangle$;
2. Manipular la función de onda de la computadora cuántica, es decir, realizar operaciones de puertas cuánticas mediante las transformaciones unitarias correspondientes U_i. Esto significa usar la mecánica cuántica para resolver el problema.
3. Medir los Qbits proyectándolos a un conjunto de bits clásicos utilizando una base computacional al finalizar el algoritmo.

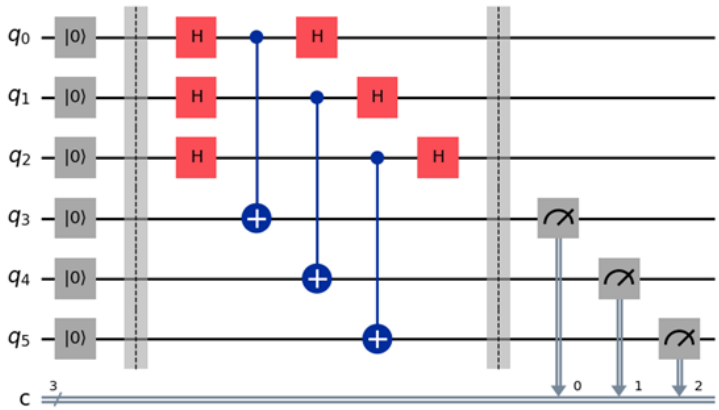


Figura 4.13: Las tres etapas del circuito cuántico. Se comienza con un dispositivo que tiene seis registros cuánticos q[0], q[1], q[2], q[3], q[4], q[5]. Se tienen también tres registros clásicos, c[0], c[1],c[2]. Los rectángulos naranja representan compuestas cuánticas Hadamard, mientras que los círculos y líneas azules corresponden a compuertas de dos Qbits llamadas CNOT, que se definirán en la próxima sección.

Estas etapas pueden apreciarse en el circuito ilustrado aquí. Se comienza con un dispositivo que tiene seis registros cuánticos q[0], q[1], q[2], q[3], q[4], q[5]. Se tienen también tres registros clásicos, c[0], c[1],c[2]. Se parte del estado inicial $|0\rangle$, aunque la gran mayoría de los algoritmos cuánticos requerirán que los Qbits se inicialicen en un estado de superposición, y posiblemente algunos de ellos deberán configurarse para el retroceso de fase, una técnica útil que se verá mas adelante. Lo siguiente que puede observarse es que

se aplica una compuerta H a q[0], q[1], q[2], tras lo cual se tiene una secuencia de 3 compuertas $CNOT$, representadas por lo símbolos en azul. La compuerta $CNOT$ se definió de manera clásica con anterioridad, indicando que requería un bit de control, que permanece constante, y un objetivo, que cambiará su valor de acuerdo al valor del bit de control. En la figura 4.13, las compuertas $CNOT$ están colocadas de la siguiente forma:

- La primera se aplica con q[0] como control y q[3] como destino.
- La segunda se aplica con q[1] como control y q[4] como destino.
- La tercera se aplica con q[2] como control y q[5] como destino.

Mas detalles sobre la versión cuántica de esta compuerta se verán en la próxima sección.

Después de este paso, se aplica una compuerta H una vez más a los Qbits q[0], q[1], q[2]. Finalmente se mide el estado de los Qbits q[3], q[4], q[5], proyectándolos a los registros clásicos c[0],c[1],c[2], lo que se representa mediante el cuadro gris con un medidor en su interior (la aguja sobre el semicírculo). Es posible demostrar que cualquier medida colectiva compleja de muchos Qbits siempre se puede realizar en la base computacional, con la condición de que esté precedida por una transformación unitaria adecuada. Si se desea realizar una medición en cualquier otra base, definida por un conjunto de estados ortonormal y completo, solo se necesita transformar de manera unitaria desde la base que se desea realizar la medición a la base computacional y finalmente se debe medir [208].

En el diseño de algoritmos cuánticos, el conjunto de transformaciones unitarias permitidas usualmente está restringido a aquellas que pueden construirse mediante productos de transformaciones unitarias que solo pueden actuar sobre uno o dos Qbits la vez [133]. Esto se debe a que la producción de compuertas cuánticas de orden superior implica dificultades técnicas aún más duras de tratar que los problemas que supone la construcción de compuertas confiables de uno y dos Qbits, que ya son suficientemente complicados [3]. Aunque esto pueda parecer limitante, en realidad no es así, debido a que toda transformación unitaria arbitraria se puede aproximar con el grado de precisión que se desee, utilizando suficientes compuertas de uno y dos Qbit [208]. A diferencia del caso clásico, en el que la construcción de operaciones lógicas reversibles requiere de al menos una compuerta de tres bits clásicos, es decir, la compuerta de Toffoli [99], en computación cuántica existe una extensión de dicha compuerta que puede elaborarse a partir de un pequeño número de compuertas de uno y dos Qbits [16].

4.3.4. Compuertas de dos Qbits y generación de entrelazamiento

A pesar de la enorme variedad de compuertas cuánticas, realizar una serie de operaciones de un solo Qbit sobre una colección de n Qbits es insuficiente para obtener ventaja respecto al uso de computadoras clásicas. En primer lugar, la realización de cálculos no triviales suele requerir de la aplicación condicional de operaciones, es decir, dependiendo de los valores que un conjunto de Qbits tenga, a otro conjunto de Qbits se le aplicarán, o no, algunas operaciones. Las compuertas que implementan este tipo de operaciones, semejantes a las sentencias condicionales típicas de los lenguajes de programación de alto nivel (if, then, else) se denominan *compuertas controladas*. Una compuerta controlada de dos bits actúa de manera selectiva. En esta compuerta, uno de los bits de entrada devolverá la misma salida y se denomina "control", mientras el otro será el "objetivo". Dependiendo del valor que el control tenga, al objetivo se le aplicara, o no, una transformación.

Una segunda razón, de mayor importancia, radica en el hecho de que todo cálculo realizado por un sistema que opere empleando exclusivamente compuertas de un Qbit puede ser *eficientemente* simulado por una computadora clásica. Sin embargo, tan pronto como se agrega una compuerta de dos Qbits a un conjunto de compuertas de un solo Qbit se puede realizar un cálculo que ya no puede simularse de manera eficiente con computadoras clásicas, debido a que el entrelazamiento, un fenómeno exclusivamente cuántico, se presenta al introducir interacciones entre Qbits.

El trabajo con compuertas do dos o mas Qbits requiere considerar en primer lugar el espacio de Hilbert correspondiente. Se parte de un sistema con dos componentes, cada uno de ellos localizado en un espacio de Hilbert $\mathscr{H}_1$ y $\mathscr{H}_2$. El sistema completo está contenido en $\mathscr{H} = \mathscr{H}_1 \otimes \mathscr{H}_2$, de manera que un vector en $\mathscr{H}$ resulta:

$$|\psi\rangle = \sum_{i,j} c_{ij} |e_{1,i}\rangle \otimes |e_{2,j}\rangle = \sum_{i,j} c_{ij} |e_{1,i} e_{2,j}\rangle \tag{4.59}$$

[3] No obstante, existe interés en la implementación de compuertas cuánticas que puedan actuar en mas de dos Qbits, por ejemplo [234]. La motivación radica en la posibilidad de simplificar la implementación de algoritmos cuánticos complejos, reemplazando una secuencia compleja de puertas de dos Qbits con una sola operación de tres Qbits, lo que a su vez promete una ejecución más rápida con una fidelidad potencialmente mayor[198].

donde $|e_{1,i}\rangle$ y $|e_{2,j}\rangle$ es una base ortonormal para $\mathscr{H}_1$ y $\mathscr{H}_2$, respectivamente y los coeficientes cumplen la condición de normalización$\sum_{i,j}|c_{ij}|^2 = 1$. Si un estado $|\psi\rangle \in \mathscr{H}$ puede escribirse como el producto tensorial de dos vectores $|\psi_1\rangle \otimes |\psi_2\rangle$, con $|\psi_1\rangle \in \mathscr{H}_1$ y $|\psi_2\rangle \in \mathscr{H}_2$, se le denomina *estado separable* o *estado producto tensorial*. Se puede interpretar a un estado de manera intuitiva y clásica, pensando que «el primer sistema *se encuentra* en el estado $|\psi_1\rangle$ y el segundo sistema *se encuentra* en el estado $|\psi_2\rangle$». Los estados que no pueden representarse como producto tensorial, se denominan *no separables*, o *estados entrelazados*, y carecen de una interpretación intuitiva.

Debe ser claro que el conjunto de estados separable tiene dimensión igual a la suma de las dimensiones de cada espacio de Hilbert ($dim\mathscr{H}_1 + dim\mathscr{H}_2$) mientras que el espacio total $\mathscr{H}$, que incluye estados separables y no separables, tiene diferente dimensión, a saber $dim\mathscr{H} = dim\mathscr{H}_1 dim\mathscr{H}_2$. Por ejemplo, sea $\mathscr{H}^2$ el espacio de Hilber de dos Qbits dado por el producto tensorial del espacio de Hilbert de un solo Qbit consigo mismo: $\mathscr{H}^2 = \mathscr{H} \otimes \mathscr{H}$. Por lo tanto, la base computacional está dada por los estados producto de los estados base de $\mathscr{H}$:

$$|0\rangle \otimes |0\rangle = |00\rangle = \begin{pmatrix}1\\0\\0\\0\end{pmatrix} \qquad |0\rangle \otimes |1\rangle = |01\rangle = \begin{pmatrix}0\\1\\0\\0\end{pmatrix} \qquad |1\rangle \otimes |0\rangle = |10\rangle = \begin{pmatrix}0\\0\\1\\0\end{pmatrix} \qquad |1\rangle \otimes |1\rangle = |11\rangle = \begin{pmatrix}0\\0\\0\\1\end{pmatrix} \tag{4.60}$$

Al trabajar con dos Qbits, el estado resultante será el producto tensorial de los estados individuales. Esta operación suele abreviarse colocando los valores que representan a los estados dentro de un mismo ket, siempre que no haya confusión. Luego, un estado genérico de dos Qbits puede escribirse en la base computacional como

$$\alpha|00\rangle + \beta|01\rangle + \gamma|10\rangle + \delta|11\rangle \tag{4.61}$$

con los coeficientes $\alpha, \beta, \gamma, \delta$ complejos, que, por otra parte, deben cumplir con la condición de normalización, por lo que $|\alpha|^2 + |\beta|^2 + |\gamma|^2 + |\delta|^2 = 1$, y dado que todos estado está definido hasta un factor de fase, se tienen seis grados de libertad, por lo que en general, no será posible considerar a un estado arbitrario como separable. El ejemplo típico es el estado $|\psi\rangle = \frac{1}{\sqrt{2}}\big(|00\rangle + |11\rangle\big)$. Si este estado trata de descomponerse como el producto de vectores

$$|\psi\rangle = \big(c_1|0\rangle + c_2|1\rangle\big) \otimes \big(d_1|0\rangle + d_2|1\rangle\big) = c_1|0\rangle \otimes \big(d_1|0\rangle + d_2|1\rangle\big) + c_2|1\rangle \otimes \big(d_1|0\rangle + d_2|1\rangle\big) =$$

$$c_1 d_1|0\rangle \otimes |0\rangle + c_1 d_2|0\rangle \otimes |1\rangle + c_2 d_1|1\rangle \otimes |0\rangle + c_2 d_2|1\rangle \otimes |1\rangle =$$

$$c_1 d_1|00\rangle + c_1 d_2|01\rangle + c_2 d_1|10\rangle + c_2 d_2|11\rangle \tag{4.62}$$

se verá que resulta imposible, debido a que se debe cumplir simultáneamente que $c_1 d_2 = c_2 d_1 = 0$ y $c_1 d_1 = c_2 d_2 = \frac{1}{\sqrt{2}}$. Por lo tanto, $|\psi\rangle = \frac{1}{\sqrt{2}}\big(|00\rangle + |11\rangle\big)$ es un estado no separable. Para espacios de Hilbert de dimensiones mayores, se tendrá que $dim\mathscr{H}_1 dim\mathscr{H}_2 \gg dim\mathscr{H}_1 + dim\mathscr{H}_2$, lo que indica que la mayoría de los estados en un espacio de Hilbert de un sistema bipartito estarán entrelazados cuando los espacios de Hilbert constituyentes sean de dimensiones más grandes. La complejidad del entrelazamiento crece exponencialmente con el número de Qbits: mientras que un estado separable de n Qbits depende solo de $2n$ parámetros reales, el estado más general (entrelazado) tiene $2(2^n—1)$ grados de libertad [21].

Las compuertas de 2 Qbits permiten que dos sistemas cuánticos interactúen y, por lo tanto, son necesarias para generar entrelazamiento. Como se mencionó en la sección referente a cómputo clásico, ninguna de las compuertas de dos bits es reversible (la entrada será de dos bits, pero su salida es a un bit), pero podrían utilizarse si se tiene en cuenta que toda función $f : \{0,1\}^m \to \{0,1\}^n$ irreversible puede "incrustarse" en una función reversible, a partir de la función $\tilde{f} : \{0,1\}^{m+n} \to \{0,1\}^{m+n}$ tal que $\tilde{f}(a,b) = (a, [b + f(a)](mod 2^n))$. Esta expresión que permite "preservar la entrada" en la salida, garantizando la reversibilidad. En el caso de funciones de 2 bits a un bit, la función $\tilde{f}$ sería de 3 bits a 3 bits. Sin embargo, existe un camino mas sencillo. Considérese la operación disyunción exclusiva, implementada mediante la compuerta XOR. En esta compuerta, el valor de la salida es cero cuando las dos entrada son iguales, y es uno cuando son distintas.

Tabla 4.8: Tabla de verdad para XOR.

Entrada		Salida
A	**B**	**Y**
0	0	1
0	1	0
1	0	0
1	1	1

Tabla 4.9: Tabla de verdad para XOR reversible.

Entrada		Salida	
A	**B**	**X**	**Y**
0	0	0	0
0	1	0	1
1	0	1	1
1	1	1	0

Si se agrega como primer bit de salida una copia del bit A, se tendrá una compuerta de dos bits que, además de ser reversible, (permite identificar a la entrada a partir de la salida), también será una compuerta controlada. Todo lo que hace falta es verla desde la perspectiva apropiada: el primer bit no cambia y será el control. Si el control es cero, el segundo bit se queda igual, en caso contrario, el segundo bit de salida se invierte respecto al valor que tenía a la entrada. Este es el origen de la compuerta CNOT antes mencionada. La versión cuántica de esta compuerta actúa sobre los estados de la base computacional, $\{|00\rangle, |01\rangle, |10\rangle, |11\rangle\}$, como la compuerta XOR :

$$CNOT\Big(|x\rangle \otimes |y\rangle\Big) = |x\rangle \otimes |x \oplus y\rangle \tag{4.63}$$

donde $x, y = 0, 1$ y $\oplus$ indica la adición módulo 2, es decir, $0 \oplus 0 = 0$, $0 \oplus 1 = 1$, $1 \oplus 0 = 1$, $1 \oplus 1 = 0$, como se muestran en la tabla de verdad.

Tabla 4.10: Tabla de verdad para $CNOT$

$Entrada$	$Salida$
$\|00\rangle$	$\|00\rangle$
$\|01\rangle$	$\|01\rangle$
$\|10\rangle$	$\|11\rangle$
$\|11\rangle$	$\|10\rangle$

Puede verse que primer Qbit en CNOT actúa como control mientras el segundo es el objetivo. La compuerta cambia el estado del Qbit de destino si el Qbit de control está en el estado $|1\rangle$ y no hace nada si el Qbit de control está en el estado $|0\rangle$. La representación matricial está dada por:

$$CNOT = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{4.64}$$

que resulta ser una matriz involutiva. Aunque en el caso clásico la compuerta CNOT carece de importancia, su contraparte cuántica resulta ser un componente básico de los circuitos cuánticos [182].

Un ligero análisis a la matriz asociada a CNOT será útil para entender mejor la lógica de las compuertas de dos Qbits. Para ello, se nombra como A a la configuración antes mencionada de CNOT, debido a que otras compuertas controladas pueden obtenerse con algunos cambios. Al aplicar A a un estado arbitrario $|\psi\rangle$, puede verse que las dos filas superiores afectan únicamente al primer Qbit, mientras que las dos filas inferiores actúan únicamente sobre el segundo Qbit, invirtiéndolo:

$$A|\psi\rangle = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{pmatrix} e_0 \\ e_1 \\ e_2 \\ e_3 \end{pmatrix} = \begin{pmatrix} e_0 \\ e_1 \\ e_3 \\ e_2 \end{pmatrix} \tag{4.65}$$

La descripción es más sencilla si se separa a la matriz A en bloques:

$$A = \left[\begin{array}{cc|cc} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ \hline 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{array}\right] = \begin{bmatrix} \mathbb{1} & \\ & \mathbb{U} \end{bmatrix}$$

El primer bloque representa la identidad. Junto con un segundo bloque lleno de ceros, garantiza que el estado del Qbit de control no se verá afectado como resultado de la transformación. Al actuar sobre las bases computacionales, únicamente uno de los e_i será uno, mientras que todos los demás son cero. Esto significa que la inversión *no se notará* cuando $e_0 = 1$ o $e_1 = 1$, porque en ambos casos $e_2 = e_3 = 0$. Por el contrario, la inversión *será evidente* si $e_2 = 1$ o $e_3 = 1$, reproduciendo las transformaciones antes mencionadas, $|00\rangle \rightarrow |00\rangle, |01\rangle \rightarrow |01\rangle, |10\rangle \rightarrow |11\rangle, |11\rangle \rightarrow |10\rangle$, es decir, si el Qbit de control está en $|0\rangle$, el Qbit objetivo se deja intacto, pero si el Qbit de control es $|1\rangle$, el estado del objetivo se *invertirá*. Esto se puede ver en el cuarto bloque, señalado con $\mathbb{U}$. Las dos últimas filas de la matriz que representan lo que sucede con los estados $|10\rangle$ y $|11\rangle$. Los unos fuera de la diagonal realizan el intercambio.

La compuerta CNOT tiene la siguiente representación en el diagrama de circuito:

Figura 4.14: Compuerta CNOT. El Qbit de control se indica con el pequeño circulo negro. El Qbit objetivo se señala con el circulo que contiene una cruz. La representación alternativa incluye explícitamente la compuerta X

La generalización cuántica natural de la compuerta CNOT es la compuerta U-controlada (CU), en la que se modifican los valores del bloque $\mathbb{U}$ por aquellos que corresponden a una compuerta arbitraria de un Qbit

$$\mathbb{U} = \begin{bmatrix} U_{11} & U_{12} \\ U_{21} & U_{22} \end{bmatrix} \tag{4.66}$$

con lo que se obtiene la matriz C_U:

$$CU = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & U_{11} & U_{12} \\ 0 & 0 & U_{21} & U_{22} \end{bmatrix} \tag{4.67}$$

De acuerdo con esta notación, la compuerta C_X es la compuerta CNOT. Al aplicar C_U sobre un estado arbitrario $|\psi\rangle$ se obtiene:

$$C_U |\psi\rangle = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & U_{11} & U_{12} \\ 0 & 0 & U_{21} & U_{22} \end{bmatrix} \begin{pmatrix} e_0 \\ e_1 \\ e_2 \\ e_3 \end{pmatrix} = \begin{pmatrix} e_0 \\ e_1 \\ U_{11}e_2 + U_{12}e_3 \\ U_{21}e_2 + U_{22}e_3 \end{pmatrix} \tag{4.68}$$

La acción de la matriz $\mathbb{U}$ no se notará, nuevamente, en los casos $e_0 = 1$ o $e_1 = 1$, que representan a $|00\rangle \rightarrow |00\rangle$ y $|01\rangle \rightarrow |01\rangle$. Por el contrario, solo se observará si $e_2 = 1$ (caso $|10\rangle$) o si $e_2 = 1$, (caso $|11\rangle$) que resultaran en:

$$C_U |10\rangle = \begin{pmatrix} 0 \\ 0 \\ U_{11} \\ U_{21} \end{pmatrix} \qquad C_U |11\rangle = \begin{pmatrix} 0 \\ 0 \\ U_{12} \\ U_{22} \end{pmatrix} \tag{4.69}$$

La compuerta C_U se representa en un diagrama de circuitos como:

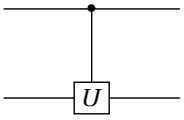


Figura 4.15: Compuerta C_U . El Qbit de control se indica con el pequeño círculo negro. El Qbit objetivo se indica con un cuadro, que representa además la operación U que se debe aplicar

Es posible que alguna aplicación en concreto requiera cambiar la condición, de tal forma que la compuerta actúe cuando el Qbit de control se encuentre en $|0\rangle$, y que no haga nada si se encuentra en $|1\rangle$. Una

configuración distinta de CNOT, conocida como compuerta CNOT negativa, permitirá realizar esto, que se identificará como $CNOT_B$:

$$CNOT_B\left|\psi\right\rangle = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} e_0 \\ e_1 \\ e_2 \\ e_3 \end{pmatrix} = \begin{pmatrix} e_1 \\ e_0 \\ e_2 \\ e_3 \end{pmatrix} \tag{4.70}$$

En este caso, la inversión no se notará cuando $e_2 = 1$, correspondiente a $|10\rangle$ o $e_3 = 1$, correspondiente a $|11\rangle$. En ambos casos $e_0 = e_1 = 0$. Por el contrario, la inversión será evidente si $e_0 = 1$, correspondiente a $|00\rangle$, o $e_1 = 1$, correspondiente a $|01\rangle$. Por tanto, se obtienen las siguientes transformaciones:

Tabla 4.11: Tabla de verdad para $CNOT_B$

Entrada	*Salida*
$\|00\rangle$	$\|01\rangle$
$\|01\rangle$	$\|00\rangle$
$\|10\rangle$	$\|10\rangle$
$\|11\rangle$	$\|11\rangle$

La compuerta $CNOT_B$ se representa en un diagrama de circuito mediante un símbolo semejante a la compuerta CNOT. Para indicar que se activa con $|0\rangle$, el bit de control se indica con un círculo hueco.

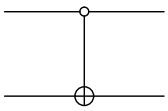


Figura 4.16: Compuerta $CNOT_B$. El Qbit de control se indica con el pequeño círculo hueco. El Qbit objetivo se señala con el círculo que contiene una cruz.

Para la implementación de la compuerta $CNOT_B$, se tiene el siguiente circuito, en el que puede verse que se realiza una inversión de bit antes y después de una compuerta CNOT ordinaria :

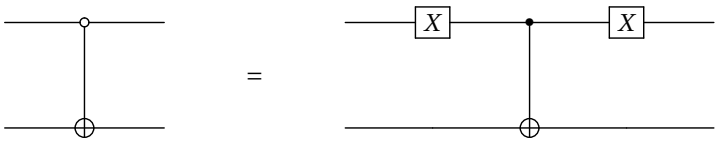


Figura 4.17: Un circuito cuántico para una compuerta $CNOT_B$. Se requieren dos compuertas X y una compuerta CNOT ordinaria

Considerando la discusión anterior sobre aplicar una compuerta arbitraria controlada, puede intentarse un análisis semejante para $CNOT_B$, en bloques

$$B = \left[\begin{array}{cc|cc} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ \hline 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{array}\right] = \begin{bmatrix} \mathbb{U} & \\ & \mathbb{1} \end{bmatrix}$$

se cambia $\mathbb{U}$ por la representación matricial de la compuerta deseada

$$C_U\left|\psi\right\rangle = \begin{bmatrix} U_{11} & U_{12} & 0 & 0 \\ U_{21} & U_{22} & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} e_0 \\ e_1 \\ e_2 \\ e_3 \end{pmatrix} = \begin{pmatrix} U_{11}e_0 + U_{12}e_1 \\ U_{21}e_0 + U_{22}e_1 \\ e_2 \\ e_3 \end{pmatrix} \tag{4.71}$$

También es posible intercambiar los papeles del Qbit de control y el objetivo. Para que el primer Qbit se invierta si el segundo Qbit se encuentra en $|1\rangle$, se usa la compuerta $CNOT_C$

Tabla 4.12: Tabla de verdad y matriz de $CNOT_C$

Entrada	*Salida*
$\|00\rangle$	$\|00\rangle$
$\|01\rangle$	$\|11\rangle$
$\|10\rangle$	$\|11\rangle$
$\|11\rangle$	$\|01\rangle$

$$\text{C}=\begin{bmatrix}1&0&0&0\\0&0&0&1\\0&0&1&0\\0&1&0&0\end{bmatrix} \tag{4.72}$$

Para la implementación de la compuerta $CNOT_C$, se tiene el siguiente circuito:

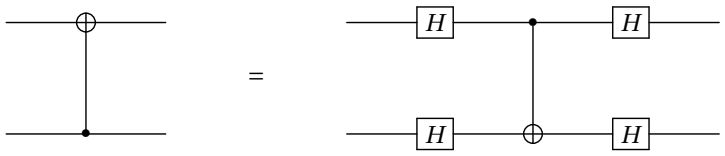


Figura 4.18: Circuito cuántico para una compuerta $CNOT_C$. Se requieren cuatro compuertas Hadamard y una compuerta CNOT ordinaria

Para que el primer Qbit se invierta si el segundo Qbit se encuentra en $|0\rangle$, se usa la compuerta $CNOT_D$

Tabla 4.13: Tabla de verdad y matriz de $CNOT_D$

Entrada	*Salida*
$\|00\rangle$	$\|00\rangle$
$\|01\rangle$	$\|11\rangle$
$\|10\rangle$	$\|10\rangle$
$\|11\rangle$	$\|01\rangle$

$$\text{D}=\begin{bmatrix}0&0&1&0\\0&1&0&0\\1&0&0&0\\0&0&0&1\end{bmatrix} \tag{4.73}$$

El entrelazamiento se utiliza ampliamente como un poderoso recurso de cálculo en computación cuántica, pero es imposible conseguirlo a partir de compuertas de un solo Qbit. Si se parte de un estado separable dado por $|\psi\rangle = |\psi_{n-1}\rangle \otimes |\psi_{n-1}\rangle \otimes \cdots \otimes |\psi_0\rangle$, se puede mover tanto como se desee cualquier Qbit sobre la esfera de Bloch, de manera que se obtenga $|\psi'\rangle = |\psi'_{n-1}\rangle \otimes |\psi'_{n-1}\rangle \otimes \cdots \otimes |\psi'_0\rangle$, donde cualquier estado del tipo $|\psi_i\rangle$ puede transformarse mediante puertas que actúan sobre el i-ésimo Qbit en cualquier superposición de los estados $|0\rangle$ y $|1\rangle$, pero el estado de n-Qbits sigue siendo separable.

Afortunadamente, la interacción de dos Qbits es suficiente para generar entrelazamiento y puede conseguirse a través de la compuerta $CNOT$. A diferencia de XOR clásica, $CNOT$ puede aplicarse a cuaquier superposición de los estados bases computacionales, lo que será útil en la generación de estados entrelazados. Para comprobar que $CNOT$ puede generar estados entrelazados, considérese primero el producto de un estado arbitrario $\alpha|0\rangle + \beta|1\rangle$ por la base $|0\rangle$, es decir, $\big(\alpha|0\rangle + \beta|1\rangle\big) \otimes |0\rangle = \alpha|0\rangle \otimes |0\rangle + \beta|1\rangle \otimes |0\rangle = \alpha|00\rangle + \beta|10\rangle$. A continuación, se aplica $CNOT$ a este producto

$$CNOT\Big[\big(\alpha|0\rangle + \beta|1\rangle\big) \otimes |0\rangle\Big] = \begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&0&1\\0&0&1&0\end{bmatrix}\begin{pmatrix}\alpha\\0\\\beta\\0\end{pmatrix} = \begin{pmatrix}\alpha\\0\\0\\\beta\end{pmatrix} = \alpha|00\rangle + \beta|11\rangle \tag{4.74}$$

que resulta ser un estado no separable, como se comprobó con anterioridad. Considérese ahora el caso especial $\alpha = \beta = \frac{1}{\sqrt{2}}$, que produce $CNOT\big(\frac{1}{\sqrt{2}}(\alpha|00\rangle + \beta|10\rangle)\big) = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$. El estado resultante es uno de los estados de la *base de Bell*, una base ortonormal para un sistema de dos Qbits dada por [205]:

$$\begin{aligned} |\phi^+\rangle &= \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle) \qquad |\phi^-\rangle = \frac{1}{\sqrt{2}}(|00\rangle - |11\rangle) \\ |\psi^+\rangle &= \frac{1}{\sqrt{2}}(|01\rangle + |10\rangle) \qquad |\psi^-\rangle = \frac{1}{\sqrt{2}}(|01\rangle - |10\rangle) \end{aligned} \tag{4.75}$$

Cada vector se llama *estado de Bell* o *vector de Bell*. Estos estados tienen una característica interesante: una medición proyectiva en uno de los dos Qbits en cualquier base ortonormal produce dos resultados con igual probabilidad, lo que indica una falta total de información sobre los subsistemas individuales [177]. Los estados de la base de Bell pueden obtenerse a partir de la base computacional estándar, utilizando una compuerta Hadamard y una compuerta $CNOT$, mediante la transformación $BELL|i_1,i_0\rangle = CNOT\big(H\otimes\mathbb{1}\big)|i_1,i_0\rangle$ (con i_0 e $i_1 \in \{0,1\}$), como puede verse en la figura 4.19. En la primera etapa de este circuito, la transformación de Hadamard convierte el Qbit superior en un estado de superposición $\frac{1}{\sqrt{2}}(|0\rangle+|1\rangle)$. En la siguiente etapa, esta superposición se mueve a la entrada de control de la compuerta $CNOT$, y el Qbit objetivo se invierte si y sólo si el valor del Qbit de control es igual a uno.

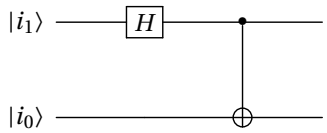


Figura 4.19: A través del uso de una sola compuerta Hadamard, es posible transformar a los estados de la base computacional $|i_1,i_0\rangle$ (es decir,$|00\rangle, |01\rangle, |10\rangle, |11\rangle$) a los estados de la base Bell.

Es posible invertir esta transformación mostrada simplemente ejecutando el circuito en la dirección opuesta, gracias a que tanto $CNOT$ como Hadamard son su propia inversa. Como resultado, cualquier estado de la base de Bell se transforma en un estado separable, esto es posible porque se ha utilizado una compuerta de dos Qbits. En este punto es posible determinar mediante una medición estándar en la base computacional cuál de los cuatro estados de Bell estaba presente al principio. Este procedimiento se conoce como *medición de Bell*. La tabla de verdad para $BELL$ se muestra a continuación:

Tabla 4.14: Tabla de verdad para $BELL$

$Entrada$	$Salida$
$\|00\rangle$	$\|\phi^+\rangle = \frac{1}{\sqrt{2}}(\|00\rangle+\|11\rangle)$
$\|01\rangle$	$\|\psi^+\rangle = \frac{1}{\sqrt{2}}(\|01\rangle+\|10\rangle)$
$\|10\rangle$	$\|\phi^-\rangle = \frac{1}{\sqrt{2}}(\|00\rangle-\|11\rangle)$
$\|11\rangle$	$\|\psi^-\rangle = \frac{1}{\sqrt{2}}(\|01\rangle-\|10\rangle)$

En circuitos clásicos, intercambiar la posición de dos bits es una operación trivial que, por lo general, no requiere de la implementación de una compuerta específica para realizarse. En el caso cuántico resulta diferente. Si se desea la interacción entre dos Qbits, por lo general se requiere que se encuentren uno al lado del otro, lo que puede requerir una reorganización de los Qbits, a través del intercambio de sus estados. La compuerta $SWAP$ permite mover información de un Qbit a otro.

$$SWAP\,|\psi\phi\rangle = |\phi\psi\rangle \tag{4.76}$$

Conviene recordar que en los diagramas de circuito, cada línea corresponde a un Qbit y cualquier secuencia de compuertas lógicas debe leerse de izquierda (entrada) a derecha (salida). De abajo hacia arriba, los Qbits van desde el menos significativo (i_o) hasta el más significativo (i_{n-1}).

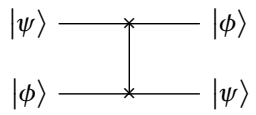


Figura 4.20: Representación de la compuerta $SWAP$ en un diagrama de circuito . La tarea que realiza es intercambiar la información contenida en ambos Qbits

Tanto $CNOT$ como $SWAP$ pueden identificarse o entenderse en términos de compuertas clásicas. Existen, no obstante, compuertas cuánticas de dos Qbits que no tienen una compuerta análoga en cómputo clásico. Un ejemplo de esto es la compuerta de corrimiento de fase controlado ($CPHASE$). Si el Qbit de control se encuentra en el estado $|1\rangle$, realizará un cambio de fase al Qbit objetivo, produciendo la salida

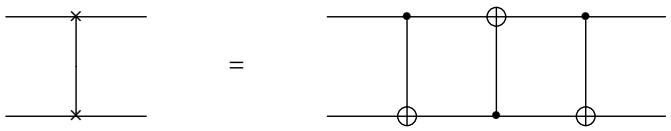


Figura 4.21: Un circuito cuántico para una compuerta *SWAP*, en donde puede observarse que su implementación requiere de tres compuertas CNOT.

$CPHASE|11\rangle = e^{i\delta}|11\rangle$. Esta compuerta puede realizarse utilizando una compuerta $CNOT$ y algunas compuertas de corrimiento de fase de un Qbit.

4.4. Compuertas cuánticas universales

La utilidad del modelo de circuito en computación clásica se debe al hecho de que una secuencia de operaciones elementales (por ejemplo, $NAND$ y $COPY$) permite construir cálculos arbitrariamente complejos. Existe un resultado similar en el cómputo cuántico: toda operación unitaria en el espacio Hilbert de n Qbits se puede descomponer en puertas $CNOT$ de un Qbit y dos Qbits.

Es posible construir un circuito cuántico para una compuerta C_U en términos de una compuerta $CNOT$ y compuertas de un solo Qbit. Si la matriz U que representa a la operación a controlar tiene una descomposición de la forma $U = e^{ia}R_Z(b)\cdot R_Y(c)\cdot R_Z(d)$, se pueden definir las matrices A, B, C como sigue 5.2:

$$A = R_Z\Big(\frac{d-b}{2}\Big) \qquad B = R_Y\Big(-\frac{c}{2}\Big)\cdot R_Z\Big(-\frac{d+b}{2}\Big) \qquad C = R_Z(b)\cdot R_Y\Big(\frac{c}{2}\Big) \qquad \Delta = diag(1, e^{ia}) \tag{4.77}$$

donde $diag(1, e^{ia})$ se refiere a una matriz diagonal, de 1×1, cuya única entrada es e^{ia}

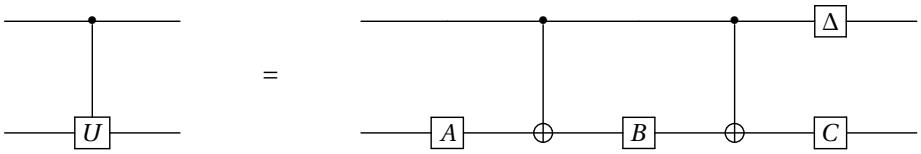


Figura 4.22: Un circuito cuántico para una compuerta C_U, donde U es una compuerta arbitraria de un Qbit

Este resultado puede extenderse al caso en el que el Qbit objetivo es $\alpha|0\rangle + \beta|1\rangle$. La compuerta C_U únicamente actuará cuando el Qbit de control esté en $|1\rangle$.

$$C_U\Big[\,|0\rangle\otimes\big(\alpha|0\rangle+\beta|1\rangle\big)\Big] = |0\rangle\otimes\big(\alpha|0\rangle+\beta|1\rangle\big) \tag{4.78}$$

$$C_U\Big[\,|1\rangle\otimes\big(\alpha|0\rangle+\beta|1\rangle\big)\Big] = |1\rangle\otimes U\Big[\big(\alpha|0\rangle+\beta|1\rangle\big)\Big] \tag{4.79}$$

Una demostración alternativa puede consultarse en [296].

4.5. Evaluación de funciones

La tarea básica de cualquier computadora clásica se refiere a la evaluación de una función clásica $f : \{0,1\}^n \rightarrow \{0,1\}^m$ que toma una cadena de n bits como entrada y produce cadena de m bits como salida. Si la entrada binaria está dada por $x = (x_0, x_1, ..., x_{n-1})$, la computadora clásica calculará f a partir de la evaluación de cada uno de los x_i hacia las salidas $f(x_i)$, utilizando las operaciones AND, OR, NOT y COPY. En cada paso del proceso computacional para una computadora clásica , el valor de la cadena cambia de un valor determinado a otro valor determinado.

En el caso de las computadoras cuánticas, las funciones se computan de una manera un tanto distinta. Debido a que las compuertas cuánticas son unitarias y, por tanto, lógicamente reversibles, no es posible calcular directamente una función f mediante una operación unitaria que transforme $|x\rangle$ en $\big|f(x)\big\rangle$. La propuesta ingenua para definir la acción de la transformación a través de $U_f|x\rangle = \big|f(x)\big\rangle$, presenta algunas inconsistencias. En primer lugar, es posible que la función f admita entradas que no tengan el mismo tamaño que las salidas, es decir, $m \neq n$, por lo que la aplicación de del operador U_f podría cambiar el número de Qbits

en el registro. Además, aún exigiendo que $m = n$, cabe la posibilidad de que la función no sea invertible: de manera que si $f(x) = f(y)$ para alguna $x \neq y$, entonces $U_f|x\rangle = U_f|y\rangle$ pero $U_f^\dagger U_f|x\rangle \neq U_f^\dagger U_f|y\rangle$, por lo que no se podría construir el operador adjunto $U_f^\dagger$, aunque los operadores deben tener un adjunto definido. En otras palabras, dos kets ortogonales $|x\rangle$ y $|y\rangle$ pueden evolucionar al mismo estado, $|f(x)\rangle = |(y)\rangle$, violando la reversibilidad lógica y, por tanto, la unitaridad. Por ejemplo, en las funciones booleanas de un solo bit:

Tabla 4.15: Funciones booleanas de un bit

x	f_0	f_1	f_2	f_3
0	0	0	1	1
1	0	1	0	1

Las funciones f_0 y f_3 proyectan el bit de entrada en 0 y 1 respectivamente y, por tanto, no son invertibles. Claramente, para que una compuerta sea reversible, debe ser posible reconstruir los bits de entrada conociendo solo el diseño de la compuerta y los bits de salida, por lo que cada bit de entrada debe preservarse de alguna forma en la salida. Con anterioridad se mencionó que la compuerta CNOT puede utilizarse para implementar una operación reversible. Además, genera una correlación específica entre el registro de entrada (que guarda el valor x) y el registro auxiliar (que guarda el valor $f(x)$). Esto significa que, al medir el registro auxiliar y obtener el valor $f(x)$, automáticamente se 'selecciona' o colapsa el registro de entrada al valor correspondiente x, de modo que ambos registros quedan asociados de manera única. Es decir, nunca ocurre una situación en la que se mida un resultado $f(x)$ en el segundo registro mientras que el primer registro quede asociado a un valor $y \neq x$, ya que la estructura del entrelazamiento garantiza que cada valor evaluado en la función permanezca ligado con su entrada original.

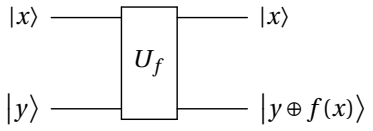


Figura 4.23: Un circuito para la evaluación de una función f.

Para la evaluación de una función, una computadora cuántica utiliza dos registros; el primer registro para almacenar los datos de entrada y el segundo para los datos de salida. Cada posible entrada es representada por $|x\rangle$, el estado cuántico del primer registro. De manera análoga, cada salida posible $y = f(x)$ está representada por $|y\rangle$ el estado cuántico del segundo registro. Luego, los estados correspondientes a distintas entradas y distinta salida serán ortogonales, $\langle x|x'\rangle = \delta_{xx'}$, $\langle y|y'\rangle = \delta_{yy'}$. Una estrategia simple para transformar funciones no invertibles en invertibles se presentó previamente: consiste en agregar algunos bits adicionales que sirvan para identificar al bit de entrada a partir del bit de salida. Por tanto, se define el efecto de la función en toda la base computacional, para todo $x \in \{0,1\}^n$ y $y \in \{0,1\}^m$,

$$U_f(|x\rangle \otimes |y\rangle) = |x\rangle \otimes |y \oplus f(x)\rangle = |x, y \oplus f(x)\rangle. \tag{4.80}$$

Una computadora cuántica implementa una función booleana f leyendo el estado del Qbit $|x\rangle$ y almacenando la salida $f(x)$ en el estado de $|y\rangle$. Si este último se inicializa en $|y\rangle$, su estado final es $|y \oplus f(x)\rangle$, en particular, si el estado inicial es $|0\rangle$, el estado final de los dos Qbits será $|x\rangle \otimes |f(x)\rangle$. La transformación U_f es invertible, y además, $U_f = U_f^\dagger$, ya que $U_f^2 = \mathbb{1}$ dado que $a \oplus b \oplus b = a$ para toda $a, b \in \{0,1\}$. Como resultado,

$$U_f|x\rangle|y \oplus f(x)\rangle = |x\rangle|y \oplus f(x) \oplus f(x)\rangle = |x\rangle|y\rangle \tag{4.81}$$

por lo que los dos problemas mencionados quedan resueltos.

Pueden considerarse, por ejemplo, las funciones f_i, que transforman dos bits en uno, presentadas con anterioridad (tabla 4.16). Al tratarse de funciones no invertibles, su evaluación en una computadora cuántica requerirá del uso de un Qbit ancilla inicializado en $|0\rangle$

$$U_{f_i}|x_1, x_0\rangle|0\rangle = |x_1, x_0\rangle|f_i(x_1, x_0)\rangle \tag{4.82}$$

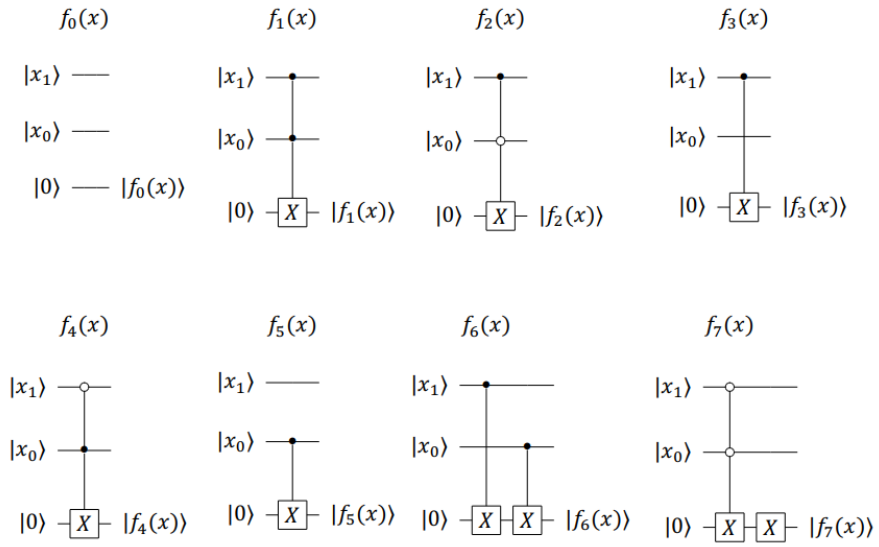


Figura 4.24: Circuitos cuánticos para funciones booleanas de dos bits [177]. Estos circuitos implementan las funciones booleanas de dos bits $f:\{0,1\}^2 \rightarrow \{0,1\}$. Los Qbits de entrada se inicializan en $|x_1\rangle$ y $|x_0\rangle$, el Qbit ancilla se inicializa en $|0\rangle$. Al final del circuito, los dos Qbits de entrada se dejan sin cambios y el Qbit ancilla almacena $f(x_1,x_0)$.

Tabla 4.16: Funciones booleanas en dos variables

x_1	x_0	f_0	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9	f_{10}	f_{11}	f_{12}	f_{13}	f_{14}	f_{15}
0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
0	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
1	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1

Para generalizar estos conceptos para el caso de funciones arbitrarias $f:\{0,1\}^n \rightarrow \{0,1\}^m$, debe recordarse que cada elemento de la cadena de Qbits de salida $f(\mathbf{x}) = (f_{m-1}(\mathbf{x}),...,f_0(\mathbf{x}))$ puede expresarse con minitérminos, de acuerdo con la expresión:

$$f_i(\mathbf{x}) = m_{x_1} \vee m_{x_2} \vee ... \vee m_{x_k} \tag{4.83}$$

y cada minitérmino puede implementarse en una computadora cuántica con una compuerta generalizada $C^n - NOT$. Toda función $f:\{0,1\}^n \rightarrow \{0,1\}^m$ puede implementarse en una computadora cuántica siempre que se proporcionen suficientes Qbits, por lo que tiene la capacidad de transformar una cadena binaria en cualquier otra cadena binaria.

Tabla 4.17: Ejemplo de una función de 2 bits a 3 bits definida en término de sus entradas y salidas. Una forma de obtener una expresión algebráica equivalente a ella consiste en construirla mediante sus minitérminos

x_2	x_1	x_0	$f_1(\mathbf{x})$	$f_0(\mathbf{x})$
0	0	0	0	0
0	0	1	1	0
0	1	0	0	0
0	1	1	0	0
1	0	0	0	1
1	0	1	0	1
1	1	0	0	1
1	1	1	0	0

Por lo tanto, las máquinas cuánticas pueden ejecutar cualquier algoritmo clásico. Por ejemplo, la función $f:\{0,1\}^3 \rightarrow \{0,1\}^2$, definida en la tabla 4.17, puede expresarse con minitérminos como: $f(\mathbf{x}) = (f_1(\mathbf{x}), f_0(\mathbf{x}))$, donde:

$$f_0(\mathbf{x}) = m_{100}(\mathbf{x}) \vee m_{101}(\mathbf{x}) \vee m_{110}(\mathbf{x}) \qquad f_1(\mathbf{x}) = m_{001}(\mathbf{x}) \tag{4.84}$$

La figura 4.25 muestra el circuito asociado con $f(\mathbf{x})$, definida en la tabla 4.17. La forma mas básica de diseñar este circuito consiste en observar los minitérminos, es decir, buscar los valores para los que $f_i(\mathbf{x})$ es diferente de cero. Cada vez que se encuentre un minitérmino, el Qbit tendrá una compuerta CNOT generalizada, controlada desde a través de la combinación de los x_i bits que producen el cambio. Si el cambio se produce mediante un uno, el circulo irá relleno, por lo que se trata de una compuerta $CNOT_A$ normal. Si el cambio se produce por un cero, se trata de una compuerta $CNOT_B$.

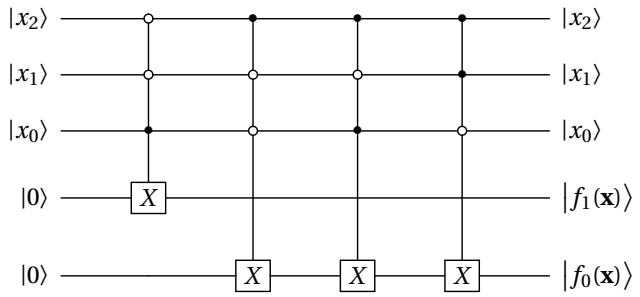


Figura 4.25: Circuito asociado con $f(\mathbf{x})$, definida en la tabla 4.17

Si se trata de una función genérica, carente de estructura, la cantidad de minitérminos aumentará de forma exponencial con el tamaño de la entrada, lo cual indica que la evaluación de f arbitraria no puede realizarse de manera eficiente. Esta situación puede resolverse en el caso clásico mediante el empleo de identidades lógicas, lo que permite encontrar un circuito lógico simplificado. Por ejemplo, si la función es $F_1(x,y) = xy' + x'y + xy$ debido a la propiedad de idempotencia $(x + x' = 0)$, es posible duplicar el tercer término para obtener $F_1(x,y) = x \cdot y' + x \cdot y + x' \cdot y + x \cdot y = x \cdot (y' + y) + y \cdot (x' + x) = x \cdot 1 + y \cdot 1 = x + y$, por lo que se reduce a una operación OR. Sin embargo, las reglas para minimizar los circuitos lógicos cuánticos son diferentes a las que se usan para minimizar los circuitos lógicos clásicos [166]. Así, para algunos casos, es posible encontrar maneras de reducir el número de operaciones controladas, por ejemplo, mediante la siguiente relación:

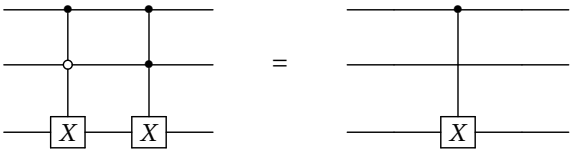


Figura 4.26: Ejemplo de una regla de simplificación para un circuito cuántico.

Esto permite simplificar el circuito propuesto en la figura 4.25 de la siguiente forma:

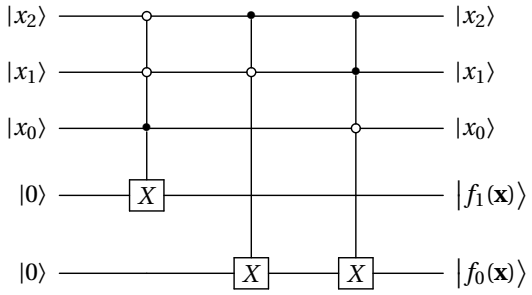


Figura 4.27: Circuito asociado con $f(\mathbf{x})$, simplificado.

Aunque en este ejemplo las funciones f_1 y f_0 son mas bien arbitrarias, el orden puede relacionarse para el cálculo de funciones mas complejas. Por ejemplo, para el cálculo de x^2 para una entrada de 2 Qbits. En general, si se necesitan $n = log_2 N$ Qbits para almacenar un entero $x \in [1, N]$, se requieren $2n = log_2 N^2$ Qbits

para almacenar $x^2 \in [1, N^2]$. Por tanto, el caso $n = 2$ requiere $2n = 4$ Qbits para almacenar la salida. Esto corresponde a 4 funciones binarias, que pueden evaluarse de forma reversible mediante 4 Qbits ancilla. Luego, la función $f(\mathbf{x})$ puede expresarse con minitérminos como $f(\mathbf{x}) = (f_3(\mathbf{x}), f_2(\mathbf{x}), f_1(\mathbf{x}), f_0(\mathbf{x}))$, donde

$$f_0(\mathbf{x}) = m_{0001}(\mathbf{x}) \vee m_{1001}(\mathbf{x}) \qquad f_1(\mathbf{x}) = 0 \qquad f_2(\mathbf{x}) = m_{0100}(\mathbf{x}) \qquad f_3(\mathbf{x}) = m_{1000}(\mathbf{x}) \tag{4.85}$$

En la tabla 4.18 se muestran las variables de entrada, su expresión binaria separada en dos bits x_1 y x_0, así como el valor resultante de aplicar la función x^2, su expresión en binario y la separación en las funciones $f_i(\mathbf{x})$.

Tabla 4.18: Tabla de verdad del circuito cuántico para la función $f(x) = x^2$, para una entrada de 2 bits

x	x_1	x_0	x^2	x^2	$f_3(\mathbf{x})$	$f_2(\mathbf{x})$	$f_1(\mathbf{x})$	$f_0(\mathbf{x})$
0	0	0	0	0000	0	0	0	0
1	0	1	1	0001	0	0	0	1
2	1	0	4	0100	0	1	0	0
3	1	1	9	1001	1	0	0	1

La figura 4.28 muestra el circuito asociado con $f(\mathbf{x})$. La figura muestra el circuito cuántico correspondiente. En el diagrama puede verse la descomposición en mintérminos. Las dos líneas superiores representan la entrada x, las cuatro líneas inferiores los Qbits auxiliares. La salida x^2 se lee de abajo hacia arriba, los Qbits van del menos significativo al más significativo.

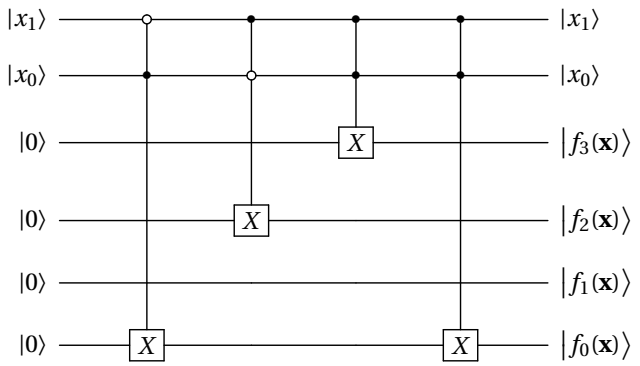


Figura 4.28: Circuito cuántico para la función $f(x) = x^2$, para una entrada de 2 bits.

Si se considera que todo el circuito describe a la transformación U, el estado final y el inicial se relacionan mediante:

$$U|x_1\rangle \otimes |x_0\rangle \otimes |0\rangle \otimes |0\rangle \otimes |0\rangle\,|0\rangle = |x_1\rangle \otimes |x_0\rangle \otimes \big|f_3(x)\big\rangle \otimes \big|f_2(x)\big\rangle \otimes \big|f_1(x)\big\rangle \otimes \big|f_0(x)\big\rangle \tag{4.86}$$

con las funciones f_i definidas como en tabla 4.18. Se puede prescindir del símbolo de producto tensorial para simplificar la notación:

$$U|x_1\rangle\,|x_0\rangle\,|0\rangle\,|0\rangle\,|0\rangle\,|0\rangle = |x_1\rangle\,|x_0\rangle\,\big|f_3(x)\big\rangle\,\big|f_2(x)\big\rangle\,\big|f_1(x)\big\rangle\,\big|f_0(x)\big\rangle \tag{4.87}$$

Si no hay confusión, es posible escribir distintos Qbits dentro del mismo ket.

$$U|x_1x_0\rangle\,|0000\rangle = |x_1x_0\rangle\,\big|f_3(x)f_2(x)f_1(x)f_0(x)\big\rangle \tag{4.88}$$

O incluso, todos los Qbits dentro de un mismo ket, aunque, en caso de que exista ambigüedad, o para mejorar la exposición, pueden separarse los valores con una coma, por ejemplo $U|x_1x_0, 0000\rangle = \big|x_1x_0, f_3(x)f_2(x)f_1(x)f_0(x)\big\rangle$. Sin embargo, aquí es conveniente dejar explícita la separación entre los Qbits que se emplean para representar la entrada x_1x_0 de aquellos que son auxiliares, los que se han inicializado en cero. De la misma forma, en la salida se ha separado los Qbits que duplican la entrada (necesarios para garantizar la reversibilidad) de

aquellos em los que se ha evaluado la función. Nuevamente, haciendo referencia a la tabla 4.18, se espera obtener los siguientes valores:

$$U|00\rangle|0000\rangle = |00\rangle|0000\rangle \qquad U|01\rangle|0000\rangle = |01\rangle|0001\rangle$$
$$U|10\rangle|0000\rangle = |10\rangle|0100\rangle \qquad U|11\rangle|0000\rangle = |11\rangle|1001\rangle$$

Para simplificar aún más la lectura, es posible usar los valores decimales que corresponden a cada conjunto de Qbits,, $00_b = 0$, $01_b = 1$, $10_b = 2$ y $11_b = 3$, de la misma forma, $0000_b = 0$, $0001_b = 1$,$0100_b = 4$ y $1001_b = 9$, para obtener:

$$U|0\rangle|0\rangle = |0\rangle|0\rangle \qquad U|1\rangle|0\rangle = |1\rangle|1\rangle$$
$$U|2\rangle|0\rangle = |2\rangle|4\rangle \qquad U|3\rangle|0\rangle = |3\rangle|9\rangle$$

A continuación, se muestra el funcionamiento de este circuito mediante *IBM Quantum Composer* (IBMQC en lo sucesivo), una herramienta de interfaz gráfica de usuario (GUI) que permite crear la creación circuitos cuánticos, a través de una interfaz que permite arrastrar y soltar compuertas, así como simular su funcionamiento, para visualizar el circuito y su salida. La figura 4.29 fue realizada con IBMQC y muestra el mismo circuito que aparece en la figura 4.28 . Las diferencias son esencialmente estéticas[4], como el hecho de que la compuerta X se represente con un círculo y un símbolo de suma en su interior, además de carecer de una forma de representar a la compuerta $CNOT_B$ de manera directa, por lo que aquí se muestra el circuito completo para la operación.

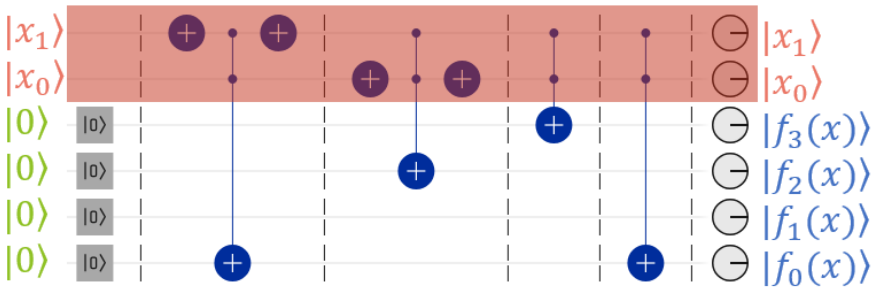


Figura 4.29: Circuito para x^2 realizado en IBMQC, para dos Qbits de entrada y 4 Qbits de salida. Los Qbits que se utilizan como entrada se han destacado en rojo, mientras que los Qbits de los que se lee la salida están en azul.

Dentro de la interfaz de IBMQC, cada Qbit finaliza está acompañado con un circulo al final, denominado *disco de fase*. Su función es proporcionar el estado local de cada Qbit al final del cálculo. La probabilidad de que el Qbit se encuentre en el estado $|1\rangle$ está representada por la altura del relleno del disco en color azul (en contraste con el gris, que representaría la probabilidad de estar en $|0\rangle$), mientras que la fase del estado está indicada por la línea que se extiende desde el centro del diagrama hasta el borde del disco (que gira en sentido antihorario alrededor del punto central).

Por ello, si al final de un cálculo, la probabilidad de que el Qbit se encuentre en el estado $|0\rangle$ es del 100%, el circulo aparecerá en gris, mientras que si la probabilidad de que el Qbit se encuentre en el estado $|1\rangle$ es del 100%, el circulo aparecerá en azul. Para el caso actual, en el que se examina caso por caso, son los únicos estados posibles, por lo que el disco de fase funciona aquí como lo haría un foco de prueba. Si el disco es de color azul, se interpreta como encendido, mientras que si es gris estará apagado. Esto puede verse en la figura 4.30, donde se muestra el resultado de aplicar U a diferentes números, es decir, a diferentes combinaciones de valores para x_0 y x_1. Para ingresar diferentes valores en los Qbits superiores se realiza los siguiente: todo Qbit en este diagrama está inicializado en cero. Si se requiere que el Qbit tenga ese valor, no es necesario hacer nada. Sin embargo, si se requiere ingresar un uno, será necesario aplicar una compuerta X para invertirlo. Después de inicializar, se realizan las transformaciones mediante las compuertas cuánticas ya mencionadas. La salida puede leerse a partir de disco de fase, que funciona aquí como lo haría un foco de prueba, azul

[4]Salvo por la dirección en que se ordenan los Qbits. Muchos libros de texto siguen la convención en la que los Qbits en un sistema de varios Qbit se ordenan desde el menos significativo al más significativo como $|q_0 q_1 \dots q_{n-1}\rangle$. IBMQC utiliza la convención opuesta, es decir $|q_{n-1} \dots q_1 q_0\rangle$ por defecto. Las matrices que corresponden a compuertas de varios Qbit se ordenan utilizando la misma convención. El orden puede cambiarse (como aquí se ha hecho) mediante el método *reverse_bits()*

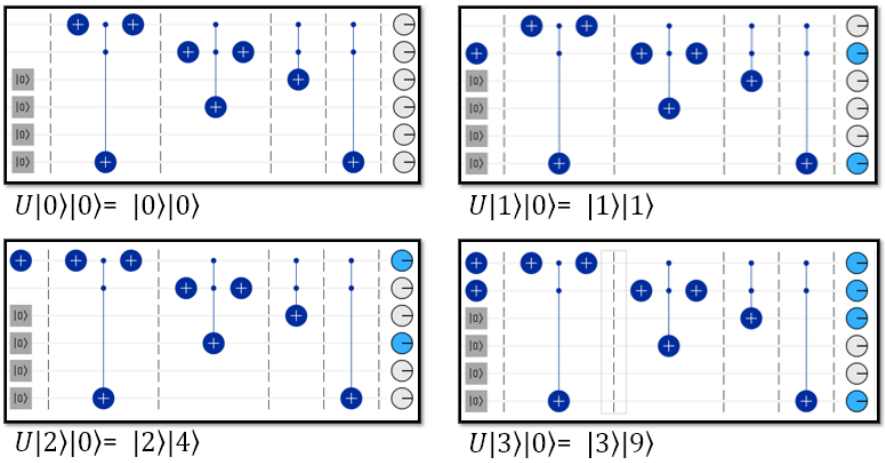


Figura 4.30: Resultado de simular la función U con IMBQC. Se aplica sobre los valores de entrada 0, 1, 2 y 3.

es encendido, mientras que gris es apagado. Debajo de cada circuito se encuentra el valor en decimal que se ingresa y el devuelto. El estado final y el inicial están relacionados mediante el producto de las matrices que componen a la transformación U, sin embargo, el tamaño de las matrices involucradas resulta poco apropiado para exhibirlas aquí. Sin embargo, al aplicar la identidad que aparece en 5.2, el circuito puede simplificarse:

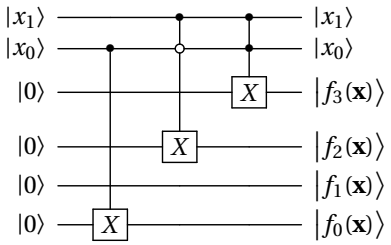


Figura 4.31: Implementación del circuito cuántico para la función $f(x) = x^2$, para una entrada de 2 bits, simplificada.

En este ejemplo se ha mostrado como una computadora cuántica puede realizar una operación de manera secuencial, es decir, se ha mostrado como introducir un valor tras otro y como leer la salida. Sin embargo este comportamiento es propio de la computado clásica. La característica principal de una computadora cuántica es el poder realizar operaciones de manera *simultánea*.

4.5.1. Paralelismo cuántico

El interés en cómputo cuántico no se limita a la evaluación de funciones entrada por entrada. Si se comienza con dos Qbits en el estado inicial $|\varphi_0\rangle = |00\rangle$ y se aplica la transformación unitaria de dos Qbis U_f tal que $U_f|xy\rangle = |x, y \oplus f(x)\rangle$, por ejemplo si se trata de la compuerta CNOT, en donde $f(x) = 0$ si $x = 0$ y $f(x) = 1$ si $x = 1$. Al aplicar la compuerta Hadamard sobre el primer Qbit se obtiene el estado:

$$|\varphi_1\rangle = H|\varphi_0\rangle = H|00\rangle = \frac{1}{\sqrt{2}}(|0\rangle|0\rangle + |1\rangle|0\rangle) \tag{4.89}$$

La compuerta Hadamard transforma cualquier estado $|x\rangle$ en una superposición de todos los valores posibles de x (es decir, 0 y 1). Después de la transformación unitaria U_f en el estado de dos Qbits $|\varphi\rangle$, el estado toma la forma: $|\varphi_2\rangle = U_f|\varphi_1\rangle = \frac{1}{\sqrt{2}}(|0, f(0)\rangle + |1, f(1)\rangle)$, estado que contiene la función evaluada $f(x)$ para las dos posibles entradas 0 y 1, que se han obtenido de un solo paso. El circuito de la figura 4.32 muestra una implementación de esto.

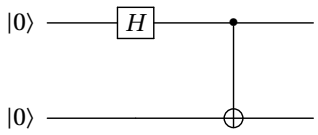


Figura 4.32: Un circuito que produce entrelazamiento.

La superposición es lo que hace posible el paralelismo cuántico. Para ver esto, se puede multiplicar n Qbits que se encuentran en n espacios de Hilbert bidimensionales distintos $H_0, H_1, ..., H_{n-1}$, que son isomorfos entre sí.

$$\left|\psi_0\right\rangle \otimes \left|\psi_1\right\rangle \otimes ... \otimes \left|\psi_{n-1}\right\rangle \in H_0 \otimes H_1 \otimes ... \otimes H_{n-1} \tag{4.90}$$

En la práctica, esto significa que los Qbits se prepararon por separado, sin ninguna interacción. Puede seleccionarse cualquier base arbitraria como base computacional para $H_i, i = 0, 1, ..., n-1$. Para facilitar la exposición, se toma $|0_i\rangle$ y $|1_i\rangle$. En consecuencia, se puede representar el Qbit i-ésimo como

$$\left|\psi_i\right\rangle = \alpha_i \left|0_i\right\rangle + \beta_i \left|1_i\right\rangle \tag{4.91}$$

Con la suposición adicional $\alpha_i = \beta_i = \frac{1}{\sqrt{2}}$ se tiene:

$$\begin{aligned}\left|\psi_0\right\rangle \otimes \left|\psi_1\right\rangle \otimes ... \otimes \left|\psi_{n-1}\right\rangle &= \frac{1}{\sqrt{2}}\left(|0_0\rangle + |1_0\rangle\right) \otimes \frac{1}{\sqrt{2}}\left(|0_1\rangle + |1_1\rangle\right) \otimes ... \otimes \frac{1}{\sqrt{2}}\left(|0_{n-1}\rangle + |1_{n-1}\rangle\right) \\ &= \frac{1}{\sqrt{2^n}}\big(|0_0\rangle \otimes |0_1\rangle \otimes ... \otimes |0_{n-1}\rangle + |0_0\rangle \otimes |0_1\rangle \otimes ... \otimes |1_{n-1}\rangle + |0_0\rangle \otimes |0_1\rangle \otimes ... \otimes |0_{n-2}\rangle \otimes |0_{n-1}\rangle ... \\ &\qquad + |1_0\rangle \otimes |1_1\rangle \otimes ... \otimes |1_{n-1}\rangle\big)\end{aligned} \tag{4.92}$$

Este sistema cuántico compuesto se llama *registro* de n Qbit y es una superposición de estados cuánticos de 2^n que existen simultáneamente. Este es un ejemplo de paralelismo cuántico. En el ámbito clásico, un aumento lineal en el tamaño corresponde aproximadamente a un aumento lineal en la potencia de procesamiento, pero en sistemas cuánticos, debido al paralelismo, un aumento lineal en el tamaño corresponde a un aumento exponencial en el poder de procesamiento. Para funciones de n variables, debido a la linealidad, se tiene:

$$U_f \sum_{x=1}^{2^n-1} c_x \left|x\right\rangle \left|y\right\rangle = \sum_{x=1}^{2^n-1} c_x \left|x\right\rangle \left|y \oplus f(x)\right\rangle \tag{4.93}$$

Lo que significa que es capaz de evaluar $f(x)$ para todas las x en una sola ejecución.

Si se considera nuevamente el ejemplo del circuito que permite evaluar $f(x) = x^2$, se deberá recordar que trabaja mediante compuertas CNOT múltiples. Sin embargo, el entrelazamiento requiere que las entradas se encuentren en estados de superposición, para lo cual es necesario agregar compuertas Hadamard. El circuito modificado se muestra a continuación:

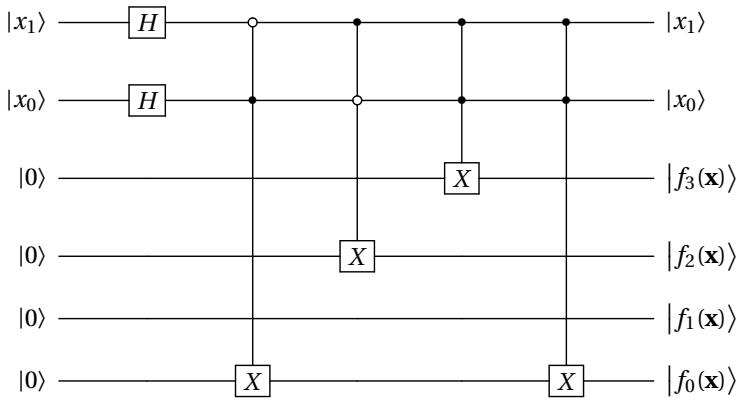


Figura 4.33: Un circuito que evalúa x^2 para entradas de dos Qbits con entrelazamiento.

En el circuito de la figura 4.33, la entrada $\frac{1}{2}\Big(|0\rangle + |1\rangle + |2\rangle + |3\rangle\Big)|0\rangle$, produce la salida $\frac{1}{2}\Big(|0\rangle|0\rangle + |1\rangle|1\rangle + |2\rangle|4\rangle + |3\rangle|9\rangle\Big)$, de manera que se ha calculado en paralelo $f(x) = x^2$ para $x = 0, 1, 2$ y 3, una posibilidad que está fuera del alcance de la computadora clásica, que sólo puede recibir como entrada un valor dado de x, no una superposición de valores de x. Esta propiedad distintiva de la computadora cuántica se conoce como *paralelismo cuántico*.

Es tentador concluir que una computadora cuántica puede evaluar una función para un número arbitrario de entradas en un solo paso. Sin embargo, no es tarea fácil extraer información útil de la superposición. El problema es que esta información está, en cierto sentido, oculta porque, para extraer información útil del estado final es necesario medirlo. Dentro de IBMQC, esta situación se representa mediante los discos de fase. Como se mencionó, la altura de la sección de azul representa la probabilidad de que el estado del Qbit sea uno. Como puede verse en el diagrama siguiente, las probabilidades ya no son ni cero ni uno, toman valores intermedios, excepto en el Qbit que representa a la salida f_1, que para todos los casos considerados en el ejemple siempre es cero.

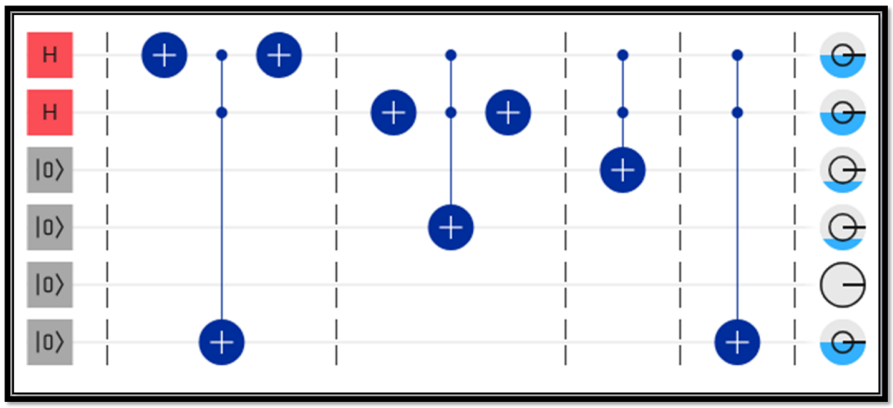


Figura 4.34: Circuito que evalúa x^2 para entrada de dos Qbits, con entrelazamiento, realizado en IBMQC. A diferencia de los diagramas anteriores, en los que se ingresaban los valores uno por uno, en esta ocasión se han ingresado los 4 valores de manera simultanea. El disco de fase indica la probabilidad de obtener, tras una medición, el estado $|1\rangle$. El Qbit correspondiente a $f_1(x)$ tiene probabilidad cero, ninguna de las compuertas CNOT está conectada con él

Una medición proyectiva del primer registro en la base computacional (es decir, una medición de la polarización del Qbit a lo largo del eje Z para todos los Qbits en este registro) produce un valor particular x'. Pero esto implica que una medición del segundo registro necesariamente dará el resultado $f(x')$. Por ejemplo, si se mide el primer registro de la función de onda, se obtiene con igual probabilidad $p_0 = p_1 = p_2 = p_3 = \frac{1}{4}$ cualquiera de los cuatro resultados posibles 0, 1, 2 y 3. Si una medida arroja $x' = 2$, entonces el estado colapsará a $|2\rangle|4\rangle$, por lo que una medición del segundo registro ahora da el resultado $f(x') = 4$ con probabilidad uno. La medición final borra por completo cualquier aparente aceleración derivada del paralelismo cuántico, por lo que, en realidad, se termina con una evaluación de la función $f(x)$ para un único valor de x. Afortunadamente, este no es el final de la historia. Existen algoritmos cuánticos que explotan la interferencia cuántica para extraer información distinta de un único valor de f eficientemente, a partir del estado de superposición.

4.5.2. Teorema de no clonación

Realizar copias múltiples de algún valor, proceso conocido como *clonación*, para poder procesar cada una de ellas de manera independiente, sin interacción, es una práctica común en cómputo clásico, no solo a nivel de aplicación del usuario final, que puede requerir la copia de un archivo, si no a nivel de programación, en dónde existen protocolos de corrección de errores que requieren realizar y comparar copias de bits en todo momento. Más aún, a nivel de componente, debe recordarse que la prueba de que toda función $f : \{0,1\}^n \rightarrow \{0,1\}^m$ pueda ser construida a partir de las compuertas elementales AND, OR, NOT, es imposible si se ignora la compuerta FANOUT, cuya función es la de tomar un bit y producir una salida de dos bits dada por $COPY : a \rightarrow (a,a)$. Sin embargo, las leyes de la mecánica cuántica tienen algo bastante diferente que decir sobre la realización de tales copias en el dominio cuántico. Por un lado, lo que parece (pero no es) una copia se convierte en la base de la comunicación cuántica. Por otro lado, la copia explícita dentro de un algoritmo

cuántico tiene enormes limitaciones.

Como punto de partida, es necesario observar que la clonación puede intentarse de diferentes formas. La más sencilla consiste en observar que si se ingresa un $|0\rangle$ puro como la entrada $\left|y\right\rangle$ de una compuerta CNOT, la salida produce una copia de la entrada $|x\rangle$:

$$CNOT(|x\rangle \otimes |0\rangle) = |x\rangle \otimes |x \oplus 0\rangle = |x\rangle \otimes |x\rangle \tag{4.94}$$

Sin embargo, esto solo funciona con estados $|0\rangle$ o $|1\rangle$ puros. Para un caso arbitrario, en el que se desea clonar a un estado en la superposición $\alpha|0\rangle + \beta|1\rangle$ con ninguno de los coeficientes cero, se produce la salida:

$$CNOT\left|\alpha,0,\beta,0\right\rangle = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \alpha \\ 0 \\ \beta \\ 0 \end{bmatrix} = \begin{bmatrix} \alpha \\ 0 \\ 0 \\ \beta \end{bmatrix} \tag{4.95}$$

que no es una copia del estado arbitrario ingresado. Al copiar sólo un estado puro, se descarta la la ventaja cuántica de procesar muchos valores posibles a la vez. Puede pensarse que una ligera modificación de CNOT podría conseguir la copia. Considerando dos Qbits u y v donde y la operación que desea sobreescribir al estado de v con el de u sin tener en cuenta el estado anterior de v. Si el estado del par se representa como $|uv\rangle$, entonces cualquier matriz que realice la copia debe transformar $|01\rangle$ (el estado donde u es 0 y v es diferente, es decir, un 1, en $|00\rangle$. De la misma forma, $|10\rangle$ debe transformarse en $|11\rangle$. Esto puede conseguirse utilizando la matriz U_c, sin embargo, no se trata de una matriz unitaria [151]:

$$U_c = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix} \qquad U_c^\dagger = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad U_c U_c^\dagger = \begin{bmatrix} 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2 \end{bmatrix} \tag{4.96}$$

Por otra parte, la idea de entrelazamiento puede sugerir que la creación de estados de Bell produce clones, forzando a dos Qbits a encontrarse en estados puros conocidos y luego aplicales una compuerta Hadamard para llevarlos a estados mixtos conocidos. Cualquier procesamiento en uno de estos Qbits que transforme su estado mixto se reflejará en cambios en el otro. Este procesamiento puede luego crear un estado mixto deseado para usarlo como entrada para algún algoritmo, y ambos Qbits tendrán el mismo estado mixto. Sin embargo, existe una gran diferencia respecto a la situación clásica, en la que dos cables salen una compuerta: es imposible desactivar el entrelazamiento y ejecutar diferentes pasos de procesamiento independientes en cada Qbit. Aunque se han clonado los estados, resulta que las copias son inseparables.

En resumen, al tratar de importar el concepto de clonación clásica al dominio cuántico, se puede tiene alguna de las siguientes situaciones:

- Se sabe que el Qbit que se está clonando es un estado puro y, por lo tanto, el valor clonado también debe ser un estado puro que puede procesarse posteriormente independientemente del original.

- Los dos Qbits reciben el mismo estado mixto y cualquier operación adicional realizada en uno de ellos cambia inmediatamente el estado del otro de la misma manera.

Si, en este último caso, se pudiera encontrar la forma de trabajar con los dos Qbits se de manera independientemente uno del otro tras la clonación, se tendría un mecanismo análogo a la idea clásica de copia perfecta. Sin embargo, esto resulta imposible. De manera similar al caso del paralelismo cuántico, la superposición es también el concepto clave detrás de la incapacidad para clonar estados cuánticos arbitrarios. Ya que le mecánica cuántica es una teoría lineal, un estado cuántico es un estado de superposición. Para un Qbit que se encuentre en un estado de la forma $\alpha|0\rangle + \beta|1\rangle$, cualquier medición que se realice sobre él provocará el colapso del estado cuántico a alguno de sus estados propios. Por lo tanto, un Qbit arbitrario no puede duplicarse sin destruir el original [299].

La imposibilidad de copiar un estado cuántico se denomina *teorema de no clonación* y tiene consecuencias importantes en el procesamiento de información cuántico, como el hecho de que una cadena de Qbits solo puede procesarse en una dirección, la seguridad de que no hay forma de que alguien realice una copia perfecta de cierta información o la posibilidad de detectar la presencia de intrusos en un canal de comunicación cuántica.

La prueba de este teorema parte de la suponer la existencia de un operador que realice copias, tomando como entrada a un determinado estado cuántico arbitrario y produciendo como salida dos estados idénticos a la entrada, que serán copias del original. Por ejemplo, si el estado de entrada es $|\psi\rangle$, entonces la salida de la copiadora es $|\psi\rangle|\psi\rangle$. Para un estado cuántico arbitrario, dicha función plantea una contradicción fundamental. Dado que la mecánica cuántica es lineal, un dispositivo cuántico que permita replicar estados debe tener asociado un operador unitario cuya función sea la de replicar estados, lo cual puede escribirse como:

$$U|\psi\rangle|\Sigma\rangle|M\rangle = |\psi\rangle|\psi\rangle|M(\psi)\rangle \tag{4.97}$$

En donde $|\Sigma\rangle$ representa un estado en blanco, aquel en el que se puede copiar un estado desconocido y $|M\rangle$ representa un estado auxiliar para la evaluación de la función. Para los estados de la base computacional, esta función produciría las salidas:

$$U\Big(|0\rangle|\Sigma\rangle|M\rangle\Big) = |0\rangle|0\rangle|M(0)\rangle \qquad U\Big(|1\rangle|\Sigma\rangle|M\rangle\Big) = |1\rangle|1\rangle|M(1)\rangle \tag{4.98}$$

Ahora, si se considera una superposición de estos dos estados dada por $|\Psi\rangle = \alpha|0\rangle + \beta|1\rangle$, dado que es un operador lineal se tiene que:

$$U|\Psi\rangle|\Sigma\rangle|M\rangle = U\Big(\alpha|0\rangle+\beta|1\rangle\Big)|\Sigma\rangle|M\rangle = \alpha U|0\rangle|\Sigma\rangle|M\rangle+\beta U|1\rangle|\Sigma\rangle|M\rangle = \alpha|0\rangle|0\rangle|M(0)\rangle+\beta|1\rangle|1\rangle|M(1)\rangle \tag{4.99}$$

Por otra parte, la superposición de estados puros es un estado puro y la clonación de $|\Psi\rangle$ puede hacerse directamente, $U|\Psi\rangle|\Sigma\rangle|M\rangle = |\Psi\rangle|\Psi\rangle|M(\Psi)\rangle$. Al desarrollar el producto se obtiene:

$$\Big(\big(\alpha|0\rangle+\beta|1\rangle\big)\big(\alpha|0\rangle+\beta|1\rangle\big)\Big)|M(\Psi)\rangle = \Big(\alpha^2|0\rangle|0\rangle+\alpha\beta|0\rangle|1\rangle+\beta\alpha|1\rangle|0\rangle+\beta^2|1\rangle|1\rangle\Big)|M(\Psi)\rangle \tag{4.100}$$

La diferencia entre las dos formas de evaluar conduce a una contradicción y, en consecuencia, muestra que un operador unitario que permita clonar $|\Psi\rangle$ para un estado arbitrario no puede existir. Esto excluye algunos casos específicos, como estados idénticos u ortogonales [81], algo que podría explicar una cuestión interesante. Si la física clásica es, después de todo, un caso particular de la cuántica, el teorema de no clonación debería aplicar en la computación clásica también, impidiendo la realización de copias perfectas. ¿Cómo es posible que se pueda copiar información clásica si no es posible copiar estados cuánticos? Para responder, es necesario entender que el teorema de no clonación no afirma que sea imposible realizar copias en todo caso, si no que esta imposibilidad aplica para estados cuánticos no ortogonales. Para decirlo con mayor precisión, si $|\psi\rangle$ y $|\phi\rangle$ son dos estados cuánticos no ortogonales, no puede existir una «copiadora cuántica» que pueda realizar copias de un estado arbitrario, es decir, un dispositivo cuántico que, al recibir la entrada $|\psi\rangle$ o $|\phi\rangle$, devuelva la entrada duplicada $|\psi\rangle|\psi\rangle$ o $|\phi\rangle|\phi\rangle$. Por otro lado, si $|\psi\rangle$ y $|\phi\rangle$ son ortogonales, entonces el teorema de no clonación no aplica. Los diferentes estados de la información clásica pueden considerarse simplemente como estados cuánticos ortogonales, de manera que puede copiarse sin violar el teorema [208].

El teorema de no clonación indica la imposibilidad de construir una máquina que pueda *clonar perfectamente* un estado cuántico desconocido mediante la evolución unitaria. Una pregunta interesante es que pasaría en caso de que se permita el empleo de dispositivos no unitarios. O que sucede en caso de que se permita copias *imperfectas*, pero que sean suficientemente buenas, de acuerdo con alguna medida de fidelidad o para un propósito específico. Estas, y otras preguntas han sido objeto de mucha investigación. El breve resumen de ello es que, incluso si se permiten dispositivos de clonación no unitarios, la clonación de estados puros no ortogonales sigue siendo imposible a menos que se esté dispuesto a tolerar una pérdida finita de fidelidad en los estados copiados[208].

5

Algoritmos Cuánticos Elementales

Los algoritmos son una serie de instrucciones que sirven para resolver un problema, mediante un conjunto básico de elementos, considerados como los bloques de construcción. En el caso de computación clásica, los bloques son circuitos booleanos, a partir de los cuales es posible componer operaciones aritméticas típicas, como funciones booleanas. Por otra parte, la computación cuántica emplea un conjunto muy distinto de bloques. Estos resultan ser, por lo general, operaciones unitarias de uno o dos Qbits, con las que se pueden construir circuitos cuánticos. Considerando que los bloques elementales de construcción para ambos tipos de computación son del todo distintos, puede entenderse que los algoritmos de computación cuántica no son solo algoritmos de computación clásica con ligeras modificaciones, sino que requieren nuevas formas de analizar los problemas. Deben explotar las propiedades de los estados cuánticos, las puertas cuánticas y los dispositivos de medición para encontrar la solución.

5.1. Nuevos algoritmos para nuevas computadoras

Tanto en el caso clásico como en el cuántico, sucedió que los algoritmos fueron desarrollados antes de que existieran las máquinas en los que pueden ejecutarse. En el caso de los algoritmos clásicos, muchos de ellos se conocieron milenios antes de la concepción de las computadoras clásicas y, de manera similar, los primeros algoritmos cuánticos se discutieron y propusieron hace décadas, principalmente con el objetivo de mostrar la manera en la que se pueden emplear las características de los estados cuánticos (desde el principio de superposición hasta los efectos de entrelazamiento e interferencia) para implementar algoritmos que resuelven problemas más rápido o de manera más eficiente que si se hiciera con una computadora convencional. La mayoría de estos algoritmos han sido diseñados para resolver problemas elaborados a modo, es decir, problemas que se han seleccionado porque son compatibles con las características propias de la computación cuántica y muestran alguna ventaja sobre los algoritmos clásicos. Por este motivo se conocen realmente pocos y la mayoría tratan temas matemáticos bastante alejados de aplicaciones inmediatas.

Todos los algoritmos de computación cuántica incluidos en esta sección comparten una estructura relativamente simple:

- Preparación de un conjunto de Qbits en algún estado inicial, generalmente uno de los estados de base computacional.
- Los estados de esos Qbits son sometidos a varias compuertas cuánticas, entre ellas, las necesarias para crear superposición de estados y las que realizan transformaciones unitarias
- Los Qbits resultantes interactúan con los dispositivos de medición, nuevamente empleando los estados básicos computacionales.
- Finalmente, interpreta los resultados de esas mediciones para responder a la pregunta que el algoritmo está diseñado para abordar.

5.2. Oráculo cuántico

En mitología, el *oráculo* era la respuesta que una deidad daba a una pregunta. El ser omnisciente atendía las consultas de manera instantánea e infalible, a través de un intermediario y en un lugar sagrado específico. La noción de ente omnisapiente que responde consultas ha sido importada al entorno computacional a través de los llamados *oráculos computacionales*, para referirse a subrutinas o circuitos que implementan alguna operación o función de la que se desconocen los detalles concernientes a su implementación. Incluso, se admite trabajar con funciones cuya implementación podría no existir en absoluto. En última instancia, un oráculo es solo una compuerta cuántica.

Las funciones oráculo actúan como una *caja negra*, en el sentido de que, dada una entrada, es posible obtener su imagen, es decir, evaluarla, aunque la forma en que la función está definida es desconocida, además, no hay manera de examinar el interior de la caja para obtener mas información. Cuando se realiza una pregunta al oráculo, es decir, al realizar una *consulta*, este devuelve la respuesta correcta al instante. Los oráculos computacionales son útiles para cuantificar la complejidad de algoritmo, especialmente cuando algunas partes del algoritmo no se comprenden del todo, ya que hacen posible la comparación entre las complejidades relativas de dos algoritmos sin la necesidad de considerar o entender la manera en que se el oráculo funciona. La diferencia entre los oráculos clásicos y los cuánticos radica en la naturaleza de las preguntas y respuestas que se pueden plantear y obtener.

Muchos de los algoritmos cuánticos conocidos se expresan como algoritmos que resuelven problemas de oráculo. Lo único que se conoce con certeza sobre f es el dominio y contradominio de la función, $f : \{0,1\}^n \to \{0,1\}^m$. El objetivo es determinar alguna propiedad de f. Cuando se proporciona alguna información adicional sobre la función, es frecuente decir que dicha propiedad de la función ha sido *prometida*. Para determinar la propiedad deseada, se tiene acceso a una transformación (el oráculo) que actúa sobre una entrada a elegir, la salida de la transformación es la función en cuestión aplicada a la entrada. Dependiendo de la forma en la que la función se codifique en el algoritmo, se tienen los oráculos XOR y de fase.

5.2.1. Oráculo cuántico XOR

De manera formal, se define a un oráculo O_f como una operación no revelada que se emplea como entrada dentro de otro algoritmo. Con frecuencia, tales operaciones se definen usando una función clásica $f : \{0,1\}^n \to \{0,1\}^m$ que toma una entrada de n bits y produce una salida de m bits. Sin embargo, debido a que los cálculos cuánticos válidos deben corresponder a transformaciones unitarias, que son invertibles, la caja negra en realidad realiza la función $f : \{0,1\}^n \times \{0,1\}^m \to \{0,1\}^n \times \{0,1\}^m : f(x,y) = (x, y \oplus f(x))$. Si la entrada binaria $x = (x_0, x_1, ..., x_{n-1})$ puede representarse mediante el Qbit dado por $|x\rangle = |x_0\rangle \otimes |x_1\rangle \otimes ... \otimes |x_{n-1}\rangle$, se define al oráculo como un operador unitario, considerando su acción sobre la base computacional, de acuerdo con:

$$O_f(|x\rangle \otimes \big|y\big\rangle) = |x\rangle \otimes \big|y \oplus f(x)\big\rangle. \tag{5.1}$$

para todo $x \in \{0,1\}^n$ y $y \in \{0,1\}^m$. Esta definición en términos de la base computacional $|x\rangle \otimes \big|y\big\rangle = |x\rangle \big|y\big\rangle$ permite definir cómo actúa O_f para cualquier otro estado $\big|\psi\big\rangle$ de $n+m$ Qbits, puesto que se puede escribir como combinación lineal de la base:

$$\big|\psi\big\rangle = \sum_{x,y} \alpha(x,y) |x\rangle \big|y\big\rangle \tag{5.2}$$

donde $\alpha : \{0,1\}^n \times \{0,1\}^m \to \mathbb{C}$ representa los coeficientes del estado $\big|\psi\big\rangle$. Así,

$$O_f \big|\psi\big\rangle = O_f \Big[\sum_{x,y} \alpha(x,y) |x\rangle \big|y\big\rangle \Big] = \sum_{x,y} \alpha(x,y) O_f \big[|x\rangle \big|y\big\rangle \big] = \sum_{x,y} \alpha(x,y) |x\rangle \big|y \oplus f(x)\big\rangle. \tag{5.3}$$

5.2.2. Oráculo cuántico de fase

El valor propio agregado por una compuerta o transformación a un Qbit se puede trasladar a un Qbit diferente a través de una operación controlada. Por ejemplo, el la base de Hadamard, el papel de Qbit de control y objetivo es efectivamente

$$CNOT\Big[|1\rangle |-\rangle \Big] = CNOT\Big[|1\rangle \Big(\frac{|0\rangle - |1\rangle}{\sqrt{2}} \Big) \Big] = |1\rangle\, NOT\Big[\Big(\frac{|0\rangle - |1\rangle}{\sqrt{2}} \Big) \Big] = |1\rangle \Big[(-1) \Big(\frac{|0\rangle - |1\rangle}{\sqrt{2}} \Big) \Big] \tag{5.4}$$

mientras que, en el caso en el que el Qbit de control se encuentre en cero, aplicar la compuerta equivale a aplicar la identidad:

$$CNOT\Big[|0\rangle |-\rangle \Big] = |0\rangle \Big[\Big(\frac{|0\rangle - |1\rangle}{\sqrt{2}} \Big) \Big] \tag{5.5}$$

dado que el Qbit objetivo es un estado propio, no cambia, por lo que el valor propio ha sido desplazado hacia el Qbit de control. Esta situación puede expresarse de manera abreviada como:

$$CNOT\Big[\,|b\rangle\,|-\rangle\,\Big] = (-1)^b\,|b\rangle\,|-\rangle \tag{5.6}$$

Por lo que para una superposición de estados de $|0\rangle$ y $|1\rangle$ se tiene

$$CNOT\Big[\,(|0\rangle+|1\rangle)\,|-\rangle\,\Big] = ((-1)^0\,|0\rangle+(-1)^1\,|1\rangle)\,|-\rangle = (|0\rangle-|1\rangle)\,|-\rangle \tag{5.7}$$

Para considerar el efecto de una compuerta mas general de 2 Qbits que implementa una función arbitraria f : $\{0,1\}\rightarrow\{0,1\}$ a través del mapeo $O_f\big[\,|x\rangle\,\big|y\big\rangle\,\big] = |x\rangle\,\big|y\oplus f(x)\big\rangle$, se mantiene fijo el Qbit objetivo en $|-\rangle$ mientras se observa la acción de la compuerta en el Qbit de control, que se encuentra en un estado base arbitrario $|x\rangle$

$$O_f\Big[\,|x\rangle\,|-\rangle\,\Big] = O_f\Big[\,|x\rangle\Big(\frac{|0\rangle-|1\rangle}{\sqrt{2}}\Big)\Big] = \frac{1}{\sqrt{2}}\,|x\rangle\Big(\big|0\oplus f(x)\big\rangle-\big|1\oplus f(x)\big\rangle\Big) \tag{5.8}$$

donde puede observarse que si $f(x)=0$ no habrá cambios, mientras que si $f(x)=1$ el signo del estado se invertirá:

$$O_f\Big[\,|x\rangle\,|-\rangle\,\Big] = \begin{cases} \frac{1}{\sqrt{2}}(|0\rangle-|1\rangle) = +\frac{|0\rangle-|1\rangle}{\sqrt{2}} = +\,|-\rangle & \text{si } f(x)=0 \\ \frac{1}{\sqrt{2}}(|1\rangle-|0\rangle) = -\frac{|0\rangle-|1\rangle}{\sqrt{2}} = -\,|-\rangle & \text{si } f(x)=1 \end{cases} \tag{5.9}$$

En ambos casos se obtiene $|-\rangle$, aunque con diferencia en la fase, que dependerá del valor de $f(x)$. Un circuito que permite implementar esta acción se muestra en la figura 5.1

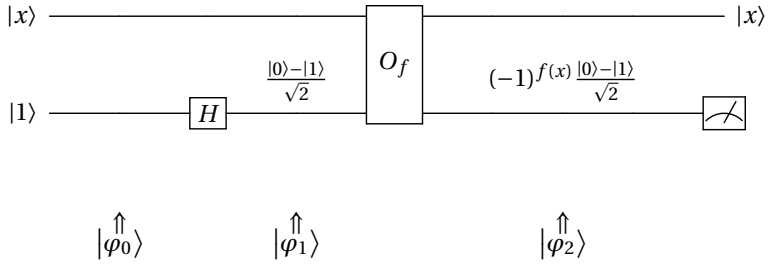


Figura 5.1: Circuito que produce un retroceso de fase. El estado $|-\rangle$ en el Qbit objetivo es un estado propio de O_f . El valor propio $(-1)^{f(x)}$ (dentro del círculo inferior) puede desplazarse al frente de la expresión que representa al estado φ_2, como si perteneciera a Qbit superior

El Qbit superior es arbitrario. El Qbit inferior se inicia en $|1\rangle$:

$$|\varphi_0\rangle = |x,1\rangle. \tag{5.10}$$

la compuerta Hadamard permite obtener la superposición $|-\rangle = \frac{|0\rangle-|1\rangle}{\sqrt{2}}$:

$$|\varphi_1\rangle = I\otimes H\,|x\rangle\,|1\rangle = |x\rangle\,|-\rangle = |x\rangle\Big(\frac{|0\rangle-|1\rangle}{\sqrt{2}}\Big) \tag{5.11}$$

Al aplicar la compuerta O_f se obtiene :

$$|\varphi_2\rangle = O_f\Big[\,|x\rangle\,|-\rangle\,\Big] = \begin{cases} +\,|x\rangle\,|-\rangle & \text{si } f(x)=0 \\ -\,|x\rangle\,|-\rangle & \text{si } f(x)=1 \end{cases} \tag{5.12}$$

que usualmente se expresa en la forma compacta:

$$|\varphi_2\rangle = (-1)^{f(x)}\,|x\rangle\,|-\rangle \tag{5.13}$$

De manera que es posible codificar f en un oráculo O_f aplicando una *fase* basada en la entrada a O_f, definiendo:

$$O_f\,|x\rangle = (-1)^{f(x)}\,|x\rangle \tag{5.14}$$

Si un oráculo de fase actúa sobre un registro inicialmente en un estado de base computacional $|x\rangle$, entonces esta fase es una fase global y por lo tanto carente de significado físico. Tal oráculo puede ser un recurso poderoso si se aplica a una superposición o como una operación controlada. Por ejemplo, un oráculo de fase O_f para una función de un solo Qbit f

$$O_f|+\rangle = O_f\left[\frac{|0\rangle+|1\rangle}{\sqrt{2}}\right] = \frac{(-1)^{f(0)}|0\rangle+(-1)^{f(1)}|1\rangle}{\sqrt{2}} = (-1)^{f(0)}\left[\frac{|0\rangle+(-1)^{f(1)-f(0)}|1\rangle}{\sqrt{2}}\right] \tag{5.15}$$

El cambio de fase está relacionado con la compuerta Z.Para mostrarlo, puede observarse que $Z^{-1} = Z^\dagger = Z$, por lo que se cumple la siguiente identidad:

$$Z^{f(0)-f(1)} = \begin{bmatrix}1 & 0\\ 0 & -1\end{bmatrix}^{|f(0)-f(1)|} \tag{5.16}$$

pero, $|f(0)-f(1)|$ solo puede tomar valores 0 o 1, por lo que será la matriz identidad o Z misma.

$$Z^{f(0)-f(1)} = \begin{cases} \begin{bmatrix}1 & 0\\ 0 & -1\end{bmatrix}^1 = Z & \text{si } f(1)\neq f(0)\\ \begin{bmatrix}1 & 0\\ 0 & -1\end{bmatrix}^0 = I & \text{si } f(1)=f(0)\end{cases} \tag{5.17}$$

Si este operador se aplica sobre sobre $|+\rangle = \frac{1}{\sqrt{2}}(|0\rangle+|1\rangle)$:

$$Z|+\rangle = \begin{bmatrix}1 & 0\\ 0 & -1\end{bmatrix}\frac{1}{\sqrt{2}}(|0\rangle+|1\rangle) = \frac{1}{\sqrt{2}}\left(\begin{bmatrix}1 & 0\\ 0 & -1\end{bmatrix}\begin{pmatrix}1\\1\end{pmatrix}\right) = \frac{1}{\sqrt{2}}\begin{pmatrix}1\\-1\end{pmatrix} = \frac{1}{\sqrt{2}}(|0\rangle-|1\rangle) \tag{5.18}$$

En resumen:

$$Z|+\rangle = \frac{1}{\sqrt{2}}(|0\rangle-|1\rangle) \qquad I|+\rangle = \frac{1}{\sqrt{2}}(|0\rangle+|1\rangle) \tag{5.19}$$

por lo que es posible escribir al oráculo O_f como:

$$O_f|+\rangle = (-1)^{f(0)}Z^{f(0)-f(1)}|+\rangle\,. \tag{5.20}$$

Tanto el oráculo cuántico XOR como el de desplazamiento de fase pueden extenderse para representar funciones clásicas, que devuelven números reales en lugar de un solo bit. Además, cada oráculo puede ser simulado por el otro, es decir, es posible construir un oráculo de fase por medio de uno XOR y recíprocamente, por lo que ambos son equivalentes[77]. La elección de la mejor manera de implementar un oráculo depende en gran medida de cómo se va a utilizar este oráculo dentro de un algoritmo determinado. Por ejemplo, el algoritmo de Deutsch-Jozsa se basa en el oráculo implementado de la primera manera, mientras que el algoritmo de Grover se basa en el oráculo implementado de la segunda manera.

5.3. Algoritmo de Deutsch (función constante o balanceada)

5.3.1. Descripción del problema

El algoritmo de Deutsch es el mas simple en la lista de algoritmos cuánticos. Fue diseñado para resolver un problema que, a pesar de ser bastante artificial, ilustra como es que la interferencia cuántica y el paralelismo cuántico pueden utilizarse en la realización de cálculos, de manera que se obtiene una ventaja respecto de los algoritmos clásicos.

El problema resuelto por el algoritmo Deutsch es el siguiente. Se tiene un circuito reversible que puede evaluar a una función desconocida de 1 bit $f:\{0,1\}\rightarrow\{0,1\}$. Esta puede ser cualquiera de las contenidas en la tabla 5.1.

Tabla 5.1: Funciones de un bit

x	f_0	f_1	f_2	f_3
0	0	0	1	1
1	0	1	0	1

Estas funciones pueden clasificarse como funciones constantes o funciones balanceadas. Como lo indica su nombre, las funciones constantes tienen el mismo valor, sin importar el argumento, es decir, f será una función constante si $f(0) = f(1)$. Por otra parte, la función será balanceada siempre que $f(0) \neq f(1)$, lo que en este caso resulta equivalente a decir que es uno a uno. A partir de la tabla 5.1 es posible notar que dos de ellas son constantes (f_0 y f_3) y dos son balanceadas (f_1 y f_2).

La función f está dada como caja negra, por lo que se puede aplicar el circuito para obtener valores de $f(x)$ para las entradas x dadas, pero no es posible obtener ninguna información sobre el funcionamiento interno del circuito para aprender sobre f. El algoritmo de Deutsch determinana si una función $f : \{0,1\} \rightarrow \{0,1\}$, es balanceada o constante, *mediante una sola evaluación de la función*, es decir, mediante *una sola consulta al oráculo*.

Para entender su funcionamiento, es necesario observar lo que sucede al realizar la operación $f(0) \oplus f(1)$. En caso de que $f(0) \oplus f(1) = 0$, se cumple que $f(0) = f(1)$, por lo que f es *constante*, aunque no se conozca el valor por separado de $f(0)$ y $f(1)$. Si, por el contrario, se determina que $f(0) \oplus f(1) = 1$, entonces $f(0) \neq f(1)$, y la función será *equilibrada*. Entonces, conocer el resultado de $f(0) \oplus f(1)$ equivale a determinar si la función f es constante o equilibrada [145].

Problema de Deutsch
Entrada: Una caja negra que pueda evaluar a la función $f : \{0,1\} \rightarrow \{0,1\}$, que no se conoce.
Tarea: Calcular el valor de $f(0) \oplus f(1)$ realizando consultas a f.

Con una computadora clásica, sería necesario evaluar primero f en una entrada, luego evaluar f en la segunda entrada para posteriormente poder comparar las salidas, por lo que la solución clásica a este problema requiere de la realización de dos consultas al oráculo. El siguiente árbol de decisión muestra lo que debe hacer una computadora clásica:

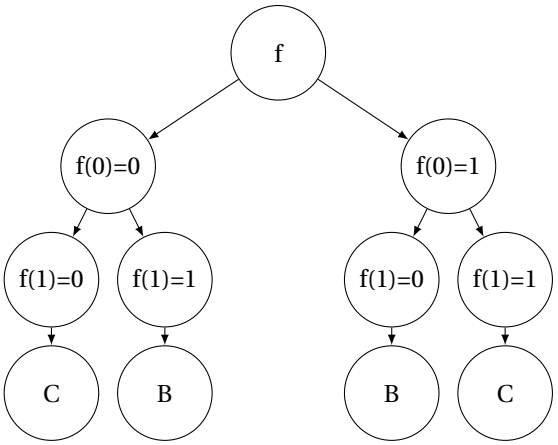


Figura 5.2: Árbol de decisión para identificar si una función de dos bits es balanceada (B) o constante (C). El algoritmo clásico requiere realizar dos evaluaciones para determinarlo.

Este problema es importante porque fue el primer algoritmo cuántico no trivial. También muestra que los algoritmos cuánticos pueden ser más eficientes que los clásicos, incluso si la ventaja parece pequeña. La solución clásica requiere dos evaluaciones de la función, mientras que la solución cuántica requiere solo una. Puede parecer una diferencia poco impresionante, pero su existencia sugiere la posibilidad de obtener mayores ventajas.

También es un excelente ejemplo del concepto de paralelismo cuántico. Si la medición del Qbit principal, que en este punto va a tener un resultado determinista, produce $|0\rangle$, resulta que se trata de una función constante, y cuando produce $|1\rangle$, se tiene uno equilibrado. Sin embargo, vale la pena enfatizar que para resolver el problema de Deutsch una computadora cuántica en realidad no realiza diferentes evaluaciones de funciones en paralelo, sino que se basa en el fenómeno de interferencia para cancelar las respuestas incorrectas se anulen entre si [298].

5.3.2. Circuito

El algoritmo de Deutsch emplea la superposición de estados en los dos Qbits. El circuito se muestra en la figura5.3.

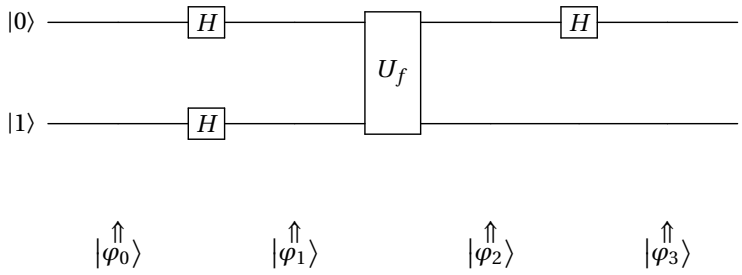


Figura 5.3: Circuito para el algoritmo de Deutsch

El cálculo comienza en:

$$|\varphi_0\rangle = |0\rangle \otimes |1\rangle. \tag{5.21}$$

Para obtener superposición en ambos Qbits se aplican las compuertas Hadamard, por lo que

$$|\varphi_1\rangle = H|0\rangle \otimes H|1\rangle = \frac{|0\rangle + |1\rangle}{\sqrt{2}} \otimes \frac{|0\rangle - |1\rangle}{\sqrt{2}} = \frac{|0\rangle}{\sqrt{2}} \otimes \frac{|0\rangle - |1\rangle}{\sqrt{2}} + \frac{|1\rangle}{\sqrt{2}} \otimes \frac{|0\rangle - |1\rangle}{\sqrt{2}} = \frac{|0\rangle}{\sqrt{2}} \otimes |-\rangle + \frac{|1\rangle}{\sqrt{2}} \otimes |-\rangle \tag{5.22}$$

Haciendo uso de la expresión:

$$U_f \left[|x\rangle |-\rangle \right] = \left\{ \begin{array}{ll} +|x\rangle |-\rangle & \text{si } f(x) = 0 \\ -|x\rangle |-\rangle & \text{si } f(x) = 1 \end{array} \right\} = (-1)^{f(x)} |x\rangle |-\rangle \tag{5.23}$$

se puede obtener

$$|\varphi_2\rangle = U_f|\varphi_1\rangle = \frac{(-1)^{f(0)}|0\rangle \otimes |-\rangle + (-1)^{f(1)}|1\rangle \otimes |-\rangle}{\sqrt{2}} = \frac{(-1)^{f(0)}|0\rangle + (-1)^{f(1)}|1\rangle}{\sqrt{2}} \otimes |-\rangle \tag{5.24}$$

$$|\varphi_2\rangle = \begin{cases} (\pm 1)\frac{|0\rangle+|1\rangle}{\sqrt{2}} \otimes |-\rangle, & \text{si } f \text{ es constante,} \\ (\pm 1)\frac{|0\rangle-|1\rangle}{\sqrt{2}} \otimes |-\rangle, & \text{si} f \text{ es balanceada.} \end{cases} \tag{5.25}$$

Ya que la compuerta Hadamard es su propia inversa:

$$|\varphi_3\rangle = H|\varphi_2\rangle = \begin{cases} (\pm 1)|0\rangle \otimes |-\rangle, & \text{si } f \text{ es constante,} \\ (\pm 1)|1\rangle \otimes |-\rangle, & \text{si } f \text{ es constante.} \end{cases} \tag{5.26}$$

A partir de ahora, solo se necesita realizar una medida sobre el Qbit superior. Si se encuentra en el estado $|0\rangle$, significa que f es una función constante, de lo contrario se trata de una función equilibrada. Únicamente hará falta una consulta al oráculo, en lugar de las dos evaluaciones que exige el algoritmo clásico. Por otra parte, el ±1 ofrece aún más información de la necesaria para resolver el problema propuesto, ya que permite identificar cuál de las dos funciones balanceadas o dos funciones constantes es f. Sin embargo, la medición no otorgará esta información. Al medir, si la función está balanceada, se obtendrá $|1\rangle$ independientemente de si el estado era (—1)$|1\rangle$ o (+1)$|1\rangle$.

Puede parecer que se ha adquirido más información de la proporcionada por la única consulta realizada. Pero en realidad no es lo que sucede. Existen cuatro funciones posibles y el árbol de decisiones mostró que una computadora clásica requiere dos bits de información para determinar de cuál de las cuatro funciones se trata, pero el problema no está preguntando eso, únicamente se refiere a una propiedad de dichas funciones. De manera que puede decirse que lo que este algoritmo hace es cambiar el orden en el que se adquiere la información. Si el objetivo fuera determinar de cuál de las cuatro funciones se trata, se puede hacer mediante dos preguntas: «¿La función es equilibrada o constante?» y «¿cuál es el valor de la función en 0?» Las respuestas a estas dos preguntas describen de manera unívoca a cada una de las cuatro funciones, algo que puede verse en el árbol de decisiones siguiente:

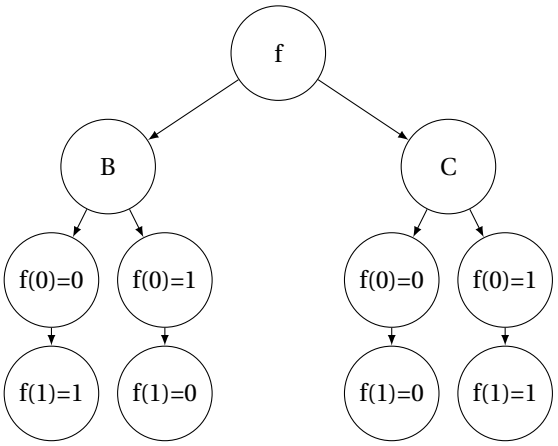


Figura 5.4: Árbol de decisión para identificar a la función f, partiendo de la pregunta ¿es una función balanceada (B) o constante (C)?

La acción de las compuertas Hadamard consiste en cambiar la base. En este contexto, equivale a cambiar la pregunta realizada. Aunque se trata de un ejemplo muy simple y sin aplicación práctica inmediata, es una evidencia de que es posible lograr algoritmos cuánticos más eficientes que los clásicos. Una extensión de este algoritmo a n variables mostrara que es posible lograr una aceleración exponencial.

5.4. Algoritmo de Deutsch-Jousa (función de n bits constante o balanceada)

5.4.1. Descripción del problema

Este algoritmo es, en realidad, una generalización del anterior. En lugar de trabajar sobre funciones booleanas $f : \{0,1\} \to \{0,1\}$, lo hace funciones con un dominio mas grande, a saber, las funciones $f : \{0,1\}^n \to \{0,1\}$, las que aceptan como entrada una cadena de n bits y generan como salida un valor, cero o uno. El dominio puede considerarse entonces como cualquier número que pueda codificarse con estos bits, es decir, cualquier número natural entre 0 y $2^n - 1$. De manera análoga al problema de Deutsch, se llamará a una función $f : \{0,1\}^n \to \{0,1\}$ *balanceada* si exactamente a la mitad de las entradas se les asigna el valor cero (y en consecuencias a la otra mitad de las entradas se les asigna el uno) mientras que la función será *constante* si todas las entradas van a cero o todas las entradas van a uno. El objetivo es identificar si la función es constante o balanceada realizando *solo una evaluación* de la función, teniendo la certeza de que la función pertenecerá a uno de estos dos conjuntos. Claramente, este objetivo es imposible en el modelo clásico de computación, ya que debe evaluarse la función en la mitad mas uno de los posibles valores de entrada, es decir, se requieren $2^{n-1} + 1$ evaluaciones de f. Sin embargo, es posible hacerlo con una sola evaluación en el modelo cuántico.

Problema de Deutsch-Jozsa
Entrada: Una caja negra que pueda evaluar a la función desconocida $f : \{0,1\}^n \to \{0,1\}$.
Promesa: f es una función constante o equilibrada.
Tarea: Determinar si f es constante o equilibrado realizando consultas a f.

De manera similar al caso de una variable, se requiere la superposición de los estados de entrada. Esta vez, se ingresara la superposición de todos los 2^n posibles valores de entrada. Esto se consigue mediante las compuertas Hadamard , aunque hará falta extender la definición para dimensiones superiores. Se pueden definir matrices Hadamard multidimensionales combinando matrices de Hadamard de un solo Qbit. El patrón para definir una matriz de Hadamard n-dimensional $H^{\otimes n}$, es a través de su producto de Kronecker consigo misma. Por ejemplo, dado

$$H = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \tag{5.27}$$

se puede obtener:

$$H^{\otimes 2} = \frac{1}{2} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix} \tag{5.28}$$

El desarrollo de potencias ⊗ puede parecer engorroso, sin embargo, es posible notar un patrón en los coeficientes, que reducirá el problema a conocer si el exponente de (-1) correspondiente a la entrada de interés es par o impar. Luego, $H^{\otimes n}$ está dada por:

$$H^{\otimes n}[i,j] = \frac{1}{\sqrt{2^n}}(-1)^{\langle \mathbf{i},\mathbf{j}\rangle} \tag{5.29}$$

donde i j son los números de renglón y columna en binario (iniciando desde 0), y la operación

$$\langle \mathbf{i},\mathbf{j}\rangle = \langle i_0 i_1 i_2 ... i_{n-1}, j_0 j_1 j_2, ..., j_{n-1}\rangle = (i_0 \wedge j_0) \oplus (i_1 \wedge j_1) \oplus ... \oplus (i_{n-1} \wedge j_{n-1}) \tag{5.30}$$

con ∧ el operador AND. Explícitamente, $H^{\otimes n}$, aplicado al ket 0 en n bits es sencillo de evaluar:

$$H^{\otimes n}|\mathbf{0}\rangle = \frac{1}{\sqrt{2^n}} \begin{bmatrix} (-1)^{\langle 0,0\rangle} & (-1)^{\langle 0,1\rangle} & (-1)^{\langle 0,2\rangle} & \dots & (-1)^{\langle 0,j\rangle} & \dots & (-1)^{\langle 0,2^n-1\rangle} \\ (-1)^{\langle 1,0\rangle} & (-1)^{\langle 1,1\rangle} & (-1)^{\langle 1,2\rangle} & \dots & (-1)^{\langle 1,j\rangle} & \dots & (-1)^{\langle 1,2^n-1\rangle} \\ (-1)^{\langle 2,0\rangle} & (-1)^{\langle 2,1\rangle} & (-1)^{\langle 2,2\rangle} & \dots & (-1)^{\langle 2,j\rangle} & \dots & (-1)^{\langle 2,2^n-1\rangle} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ (-1)^{\langle i,0\rangle} & (-1)^{\langle i,1\rangle} & (-1)^{\langle i,2\rangle} & \dots & (-1)^{\langle i,j\rangle} & \dots & (-1)^{\langle i,2^n-1\rangle} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ (-1)^{\langle 2^n-1,0\rangle} & (-1)^{\langle 2^n-1,1\rangle} & (-1)^{\langle 2^n-1,2\rangle} & \dots & (-1)^{\langle 2^n-1,j\rangle} & \dots & (-1)^{\langle 2^n-1,2^n-1\rangle} \end{bmatrix} \begin{pmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \\ \vdots \\ 0 \end{pmatrix} \tag{5.31}$$

Es claro que todos los elementos de la primer columna son +1, por lo que

$$H^{\otimes n}|\mathbf{0}\rangle = \frac{1}{\sqrt{2^n}} \begin{pmatrix} 1 \\ 1 \\ 1 \\ \vdots \\ 1 \\ \vdots \\ 1 \end{pmatrix} = \frac{1}{\sqrt{2^n}} \left[\begin{pmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \\ \vdots \\ 0 \end{pmatrix} + \begin{pmatrix} 0 \\ 1 \\ 0 \\ \vdots \\ 0 \\ \vdots \\ 0 \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ 1 \\ \vdots \\ 0 \\ \vdots \\ 0 \end{pmatrix} + ... + \begin{pmatrix} 0 \\ 0 \\ 0 \\ \vdots \\ 1 \\ \vdots \\ 0 \end{pmatrix} + ... + \begin{pmatrix} 0 \\ 0 \\ 0 \\ \vdots \\ 0 \\ \vdots \\ 1 \end{pmatrix} \right] = \frac{1}{\sqrt{2^n}} \sum_{y \in \{0,1\}^n} |\mathbf{y}\rangle \tag{5.32}$$

Para un estado arbitrario $|\mathbf{x}\rangle$, que puede ser representado por un vector columna con un solo 1 en la posición x y 0 en todas las demás, se puede *extraer* el renglón x-ésimo de $H^{\otimes n}$ (y dado que se trata de una matriz simétrica, es igual a la x-esima columna de $H^{\otimes n}$).

$$H^{\otimes n}|\mathbf{x}\rangle = \frac{1}{\sqrt{2^n}} \begin{bmatrix} (-1)^{\langle 0,0\rangle} & (-1)^{\langle 0,1\rangle} & (-1)^{\langle 0,2\rangle} & \dots & (-1)^{\langle 0,j\rangle} & \dots & (-1)^{\langle 0,2^n-1\rangle} \\ (-1)^{\langle 1,0\rangle} & (-1)^{\langle 1,1\rangle} & (-1)^{\langle 1,2\rangle} & \dots & (-1)^{\langle 1,j\rangle} & \dots & (-1)^{\langle 1,2^n-1\rangle} \\ (-1)^{\langle 2,0\rangle} & (-1)^{\langle 2,1\rangle} & (-1)^{\langle 2,2\rangle} & \dots & (-1)^{\langle 2,j\rangle} & \dots & (-1)^{\langle 2,2^n-1\rangle} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ (-1)^{\langle i,0\rangle} & (-1)^{\langle i,1\rangle} & (-1)^{\langle i,2\rangle} & \dots & (-1)^{\langle i,j\rangle} & \dots & (-1)^{\langle i,2^n-1\rangle} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ (-1)^{\langle 2^n-1,0\rangle} & (-1)^{\langle 2^n-1,1\rangle} & (-1)^{\langle 2^n-1,2\rangle} & \dots & (-1)^{\langle 2^n-1,j\rangle} & \dots & (-1)^{\langle 2^n-1,2^n-1\rangle} \end{bmatrix} \begin{pmatrix} 0 \\ 0 \\ 0 \\ \vdots \\ 1 \\ \vdots \\ 0 \end{pmatrix} \tag{5.33}$$

que suele abreviarse como

$$H^{\otimes n}|\mathbf{x}\rangle = \frac{1}{\sqrt{2^n}} \sum_{z \in \{0,1\}^n} (-1)^{\langle \mathbf{z},\mathbf{x}\rangle} |\mathbf{z}\rangle \tag{5.34}$$

5.4.2. Circuito

El circuito resulta similar al caso de entrada de un solo bit, solo que ahora debe considerarse un circuito donde la entrada superior es una cadena binaria n-dimensional.

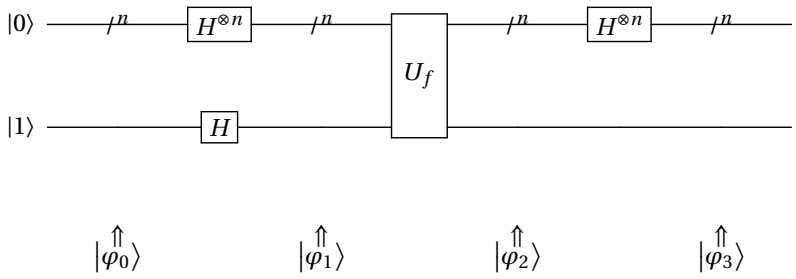


Figura 5.5: Circuito para el algoritmo Deutsch-Jozsa, generalización del algoritmo Deutsch para el caso binario multivariable. Para representar una cadena de n Qbits se usa un cable cuántico con una diagonal. El resultado final es que si se mide el Qbit superior en el circuito anterior, solo se obtendrá $|\mathbf{0}\rangle$ si la función es constante

El cálculo comienza en el estado$|\varphi_0\rangle = |\mathbf{0}\rangle \otimes |1\rangle$. Al aplicar el primer conjunto de compuertas Hadamard se obtiene

$$|\varphi_1\rangle = H^{\otimes n+1}|\mathbf{0}\rangle \otimes |1\rangle = H^{\otimes n}|\mathbf{0}\rangle \otimes H|1\rangle = \frac{1}{\sqrt{2^n}} \sum_{x\in\{0,1\}^n} |\mathbf{x}\rangle\,|-\rangle \tag{5.35}$$

Al aplicar la matriz unitaria U_f, se tiene

$$|\varphi_2\rangle = \frac{1}{\sqrt{2^n}} \sum_{x\in\{0,1\}^n} (-1)^{f(\mathbf{x})}\,|\mathbf{x}\rangle\,|-\rangle \tag{5.36}$$

En la última etapa del circuito, se aplica de nuevo una compuerta Hadamard $H^{\otimes n}$ a los Qbits superiores, que se encuentran en una superposición de diferentes estados $\mathbf{x}$

$$\begin{aligned}
|\varphi_3\rangle = H^{\otimes n} \otimes I \frac{1}{\sqrt{2^n}} \sum_{x\in\{0,1\}^n} (-1)^{f(\mathbf{x})}\,|\mathbf{x}\rangle\,|-\rangle &= \frac{1}{\sqrt{2^n}} \sum_{x\in\{0,1\}^n} (-1)^{f(\mathbf{x})} \left[H^{\otimes n}\,|\mathbf{x}\rangle\right] \otimes I\,|-\rangle \\
&= \frac{1}{\sqrt{2^n}} \sum_{x\in\{0,1\}^n} (-1)^{f(\mathbf{x})} \left[\frac{1}{\sqrt{2^n}} \sum_{z\in\{0,1\}^n} (-1)^{\langle \mathbf{z},\mathbf{x}\rangle}\,|\mathbf{z}\rangle\right] |-\rangle \\
&= \frac{1}{2^n} \sum_{x\in\{0,1\}^n} \sum_{z\in\{0,1\}^n} (-1)^{f(\mathbf{x})}(-1)^{\langle \mathbf{z},\mathbf{x}\rangle}\,|\mathbf{z}\rangle\,|-\rangle
\end{aligned} \tag{5.37}$$

la suma de los exponentes ha de realizarse módulo 2:

$$|\varphi_3\rangle = \frac{1}{2^n} \sum_{x\in\{0,1\}^n} \sum_{z\in\{0,1\}^n} (-1)^{f(\mathbf{x})\oplus\langle \mathbf{z},\mathbf{x}\rangle}\,|\mathbf{z}\rangle\,|-\rangle \tag{5.38}$$

Finalmente, se miden los Qbits del estado $\left|\varphi_3\right\rangle$. Si se toma **z=0**

$$|\varphi_3\rangle = \frac{1}{2^n} \sum_{x\in\{0,1\}^n} (-1)^{f(\mathbf{x})\oplus\langle \mathbf{0},\mathbf{x}\rangle}\,|\mathbf{0}\rangle\,|-\rangle = \frac{1}{2^n} \sum_{x\in\{0,1\}^n} (-1)^{f(\mathbf{x})}\,|\mathbf{0}\rangle\,|-\rangle \tag{5.39}$$

puede verse que la probabilidad de colapsar $|\mathbf{0}\rangle$ totalmente depende de $f(\mathbf{x})$, por lo que si $f(\mathbf{x})$ es constante en 1, los Qbits superiores se encontrarán en

$$\frac{1}{2^n} \sum_{x\in\{0,1\}^n} (-1)^1\,|\mathbf{0}\rangle = -1\,|\mathbf{0}\rangle \tag{5.40}$$

mientras que si es constante en 0, se tendrá

$$\frac{1}{2^n} \sum_{x\in\{0,1\}^n} (-1)^0\,|\mathbf{0}\rangle = +1\,|\mathbf{0}\rangle \tag{5.41}$$

En caso de que f sea balanceada, la mitad de las $\mathbf{x}$ deberán cancelar a la otra mitad :

$$\frac{1}{2^n} \sum_{x\in\{0,1\}^n} (-1)^{f(\mathbf{x})}\,|\mathbf{0}\rangle = 0\,|\mathbf{0}\rangle \tag{5.42}$$

por lo que, al medir los Qbits superiores en $|\varphi_3\rangle$, solo será posible obtener $|\mathbf{0}\rangle$ si la función es constante. En caso de encontrar cualquier otra cosa, se trata de una función balanceada. Por lo tanto, se ha resuelto un problema en una sola evaluación, en comparación con las $2^{n-1}+1$ evaluaciones requeridas por por una computadora clásica, lo cual es una mejora exponencial.

Si el algoritmo de Deutsch se considera importante por ser el primer algoritmo cuántico, aunque apenas puede decirse que superó la solución clásica, el algoritmo de Deutsch-Jozsa muestra que la mejora puede ser exponencialmente grande. Se observa un gran avance sobre la mera sustitución de dos operaciones por una sola. Sin embargo, debe entenderse que la mejora exponencial ocurre en comparación con un algoritmo clásico determinista que podría requerir la realización de un número exponencial de evaluaciones de f antes de decidir si está equilibrado, en el peor de los casos. Por otra parte, un algoritmo aleatorio podría hacer esta distinción en un número constante de evaluaciones, siempre que sea posible admitir una pequeña probabilidad de cometer un error. Por lo tanto, este algoritmo es un paso importante, pero aún no llega a demostrar que los algoritmos cuánticos pueden ser exponencialmente más rápidos que los algoritmos clásicos, si se permite que estos últimos sean aleatorios

5.5. Algoritmo de Simon (Búsqueda de periodos)

5.5.1. Descripción del problema

Los dos algoritmos anteriores son en realidad atípicos porque se obtiene la respuesta final *con certeza* después de *una sola* consulta. En realidad, la mayoría de los algoritmos cuánticos utilizan una combinación de partes cuánticas y partes clásicas; requieren el uso del circuito cuántico en más de una ocasión; e implican probabilidad. El algoritmo de Simon es un algoritmo cuántico que usa lo anterior para resolver un problema informático clásico con una aceleración cuántica exponencial. Daniel Simon, el creador del algoritmo de Simon, dice que observó los primeros algoritmos de computación cuántica y no quedó impresionado. Era un escéptico de la computación cuántica y no creía que estos algoritmos (o, tal vez, cualquier algoritmo) resultaran en una aceleración cuántica respecto de la computación clásica. Simon se propuso demostrar esto matemáticamente, pero en lugar de eso, creó un algoritmo cuántico que es una herramienta importante y un componente básico en la carrera por demostrar una ventaja cuántica de base amplia.

Este algoritmo se usa para encontrar patrones en funciones booleanas, a través de la búsqueda del periodo. Dada una función booleana $f : \{0,1\}^n \to \{0,1\}^n$, que es posible evaluar pero que se proporciona como caja negra. Se asegura que existe una cadena binaria c cuyo valor no se revela, tal que para todas las cadenas $y, z \in \{0,1\}^n$

$$f(z) = f(y) \iff z = y \oplus c \tag{5.43}$$

donde $\oplus$ es la operación OR Exclusiva bit a bit. En otras palabras, los valores de f se repiten con algún patrón y el patrón está determinado por c, el período de f. El objetivo es encontrar c a través de la evaluación de f sin conocer los detalles de su implementación

El objetivo del algoritmo de Simon es determinar este periodo. Por ejemplo, si $n = 3$ y $c = 011$, puede observarse que :

$$000 \oplus 011 = 011 \to f(000) = f(011) \qquad 100 \oplus 011 = 111 \to f(100) = f(111)$$
$$001 \oplus 011 = 010 \to f(001) = f(010) \qquad 101 \oplus 011 = 110 \to f(101) = f(110)$$
$$010 \oplus 011 = 001 \to f(010) = f(001) \qquad 110 \oplus 011 = 101 \to f(110) = f(101)$$
$$011 \oplus 011 = 000 \to f(011) = f(000) \qquad 111 \oplus 011 = 100 \to f(111) = f(100)$$

En el caso de que c sea el vector con todas las entradas iguales a cero, f será una función uno a uno, mientras que para todos los demás c la promesa obliga a f a ser dos a uno. En este último caso, se dice que f es periódica con periodo c.

Problema de Simon
Entrada: Una caja negra que pueda evaluar a la función desconocida $f : \{0,1\}^n \to X$, donde X es un conjunto finito.
Promesa: Existe una cadena $c = c_1 c_2 ... c_n$ tal que $f(x) = f(y)$ si y sólo si $x = y$ o $x = y \oplus c$.
Tarea: Determinar la cadena c realizando consultas a f.

Todo lo que se requiere para realizar esta tarea es encontrar dos elementos distintos en el dominio, a y b, que se correspondan con el mismo valor, es decir, $f(a) = f(b)$. Una vez que se encuentre dicho par, será posible calcular fácilmente c usando que $a = b \oplus s \implies s = a \oplus b$, por lo que resolver este problema equivale a encontrar cualquier par de elementos con la misma imagen. Debería haber muchos pares de este tipo en el dominio gracias a la condición previa de 2 a 1. Específicamente, hay 2^n elementos en el dominio, por lo que si se calcula f para la mitad de ellos y se encuentra que todas sus imágenes son distintas, que es el peor de los casos, entonces calcular un elemento más seguramente conducirá a una *colisión* [44] (un valor igual a uno de los valores previamente calculados), lo que explica el crecimiento explosivo del algoritmo clásico. El número de veces que es necesario consultar al oráculo crece exponencialmente con n, el tamaño de la entrada. Por lo tanto, este problema es intratable, no existe ningún algoritmo determinista (clásico) eficiente para resolverlo.

El teorema de Simon establece que c puede hallarse usando un algoritmo cuántico en tiempo polinomial [254]. Este algoritmo distingue el caso en el que f es uno a uno de los casos particulares donde f es dos a uno. Los algoritmos clásicos factibles, aún los aleatorios, son incapaces de realizar esto, por lo que el algoritmo de Simon es importante para mostrar el poder del cómputo cuántico.

El algoritmo de Simon se puede separar en dos partes principales: la parte cuántica y la parte clásica. La parte cuántica del algoritmo utiliza operaciones cuánticas para obtener una superposición de todos los valores posibles del período, lo que permite alimentar al oráculo con una superposición de todas esas 2^n posibilidades y obtener un rendimiento que captura todos los valores correspondientes de f en una sola consulta del oráculo. Por lo tanto, si de alguna manera se puede manipular el retorno para obtener una colisión, será posible calcular c en una complejidad de consulta de $\mathcal{O}(1)$ en lugar de la clásica $\mathcal{O}(2^n)$. La complejidad real es $\mathcal{O}(n)$, pero sigue siendo exponencialmente mejor que el algoritmo clásico.

La parte clásica del algoritmo utiliza álgebra lineal para extraer el período de la superposición. El algoritmo de Simon es un buen ejemplo de muchos procesos que necesitan una rápida interconexión. Acción entre computadoras cuánticas y clásicas. Actualmente se producen supercomputadoras con interconexiones de alta velocidad integradas. Estos permitirán que las futuras computadoras cuánticas se conecten e interopereren a velocidades muy altas para resolver problemas complejos, utilizando lo mejor de las computadoras clásicas y cuánticas.

5.5.2. Circuito

Calcular f en una superposición de estados efectivamente lo evalúa en todos esos estados de una sola vez, pero los valores de las funciones individuales son inaccesibles porque aparecen en una superposición, y una vez que mida, solo será posible conocer uno de ellos. Por lo tanto, es necesario manipular el estado cuántico, antes de medir, para que se pueda encontrar la colisión buscada.

Se introduce en el oráculo la superposición de los 2^n valores posibles. A diferencia del algoritmo de Deutsch, en el que f tiene un valor de un solo Qbit, la función en el problema de Simon tiene tantos Qbits como el tamaño de la entrada. Por lo tanto, la entrada inferior debe constar de n Qbits que se inicializan en cero. La salida del oráculo tendrá la forma $\sum_{x\in\{0,1\}^n} |\mathbf{x}\rangle \left|f(x)\right\rangle$, una superposición de 2^n estados, cada uno de los cuales tiene un posible valor para x entrelazado con un valor correspondiente para $f(x)$. Si se mide el Qbit inferior, que se encuentra en el estado $\left|f(x)\right\rangle$, colapsará a alguno de los valores que f puede tomar, por ejemplo $F = f(a)$, para algún a. Pero, dado que f es 2 a 1, solo existen dos de esos valores, por lo que la entrada superior colapsará a una superposición de los estados $|a\rangle$ y $|b\rangle$, donde $f(a) = f(b)$ y $b = a \oplus c$

Aunque esto resuelve el problema a nivel cuántico, no hay forma de conocer a y b . Todo lo que se puede hacer es medir la salida superior, pero, al hacerlo, se obtendrá a o b, pero no ambos. Debido al teorema de no clonación, es imposible copiar el estado, con la esperanza de obtener a y b por aparte. Si el proceso se repite, se obtendrán las colisiones de otros pares de números, que dependerán de cuál de las $\left|f(x)\right\rangle$ colapsó. Por lo tanto, se vuelve necesario manipular el estado antes de medirlo. Para ello, se agrega una compuerta Hadamard en cada uno de los Qbits superiores para permitir que los componentes interfieran. Dado que a y b están conectados a través de c, su patrón de interferencia revelará información sobre c. Si esta información es suficiente para determinar c, se ha terminado. En caso contrario, se repite todo el proceso para obtener más información sobre c, que es independiente de qué valor $\left|f(x)\right\rangle$ colapsa, por lo que la repetición es significativa aquí. La parte cuántica de este algoritmo consiste, básicamente, en realizar las operaciones indicadas en el circuito bastantes veces:

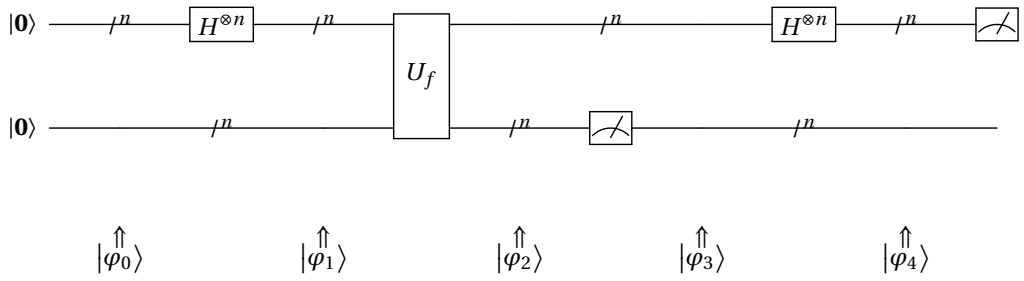


Figura 5.6: Circuito para el algoritmo de Simon

En términos de matrices, este circuito es equivalente a aplicar las siguientes matrices:

$$[H^{\otimes n} \otimes \mathbf{1}] U_f [H^{\otimes n} \otimes \mathbf{1}] |\mathbf{0}\rangle |\mathbf{0}\rangle \tag{5.44}$$

El cálculo comienza en el estado $|\varphi_0\rangle = |\mathbf{0}\rangle |\mathbf{0}\rangle$. Al aplicar el primer conjunto de compuertas Hadamard se obtiene

$$|\varphi_1\rangle = H^{\otimes n} \otimes \mathbf{1} |\mathbf{0}\rangle |\mathbf{0}\rangle = H^{\otimes n} |\mathbf{0}\rangle \otimes |\mathbf{0}\rangle = \frac{1}{\sqrt{2^n}} \sum_{x \in \{0,1\}^n} |\mathbf{x}\rangle |\mathbf{0}\rangle \tag{5.45}$$

Al aplicar la matriz unitaria U_f, se tiene

$$|\varphi_2\rangle = \frac{1}{\sqrt{2^n}} \sum_{x \in \{0,1\}^n} |\mathbf{x}\rangle |f(x)\rangle \tag{5.46}$$

La medición es necesaria para filtrar los valores de x que dan el mismo valor de $f(x)$, sin embargo, es más sencillo comprender la acción de la segunda compuerta Hadamard omitiendola de momento como si la medición no se efectuara. En ese caso:

$$|\varphi_4\rangle = H^{\otimes n} \left[\frac{1}{\sqrt{2^n}} \sum_{x \in \{0,1\}^n} |\mathbf{x}\rangle |f(x)\rangle \right] = \frac{1}{\sqrt{2^n}} \sum_{x \in \{0,1\}^n} H^{\otimes n} |\mathbf{x}\rangle \otimes |f(x)\rangle \tag{5.47}$$

$$= \frac{1}{\sqrt{2^n}} \sum_{z \in \{0,1\}^n} \frac{1}{\sqrt{2^n}} \sum_{z \in \{0,1\}^n} (-1)^{\langle \mathbf{z},\mathbf{x} \rangle} |\mathbf{x}\rangle \otimes |f(x)\rangle \tag{5.48}$$

$$|\varphi_4\rangle = \frac{1}{2^n} \sum_{z \in \{0,1\}^n} \sum_{x \in \{0,1\}^n} (-1)^{\langle \mathbf{z},\mathbf{x} \rangle} |\mathbf{z}\rangle |f(x)\rangle \tag{5.49}$$

donde habrá que realizar la suma sobre todas las x. Sin embargo, al efectuar la medición $|f(x)\rangle$, colapsará a alguno de los valores que f puede tomar, por ejemplo $F = f(a)$, para algún a, pero, dado que f es 2 a 1, según la promesa, solo existen dos de esos valores, por ejemplo a y b según la promesa, deben estar conectados por c, por lo que la entrada superior colapsará a una superposición de los estados $|a\rangle$ y $|b\rangle$, donde $f(a) = f(b)$ y $b = a \oplus c$.

$$|\varphi_3\rangle = \frac{1}{\sqrt{2}} (|a, F\rangle + |b, F\rangle) = \frac{1}{\sqrt{2}} (|a, F\rangle + |a \oplus c, F\rangle) \tag{5.50}$$

Recordando que

$$H^{\otimes n} |\mathbf{x}\rangle = \frac{1}{\sqrt{2^n}} \sum_{z \in \{0,1\}^n} (-1)^{\langle \mathbf{z},\mathbf{x} \rangle} |\mathbf{z}\rangle \tag{5.51}$$

al aplicar en $\mathbf{x} \oplus \mathbf{c}$ se obtiene:

$$H^{\otimes n} |\mathbf{x} \oplus \mathbf{c}\rangle = \frac{1}{\sqrt{2^n}} \sum_{z \in \{0,1\}^n} (-1)^{\langle z, x \oplus c \rangle} |\mathbf{z}\rangle = \frac{1}{\sqrt{2^n}} \sum_{z \in \{0,1\}^n} (-1)^{\langle \mathbf{z},\mathbf{x} \rangle} (-1)^{\langle \mathbf{z},\mathbf{c} \rangle} |\mathbf{z}\rangle \tag{5.52}$$

por lo que se puede simplificar

$$|\varphi_4\rangle = \frac{1}{\sqrt{2^n}} \sum_{z \in \{0,1\}^n} \frac{1}{\sqrt{2}} ((-1)^{\langle z,a \rangle} |z, f(a)\rangle + (-1)^{\langle z, a \oplus c \rangle} |z, f(a \oplus c)\rangle) \tag{5.53}$$

$$= \frac{1}{\sqrt{2^n}} \sum_{z\in\{0,1\}^n} \frac{1}{\sqrt{2}}((-1)^{\langle z,a\rangle}|z,F\rangle + (-1)^{\langle z,a\oplus c\rangle}|z,F\rangle) \tag{5.54}$$

$$= \frac{1}{\sqrt{2^{n+1}}} \sum_{z\in\{0,1\}^n} \left((-1)^{\langle z,a\rangle} + (-1)^{\langle z,a\rangle}(-1)^{\langle z,c\rangle}\right)|z,F\rangle \tag{5.55}$$

$$= \frac{1}{\sqrt{2^{n+1}}} \sum_{z\in\{0,1\}^n} (-1)^{\langle z,a\rangle}\left(1 + (-1)^{\langle z,c\rangle}\right)|z,F\rangle \tag{5.56}$$

El estado final expone el efecto de interferencia: si $\langle z,c\rangle$ es impar para ciertos componentes $|z\rangle$ entonces esos componentes interfieren destructivamente y se traduce en probabilidad cero de obtener estos componentes cuando se mide. Pero si es par, entonces los componentes interfieren constructivamente y se tiene un 100% de probabilidad de obtener estos componentes. Por lo tanto, si se mide la salida superior y la se encuentra en el estado $|z\rangle$ entonces $\langle z,c\rangle$ es par. Esto da cierta información sobre el c buscado; de hecho, reduce a la mitad el número de valores posibles de c. Para conocer más al respecto, se repite todo el proceso suficientes veces para determinarlo por completo. Dado que c contiene n bits, repetir el cálculo $\mathcal{O}(n)$ veces debería ser suficiente para determinar c por completo. Si esto se repite $n-1$ veces, podrá obtenerse un sistema lineal hoomogéneo de $n-1$ ecuaciones con n incógnitas (los bits de c). Luego, se pueden colocar estos resultados en un algoritmo clásico que resuelve ecuaciones lineales (aunque en lugar de utilizar la operación de suma habitual, se emplea $\oplus$ en cadenas binarias). Se puede demostrar que este sistema tendrá una solución única distinta de cero con probabilidad $p > 1/4$. Por lo tanto, si se repite m veces (para un total de $m(n-1)$ repeticiones), la probabilidad de falla $(1-1/4)^m < e^{-m/4}$ cae exponencialmente con m. Por lo tanto, las consultas de oráculo $\mathcal{O}(n)$ son suficientes para determinar c. Esta es una aceleración exponencial con respecto al cálculo clásico. En conclusión, para una función periódica dada f, es posible encontrar el período c en n evaluaciones de la función, en contraste con las $2^{n-1}+1$ evaluaciones necesarias con el algoritmo clásico.

5.6. Estimación de fase cuántica

5.6.1. Descripción del problema

Al final de los algoritmos de Deutsch y de Deutsch-Jousa, se empleo una compuerta Hadamard para obtener información codificada en las fases relativas de un estado. La compuerta Hadamard es su propia inversa y, por lo tanto, también hace lo contrario, es decir, puede ser utiliza para codificar información en las fases. Considerando que

$$H^{\otimes n}|\mathbf{x}\rangle = \frac{1}{\sqrt{2^n}} \sum_{z\in\{0,1\}^n} (-1)^{\langle \mathbf{z},\mathbf{x}\rangle}|\mathbf{z}\rangle \tag{5.57}$$

Se puede pensar que la compuerta de Hadamard de n Qbits tiene información *codificada* sobre el valor de x en las fases $(-1)^{\langle \mathbf{z},\mathbf{x}\rangle}$ de los estados base $|\mathbf{z}\rangle$. Si se aplica $H^{\otimes n}$ a este estado se vuelve a obtener $|\mathbf{x}\rangle$ nuevamente.

$$H^{\otimes n}\frac{1}{\sqrt{2^n}} \sum_{z\in\{0,1\}^n} (-1)^{\langle \mathbf{z},\mathbf{x}\rangle}|\mathbf{z}\rangle = H^{\otimes n}H^{\otimes n}|\mathbf{x}\rangle) = |\mathbf{x}\rangle \tag{5.58}$$

Aquí se puede considerar que la puerta Hadamard de n Qbits actúa *decodificando* la información sobre el valor de x que se codificó en las fases. En este caso, las fases $(-1)^{\langle \mathbf{z},\mathbf{x}\rangle}$ tienen una forma especial, pero, en general, una fase es una cantidad compleja, comunmente expresada en la forma compleja $e^{2\pi i\omega}$, para cualquier número real $\omega \in (0,1)$. La fase -1 corresponde a $\omega = \frac{1}{2}$. La transformación Hadamard de n Qbits no puede acceder completamente a información codificada de manera más general.

Suponiendo que se tiene un estado de la forma $\frac{1}{\sqrt{2^n}}\sum_{z=0}^{2^n-1} e^{2\pi i\omega z}|\mathbf{z}\rangle$ donde $\omega \in (0,1)$ se está considerado en determinar ω.. Las cadenas $\mathbf{z}$ de n bits se consideran como números enteros de 0 a 2^n-1

Problema de la estimación de fase y Transformada Cuántica de Fourier

Entrada: El estado $\frac{1}{\sqrt{2^n}}\sum_{y=0}^{2^n-1} e^{2\pi i\omega z}|\mathbf{z}\rangle$.

Tarea: Obtener una estimación del parametro ω

Existe un algoritmo cuántico para resolver el problema de estimación de fase. Si la entrada es un estado de 1 Qbit, entonces $n=1$, y si $\omega = 0.x_1$, el estado puede escribirse como [1]:

[1] Es necesario indicar algo de notación, por ejemplo, ω se puede escribir en binario como $\omega = 0.x_1x_2x_3...$, que significa $x_1\cdot 2^{-1} + x_2\cdot 2^{-2} +$

$$\frac{1}{\sqrt{2^n}}\sum_{z=0}^{2^n-1} e^{2\pi i(0.x_1)z}|\mathbf{z}\rangle = \frac{1}{\sqrt{2^n}}\sum_{z=0}^{2^n-1} e^{2\pi i\frac{x_1}{2}z}|\mathbf{z}\rangle = \frac{1}{\sqrt{2^n}}\sum_{z=0}^{2^n-1} e^{\pi i x_1 z}|\mathbf{z}\rangle = \frac{1}{\sqrt{2^n}}\big(|0\rangle + (-1)^{x_1}|1\rangle\big) \tag{5.60}$$

Puede observarse que se puede usar una compuerta Hadamard de un solo Qbit para determinar el valor de x_1, que en este caso representa ω

$$H\Big(\frac{1}{\sqrt{2^n}}\big(|0\rangle + (-1)^{x_1}|1\rangle\big)\Big) = |x_1\rangle \tag{5.61}$$

$$\frac{1}{\sqrt{2^n}}\sum_{z=0}^{2^n-1} e^{2\pi i(0.x_1)z}|\mathbf{z}\rangle = \frac{1}{\sqrt{2^n}}\sum_{z=0}^{2^n-1} e^{2\pi i\frac{x_1}{2}z}|\mathbf{z}\rangle \tag{5.62}$$

Para tratar con casos más complicados, se tiene la identidad

$$\frac{1}{\sqrt{2^n}}\sum_{z=0}^{2^n-1} e^{2\pi i(0.x_1)z}|\mathbf{z}\rangle = \Big(\frac{|0\rangle + e^{2\pi i(2^{n-1}\omega)}|1\rangle}{\sqrt{2}}\Big)\otimes\Big(\frac{|0\rangle + e^{2\pi i(2^{n-2}\omega)}|1\rangle}{\sqrt{2}}\Big)\otimes\ldots\otimes\Big(\frac{|0\rangle + e^{2\pi i(\omega)}|1\rangle}{\sqrt{2}}\Big) \tag{5.63}$$

$$= \Big(\frac{|0\rangle + e^{2\pi i(0.x_n x_{n+1}\ldots)}|1\rangle}{\sqrt{2}}\Big)\otimes\Big(\frac{|0\rangle + e^{2\pi i(0.x_{n-1}x_n x_{n+1}\ldots)}|1\rangle}{\sqrt{2}}\Big)\otimes\ldots\otimes\Big(\frac{|0\rangle + e^{2\pi i(0.x_1 x_2\ldots)}|1\rangle}{\sqrt{2}}\Big) \tag{5.64}$$

por ejemplo, para un estado de 2 Qbits:

$$\frac{1}{\sqrt{2^2}}\sum_{z=0}^{2^2-1} e^{2\pi i(0.x_1x_2)z}|\mathbf{z}\rangle = \Big(\frac{|0\rangle + e^{2\pi i(0.x_2)}|1\rangle}{\sqrt{2}}\Big)\otimes\Big(\frac{|0\rangle + e^{2\pi i(0.x_1x_2)}|1\rangle}{\sqrt{2}}\Big) \tag{5.65}$$

Se puede obtener x_2 a partir del primer Qbit, aplicando una puerta de Hadamard, de la misma forma en que se hizo en el caso anterior, pero ahora se debe determinar x_1, algo que debe hacerse a partir del segundo Qbit. Si que $x_2 = 0$, entondes el segundo Qbit esta en el estado $\frac{1}{\sqrt{2}}(|0\rangle + e^{2\pi i(0.x_2)}|1\rangle)$ y es posible determinar x_1 usando una puerta de Hadamard. Pero si $x_2 = 1$ se necesita hacer algo más. Para esto se define un operador de rotación de fase de 1 Qbit R_2 respecto a la base computacional como sigue:

$$R_2 = \begin{bmatrix} 1 & 0 \\ 0 & e^{\frac{2\pi i}{2^2}} \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & e^{2\pi i(0.01)} \end{bmatrix} \tag{5.66}$$

en donde 0.01 es el exponente escrito en base 2. La inversa de R_2 es

$$R_2 = \begin{bmatrix} 1 & 0 \\ 0 & e^{-2\pi i(0.01)} \end{bmatrix} \tag{5.67}$$

Si $x_2 = 1$ se debe considerar el efecto de aplicar R_2^{-1} al segundo Qbit:

$$R_2^{-1}\Big(\frac{|0\rangle + e^{2\pi i(0.x_1 1)}|1\rangle}{\sqrt{2}}\Big) = \Big(\frac{|0\rangle + e^{2\pi i(0.x_1 1 - 0.01)}|1\rangle}{\sqrt{2}}\Big) = \Big(\frac{|0\rangle + e^{2\pi i(0.x_1)}|1\rangle}{\sqrt{2}}\Big) \tag{5.68}$$

Como puede verse, después de aplicar R_2^{-1}, la compuerta Hadamard se puede utilizar para determinar x_1. La aplicación deR_2^{-1} al segundo Qbit antes de aplicar la compuerta Hadamard dependerá si $x_2 = 1$ o $x_2 = 0$. Después de aplicar la compuerta Hadamard al primer Qbit, el estado del primer Qbit pasó a ser $|x_2\rangle$. Por eso es necesario utilizar una compuerta R_2^{-1} controlada. El objetivo será el segundo Qbit, el control será el estado del primer Qbit. En resumen, para el caso de un estado de 2 Qbits con $\omega = 0.x_1x_2$, el circuito que se muestra a continuación resuelve el problema de estimación de fase (aunque, en este caso, la estimación es exacta).

$x_3\cdot 2^{-3} + \ldots$, de manera similar, se escribe los múltiplos potencia par de ω como $2^k\omega = x_1x_2x_3\ldots x_k.x_{k+1}\ldots$, como $e^{2\pi ik} = 1$ para todo entero k, se tiene

$$e^{2\pi i(2^k\omega))} = e^{2\pi i(x_1x_2x_3\ldots x_k.x_{k+1}\ldots)} = e^{2\pi i(x_1x_2x_3\ldots x_k)}e^{2\pi i(0.x_{k+1}x_{k+2}\ldots)} = e^{2\pi i(0.x_{k+1}x_{k+2}\ldots)} \tag{5.59}$$

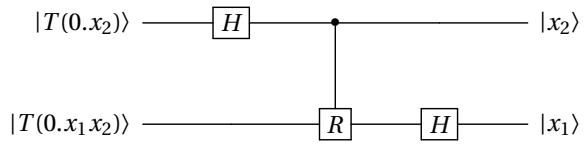


Figura 5.7: Circuito para el algoritmo cuántico para la estimación de fase cuántica, caso de un estado de 2 Qbits con $\omega = 0.x_1x_2$. Para abreviar la notación, se usa aquí $|T(\omega)\rangle = \frac{|0\rangle+e^{2\pi i(\omega)}|1\rangle}{\sqrt{2}}$

Para ilustrar la forma en que este procedimiento puede generalizarse, el caso $\omega = 0.x_1x_2x_3$ para un estado de 3 Qbits es de utilidad. Dada la identidad mencionada $\frac{1}{\sqrt{2^3}}\sum_{z=0}^{2^3-1} e^{2\pi i(0.x_1x_2x_3)z}|\mathbf{z}\rangle = \left(\frac{|0\rangle+e^{2\pi i(0.x_2)}|1\rangle}{\sqrt{2}}\right)\otimes\left(\frac{|0\rangle+e^{2\pi i(0.x_1x_2)}|1\rangle}{\sqrt{2}}\right)\otimes\left(\frac{|0\rangle+e^{2\pi i(0.x_1x_2x_3)}|1\rangle}{\sqrt{2}}\right)$, el diagrama de circuito puede verse como sigue:

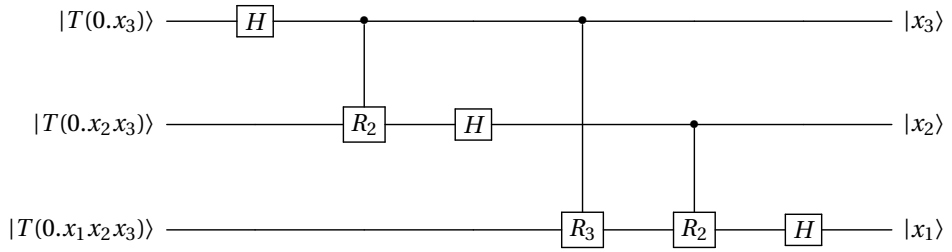


Figura 5.8: Circuito para el algoritmo cuántico para la estimación de fase cuántica, caso de un estado de 3 Qbits con $\omega = 0.x_1x_2x_3$.

Se define a la compuerta general de rotación de fase de 1 Qbit R_k como :

$$R_k = \begin{bmatrix} 1 & 0 \\ 0 & e^{\frac{2\pi i}{2^k}} \end{bmatrix} \tag{5.69}$$

cuya inversa R_k^{-1} tiene los siguientes efectos en los estado base:

$$R_k^{-1}|0\rangle = |0\rangle \tag{5.70}$$

$$R_k^{-1}|1\rangle = e^{-2\pi i(0.0_1 0_2 \ldots 0_{k-1} 1_k 0_{k+1} \ldots)}|0\rangle \tag{5.71}$$

En donde a cada dígito se le ha colocado como subíndice la posición que ocupa después del punto, es decir, el 1 en el exponente se encuentra en la k-ésima posición. Esta vez, para obtener el tercer Qbit, será necesario rotar tanto x_2 como x_3. El circuito de la figura 5.8 muestra una implementación del algoritmo de estimación de fase para un estado de 3 Qbits. Una medida en el estado a la salida del circuito indicará $\omega = 0.x_1x_2x_3$.

El algoritmo de estimación de fase anterior funciona para el estado de n Qbits

$$\frac{1}{\sqrt{2^n}}\sum_{y=0}^{2^n-1} e^{2\pi i\omega z}|\mathbf{z}\rangle \tag{5.72}$$

cuando la fase tiene la forma $\omega = 0.x_1x_2\ldots x_n$. Es decir, el algoritmo de estimación de fase devuelve x cuando ω tiene la forma $\frac{x}{2^n}$ de algún número entero n. Pero, para ω arbitrario devolverá x tal que $\frac{x}{2^n}$ sea el más cercano a ω con alta probabilidad, motivo por el que se usa la palabra *estimación* en este algoritmo. Lo que hay que hacer es decidir la cantidad n de Qbits que se usarán para que la estimación sea tan cercana como se desee.

Puede verse que el resultado del diagrama de la figura 5.8 es el estado$|x_3x_2x_1\rangle$. Para el circuito análogo en n Qbits que estima la fase $x = 0.x_1x_2\ldots x_n$, la salida del circuito sería el estado $|x_1x_2\ldots x_n\rangle$. Si se agregan algunas compuertas para invertir el orden de los Qbits al final, se tendrá un circuito eficiente, con $\mathcal{O}(n^2)$ puertas, que implementa

$$U\left(\frac{1}{\sqrt{2^n}}\sum_{y=0}^{2^n-1} e^{2\pi i\frac{x}{2^n}z}|\mathbf{z}\rangle\right) = |x\rangle \tag{5.73}$$

Si se toma la transformación inversa:

$$U^{-1}|x\rangle = \frac{1}{\sqrt{2^n}} \sum_{y=0}^{2^n-1} e^{2\pi i \frac{x}{2^n} z} |\mathbf{z}\rangle \tag{5.74}$$

El circuito para esta transformación se obtiene remplazando cada compuerta por su inversa y recorriendo el circuito resultante de derecha a izquierda. Como puede observarse, existe mucha similitud la transformada discreta de Fourier con la transformación resultante, por lo que se le denomina *transformada cuántica de Fourier* en n Qbits [303], o *QFT*, por sus siglas en inglés.

$$QFT|x\rangle = \frac{1}{\sqrt{N}} \sum_{z=0}^{N-1} e^{2\pi i \frac{xz}{2^n}} |z\rangle \tag{5.75}$$

con $N = 2^n$. La QFT se extiende linealmente a superposiciones arbitrarias de estados base. Dado que QFT es la inversa de la operación de estimación de fase, se tiene un circuito eficiente para realizar la QFT (a partir del circuito de estimación de fase, lo único que se debe hacer es recorrerlo al revés). Como referencia, en la Figura 5.9 se muestra un circuito cuántico para QFT.

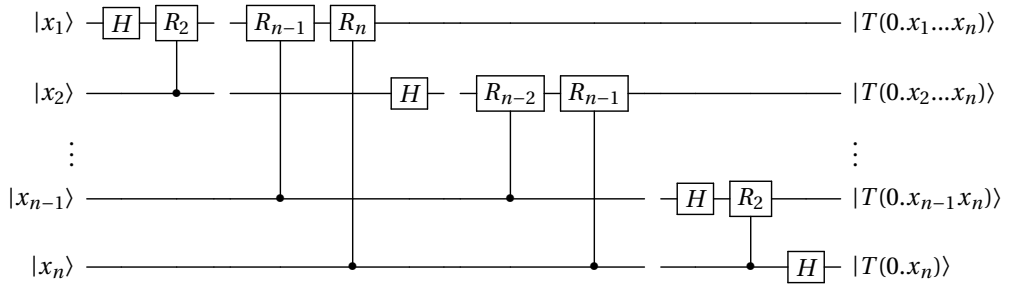


Figura 5.9: Un circuito para QFT, hasta la permutación de los Qbits de salida para invertir su orden. En la práctica, no es necesario implementar físicamente esta inversión, es posible lograr el resultado deseado simplemente reasignando etiquetas a los Qbits.

5.7. Algoritmo de Grover (Búsqueda cuántica)

5.7.1. Descripción del problema

Una de las tareas más comunes en computación es la *búsqueda*. Existe una clase grande de problemas que se pueden identificar como *problemas de búsqueda*, para los que no se conocen técnicas que hagan mucho uso de la estructura del problema, y el mejor algoritmo que se conoce para resolverlos resulta ser la búsqueda *ingenua* entre las posibles soluciones hasta encontrar una. Tales problemas pueden enunciarse de la forma: «Encontrar alguna x en un conjunto de posibles soluciones tales que el enunciado $P(x)$ resulte verdadero» [239], y el mecanismo propuesto requiere que sea posible reconocer de manera eficiente la solución, buscando dentro de una lista de posibles soluciones para encontrar la correcta. Por ejemplo, dado un número entero grande N, se puede reconocer eficientemente si un número entero p es un factor no trivial de N y, por lo tanto, una estrategia ingenua para encontrar factores no triviales de N es simplemente buscar en el conjunto $\{2,3,4,...,\sqrt{N}\}$ hasta encontrar un factor. Problemas de este estilo van desde la búsqueda en bases de datos hasta la clasificación y la coloración de gráficos, que verse como la búsqueda de una asignación de colores a los vértices de modo que la afirmación «todos los vértices adyacentes tienen colores diferentes» resulte verdadera. De manera similar, un problema de ordenación puede verse como una búsqueda de una permutación para la cual la afirmación «la permutación x lleva el estado inicial al estado ordenado deseado» es verdadera.

Problemas como estos pueden resolverse enumerando las posibles soluciones y luego buscándolas, ya sea de manera sistemática o de forma totalmente aleatoria, para determinar cuáles son las correctas. En algunos casos, determinar que ciertas posibilidades son incorrectas permite eliminar algunas de las opciones y, por lo tanto, hace posible reducir la búsqueda de la verdadera solución. Estos problemas de búsqueda se dice que son *estructurados*. Alternativamente, hay otros problemas de búsqueda en los que no se aprende nada útil al descubrir que ciertas posibilidades son incorrectas, aparte de la inutilidad de intentar esas posibilidades nuevamente, las consultas no proporciona información sobre la respuesta hasta que realmente se encuentra. Estos problemas de búsqueda se dice que son *no estructurados*.

La búsqueda no estructurada es el problema computacional equivalente a *encontrar la aguja en el pajar*, en donde es necesario inspeccionar cuidadosamente cada una de las fibras en la pila de heno, para verificar si se trata de la aguja en cuestión. Un ejemplo práctico de búsqueda en una base de datos desordenada sería encontrar dentro de una guía telefónica un cierto número y se desea conocer el nombre de la persona a la cuál pertenece. Lo mejor que puede hacerse con una computadora clásica es revisar la guía, número por número, hasta encontrar el buscado y se tendrá la solución (el nombre) justo al lado. Esto quiere decir que reconocer la solución es fácil, la parte difícil consiste en encontrarla. Esta es una característica de los problemas de la clase computacional **NP** y para muchos de estos casos no hay mejor algoritmo clásico que la búsqueda exhaustiva de todas las soluciones posibles. Normalmente, el número de soluciones potenciales es exponencial en el tamaño de la instancia del problema, por lo que el algoritmo ingenuo no es eficiente. A menudo, la búsqueda clásica más conocida hace un uso muy limitado de la estructura del problema, tal vez para descartar algunos candidatos obviamente imposibles o para priorizar algunos candidatos más probables, pero la complejidad general de la búsqueda sigue siendo exponencial.

La formulación computacional de este problema se refiere a una matriz desordenada de N elementos en la que debe hallarse un elemento particular. Clásicamente, el peor de los casos requiere realizar N consultas, es decir, examinar hasta el último elemento antes de encontrar el valor objetivo. En promedio, el elemento deseado se encontrará tras realizar $N/2$ consultas. Parece claro que si los elementos de la base de datos se almacenan en orden aleatorio, el único método disponible para encontrar un elemento específico es una búsqueda exhaustiva, que no será, por lo general, la mejor manera de utilizar bases de datos, especialmente si se consultan varias veces. Sería mejor clasificar los elementos, lo cual es una tarea costosa, pero que se realiza solo una vez. Sin embargo, en el contexto de la computación cuántica, almacenar datos en superposición o en estado entrelazado durante un largo período de tiempo no es una tarea fácil[231].

El algoritmo de Grover es un algoritmo de búsqueda diseñado originalmente para buscar un elemento en una base de datos desordenada y sin elementos repetidos, equivalente a encontrar la única entrada para una función dada que producirán una determinada salida. De manera formal, si f es una función binaria eficientemente computable que opera en cadenas de bits de longitud n, encontrar x tal que $f(x) = 1$. Este problema puede reformularse como un problema de oráculo, en el que los elementos de la base de datos estarán dados como $\{0,1,2,...,N-1\}$ y x_0 es un elemento desconocido pero marcado. El oráculo estará definido como $f : \{0,1\}^n \to \{0,1\}$ tal que

$$f(x) = \begin{cases} 1, & \text{si } x = x_0, \\ 0, & \text{si } x \neq x_0. \end{cases} \tag{5.76}$$

El problema es encontrar x_0, el *elemento marcado*, con el mínimo número de consultas del oráculo f. Ya que no se conocen soluciones en tiempo polinómico, este es un ejemplo básico de búsqueda NP. En ausencia de información sobre la naturaleza de la función, el algoritmo clásico más rápido conocido para este problema es la búsqueda exhaustiva o exploración de todas las entradas posibles para encontrar la respuesta, un proceso que requiere $\mathcal{O}(N) = \mathcal{O}(2^n)$ pasos, donde n es el número de bits necesarios para representar la entrada. El algoritmo de Grover resuelve este problema en $\mathcal{O}(\sqrt{N})$ pasos. Si bien esto es sólo una aceleración polinómica con respecto al mejor enfoque clásico, podría resultar significativa en la práctica.

Problema de la búsqueda
Entrada: Una caja negra U_f que permita evaluar a una función desconocida $f : \{0,1\}^n \to \{0,1\}$.
Tarea: Encontrar una entrada $x \in \{0,1\}^n$ tal que $f(x) = 1$.

Como el mismo Grover describe [117], si se inicia con una superposición lineal uniforme y se permite que evolucione en presencia de una función potencial, «gravitará» hacia puntos en los que el potencial es menor. Luego, si se trata de diseñar un algoritmo que permita llegar a determinados estados, dichos estados deberían tener un valor de potencial menor. Además, el algoritmo requerirá varias iteraciones de las transformaciones, de tal forma que el vector de estado pueda rotar desde un estado inicial hasta el objetivo.

Dado el oráculo U_f que implementa la función $f(x)$, se tiente que $U_f|x,0\rangle = |x, f(x)\rangle$. Como ya se ha visto, es de utilidad emplear la superposición para evaluar f en todas las posibles entradas. Para hacerlo, se aplica la transformación U_f sobre $\frac{1}{\sqrt{2^n}} \sum_{x=0}^{2^n-1} |x\rangle |0\rangle$, obteniendo

$$|\psi_1\rangle = \frac{1}{\sqrt{2^n}} \sum_{x=0}^{2^n-1} |x, f(x)\rangle \tag{5.77}$$

Al igual que antes, el paso difícil es obtener un resultado útil a partir de esta superposición. Pero, puede observarse que para cualquier x_0 tal que $f(x_0) = 1$, el estado $|x_0, 1\rangle$ será parte de la superposición de la ecuación 5.77. Dado que la amplitud de tal estado es $\frac{1}{\sqrt{2^n}}$, la probabilidad de que una medición aleatoria de la superposición produzca x_0 es sólo 2^{-n}. El algoritmo de Grover consiste en cambiar el estado cuántico de la ecuación 5.77 para aumentar considerablemente la amplitud de los vectores $|x_0, 1\rangle$ y disminuir la amplitud de los vectores $|x, 0\rangle$. Una vez que se ha realizado dicha transformación, sera posible medir el último Qbit, que representa a $f(x)$. Como la amplitud ha sido modificada, existirá una probabilidad muy alta de que el resutltado sea 1, caso en el que la medición ha proyectado al estado de la ecuación 5.77 en el subespacio $\frac{1}{\sqrt{k}} \sum_{i=1}^{k} |x_i, 1\rangle$ donde k es el número de soluciones. Una medición adicional de los Qbits restantes proporcionará una de estas soluciones. Si la medición del Qbit $f(x)$ arroja 0, entonces todo el proceso se reinicia y la superposición de la ecuación 5.77 debe calcularse nuevamente.

Antes de abordar el circuito, será necesario identificar algunas de las operaciones realizadas por sus componentes. En primer lugar, se tiene al *operador del oráculo*, la compuerta cuántica equivalente a la función de caja negra $f(x)$. El oráculo actuará sobre un estado cuántico de n Qbit $|x\rangle$ y agregará una fase negativa al estado si es igual al estado objetivo $|x_0\rangle$ y, en caso contrario, lo dejará sin cambios:

$$U_f |x\rangle = \begin{cases} -|x\rangle, & \text{si } x = x_0, \\ +|x\rangle, & \text{si } x \neq x_0. \end{cases} \tag{5.78}$$

que, de acuerdo con la ecuación 5.14, también como $U_f |x\rangle = (-1)^{f(x)} |x\rangle$. Hay una forma más de visualizar este operador. Dado que los estados $|x\rangle$ son vectores columna, si se desea cambiar el signo a la j-ésima entrada, respetando toda las demás, es puede multiplicar por la siguiente matriz

$$M_j = \begin{pmatrix} 1 & 0 & \dots & 0 & \dots & 0 \\ 0 & 1 & \dots & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & -1 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 0 & \dots & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & \dots & 0 & \dots & 0 \\ 0 & 1 & \dots & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 1 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 0 & \dots & 1 \end{pmatrix} - 2 \begin{pmatrix} 0 & 0 & \dots & 0 & \dots & 0 \\ 0 & 0 & \dots & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 1 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 0 & \dots & 0 \end{pmatrix} \tag{5.79}$$

que es la identidad, excepto por un uno cuyo signo está invertido. Los estados $|x\rangle$ son vectores columna, con la característica adicional de que una sola de sus entradas es 1 , todas las demás son nulas, por lo que los productos de la forma $|x\rangle \langle x|$ son, en realidad, matrices con un solo elemento ($[M_j]_{j,j}$) igual a uno, y todas las demás entradas en cero. Luego, la acción sobre un estado marcado puede considerarse como una disminución en dos veces su valor. La primera le llevaría al cero, la segunda a tener el mismo valor original, pero con signo opuesto. Por tanto, $U_f |x\rangle = I - 2 |x_0\rangle \langle x_0|$ es una manera alternativa de escribir al operador del oráculo .

Un ejemplo simple en el que puede observarse la acción de esta operación unitaria es el siguiente. Si $|x\rangle$ parte de una superposición de cuatro estados diferentes, dados por $[\frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}]^T$, y U_f es tal que selecciona la tercer cadena, es decir $x_0 = 10$, tras realizar una inversión de fase, el estado se verá como $[\frac{1}{2}, \frac{1}{2}, -\frac{1}{2}, \frac{1}{2}]^T$. Aunque el estado de interés ha sido marcado con un signo negativo, una medición en $|x\rangle$ no proporciona ninguna información sobre ella, tanto $|+\frac{1}{2}|^2$ como $|-\frac{1}{2}|^2$ son iguales a $+\frac{1}{4}$. Cambiar la fase de positiva a negativa separa las fases, pero no de una manera que pueda medirse.

Para incrementar la separación de la fase de la cadena binaria marcada de las correspondientes fases del resto de las cadenas binarias, se realiza una operación que se denomina *inversión respecto a la media*. Consiste en tomar la distancia d_i de cada valor x_i que conforma el promedio $\bar{x}$ al promedio mismo. Si se está por encima del promedio, al promedio se le restará d_i y se obtendrá el valor invertido x'_i. Y de la misma forma, si x_i está por debajo del promedio, al promedio se le suma d_i para obtener x'_i. El promedio de los valores invertidos es idéntico al original, ya que $x'_i = \bar{x} - d_i = \bar{x} - (x_i - \bar{x}) = 2\bar{x} - x_i$. Por ejemplo, en la secuencia de números enteros 53, 38, 17, 23 y 79, el promedio es $\bar{x} = 42$. El primer número, $x_1 = 53$, está a $d_1 = -11$ unidades del promedio, por lo que al sumar $\bar{x}$ a d_1 se obtiene $x'_1 = 2\bar{x} - x_1 = 31$. El resto de los inversos se calcula de manera similar. Para obtener una implementación de esta operación en términos de matrices, conviene observar que el promedio puede obtenerse si las cantidades a promediar se colocan en un vector renglón $\mathbf{x} = \begin{bmatrix} x_1, & x_2, & \dots & , x_N \end{bmatrix}^T$, por ejemplo: $[53, 38, 17, 23, 79]^T$. Multiplicar por una matriz A de $N \times N$, con todas sus entradas igual a $1/N$ devolverá una matriz con todas sus entradas iguales al promedio. Luego,

la inversión estará dada por **x'** $= 2A\mathbf{x} - \mathbf{x} = (2A - I)\mathbf{x}$. Dados n Qbits, exiten 2^n estados posibles ($N = 2^n$), por lo que, para promediar, se construye la matriz A de $N \times N$ con todas sus entradas igual a $\frac{1}{2^n}$. Con estos elementos se presenta la definición para el *operador de difusión* D [116] :

$$D = 2\begin{pmatrix} \frac{1}{2^n} & \frac{1}{2^n} & \cdots & \frac{1}{2^n} \\ \frac{1}{2^n} & \frac{1}{2^n} & \cdots & \frac{1}{2^n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{1}{2^n} & \frac{1}{2^n} & \cdots & \frac{1}{2^n} \end{pmatrix} - \begin{pmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{pmatrix} = \begin{pmatrix} \frac{2}{2^n} - 1 & \frac{2}{2^n} & \cdots & \frac{2}{2^n} \\ \frac{2}{2^n} & \frac{2}{2^n} - 1 & \cdots & \frac{2}{2^n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{2}{2^n} & \frac{2}{2^n} & \cdots & \frac{2}{2^n} - 1 \end{pmatrix} \tag{5.80}$$

que realiza la inversión respecto al promedio. Además de realizar la operación solicitada, la compuerta $D = 2A - I$ debe ser reversible. Para comprobarlo, debe observarse que, si $A = \frac{1}{2^n}[1]_{i,j}$ entonces

$$A^2 = (\frac{1}{2^n}[1]_{i,j})(\frac{1}{2^n}[1]_{i,j}) = \frac{1}{(2^n)^2}[1 + 1 + ... + 1]_{i,j} = \frac{1}{(2^n)^2}[2^n]_{i,j} = \frac{1}{2^n}[1]_{i,j} = A \tag{5.81}$$

Por tanto, al aplicar dos veces el operador D, se obtiene $D^2 = (2A - I)(2A - I) = 4A^2 - 2AI - 2IA + I^2 = 4A - 4A + I = I$, lo cuál implica que D es su propia inversa. La implementación de esta operación requiere descomponer en compuertas de uno y dos Qbits. Para hacerlo, puede observarse que el *operador de inversión de fase condicional* es similar al operador del oráculo, excepto que en lugar de agregar una fase negativa al estado si es igual al estado objetivo $|x_0\rangle$, actúa si es igual al estado cero de n Qbits $|\mathbf{0}\rangle$. Esto es necesario porque el estado cero tiene la fase equivocada tras el paso anterior.

Por lo demás, el estado se mantiene sin cambios, de acuerdo con

$$U_0 |x\rangle = \begin{cases} -|x\rangle & \text{si } x = |\mathbf{0}\rangle \\ +|x\rangle & \text{si } x \neq |\mathbf{0}\rangle \end{cases} \tag{5.82}$$

Como caso especial de la expresión $U_f |x\rangle = I - 2|x_0\rangle\langle x_0|$, también se puede expresar como $U_0 = I - 2|\mathbf{0}\rangle\langle\mathbf{0}|$. Para evaluar en todas las 2^n posibles entradas, deberá introducirse una superposición $\frac{1}{\sqrt{2^n}}\sum_{x=0}^{n-1}|x\rangle$, por lo que será necesario aplicar una compuerta Hadamard $H^{\otimes n}$ antes de U_0. El operador D se obtiene aplicando un operador de Hadamard en todos los n Qbits antes y después de aplicar el operador de inversión de fase y luego agregando una fase negativa:

$$D = -H^{\otimes n} U_0 H^{\otimes n} \tag{5.83}$$

Es posible sustituir en la representación alternativa el operador inversor de fase para obtener:

$$D = -H^{\otimes n}(I - 2|\mathbf{0}\rangle\langle\mathbf{0}|)H^{\otimes n} = -\left(I - 2|+\rangle\langle+|\right) \tag{5.84}$$

5.7.2. Interpretación geométrica

Los operadores mencionados pueden expresarse en términos de transformaciones de inversión y reflexión. Un estado arbitrario $|\phi\rangle$, puede invertirse o reflejarse respecto a algún otro estado cuántico $|\psi\rangle$ mediante los siguiente operadores:

$$I_{|\psi\rangle} = I - 2|\psi\rangle\langle\psi| \qquad R_{|\psi\rangle} = -I_{|\psi\rangle} = -I + 2|\psi\rangle\langle\psi| \tag{5.85}$$

La manera en que actúan puede observarse si el estado arbitrario se descompone en expresa en términos de sus proyecciones respecto al sistema formado por un vector y su ortogonal, es decir, $|\phi\rangle = \alpha|\psi\rangle + \beta|\psi^{\perp}\rangle$. Al aplicar el operador de inversión se obtiene $I_{|\psi\rangle}|\phi\rangle = -\alpha|\psi\rangle + \beta|\psi^{\perp}\rangle$, en donde se observa que el signo delante de la componente $|\psi\rangle$ ha cambiado, lo que corresponde a un reflejo del estado general $|\phi\rangle$ respecto del estado ortogonal a $|\psi\rangle$, como puede verse en la figura

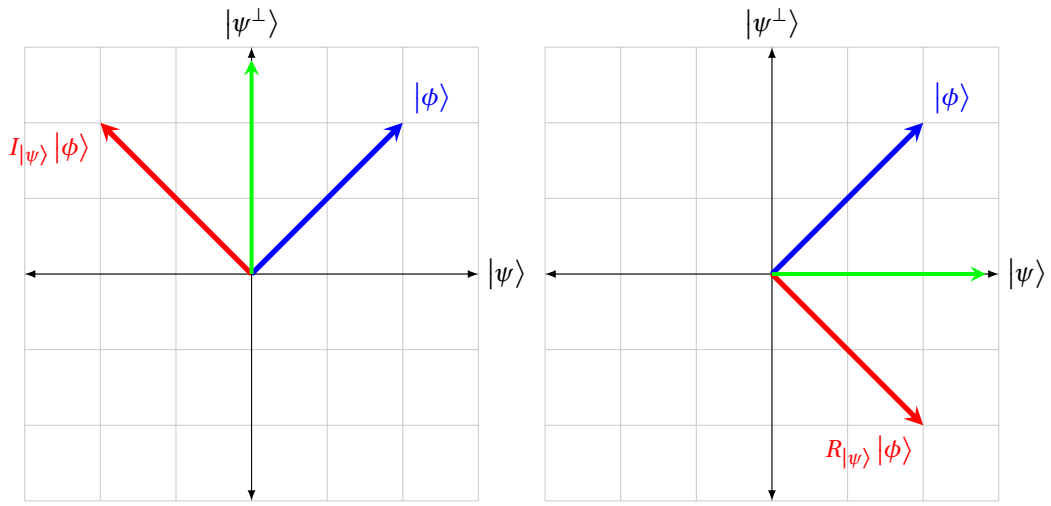


Figura 5.10: Izquierda. Operación de inversión en un estado arbitrario $|\phi\rangle$ respecto al vector $|\psi\rangle$. Derecha. Operación de reflexión de un vector de estado arbitrario $|\phi\rangle$ respecto al vector $|\psi\rangle$. La inversión respecto a $|\psi\rangle$ es una reflexión respecto del vector ortonormal $|\psi^\perp\rangle$. El eje de simetría se indica mediante la flecha verde

De manera similar, si se aplica el operador de reflexión al estado arbitrario, se obtiene $R_{|\psi\rangle}|\phi\rangle = \alpha|\psi\rangle - \beta|\psi^\perp\rangle$, donde se encuentra que el signo correspondiente a la componente ortogonal ha cambiado, lo que corresponde a un reflejo del estado general $|\phi\rangle$ sobre el estado $|\psi\rangle$, como puede verse en la figura 5.11.

Con estas definiciones, el operador de oráculo, el operador inversor de fase y el operador de difusión pueden reescribirse como:

$$U_f = I_{|x_0\rangle} = -R_{|x_0\rangle} \qquad U_0 = I_{|\mathbf{0}\rangle} = -R_{|\mathbf{0}\rangle} \qquad D = -I_{|+\rangle} = R_{|+\rangle} \tag{5.86}$$

De manera que la aplicación de la inversión de fase y el operador de difusión puede verse como $G = DU_f = -I_{|+\rangle}I_{|x_0\rangle} = (R_{|+\rangle})(-R_{|x_0\rangle}) = (-R_{|+^\perp\rangle})(-R_{|x_0\rangle}) = R_{|+^\perp\rangle}R_{|x_0\rangle}$. Un estado arbitrario $|\zeta\rangle$ en el espacio bidimensional generado por $|x_0\rangle$ y $|+\rangle$ puede escribirse en términos del estado $|+\rangle$ y su contraparte ortogonal $|+^\perp\rangle$ como $|\zeta\rangle = \alpha|+\rangle + \beta|+^\perp\rangle$, mientras que el estado objetivo $|x_0\rangle = a|+\rangle + b|+^\perp\rangle$.

El operador G puede verse como la reflexión del estado $|\zeta\rangle$ respecto al estado $|x_0\rangle$, el estado resultante se reflejara respecto al estado ortogonal a $|+\rangle$

Así, si inicialmente se tiene un ángulo θ entre el estado arbitrario y el objetivo, se debe sustraer dos veces ese ángulo tras la reflexión respecto a $|x_0\rangle$ y luego agregar $2(\theta-\gamma)$ tras la reflexión respecto al eje ortogonal (vertical).

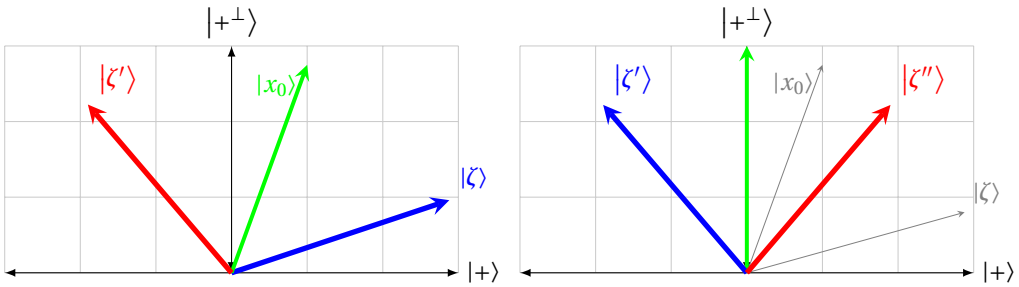


Figura 5.11: Izquierda. El estado $|\zeta'\rangle$ resulta de reflejar a $|\zeta\rangle$ respecto al estado objetivo $|x_0\rangle$ Derecha. El estado $|\zeta''\rangle$ se obtiene de invertir a $|\zeta'\rangle$ respecto al ortogonal al estado $|+\rangle$

El ángulo final será $\theta' = \theta - 2\theta + 2(\theta-\gamma) = \theta - 2\gamma$ por lo que en cada iteración, el ángulo entre el estado arbitrario $|\zeta\rangle$ y el estado objetivo $|x_0\rangle$ disminuye en 2γ. Para que el estado inicial alcance al objetivo es necesario aplicar varias veces el operador, hasta que se encuentra tan cerca como sea posible. El estado inicial, en realidad, no es arbitrario, sino que está determinado por $|\zeta\rangle = H^{\otimes n}|\mathbf{0}\rangle = |+\rangle$, de manera que el ángulo inicial

θ entre el estado inicial y el objetivo está dado por:

$$\langle +|x_0\rangle = cos\theta = \frac{1}{2^{n/2}} \sum_{x\in\{0,1\}^n} \langle x|x_0\rangle = \frac{1}{2^{n/2}} \sum_{x\in\{0,1\}^n} \delta_{x,x_0} = \frac{1}{2^{n/2}} \tag{5.87}$$

Y de forma similar, el ángulo γ entre el estado objetivo y el eje ortonormal está dado por $\cos\theta = \cos\left(\frac{\pi}{2} - \gamma\right) = \sin\gamma = \frac{1}{2^{n/2}}$. Para el caso de una base de datos muy grande $n \to \infty$, $\theta \to \pi/2$, $\gamma \to 0$, por lo que se puede aplicar la aproximación de ángulos pequeños $\gamma = \frac{1}{2^{n/2}}$. Esto significa que el número de iteraciones requeridos en el algoritmo de Grover estará dado por $\sqrt{2^n}$.

5.7.3. Circuito

La idea esencial de este algoritmo es iniciar con un vector que se encuentre a la misma distancia de todas las posibles soluciones para después rotarlo de manera gradual en dirección de x_0 [15]. En contra de toda intuición clásica, es posible realizar este giro aun sin conocer la solución, y sin ver la matriz que realiza esta acción. El vector $|+\rangle = \frac{1}{2^{n/2}} \sum_{x\in\{0,1\}^n} |x\rangle$ es el promedio de todos los estados clásicos observables $|+\rangle = \frac{1}{2^{n/2}} \sum_{x\in\{0,1\}^n} |x\rangle$.

El cálculo comienza en

$$|\varphi_1\rangle = |\mathbf{0}, 1\rangle. \tag{5.88}$$

Al igual que antes, se prepara la superposición de todos los posibles estados mediante $H^{\otimes n}$ aplicada sobre $|\mathbf{0}\rangle = |0...0\rangle$.

$$|\varphi_2\rangle = H^{\otimes n+1}|\mathbf{0}, 1\rangle = H^{\otimes n}|\mathbf{0}\rangle \otimes H|1\rangle = \frac{1}{\sqrt{2^n}} \sum_{x\in\{0,1\}^n} |\mathbf{x}\rangle\,|-\rangle = \frac{1}{\sqrt{2^n}} \Big(|x_0\rangle + \sum_{x\neq x_0} |\mathbf{x}\rangle \Big) |-\rangle \tag{5.89}$$

El vector resultante $|+\rangle$ se encuentra a la misma distancia de todos posibles vectores de solución $\langle +|a\rangle = \frac{1}{2^{n/2}}$. El bloque U_f que sigue se repetirá $2^{n/2}$ veces. En términos de computación clásica, esto se puede considerar un ciclo, donde la salida del difusor se usa nuevamente en la entrada al bloque, algo que sugiere que la salida y la entrada están conectadas, aunque esto no es lo que sucede físicamente. Luego

$$|\varphi_3\rangle = \begin{cases} |\varphi_2\rangle & \text{si es la primera iteración} \\ |\varphi_{3b}\rangle & \text{si no es la primera iteración} \end{cases} \tag{5.90}$$

Para marcar el vector buscado, se aplica la matriz unitaria U_f, que refleja al vector, respecto al vector ortogonal al estado objetivo, obteniendo

$$|\varphi_{3a}\rangle = \frac{1}{\sqrt{2^n}} \sum_{x\in\{0,1\}^n} (-1)^{f(\mathbf{x})} |\mathbf{x}\rangle\,|-\rangle \tag{5.91}$$

Se desea rotar $|+\rangle$ hacia la desconocida x_0. Una rotación es producto de dos reflexiones. Para realizar la rotación Rot que mueve $|+\rangle$ hacia $|x_0\rangle$, primero se refleja en el hiperplano ortogonal a $|x_0\rangle$. El estado $|-\rangle$ en el registro inferior se utiliza para aplicar la fase negativa. Para mostrar la forma en que esta operación afecta a la amplitud, debe notarse que el registro superior de $|\varphi_2\rangle$ consiste en una superposición de todos los posibles valores para x, todos con la misma amplitud $\frac{1}{\sqrt{2^n}}$, debido a que el estado se inicializó de manera balanceda. Luego, el bloque amplificará la amplitud de $|x_0\rangle$, mientras suprime las $(N-1)$ amplitudes restantes por igual. Esto permite reescribir al estado $|\varphi_2\rangle$ de una manera que se ajuste a amplitudes desiguales, lo que permitirá que sea válido en todas las iteraciones, no solo en la primera:

$$|\varphi_2\rangle = \Big(\alpha\,|x_0\rangle + \beta \sum_{x\neq x_0} |\mathbf{x}\rangle \Big) |-\rangle \tag{5.92}$$

Tanto α como β son cantidades que irán cambiando su valor en cada iteración del ciclo. Dado que el valor que tienen al inicio es un número real, $\frac{1}{\sqrt{2^n}}$, y que las dos transformaciones que componen el ciclo únicamente cambian signos y magnitudes, se trata de números reales, para los que se tendrá la condición de normalización $\alpha^2 + (2^n - 1)\beta^2 = 1$. También cabe señalar que solo hay un β para todos los $N-1$ vectores que no son el objetivo. Esto se debe a que las amplitudes de los vectores que no son el objetivo comienzan siendo iguales y el ciclo las amplifica o suprime por igual, por lo que es posible escribir $|\varphi_3\rangle$ de una manera que sea válida en todas las iteraciones:

$$|\varphi_3\rangle = \left(\alpha\,|x_0\rangle + \beta \sum_{x\neq x_0} |\mathbf{x}\rangle\right)|-\rangle \tag{5.93}$$

Cuando esto se envía a U_f, debido al traslado de fase:

$$|\varphi_{3a}\rangle = \left(\alpha(-1)^{f(x_0)}\,|x_0\rangle + \beta \sum_{x\neq x_0} (-1)^{f(x)}\,|\mathbf{x}\rangle\right)|-\rangle \tag{5.94}$$

La condición de búsqueda establece que $f(x_0) = 1$ y $f(x) = 0$ para $x \neq x_0$, por lo que:

$$|\varphi_{3a}\rangle = \left(\alpha(-1)\,|x_0\rangle + \beta \sum_{x\neq x_0} |\mathbf{x}\rangle\right)|-\rangle \tag{5.95}$$

El promedio después de aplicar U_f será $\bar{x} = (-\alpha + (2^n - 1)\beta)/N$. Al aplicar la inversión respecto de la media se obtendrá:

$$|\varphi_{3a}\rangle = \left((2\bar{x} + \alpha)\,|x_0\rangle + (2\bar{x} - \beta) \sum_{x\neq x_0} |\mathbf{x}\rangle\right)|-\rangle \tag{5.96}$$

La comparación de $|\varphi_{3a}\rangle$ y $|\varphi_3\rangle$ permite establecer una relación de recurrencia para las amplitudes:

$$\alpha_{i+1} = 2\bar{x} + \alpha_i \qquad \beta_{i+1} = 2\bar{x} - \beta_i \tag{5.97}$$

sustituyendo el promedio

$$\alpha_{i+1} = 2[-\alpha_i + (2^n - 1)\beta_i]/2^n + \alpha_i \tag{5.98}$$

de la condición de normalización $\beta_i = \sqrt{\frac{1-\alpha_i^2}{2^n-1}}$,

$$\alpha_{i+1} = 2\frac{\sqrt{2^n-1}}{2^n}\sqrt{1-\alpha_i^2} + \alpha_i(1 - \frac{2}{2^n}) \tag{5.99}$$

usando la sustitución trigonométrica $\cos(\phi) = 1 - \frac{2}{2^n}$, $\sin(\phi) = \sqrt{1-\cos^2(\phi)} = \sqrt{1-(1-\frac{2}{2^n})^2} = 2\frac{\sqrt{2^n-1}}{2^n}$. Por tanto:

$$\alpha_{i+1} = \sin(\phi)\sqrt{1-\alpha_i^2} + \alpha_i \cos(\phi) \tag{5.100}$$

y luego en el hiperplano ortogonal a $|+\rangle$, como en la figura, lo cual es válido para rotar cualquier vector $|v\rangle$ hacia $|x_0\rangle$.

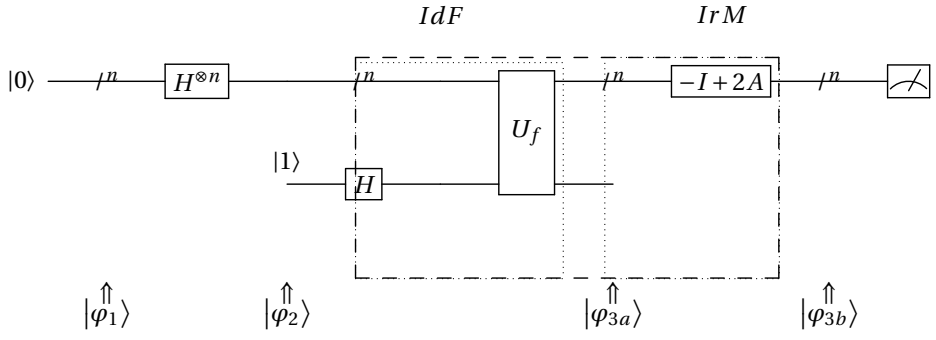


Figura 5.12: Circuito para el algoritmo de Grover. La parte enmarcada entre guiones debe repetirse $\sqrt{2^n}$ veces, está compuesta, a su vez, por dos subprocesos, la inversión, o traslado, de fase (IdF) y la inversión respecto a la media (IrM)

Es importante notar que, debido a que el operador de difusión produce una rotación constante, realizar una mayor cantidad de iteraciones no necesariamente mejorará la aproximación del resultado, se debe realizar la medición en el número de iteración óptimo, de lo contrario, la probabilidad de encontrar el elemento marcado disminuirá. Esto se conoce como «sobrecocción»[43]. Además, el algoritmo original de Grover no suele transformar exactamente el vector de estado de la computadora cuántica al estado marcado o, en otras

palabras, la búsqueda no tiene un éxito del 100 por ciento. Para evitar esto, se requiere que en la última iteración, el operador de búsqueda estándar de Grover sea reemplazado por uno que utilice un paso menos para que el vector de estado de la computadora cuántica sea exactamente el estado marcado. Además, en muchas aplicaciones del algoritmo de búsqueda cuántica, es necesario utilizar específicamente un algoritmo de búsqueda cuántica que tenga pasos más cortos [130].

6

Algoritmo de Shor (Factorización de enteros)

6.1. Introducción

Se trata de uno de los desarrollos mas notables en cómputo cuántico. Fue descubierto por Peter Shor en 1994, como una propuesta para resolver el problema de factorización de números grandes. Aunque en apariencia se trata de una cuestión de interés académico, ha cobrado importancia e interés debido a que la factorización eficiente de enteros grandes permite vulnerar protocolos modernos de seguridad. Por ejemplo, el sistema de criptografía RSA es un protocolo de clave pública ampliamente utilizado en la industria y el gobierno para cifrar información confidencial. La seguridad de RSA se basa en el supuesto de que a las computadoras les resulta *difícil* descomponer a un número entero grande en sus factores, en el sentido de que ningún algoritmo clásico puede encontrar los factores en un tiempo que sea polinómico en n, para un número de n bits. De encontrarse dicho algoritmo, el sistema RSA estaría comprometido, por lo que el descubrimiento de que una computadora cuántica sería, en principio, capaz de factorizar grandes números eficientemente, despertó un gran entusiasmo, disparando tanto el interés en el potencial de la computación cuántica y trajo consigo sustanciales inversiones tanto de la iniciativa privada como de distintos gobiernos en todo el mundo.

El algoritmo de Shor se basa en el siguiente hecho: el problema de factorización se puede reducir a encontrar el período de una determinada función. Una explicación detallada del algoritmo requiere algunas nociones de la Teoría de Números.

6.2. Criptografía

La criptografía es el estudio de los métodos para garantizar que la comunicación entre dos partes es segura[243]. Esto significa que las partes interesadas quieren enviar y recibir mensajes entre sí, evitando la posibilidad de que un tercero pueda entender los mensajes y que la información que contienen caiga en manos equivocadas. La privacidad se provee a través de *métodos de encriptación*, en los que se toma el *mensaje*, (que puede ser algún texto, datos numéricos e incluso un programa ejecutable, en general, cualquier tipo de información) en forma de *texto simple*, es decir, información que, al menos en principio, cualquier persona podría leer, entender, utilizar o ejecutar, y se le convierte en un *texto cifrado* o *encriptado*, es decir, información que sólo un grupo de personas, las autorizadas, podrán convertir nuevamente en texto simple y conocer su contenido. Los primeros sistemas de criptografía tenían la característica de ser ellos mismos secretos. En el momento en el que se conocía la forma en que trabajaba, era sencillo descifrar los mensajes que pretendía ocultar. La fortaleza de un método de encriptación no puede radicar en que la forma en que trabaja no se conozca. Por el contrario, es mejor asumir que los algoritmos de cifrado y descifrado son de dominio público, ya que eventualmente podría revelarse la manera en que operan. El secreto debe residir en los parámetros de los algoritmos, denominados *claves*.

El *cifrado*, consiste en transformar al texto simple P en texto encriptado C. Esto se obtiene a través de la ejecución de un algoritmo o función de cifrado E. Para que este algoritmo opere, se le deben proporcionar los parámetros con el mensaje original, es decir, el texto simple P y la clave de cifrado K_E, es decir $C = E(P, K_E)$. Para que una persona autorizada pueda entender el contenido del mensaje, será necesario volver a obtener el

texto simple P, utilizando el algoritmo de descifrado D, al que se le asignan el mensaje cifrado C y la clave de descifrado K_D como parámetros, lo que significa $P = D(C, K_D)$ [273]. La figura 6.1 muestra la relación entre los elementos de cifrado y descifrado.

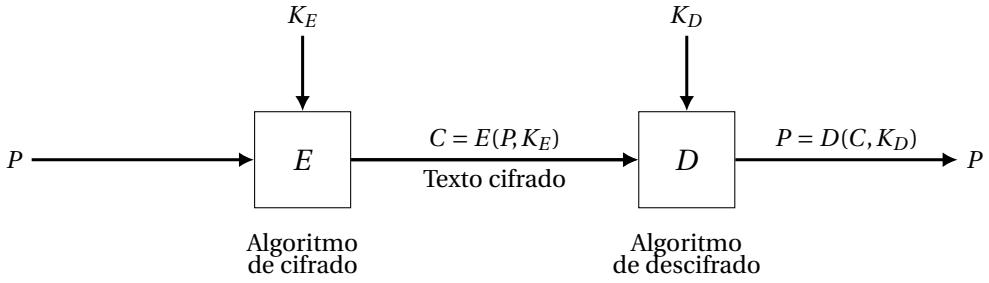


Figura 6.1: Elementos del cifrado y descifrado de un mensaje. El texto simple P ingresa en el algoritmo E. Mediante la clave de cifrado K_E es transformado en el texto cifrado C. Cuando este texto cifrado ingresa al algoritmo de descifrado, con la clave de descifrado K_D se recupera el texto simple P [273].

La encriptación es muy antigua. En los primeros esquemas de encriptación, los algoritmos de cifrado y descifrado dependían de la misma clave K, por lo que son conocidos como *métodos simétricos*. Sería hasta 1976 que se desarrollaría un esquema que permite usar claves distintas, introduciendo los llamados *métodos asimétricos*. Por ello, hoy en día se cuenta con dos sistemas de criptografía [50]:

- **De clave secreta o simétrico.** Su característica principal es que se usa una clave única para cifrar y descifrar datos. Una de las ventajas que poseen es su eficiencia, dado que la cantidad de operaciones matemáticas que se necesarias para cifrar o descifrar un mensaje es pequeña (en comparación con otros esquemas). Su principal desventaja radica en que tanto el emisor como el receptor deben compartir la clave. Un ejemplo de esto es el algoritmo en el que cada letra se sustituye por otra. Todo el mundo puede conocer como trabaja el sistema, pero la clave radica en conocer el valor de cada letra. Si alguna persona no autorizada adquiere por el medio que sea la clave, será capaz de conocer el contenido del mensaje. El hecho de que la clave se deba compartir implica el riesgo de que sea interceptada.
- **De clave pública o asimétrico.** Indica que utilizan distintas claves para el cifrado y el descifrado. La clave de cifrado puede publicarse, de manera que todo el mundo la conozca y, con ella, cualquier persona puede cifrar un mensaje, pero solo el poseedor de la clave de descifrado podrá recuperar el contenido del mensaje, por lo que hay que mantenerla oculta. La clave pública no sirve para descifrar, y, si se elige de manera apropiada, es casi imposible descubrir la clave de descifrado correspondiente. Como desventaja, las operaciones que involucra suelen requerir cálculos mas complejos y resultan menos eficientes.

Además de proporcionar confidencialidad, la criptografía también aborda otros problemas importantes relacionados con la comunicación[73]:

1. Integridad de los datos: La criptografía garantiza que los datos no se alteren durante la transmisión. Si alguien intenta modificar el mensaje cifrado, se detecta. Esto es crucial para evitar manipulaciones accidentales o maliciosas.
2. Autenticación:
    - del origen de datos: Permite al receptor verificar la fuente del mensaje. Por ejemplo, las firmas digitales aseguran que el remitente es quien dice ser.
    - de entidad: Al iniciar una comunicación, las partes involucradas pueden identificarse mutuamente. Esto evita que alguien pretenda ser otra persona.

3. No repudio: La criptografía permite demostrar que un mensaje fue enviado por un remitente específico. Así, el remitente no puede negar posteriormente haberlo enviado.

A diferencia de la encriptación, que tiene una larga historia, la necesidad de verificar integridad y autenticación no habían surgido hasta el uso pleno de comunicaciones electrónicas, puesto que los documentos en papel permitían verificar la ausencia de manipulación e integridad, mientras las firmas garantizaban autenticación y no repudio. En la actualidad, se han diseñado medidas para asegurar la integridad de los mensajes, a través del llamado *Código de autenticación de Mensajes* (MAC, por sus siglas en inglés), mientras que las *firmas digitales* procuran asegurar la autenticación y no repudio que garantizaría una firma escrita a mano. En la práctica, antes de que se pueda firmar un mensaje, se debe pasar a través de una *función de resumen*, (el término usado en inglés es «hash», cuya traducción literal equivale a hacer picadillo), con el objetivo de comprimir al mensaje, de longitud arbitraria, en un resumen del mensaje, un texto corto, de apariencia aleatoria y de longitud fija. La función de resumen debe considerarse de dominio público [263].

Los algoritmos de cifrado y descifrado, las funciones resumen criptográficas o los generadores pseudoaleatorios son los componentes básicos para resolver problemas relacionados con la privacidad, la autenticación o la integridad de los datos. En muchos casos, un solo componente no basta para dar solución a un problema dado, sino que es necesario combinarlos, a través de la ejecución de una serie de pasos realizar una tarea determinada. Tal serie de pasos, que involucra una secuencia específica de mensajes intercambiados entre dos (o posiblemente más) partes, recibe el nombre de *protocolo criptográfico* [73].

La seguridad de un sistema criptográfico de llave pública está basada en la existencia de parejas de problemas computacionales, inversos entre si, tal que uno de ellos sea muy sencillo de resolver, mientras el otro carezca de un algoritmo eficiente para ser resuelto. En términos matemáticos, se basa en una *función de una vía* o *función trampilla* f, tal que, para una x dada el valor correspondiente a *f(x)* puede calcularse de manera eficiente pero, obtener x a partir de únicamente $f(x)$ no es factible. La definición para función de una via puede escribirse en términos del uso de algoritmos polinomiales probabilísticos. La probabilidad de invertir exitosamente la función mediante un algoritmo polinomial probabilístico es insignificante, y se entiende por insignificantemente pequeña que es asintóticamente menor que cualquier límite polinomial dado. Entre los ejemplos mas importantes se encuentra el problema de factorización, que se discutirá a continuación con mayor detalle.

6.3. Aritmética modular

Los sistemas de cifrado, tanto simétricos como asimétricos están basados en aritmética con un número finito de elementos. La aritmética modular es una forma simple de desarrollar aritmética en un conjunto finito de N enteros, mediante la identificación de todos los números enteros que difieren en múltiplos de N. Debido a ello, solo se tendrán N cantidades distintas, que pueden ser representadas por $0, 1, \ldots, N-1$. Esta identificación se precisa mediante la definición de la relación de congruencia.

> Sea m un entero positivo. Entonces un entero a es congruente con un entero b módulo m si m divide a $(b-a)$. En símbolos, se escribe $a \equiv b \pmod{m}$; m es el módulo de la relación de congruencia [210].

Por ejemplo, $30 \equiv 2 \pmod 7$ ya que $30 - 2 = 28$ (un múltiplo de 7). Otros ejemplos serían: $30 \equiv 23 \pmod 7$, $23 \equiv 16 \pmod 7$, $9 \equiv 2 \pmod 7$.

La relación de congruencia resulta ser una relación de identidad. Conceptualmente, la relación de congruencia nace de la observación de los residuos al dividir distintos números entre una cierta cantidad dada, en este caso N. El algoritmo de Euclides garantiza que estos residuos nunca serán mayores a N. Luego $a \equiv b \pmod m \iff a$ y b dejan el mismo residuo al dividirse entre m. Además, si r es el residuo de dividir a a entre m, entonces $a \equiv r \pmod m$. En el caso de que $0 \leq r < m$, r recibe el nombre de *residuo mínimo* [152]. Esto es útil porque permite disminuir el tamaño de los números resultantes de una determinada operación aritmética. Por ejemplo:

$$30 \equiv 2 \pmod 7 \implies 5 \times 6 \equiv 2 \pmod 7 \tag{6.1}$$

$$16 \equiv 2 \pmod 7 \implies 2^4 \equiv 2 \pmod 7 \tag{6.2}$$

Es posible indicar esta reducción módulo m para una función f de la siguiente forma:

$$f(n) \pmod m = r \text{ si: } f(n) \equiv r \pmod m \text{ y } 0 \leq r < m \tag{6.3}$$

El tipo de operación que interesa en criptografía es encontrar las potencias de a reducidas módulo N [213]. Por ejemplo, si $a = 2$ y $m = 15$ se tiene

x	0	1	2	3	4	5	6	7	8	9
a^x	1	2	4	8	16	32	64	128	256	512
$f(x) = a^x (\text{mod } m)$	1	2	4	8	1	2	4	8	1	2

mientras que , si $a = 4$ y $m = 15$ se tiene

x	0	1	2	3	4	5	6	7	8	9
a^x	1	4	16	64	256	1024	4096	16384	65536	262144
$f(x) = a^x (\text{mod } m)$	1	4	1	4	1	4	1	4	1	4

Para ilustrar la manera en que esta operación puede utilizarse para encriptar información, puede considerarse el siguiente ejemplo. Dado que $2^{21} = 2097152 \equiv 2 (\text{mod } 15)$, se tiene

$$2^{21} (\text{mod } 15) = 2 \tag{6.4}$$

Puede verse que en ambos lados de la relación de congruencia aparece el dos. Este será el mensaje a. El proceso de elevar este número a la potencia 21 (módulo 15) equivale a codificar y decodificar la información, pero, para que resulte útil como método de encriptación, debe realiza por partes. La operación 2^{21} puede escribirse también como $2^{7\times 3}$, de manera que:

$$2^{21} (\text{mod } 15) = 2^{7\times 3} (\text{mod } 15) = 2. \tag{6.5}$$

Si se toma $b = 2^7 (\text{mod } 15) = 128 (\text{mod } 15) = 8$, se tendrá una forma de cifrar el mensaje a. En este caso, la clave de cifrado c fue el número 7. Todo lo que hará falta para descifrar es completar la operación $b^3 (\text{mod } 15) = 8^3 (\text{mod } 15) = 2 = a$, por lo que la clave de descifrado d es el número 3.

En general, dado que $a^{cd} (\text{mod } X) = a$, el mensaje cifrado es $b = a^c (\text{mod } X)$. El mensaje original puede recuperarse realizando $b^d (\text{mod } X)$. La clave de cifrado es pública, cualquiera que desee encriptar un mensaje puede hacerlo. Pero esta clave es inútil para recuperar el contenido del mensaje. Se requiere la clave privada, que debe mantenerse en secreto. El número X ha de conocerse también, para que pueda realizarse la reducción de la potencia en módulo X al momento de codificar. En este ejemplo, es evidente que X es el producto de dos números primos y es sencillo obtener su factorización. En una implementación real, aunque se conoce el valor de X y se sabe que es el producto de primos, X es un número muy grande, su factorización desconocida y no existe algoritmo capaz de obtenerla de manera práctica.

Un detalle que se ha omitido para hacer más ágil la explicación del mecanismos de codificación y decodificación es la existencia de una relación entre las claves de cifrado c, descifrado d y el enigmático número X. Sucede que $a^{cd} (\text{mod } X) = a$ si $X = pq$ con p y q primos y $cd = 1 + s(p-1)(q-1)$. Es posible explicar esto en términos de la Teoría de los Números (por ejemplo, en el lema 7.2 y el teorema 7.3 de [152]), sin embargo, una ruta alternativa y algo mas corta se basa en Teoría de Grupos (aquí, se sigue la extremadamente suscita explicación que aparece en en el texto principal de [191]).

6.3.1. Grupos y el Pequeño Teorema de Fermat

Un grupo es un conjunto G, cerrado bajo alguna operación binaria $*$ que satisface los axiomas de asociatividad de $*$, la existencia de elemento idéntico para $*$ y la existencia de un inverso para cada elemento del conjunto[95]. En el caso del cifrado RSA, se requieren grupos finitos, cuya operación de grupo esté definida como el producto reducido en módulo N, donde N es algún entero fijo. Dicha operación se denotará por $\circledcirc_N$ y se define como :

$$a \circledcirc_N b = ab (\text{mod } N) \tag{6.6}$$

lo cual significa calcular el producto y luego obtener el residuo mínimo. Por ejemplo, si N=7 y se toman los enteros 1, 2, 3, 4 y 5

$$1 \circledcirc_7 5 = 5 \quad 2 \circledcirc_7 4 = 8 (\text{mod } 7) = 1 \quad 3 \circledcirc_7 4 = 12 (\text{mod } 7) = 5 \quad 4 \circledcirc_7 5 = 20 (\text{mod } 7) = 6 \tag{6.7}$$

Dado un entero N, es posible construir el conjunto $G_N^{\circledcirc}$ compuesto por los enteros x que son primos relativos con N y menores que él, con la operación producto reducido en módulo N.

$$G_N^{\circledcirc} = \{x : (x, N) = 1 \text{ y } x < N\} \tag{6.8}$$

Por ejemplo, para $N = 12$ se tiene $G_{12}^{\circledcirc} = \{1,5,7,11\}$ y para $N = 7$ se tiene $G_{12}^{\circledcirc} = \{1,2,3,4,5,6\}$. Es posible demostrar que $G_N^{\circledcirc}$ es un grupo ([152],[191]).

Dado un grupo arbitrario G y un elemento a dentro de el, se define al *orden* k de a como la potencia mínima que, bajo la operación de grupo, satisface $a^k = 1$. En el caso de $G_N^{\circledcirc}$, se busca la mínima potencia tal que $a^k (\text{mod } N) = 1$. Por ejemplo, si $a = 3 \rightarrow 3^6 (\text{mod } 7) = 729 (\text{mod } 7) = 1$, por lo que el orden de 3 es 6.

Una propiedad importante todo grupo es que el orden k es un divisor de la cantidad de elementos de G, es decir, k divide al la cardinalidad de G. Este hecho resulta de interés porque, si se construye un grupo con la operación $\circledcirc$ sobre un primo, $G_p^{\circledcirc}$ tendrá $p-1$ elementos, por lo que resultará posible expresar a $p-1$ como un múltiplo de k, es decir $p-1 = bk$. De ello se sigue que $a^{p-1} (\text{mod } p) = a^{bk} (\text{mod } p) = 1$. El hecho de que $a^{p-1} (\text{mod } p) = 1$ o, equivalentemente, $a^{p-1} \equiv 1 (\text{mod } p)$ es conocido como *Pequeño Teorema de Fermat* ([152] para un enfoque basado en la Teoría de Números, ver teorema 7.3). Para utilizar este teorema en criptografía, es necesario llevarlo a la expresión $a^{1+s(p-1)(q-1)} \equiv a (\text{mod } pq)$, donde p y q son dos números primos diferentes y a no es divisible entre ellos. En tal caso, dada la unicidad de la factorización canónica, ninguna potencia de a resultará divisible por ellos. Puede elegirse una potencia conveniente, tal como a^{q-1}, para obtener un número que no será divisible por p y, en tal caso, el Pequeño Teorema de Fermat es aplicable: $[a^{q-1}]^{p-1} \equiv 1 (\text{mod } p)$. De manera análoga, dado que q no divide a a, ninguna potencia de a, por ejemplo $a^(p-1)$, será divisible entre q, por lo que es posible aplicar el Pequeño Teorema de Fermat y obtener que $[a^{p-1}]^{q-1} \equiv 1 (\text{mod } q)$. Para el paso final, debe recordarse el significado congruencia, que resulta ser «más que solo una notación conveniente» [210], se trata de una afirmación acerca de la divisibilidad:

- $[a^{q-1}]^{p-1} \equiv 1 (\text{mod } p)$ equivale a decir que $a^{(q-1)(p-1)} - 1$ es múltiplo de p,
- $[a^{p-1}]^{q-1} \equiv 1 (\text{mod } q)$ equivale a decir que $a^{(p-1)(q-1)} - 1$ es múltiplo de q.

Dado que p y q son primos distintos, $a^{(p-1)(q-1)} - 1$ es múltiplo de ambos, por lo que $a^{(p-1)(q-1)} \equiv 1 (\text{mod } pq)$. A partir de esta última relación es inmediato que

$$a^{1+s(p-1)(q-1)} \equiv a (\text{mod } pq) \tag{6.9}$$

que resulta útil porque se mantiene aún si a es divisible por p o q [191]. Para concluir, debe tomarse c de tal forma que no tenga factores en común con $(p-1)(q-1)$, es decir, $c \in G_{(p-1)(q-1)}$. Dado que se trata de un grupo, existe d, el inverso de c bajo la operación de grupo, de tal forma que $c \circledcirc_{(p-1)(q-1)} d = 1$, es decir, que $cd \equiv 1 (mod\ (p-1)(q-1))$ de donde pueda escribirse $cd = 1 + s(p-1)(q-1)$ para alguna s entera. Se recordará que el mecanismo de codificación y decodificación requería de una expresión de la forma

$$a^{cd} \equiv a (\text{mod } pq) \tag{6.10}$$

6.3.2. Cifrado y descifrado de mensajes

Con todo lo anterior, se tiene el mensaje a y un mecanismo para recuperarlo, a través de $X = pq$. Si la expresión anterior se descompone en dos etapas, será posible codificar y, posteriormente, decodificar la información, es decir, a través del uso de la clave pública y la privada. Para determinarlas, es necesario conocer a los factores de X, por separado y deben mantenerse en secreto, porque de otra forma se podrá descifrar la información. Por otra parte, aunque X es público, no existe, al momento, un algoritmo que permita factorizar enteros de manera eficiente, por lo que resulta imposible obtener a p y q a partir de él. Para quien conozca p y q, es sencillo calcular $(p-1)(q-1)$ y, a partir de él, tomar cualquier c del grupo. Dado c, se garantiza la existencia del inverso correspondiente d, que también puede calcularse. Esto puede entenderse mejor con el siguiente ejemplo:

- Para generar el par de claves:
  - Seleccionar dos números primos p y q , por ejemplo $p = 3, q = 5$
  - Calcular el producto $X = pq$, en este caso $X = 3 \times 5 = 15$
  - Elegir un entero de codificación c, coprimo con $(p-1)(q-1)$. Dado que $(p–1)(q–1) = 8$, se puede tomar $c = 7$

  - Buscar d, un inverso multiplicativo de $c(\text{mod } (p-1)(q-1))$. Puede tomarse, por ejemplo $d = 23$, ya que $7 \times 23(\text{mod } 8) = 161(\text{mod } 8) = 1$
  - Se publican X y c, se guarda d.
- Para codificar un mensaje:
  - Se codifica el mensaje como cadena de dígitos, por ejemplo, sea $a = 2$.
  - Usando c y X, se calcula $b = a^c(\text{mod X})$. En este caso $2^7(\text{mod } 15) = 128(\text{mod } 15) = 8$
  - Se envía el mensaje 8.
- Finalmente, para descifrar:
  - Al recibirlo se calcula $a = b^d(\text{mod } X)$, es decir $8^{23}(\text{mod } 15) = 2$
  - Si se pudiera encontrar p y q a partir de X, se podría calcular $(p-1)(q-1)$ y encontrar el entero decodificador d correspondiente.

Aparte de la necesidad de almacenar en secreto el valor de los números primos p y q hay otra posible forma de conocer el contenido del mensaje. El inverso multiplicativo en $G_X^{\circledcirc}$ no es único. Esto significa que, aunque se guarde celosamente el valor del número que sirve para decodificar, en realidad existen otros que pueden realizar el trabajo. En el ejemplo mostrado, puede usarse a 31 en lugar de 23 y volverá a obtenerse a. De hecho, cualquier otro número de la forma $23 + 8s$, con s entero, funcionará como clave para decodificar el mensaje, dada la periodicidad de la función $f(x) = a^x(\text{ mod} m)$. El espacio entre ellos, el periodo, puede permitir la decodificación sin conocer explícitamente p y q. Dado que se ha seleccionado a c de tal forma que no tenga factores en común con $(p-1)(q-1)$, y dado que el periodo r divide al orden $(p-1)(q-1)$ de $G_{pq}^{\circledcirc}$, el entero codificador c no tiene factores en común con r, luego se tendrá que $c \equiv c' mod r$, para alguna c' miembro de $G_r^{\circledcirc}$. Dado que este último es un grupo, existirá d', el inverso de c', que a su vez, también será inverso de c, es decir $cd' \equiv 1 mod r$

En aplicaciones prácticas, obtener el periodo, o cualquiera de los números decodificadores sin la factorización de X resulta igual de complicado que el problema original. No se conocen algoritmos clásicos eficientes para realizar ninguna de esas tareas. Pero un algoritmo cuántico puede encontrar el periodo de una manera eficiente. A continuación, se indica la manera en que un problemas e transforma en el otro.

6.4. Descripción del problema

Dado que la seguridad del protocolo RSA está basada en la imposibilidad de encontrar a los números p y q a partir de X, el problema que resuelve el algoritmo de Shor es la descomposición de números enteros en sus factores primos, o de manera abreviada, factorización. Además de la importancia para la criptografía, el problema de factorización ha estado en el centro de atención de la Teoría de Números[13]. Parece imposible encontrar mejores palabras para describir este problema, que las que se pueden encontrar en las *Disquisitiones arithmeticae*, de Carl Friedrich Gauss:

> “El problema de distinguir números primos de números compuestos y de resolver estos últimos en sus factores primos es conocido como uno de los más importantes y útiles en aritmética. Ha ocupado la industria y la sabiduría de geómetras antiguos y modernos a tal grado que sería superfluo discutir el problema detenidamente. No obstante debemos admitir que todos los métodos que han sido propuestos hasta ahora son o restringidos a casos muy especiales o tan laboriosos y prolijos que aún para números que no exceden los límites de tablas construidas por hombres estimables, i.e., para números que no requieren métodos ingeniosos, ponen a prueba la paciencia hasta de los calculistas experimentados. Y estos métodos a duras penas pueden ser usados para números grandes. Aún cuando las tablas, que están disponibles para quien quiera y las cuales esperamos continuarán siendo extendidas, son realmente suficientes para la mayoría de los casos ordinarios, frecuentemente sucede que el calculista entrenado obtendrá la suficiente ganancia de la reducción de números grandes a sus factores de modo que esto lo compensará por el tiempo consumido.” ”

> Luego, la dignidad de la ciencia misma parece requerir que todos los medios posibles para la solución de un problema tan elegante y tan célebre sean explorados. Por esta razón, no dudamos que los dos métodos siguientes, cuya eficacia y brevedad podemos confirmar a partir de una larga experiencia, resultarán gratificantes a los aficionados a la aritmética. Está en la naturaleza del problema que cualquier método se hará más elaborado a medida que los números se hacen mayores. No obstante, en los siguientes métodos, las dificultades crecen algo lentamente, y números con siete, ocho o aún más dígitos han sido manipulados con éxito y rapidez más allá de la esperada, especialmente por el segundo método. Las técnicas que se conocían hasta ahora requerirían un trabajo intolerable aún para el calculista más infatigable [106].

Hoy en día se dispone del «calculista más infatigable», literalmente, se trata de una máquina construida con ese propósito específico y, sin embargo, el trabajo seguirá siento «intolerable», en la medida que los tiempos de ejecución del algoritmo escrito para tal fin no se encuentre en orden polinomial, de manera que la creación de la computadora no zanjó la cuestión.

6.4.1. Pruebas de primalidad

La primera parte del problema mencionado por Gauss, la búsqueda de algoritmos que corran en tiempo polinomial para identificar si un número es primo se completó, en cierta forma, en la década de 1970 cuando aparecieron un par de propuestas de dos métodos muy eficientes para probar primalidad [232], [259]. La característica común de estos algoritmos es el empleo de métodos aleatorios, por lo que entran en la categoría de algoritmos aleatorios. Para ambos algoritmos se cumple lo siguiente:

- Si la entrada es un número primo, la salida es 0.
- Si la entrada es es un número compuesto, la salida es 0 o 1. La probabilidad de que el resultado sea 0 está acotada por 1/2.

Para una entrada n, ambos algoritmos utilizan como máximo $c \log n$ operaciones aritméticas en números inferiores a n^2, para alguna constante c. Si la salida es 1, el número de entrada es definitivamente compuesto, caso en que se afirma que el cálculo prueba que n es compuesto y arroja un certificado por ese hecho. Si el resultado es 0, en realidad no es posible saber si el número es un primo o no. Si bien, una cota de error de hasta un medio puede parecer insatisfactoria, es posible repetir el algoritmo hasta g veces, gastando así $lc \log n$ operaciones aritméticas en la entrada n, el límite de error se puede reducir a $\frac{1}{2^g}$, para g arbitrario [76].

Estos algoritmos aleatorios, entre otros, resultan suficiente para dar por resuelto el el problema de determinar la primalidad para n lo suficientemente grande para todo propósito práctico, y son los que se utilizan en las aplicaciones. En la práctica, no existe razón para preocuparse por el riesgo de que un 0 resulte de una entrada compuesta, siempre que el límite de error se haya ajustado a las necesidades específicas de la aplicación y que el algoritmo realmente se comporte de manera aleatoria. La probabilidad de error resulta insignificante si se le compara con otros riesgos de error debidos a software o hardware que, de cualquier forma, son inevitables en un sistema informático real.

A pesar de resultar útil en el aspecto práctico, desde el punto de vista teórico, persistía la cuestión de si era posible encontrar un algoritmo sin errores que permitiera resolver el problema de primalidad con un cota de tiempo mínima. En 2002, Agrawal, Kayal, y Saxena propusieron una prueba determinista de tiempo polinomial [3] (*prueba AKS*, para abreviar) para comprobar la primalidad, que no depende de conjeturas matemáticas cuya demostración aún no se tiene, a diferencia del algorimo de Miller, que requiere de la hipótesis generalizada de Riemann [193]. Aparentemente, nadie se sorprendió de que una prueba de primalidad que corriera en tiempo polinomial existiera [80], pero lo que si resultó ser una gran sorpresa fue la relativa simplicidad tanto del algoritmo como de su demostración, posiblemente el algoritmo mas corto conocido (solo 6 declaraciones) para un problema tan importante, que permaneció sin resolver por tanto tiempo [302]. La clave detrás del funcionamiento de la prueba AKS se encuentra, de hecho, en una extensión del pequeño teorema de Fermat [3], donde si x es indeterminado y a y n son *primos relativos* [1] con $n > 1$, entonces n es primo $\iff (x-a)^n \equiv (x^n - a) (\text{mod } n)$. Una exposición detallada de la prueba, junto con las

[1] También llamados *coprimos*, se refiere a un número que no tiene divisores comunes, lo que equivale a decir que el entero mas grande que divide tanto a a como a n es 1. En general, el entero más grande que divide a dos números enteros a y b se denomina *máximo común divisor* de a y b, y se denota como $MCD(a,b)$, luego, la condición para que un par de números sean primos relativos es que $MCD(a,b) = 1$

demostraciones que conducen al algoritmo, así como el contexto histórico y referencias adicionales, pueden encontrarse en [113].

6.4.2. Reducción del problema de factorización

Si la primera parte del problema mencionado por Gauss ha sido *efectivamente* resuelta, en el sentido de que existe un algoritmo polinomial determinista para tratarlo, la segunda parte no se resuelve de manera trivial ni automática a partir de la primera. La existencia de un algoritmo que permita saber si un entero dado es primo no implica que se tenga una manera eficiente para separar a ese entero en sus componentes. En resumen, el problema de descomponer un número en sus factores primos, que por otra parte vulneraría el sistema de encriptación RSA, puede enunciarse como sigue:

Problema de factorización de números enteros
Entrada: Un número entero $N > 1$.
Tarea: Generar el conjunto de números enteros positivos $P_1, P_2, ..., P_l, r_1, r_2, ..., r_l$ tales que

$$N = \prod_{i=1}^{l} P_i^{r_i}$$

con P_i números primos distintos

El mecanismo para realizar esta separación se conoce, pero resulta ser, a la vez, sencillo y poco práctico. En esencia, se trata de usar fuerza bruta, intentando divisiones sucesivas entre cada uno de los números primos menores que la cantidad objetivo. Los que arrojen un residuo cero se toman como factores, los que no, se descartan. Dado que en la práctica , N podría ser un número grande, tal vez de cientos de dígitos, el proceso se vuelve lento, por lo que algunas mejoras podrían a ayudar a su implementación. Si se desea factorizar un número natural $N > 1$ debe determinarse en primer lugar si se trata de un número primo o no, mediante alguna prueba de primalidad, como la antes comentada. Una vez que se ha comprobado que N no es un número primo, se realiza la *descomposición*, es decir, se debe encontrar un factor no trivial M y recursivamente factorizar al cociente N/M y al factor M, hasta que solo queden números primos y la unidad. No es necesario evaluar todos los números hasta N, basta con investigar hasta $\sqrt{N}$ (un hecho que, de acuerdo con Gauss en la misma obra citada arriba, «todo el mundo sabe»[106], pero que se puede verificar fácilmente [152]), por lo que este algoritmo trivial para la descomposición de N es de $\mathcal{O}(\sqrt{N})$.

La sencillez de este algoritmo, no obstante, le impide resolver el problema en una aplicación de interés real. En 1991 la compañía RSA Security, dedicada a la criptografía y al software de seguridad (fundada por los mismos creadores del protocolo de seguridad RSA), lanzó al público el desafío de factorizar una lista de números con el objeto de mostrar la seguridad de su código. Cada uno de los números fue creado multiplicando dos números primos muy grandes generados aleatoriamente, tras lo cual se destruyó toda huella de ellos. El tamaño de los números variaba desde las 100 cifras decimales del llamado «RSA-100» hasta las 617 cifras decimales del « RSA-617»[61]. Algunos de los primos de esta lista ya han sido factorizados, aunque por otros métodos. Si se deseara factorizar un número de de 400 dígitos por fuerza bruta, será necesario probar uno a uno los primos hasta $\sqrt{N}$), que será un número que tendrá, aproximadamente, 200 dígitos. Suponiendo que cada átomo del universo entero fuera una computadora (es decir, disponiendo de 10^{90} equipos) cada uno con la capacidad de revisar 10^{40} números primos por segundo (cercano al valor de las computadoras clásicas actuales), entonces se requeriría de un tiempo $\frac{10^{200}}{10^{90}\times 10^{40}} = 10^{70}\,s$, valor que supera por mucho la edad del universo (alrededor de $10^{20}\,s$)[70]. Incluso con los mejores algoritmos clásicos disponibles, factorizar un número de 232 dígitos requirió alrededor de 1000 núcleos y 2 años de cálculos, presentandose en el 2010[149]. Creer que la distancia entre 232 bits y 400 bits es algo que se alcanzará eventualmente es un tanto ingenuo. El siguiente hito llegaría hasta 2019 y se trata de la factorización de un número de 240 digitos[40].

Dado que la intención es factorizar un número que se sabe que no es primo, el siguiente paso más simple consiste es retirar todos los factores de dos, por lo que puede suponerse que N es impar. Además, si N es de la forma P^α, es decir, una *potencia perfecta* de un primo , existen algorimos que lo determinan [14], y, además, la prueba de primalidad de AKS también tiene la capacidad de hacerlo, por lo que es posible suponer que la factorización de N contiene al menos dos factores primos impares distintos. Si fuera posible descomponer cualquier número entero impar que no sea potencia perfecta de un primo en dos factores no triviales, entonces sería posible factorizar completamente N en sus factores primos utilizando a lo sumo $\log N$ separaciones.

Al realizar la separación de un número en dos factores, se tendrá que actuar recursivamente en cada uno de ellos, evaluando si son primos, potencia de primos o números compuestos y, cuando corresponda, se deberá descomponer nuevamente. En principio, esto puede realizarse a través de la prueba de primalidad AKS, sin embargo, posee una gran desventaja. A pesar de ser determinística en tiempo polinomial tiene un grado bastante alto ($\mathcal{O}(\sqrt{n}^{13})$) [3]. Por otra parte, existen algoritmos probabilísticos clásicos eficientes para probar la primalidad, como Solovay-Strassen. Este algoritmo no ofrece total seguridad de que un número sea primo, pero da una probabilidad tan alta como se desee [259]. Tiene la ventaja de que se puede llevar a la práctica para números grandes, porque su complejidad es $\mathcal{O}(log_2 n)^2$, siendo n el entero que se desea probar, lo que significa que su complejidad es cuadrática al número de bits[111]. Otro ejemplo de algoritmo probabilista es la prueba de Miller-Rabin, cuya complejidad algorímica resulta aún menor que el anterior. Si el número falla la prueba, se tiene certeza total de que se trata de un número compuesto, en tanto que si la pasa un elevado número de veces, disminuye la probabilidad de que no sea un primo [232]. Los algoritmos deterministas clásicos de tiempo polinomial no pueden competir contra los algoritmos probabilísticos clásicos[118], por lo que estos últimos suelen implementarse en la práctica.

Ya sea que se usen pruebas de primalidad deterministas o no deterministas, estás permitirán establecer el momento en que debe dejar de intentarse la separación de un número en factores y declarar que se trata de un primo. Luego, el problema de factorizar N se puede reducir a $\mathcal{O}(\log(N))$ casos del problema de separar a un entero impar que no es potencia perfecta de un primo [145].

Problema de separar a un entero impar que no es potencia perfeceta de un primo
Entrada: Un número entero impar N que tiene al menos dos factores primos distintos.
Tarea: Generar dos números enteros N_1, N_2 pertenecientes al intervalo ente 1 y N, tales que $N = N_1 N_2$

En la discusión sobre aritmética modular, se mencionó que la función exponenciación modular $f(x) = a^x \bmod N$ es periódica. Esto es importante porque el periodo r de $a^x \bmod N$ coincide con el orden de a, es decir, r es el menor entero tal que $a^r \equiv 1 (\bmod N)$ [21]. Si a se selecciona aleatoriamente, con $MCD(a, N) = 1$, entonces el orden r de a será par con probabilidad al menos $\frac{1}{2}$. En caso de que sea impar, se elije otra a.

- Si r es par, entonces se satisface $a^r \equiv 1 \bmod N$.
- Transponiendo se tiene que $a^r - 1 \equiv 0 \bmod N$,
- Al factorizar se tiene que $(a^{r/2} - 1)(a^{r/2} + 1) \equiv 0 \bmod N$
- Esto significa que N divide al producto $(a^{r/2} - 1)(a^{r/2} + 1) = a^r - 1$
- Dado que N divide $a^r - 1$, existe un entero l tal que $a^r - 1 = lN$, por lo que $(a^{r/2} - 1)$ debe ser, o contener, a un factor de N y lo mismo aplica para $(a^{r/2} + 1)$. En este punto es posible preguntarse si N divide a los factores $(a^{r/2} - 1)$, $(a^{r/2} + 1)$:
  - Por una parte, se sabe que $(a^{r/2} - 1)$ no es congruente con cero en módulo N porque r era el menor entero que satisfacía esa congruencia para el entero a, luego $(a^{r/2} - 1) \not\equiv 0 \bmod N$ y, por tanto, N no divide a $(a^{r/2} - 1)$
  - Respecto a $(a^{r/2} + 1)$ no se puede afirmar algo semejante, pero en caso de que $(a^{r/2} - 1) \equiv 0 \bmod N$ el procedimiento falla porque se obtendrían los factores triviales N y 1. Supóngase por ahora que *se tiene suerte* y $(a^{r/2} + 1) \not\equiv 0 \bmod N$
- En tal caso ni $(a^{r/2} - 1)$ ni $(a^{r/2} + 1)$ son divisibles por $N = pq$, pero el producto si lo es
- Dado que p y q son primos, esto solo será posible solo si alguno de ellos, por ejemplo p, divide a $(a^{r/2} - 1)$ mientras el otro, q, divide a $(a^{r/2} + 1)$
- Dado que los únicos divisores de N son p y q, se sigue que p sería el máximo común divisor de N y $(a^{r/2} - 1)$ mientras que q sería el máximo común divisor de N y $(a^{r/2} + 1)$
- A continuación se calcula $MCD(a^{r/2} - 1, N)$ y $MCD(a^{r/2} + 1, N)$. Si las suposiciones que se han hecho se cumplen, se podrá encontrar a p o a q mediante la aplicación del algoritmo de Euclides.

En síntesis, la esperanza recae en que $MCD(a^{r/2}-1 \bmod N, N)$ sea un factor no trivial de N. Si N tiene al menos dos factores primos distintos, entonces, para un número entero a seleccionado uniformemente al azar con orden r par, la probabilidad de que $MCD(a^{r/2}-1 \bmod N, N)$ sea un factor no trivial de N es, al menos $\frac{1}{2}$. El procedimiento falla si el orden r de a resulta ser impar, porque en tal caso $\frac{r}{2}$ no es entero, o cuando $a^{r/2} \pm 1 \equiv 0 \bmod N$, en donde se obtendrán los factores triviales 1 y N. Cuando el procedimiento falla, debe repetirse con un número aleatorio diferente $a < N$. Por lo tanto, solo es necesario probar un número constante de valores para dividir N con éxito con alta probabilidad.

El problema de factorización de enteros se reduce, mediante una transformación de tiempo polinomial [193], al problema de encontrar el orden de un elemento a primo relativo con N, es decir, a pertenece al grupo $G_N^{\circledcirc}$. De manera que es necesario contar con un mecanismo para encontrar a, es decir, los coprimos de N. Se puede elegir alguna $a < N$ al azar. Si a no es primo relativo con N, entonces el máximo común divisor de a y N, $MCD(a,N)$, será un factor no trivial de N, que puede encontrarse utilizando el algoritmo de Euclides. Por lo tanto, se pueden muestrear uniformemente elementos primos relativos con N muestreando uniformemente $\{2,3,...,N-2\}$ y luego aplicar el algorimo de Euclides para probar si el entero muestreado es primo relativo de N. (Si $MCD(a,N) > 1$, entonces habrá encontrado un factor no trivial de N).

Problema de búsqueda de órdenes
Entrada: Dos enteros a y N tales a y N son primos relativos.
Tarea: Encontrar el orden r de a módulo N (encontrar el periodo r de la función $f(x) = a^x \bmod N$).

Hasta ahora, todos cada paso que permita reducir el problema de factorización al problema de la búsqueda del período, puede realizarse de manera eficiente usando computadoras clásicas. En el caso de la función de exponenciación modular, también existen algoritmos clásicos eficientes [197]. Aunque existe una implementación cuántica que permite calcular la función a^x para todo x en una sola ejecución [16], queda aún algún trabajo por realizar.

Dado el requisito de que a y N sean primos relativos, es posible considerar la cuestión de que tan probable puede ser que esto suceda, si los números son seleccionados al azar. La probabilidad de que dos enteros k_1 y k_2, elegidos de manera uniforme entre 1 y $r-1$, sean primos relativos es $\frac{6}{\pi^2}$. La demostración rigurosa de este hecho requiere establecer con precisión el significado de *aleatorio* y también amerita una discusión sobre algunas matemáticas que están más allá del alcance de este trabajo (la demostración completa de esto puede consultarse en [150]), pero puede hacerse plausible considerando lo siguiente[114]: Si p es la probabilidad de que k_1 y k_2 sean primos relativos, es decir $MCD(k_1,k_2) = 1$ y, para cualquier entero positivo d, se considera la probabilidad de que $MCD(k_1,k_2) = d$. Esto ocurre cuando k_1 es múltiplo de d, k_2 es múltiplo de d y $MCD(k_1/d, k_2/d) = 1$. La probabilidad de que d divida a k_1 es $1/d$. Por lo tanto, la probabilidad de que $MCD(k_1,k_2) = d$ es $(1/d) \times (1/d) \times p$, es decir, p/d^2. La suma de estas probabilidades sobre todos los valores posibles de d debe ser uno:

$$p\Big(\frac{1}{1^2}+\frac{1}{2^2}+\frac{1}{3^2}+\frac{1}{4^2}+...\Big) = 1 \tag{6.11}$$

Dado que la suma $\Sigma_{i=1}^{\infty}\frac{1}{d^2} = \frac{\pi^2}{6}$ [58], se tendrá que p, la probabilidad de que dos números enteros tomados de forma aleatoria en el intervalo entre 1 y $r-1$ es $\frac{6}{\pi^2}$, alrededor del 60% . Usando algunas propiedades del MCD y el mcm[2] puede establecerse el siguiente teorema, que aquí se da sin demostración:

Teorema
Sea r un entero positivo. Si los enteros k_1 y k_2 son seleccionados aleatoriamente de $\{0,1,...,r—1\}$ y se se tienen los enteros c_1, r_1, c_2, r_2, tales que $MCD(r_1,c_1) = MCD(r_2,c_2) = 1$ y $\frac{k_1}{r} = \frac{c_1}{r_1}$, $\frac{k_2}{r} = \frac{c_2}{r_2}$

Entonces se tendrá que r será el *mínimo común múltiplo*de r_1 y r_2 con probabilidad $\frac{6}{\pi^2}$. El orden de tiempo del algoritmo para calcular r es $\mathcal{O}(\log^2 r)$ [145].

Este teorema indica que el problema de encontrar el orden se reduce a la tarea de muestrear fracciones $\frac{k}{r}$ para números enteros k elegidos uniformemente al azar entre 1 y $r-1$. La reducción del problema puede continuar un nivel más, sin embargo, el desarrollo completo requiere de una revisión profunda de fracciones

[2] El mínimo común múltiplo, denotado $mcm(x,y)$, es el menor entero que cumple con ser divisible tanto por x como por y. Puede encontrarse a partir del MCD, a través de la propiedad $mcm(x,y) = \frac{xy}{MCD(x,y)}$

continuas [3] y otros elementos de la teoría de los números, (como puede verse en [17]). El siguiente teorema contiene los hechos básicos sobre fracciones continuas que son usados en el algoritmo de Shor:

Teorema Todo número racional tiene una secuencia de $\mathcal{O}(n)$ aproximaciones sucesivas, llamadas *convergentes* $\frac{a_1}{b_1}, \frac{a_2}{b_2}, \frac{a_3}{b_3} ... \frac{a_n}{b_n}$, con $\frac{a_n}{b_n} = \frac{x}{2^n}$ con las propiedades:

- Los numeradores a_i y denominadores b_i cumplen $a_i < a_{i+1}, b_i < b_{i+1}$ para toda $i \in \{1,2,...,m\}$
- La lista de convergentes de $\frac{x}{2^n}$ se calcula en tiempo polinomial en n
- Si alguna fracción $\frac{k}{r}$ satisface

$$\left| \frac{x}{2^n} - \frac{k}{r} \right| \leq \frac{1}{2r^2}$$

entonces $\frac{k}{r}$ se encontrará en el conjunto de convergentes de $\frac{x}{2^n}$.

Aunque la forma aquí presentada es la mas adecuada para su aplicación en al algoritmo de Shor, tal como se encuentra en [145], se enuncia sin demostración en esa obra. El primer punto es solo la definición de *convergente*, el segundo es una afirmación sobre la posibilidad de realizar la operación en tiempo polinomial y el tercer punto se conoce como *Criterio de Lagrange* [4].

Como consecuencia de este teorema, la tarea de determinar exactamente la fracción $\frac{k}{r}$ se puede reducir a la tarea de encontrar una estimación $\frac{x}{2^n}$ con $|\frac{k}{r} - \frac{x}{2^n}| \leq \frac{1}{2r^2}$. Finalmente, encontrar el orden r de un módulo N con error acotado se puede reducir al siguiente problema de muestreo [145].

Problema de estimaciones muestrales para un entero casi uniformemente aleatorio múltiplo de $1/r$
Entrada: Enteros a y N tales que $MCD(a,N) = 1$. Sea r el orden de a (desconocido).
Tarea: Generar un número $x \in \{0,1,2,...,2^{n-1}\}$ tal que para cada $k \in \{0,1,...,r—1\}$ se tenga

$$Pr\left(\left| \frac{x}{2^n} - \frac{k}{r} \right| \leq \frac{1}{2r^2}\right) \geq \frac{c}{r} \tag{6.13}$$

para alguna constante $c > 0$

6.5. Circuito

Se puede resumir la sección anterior indicando que la solución al problema de factorizar cualquier número natural, se obtiene mediante la siguiente cadena de reducciones:

- Factorizar cualquier número entero
- Separar a un entero impar que no es potencia perfeceta de un primo
- Encontrar el orden de los números enteros módulo N
- Estimaciones de muestreo para un múltiplo entero aleatorio de $\frac{1}{r}$, (donde r es el orden de algún número entero a mod N).

El algoritmo cuántico se usa justamente en este último problema del muestreo. Para simplificar la discusión, se considera aquí el caso particular en el que r divide exactamente el número de puntos $N = 2^n$ sobre los que se evalúa la función $f(x)$, es decir $N/r = M$, con M entero. Además, se supone que $f(x) = f(y)$ si y sólo si $x = y$ mod r. El caso general agrega algunas complicaciones pero no cambia las ideas generales discutidas a continuación. Se requieren dos registros, el primero se prepara en la superposición igual de todos

[3] Una expresión de la forma

$$a_0 + \cfrac{1}{a_1 + \cfrac{1}{a_2 + \cfrac{1}{a_3 + \ddots + \cfrac{1}{a_n}}}} \tag{6.12}$$

es denominada *fracción continua regular* o *simple* [147] y se pueden usar para encontrar aproximaciones sucesivas a un número real.

[4] La demostración de este último puede encontrarse, por ejemplo, en [275] en el teorema 7.23 y su corolario.

los estados de la base computacional y el segundo almacena la función $f_{a,N}(x) = a^x (\text{mod N})$ (en lo sucesivo $f(x)$ para abreviar). La salida de esta función siempre será menor que N, por lo que se necesitarán $n = \log_2 N$ bits de salida. Además, es necesario evaluar $f_{a,N}(x)$ para al menos los primeros valores N^2 de x por lo que se requiere contar con al menos $m = \log N^2 = 2\log N = 2n$ Qbits de entrada. El circuito cuántico será el operador $U_{f_{aN}}$, que puede verse como una compuerta de varios Qbits en el diagrama que representa a todo el algoritmo. La forma explícita de este circuito se considera como caja negra.

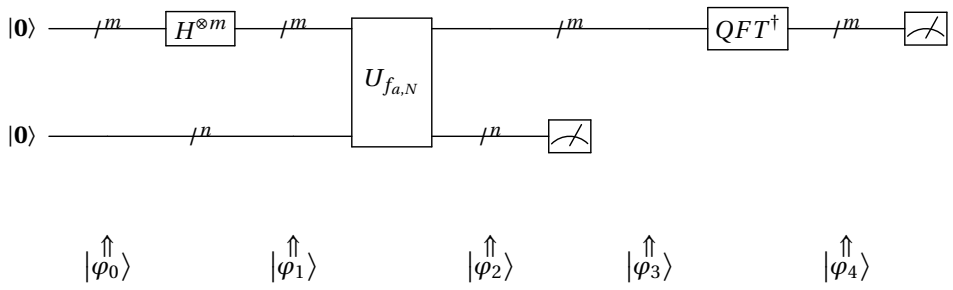


Figura 6.2: Circuito para el algoritmo de Shor

A partir de

$$\left|\varphi_0\right\rangle = |0_m, 0_n\rangle \tag{6.14}$$

se coloca la entrada en una superposición.

$$\left|\varphi_1\right\rangle = H^{\otimes m} \otimes I\,|0_m, 0_n\rangle = \frac{1}{\sqrt{2^n}} \sum_{x=0}^{2^n-1} |x, 0_n\rangle \tag{6.15}$$

El paralelismo cuántico permite calcular la función $f(x)$ para todo x en una sola ejecución. Después de la evaluación de la función, los dos registros se entrelazan.

$$\left|\varphi_2\right\rangle = \frac{1}{\sqrt{2^n}} \sum_{x=0}^{2^n-1} \left|x, f(x)\right\rangle \tag{6.16}$$

Las salidas para una función $f(x)$ periódica, como $a^x (\text{mod N})$, se repetirán. Todo lo que queda por hacer es averiguar cuál es el período. Aunque el paralelismo cuántico ha permitido la evaluación de todos los valores necesarios a la vez, no es posible tener acceso directo a todos los valores de $f(x)$. Por el contrario, después de una medición del segundo registro, este colapsa a un estado particular $\left|f(x_0)\right\rangle$. Por lo tanto, la función de onda de la computadora cuántica se convierte en:

$$\left|\varphi_3\right\rangle = \frac{1}{\sqrt{M}} \sum_{j=0}^{M-1} \left|x_0 + jr, f(x_0)\right\rangle \tag{6.17}$$

con $M = N/r$ el número de valores tales que $f(x) = f(x_0)$.

Dado que se trata de una función con periodo r, se tiene que $f(x_0) = f(x_0 + r) = f(x_0 + 2r) = ... = f(x_0 + [M-1]r)$. Para la función que se considera aquí, se tendrá que para todo j entero, $a^{x_0} = a^{x_0 + jr}(mod N)$. El valor de x_0 dependerá de la medición. Cada vez que se realice una medición, se obtendrá $f(x)$ evaluada en un número, generalmente, distinto. La forma de resolver esta situación consiste en aprovechar las propiedades de la transformada de Fourier. Al aplicarse sobre una función desplazada en el tiempo, se obtiene la transformada de Fourier de la función sin desplazar, multiplicada por un exponencial complejo cuyo argumento es igual al factor de desplazamiento [230], lo que, en términos de cómputo cuántico, se traduce a una fase sin importancia física. Además, si f tiene periodo r, su transformada $\tilde{f}$ tiene periodo N/r [303]. Recordando que la transformada cuántica de Fourier QFT está dada por

$$QFT|x\rangle = \frac{1}{\sqrt{N}} \sum_{z=0}^{N-1} e^{2\pi i \frac{xz}{2^n}} |z\rangle \tag{6.18}$$

se aplica al primer registro, obteniendo

$$\left|\varphi_4\right\rangle = QFT\left|\varphi_3\right\rangle = F\left(\frac{1}{\sqrt{M}}\sum_{j=0}^{M-1}\left|x_0 + jr, f(x_0)\right\rangle\right) = \frac{1}{\sqrt{M}}\sum_{j=0}^{M-1}\frac{1}{\sqrt{2^n}}\sum_{k=0}^{2^n-1} e^{2\pi i([x_0+jr]k)/2^n}\left|k\right\rangle\left|f(x_0)\right\rangle \tag{6.19}$$

El segundo estado está factorizado y no es de interés para el resto del análisis por lo que se omite de aquí en adelante. Por otra parte, el primer estado puede escribirse también como;

$$\left|\varphi_4'\right\rangle = \frac{1}{\sqrt{M2^n}}\sum_{k=0}^{2^n-1} e^{2\pi i x_0 k/2^n}\sum_{j=0}^{M-1} e^{2\pi i jrk/2^n}\left|k\right\rangle \tag{6.20}$$

La segunda suma tiene un comportamiento interesante que conviene destacar aquí:

$$\sum_{j=0}^{M-1} e^{2\pi i jrk/2^n} = \sum_{j=0}^{M-1}[e^{2\pi irk/2^n}]^j = \sum_{j=0}^{M-1}\omega^j \tag{6.21}$$

con $\omega = e^{2\pi irk/2^n}$. Si $\omega \neq 1 \implies \sum_{j=0}^{M-1}\omega^j = \frac{1-\omega^{[m-1]+1}}{1-\omega} = \frac{1}{1-\omega}\left(1 - e^{(2\pi i(rk/2^n))m}\right) = 0$. La interferencia cuántica ha seleccionado unas pocas frecuencias específicas. Por lo tanto, los únicos valores que sobrevivirán en la suma doble serán aquellos en los que $\omega = 1$, es decir, cuando k sea un múltiplo de N/r, lo que ocurre M veces:

$$\frac{1}{\sqrt{M2^n}}\sum_{k=0}^{2^n-1} e^{2\pi i x_0 k/2^n}\sum_{j=0}^{M-1} e^{2\pi i jrk/2^n}\left|k\right\rangle = \frac{M}{\sqrt{M2^n}}\sum_{k=0}^{2^n-1} e^{2\pi i x_0 k/2^n}\left|k\right\rangle \tag{6.22}$$

Dado que $M = N/r = 2^n/r$, es posible escribir

$$\left|\varphi_4'\right\rangle = \frac{1}{\sqrt{r}}\sum_{k=0}^{2^n-1} e^{2\pi i x_0 k/2^n}\left|k\right\rangle \tag{6.23}$$

En realidad, una medición cuántica de la función de onda $\left|\varphi_4'\right\rangle$ dará uno de los r resultados de la forma lN/r, con $l = 0, 1, ..., r-1$ con igual probabilidad (siendo irrelevante el factor de fase $e^{2\pi i x_0 k/2^n}$), así, si se denota por c el valor medido de k, se tendrá $c/N = l/r$, siendo l un entero desconocido. Si l y r no tienen factores comunes, entonces se reduce c/N a una fracción irreducible y así se obtiene tanto l como r. Esto ocurre con una probabilidad de al menos $1/\log(\log(r))$ [253]. De lo contrario, el algoritmo falla y debe repetirse. Se puede demostrar que el algoritmo tiene éxito con una probabilidad arbitrariamente cercana a uno después de un número de corridas $\mathcal{O}(\log(\log(r)))$.

Se ha visto que la aceleración de la computación cuántica, respecto a todo algoritmo clásico conocido a la fecha resulta exponencial en el problema de factorizar y cuadrática en el problema de la búsqueda. La explicación a esta diferencia puede describirse en términos de la estructura subyacente en cada problema. Mientras que el objeto buscado en la base de datos puede encontrarse en cualquier sitio, por lo que la mecánica cuántica solo puede ayudar en esta búsqueda dando una aceleración cuadrática significativa que, desafortunadamente, resulta la máxima posible, encontrar la factorización a un número no es una búsqueda a ciegas, se trata en realidad de explotar al periodo de una función específica, una cantidad desconocida y difícil de determinar por medios clásicos, pero que existe. La estructura permite reducir el problema de factorización de enteros al problema de encontrar el período de una función particular, y como es natural en los problemas relacionados con periodos, se puede encontrar a través del cálculo de su transformada de Fourier que puede implementarse de manera eficiente en una computadora cuántica.

7

Conclusiones y perspectivas

A lo largo de este trabajo se ha analizado cómo la necesidad de calcular impulsó el desarrollo de la computación. Las herramientas computacionales han alcanzado, en gran medida, los límites físicos de su capacidad, proporcionando soluciones para una amplia variedad de problemas. Sin embargo, aún existen desafíos que superan el alcance del cómputo clásico. Ya sea por un interés puramente académico o por su enorme potencial tecnológico, abordar estas limitaciones requiere aprovechar propiedades cuánticas, lo que implica un replanteamiento del paradigma computacional tradicional. Si bien, las leyes de la lógica y matemáticas permanecen invariantes, la forma en que pueden procesarse es muy diferente, al punto de que la teoría de la computación reversible, aunque no es completamente nueva, cobra una relevancia crucial en el diseño de algoritmos y arquitecturas cuánticas.

Precisamente, esta relevancia de la computación reversible nos lleva a reflexionar sobre aspectos fundamentales que conectan la lógica con las limitaciones físicas de los sistemas computacionales. A manera de conclusión, me gustaría mencionar algunas cuestiones sobre la relación entre reversibilidad lógica y computación cuántica. En computación clásica, la importancia del trabajo de Landauer se debe a las importantes consecuencias tecnológicas relacionadas con la gran cantidad de calor generado por las computadoras. Como investigador en una empresa lider en tecnología informática (IBM), la motivación original detrás del trabajo de Landauer estaba muy lejos de buscar establecer un vínculo entre la información y la termodinámica, en realidad, perseguía encontrar la cantidad mínima de calor disipado por una computadora ideal, si la hubiera. El desarrollo de circuitos integrados está limitado por la disipación de calor. Se sabe que la producción de calor en los microprocesadores utilizados en las computadoras modernas es un factor importante que dificulta su miniaturización, ya que se vuelve cada vez más difícil evacuar el exceso de calor cuando se reduce el tamaño y, por lo tanto, la superficie. Por otra parte, la potencia disipada en los dispositivos electrónicos es directamente proporcional a la frecuencia de trabajo. Por tanto, cuanto mayor es la frecuencia con la que se realizan cálculos, más calor se disipa. La cantidad total de energía que se puede disipar del chip impone un límite en la frecuencia operativa de alrededor de 5 GHz para computadoras reales. En otras palabras, la tecnología para construir un solo transistor que funcione a frecuencias tan altas como 1 THz está bien desarrollada, pero no la tecnología para extraer la cantidad de calor generada en un chip con 10^{10} de dichos transistores [64]

En la actualidad, todo mecanismo de pérdida de energía se encuentra varios órdenes de magnitud por encima del valor implicado por la tesis de Landauer, pero a medida que se controlen estos otros mecanismos de pérdida, eventualmente las pérdidas de energía debidas únicamente al borrado de información, que podrían resultar del uso de compuertas lógicas irreversibles, se convertirían en la contribución dominante. Uno de los mecanismos de pérdida en la computadoras es la energía mínima de conmutación de un transistor CMOS/FET. La tendencia observada indica que la cantidad predicha por Landauer podría alcanzarse en 2035 (o incluso antes)[96], por lo que pronto se debería enfrentar lo que aparentemente resulta una limitante impuesta por la segunda ley de la termodinámica. Si el principio de Landauer es correcto, puede haber formas de construir computadoras que se disipen mucho menos que los dispositivos actuales. Pero en caso contrario, hay un límite inferior de calor que una computadora debe disipar en cada paso, lo que establece un límite inevitable en la cantidad de calor que debe producir una computadora.

Si la disipación en calor debe ocurrir solo si la información se borra (destruye), disipando una energía de al menos $k_B T Ln(2)$, la cuestión debería ser observada en el laboratorio. Sin embargo, la energía mencionada

en la tesis de Landauer, aproximadamente $3 \times 10^{-21} J$ a temperatura ambiente, es una cantidad muy pequeña de energía, insignificante en comparación con la energía disipada por las computadoras reales. Por ello, durante décadas las discusiones sobre el tema han sido en gran parte académicas y siempre teóricas. Para probar la validez del principio de Landauer se requieren pruebas del mundo real, pero solo en los últimos años se han realizado experimentos a niveles de energía que pueden proporcionar una prueba válida [216], aunque la discusión sigue en curso [159].

Transformaciones unitarias y lógica reversible
Las leyes de la física se conservan bajo las siguientes transformaciones en el espacio y el tiempo:

- Traslaciones (por ejemplo, a lo largo del eje X),
- Rotaciones (por ejemplo, alrededor del eje Z),
- Transformaciones de velocidad (es decir, respecto a un marco de referencia en movimiento uniforme)
- Traslaciones de tiempo (evolución temporal).

Cada transformación está especificada mediante una serie de coeficientes numéricos que dependerán del sistema de coordenadas al que se aplica. Al tratarse de transformaciones lineales del espacio euclidiano en él mismo, la estructura algebraica resultante es un operador y se representan mediante matrices. Los operadores que preservan norma y producto escalar en $\mathbb{R}^3$ se denominan ortogonales, sin embargo, los vectores de estado en mecánica cuántica requieren trabajar en un espacio de Hilbert, por lo que se requiere una extensión del concepto de ortogonalidad, en el sentido de que ni las longitudes ni los «ángulos» complejos deben cambiar.

Si se desea determinar la forma del operador que puede efectuar una traslación de una función, es necesario identificar los elementos geométricos que preserva. Por ejemplo, en el caso de la traslación de un estado $|\Psi\rangle$ normalizado, es decir $\langle\Psi|\Psi\rangle = 1$, se espera que el operador $U = U(x)$ traslade al estado cuántico a lo largo del eje x, sin alterar el estado de ninguna otra forma. Luego $\left|\Psi'\right\rangle = U|\Psi\rangle$ es el ket que representa al estado trasladado. Por definición del adjunto de un operador $\left\langle\Psi'\right| = \langle\Psi| U^\dagger$ es el estado bra trasladado de manera similar correspondiente a $\left|\Psi'\right\rangle$. Dado que no se ha hecho otra cosa que trasladar $\langle\Psi|$ y $|\Psi\rangle$, el producto interior de estos estados trasladados debe permanecer sin alteraciones, es decir, debe ser invariante ante la traslación, por lo que

$$\left\langle\Psi'\middle|\Psi'\right\rangle = \langle\Psi|\Psi\rangle \tag{7.1}$$

pero al escribir a los estados transformados como producto de los originales y el operador que representa a la transformación se tiene

$$\left\langle\Psi'\middle|\Psi'\right\rangle = \Big(\langle\Psi| U^\dagger\Big)\Big(U|\Psi\rangle\Big) = \langle\Psi| U^\dagger U|\Psi\rangle = \langle\Psi|\Big(UU^\dagger|\Psi\rangle\Big) \tag{7.2}$$

Como lo indica la última expresión de la ecuación, si se considera que $U^\dagger$ actúa hacia la derecha, la igualdad aún debe mantenerse. Pero entonces debe cumplirse que $UU^\dagger = \mathbf{1}$. Los operadores que cumplen esta última condición se conocen como unitarios y constituyen la generalización de operadores ortogonales a espacios complejos, con un número arbitrario de dimensiones. Es posible emplear argumentos similares para tratar las otras transformaciones espacio-temporales [41]. Por tanto, se concluye que tales transformaciones deben ser implementadas por operadores unitarios. De manera que las transformaciones unitarias en el espacio de Hilbert representan el equivalente a las rotaciones rígidas de un sistema físico en un espacio tridimensional. Una rotación de este tipo es, por supuesto, equivalente a una rotación del sistema de referencia en el sentido opuesto [7].

El estado de una computadora cuántica, como sistema físico, está descrito a través del vector de estado. La evolución de dicho estado a lo largo del tiempo está determinada por un operador unitario. Por tanto, las operaciones que una computadora cuántica puede realizar sobre un solo Qbit para transformarlo están representadas por la acción sobre el estado del Qbit de cualquier transformación lineal que convierta vectores unitarios en vectores unitarios, es decir, por transformaciones unitarias, lo que se traduce en el hecho de que las compuertas lógicas cuánticas se representan mediante matrices unitarias

Si se comienza con un Qbit en estado $|\Psi\rangle$ y aplicar una operación unitaria U, el estado final del Qbit $\left|\Psi'\right\rangle$ será

$$\left|\Psi'\right\rangle = U|\Psi\rangle \tag{7.3}$$

Pero, dado que cualquier transformación unitaria tiene una inversa unitaria, $UU^\dagger = \mathbf{1}$,

$$U^\dagger\left|\Psi'\right\rangle = U^\dagger U|\Psi\rangle = |\Psi\rangle \tag{7.4}$$

por lo que cualquier transformación se puede revertir aplicando su conjugado complejo en el nuevo estado, permitiendo que cada operación cuántica sea reversible. Entonces, las compuertas cuánticas realizan operaciones unitarias sobre los Qbits y son *físicamente reversibles*.

Reversibilidad lógica y reversibilidad física

Sin embargo, hasta este punto no se ha establecido la necesidad de que sean *lógicamente reversibles*. Para entender este requisito, es necesario suponer que la tesis de Landauer es correcta. En uno de los primeros artículos que se escribieron sobre compuertas cuánticas, de donde se toma la notación actualmente utilizada, puede encontrarse:

> Históricamente, la idea de que la mecánica cuántica de sistemas aislados debería estudiarse como un nuevo sistema formal de computación surgió del reconocimiento hace veinte años de que la computación podía hacerse reversible dentro del paradigma de la física clásica. Es posible realizar cualquier cálculo de una manera que sea reversible tanto *lógicamente* (es decir, el cálculo es una secuencia de transformaciones biyectivas) como *termodinámicamente* (el cálculo podría, en principio, realizarse mediante un aparato físico que disipe arbitrariamente poca energía)[16].

En donde se observa una clara distinción entre dos tipos de reversibilidad (la cursiva está incluida en el original). Sin embargo, al terminar este párrafo, cita el trabajo de Landauer, en el que claramente puede leerse «Creemos que los dispositivos que exhiben irreversibilidad lógica son esenciales para la informática. Creemos que la irreversibilidad lógica implica a su vez irreversibilidad física, y esta última va acompañada de efectos disipativos » [161]. A partir de esta cita, ya no sorprende que, en la próxima sección, indique que la computación reversible ya está incluida en la mecánica cuántica, como un «pequeño subconjunto»:

> La física cuántica también es reversible, porque la evolución en el tiempo inverso especificada por el operador unitario $U^{-1} = U^{\dagger}$ siempre existe; como consecuencia, varios investigadores reconocieron que se podía ejecutar computación reversible dentro de un sistema mecánico cuántico. Se han discutido las máquinas de Turing mecánicas cuánticas, los conjuntos de compuertas y los autómatas celulares, y se han discutido las realizaciones físicas de las compuertas universales de Toffoli y Fredkin dentro de varios sistemas físicos de mecánica cuántica.
> Si bien la computación reversible está incluida en la mecánica cuántica, es un subconjunto pequeño: la evolución temporal de una computadora reversible clásica se describe mediante operadores unitarios cuyos elementos de la matriz son sólo cero o uno; no se permiten números complejos arbitrarios[...] El conjunto de puertas cuánticas es la generalización cuántica natural de los circuitos lógicos combinacionales acíclicos estudiados en la teoría de la complejidad computacional convencional[16].

Esto explica que, para garantizar la reversibilidad a todos los niveles, se exija que las compuertas sean lógicamente reversibles. Si no cumplen con esto, la tesis de Landauer implicaría disipación de energía por cada bit borrado. Siguiendo este argumento, incluso si se tuviera éxito eliminado cualquier otro mecanismo de pérdida de energía de todo circuito basado en NAND, el circuito seguiría disipando energía mientras se encuentre funcionando, debido a las pérdidas «inevitables» de energía que se producen al borrar información. La mayoría de las compuertas lógicas clásicas son lógicamente irreversibles. Las funciones booleanas $f : \{0,1\}^2 \rightarrow \{0,1\}$ eliminan un bit de información. De acuerdo con Landauer, ello implicaría que no serían físicamente reversibles y no serían de utilidad en cómputo cuántico porque no podrían ser unitarias.

La implementación de lógica reversible no es gratuita. Debe considerarse que la existencia de un mapeo uno a uno único entre los conjuntos de bits de entrada y de salida implica que habrá bits de salida que no serán usados en el cálculo pero que se requieren porque su propósito es mantener la reversibilidad (bits basura), así como entradas constantes que se utilizan en circuitos reversibles para almacenar valores intermedios durante el cálculo (bits ancilla). Considerando que la realización de un circuito cuántico de muchos Qbits es extremadamente difícil, la cantidad de Qbits en una implementación real es limitada, por lo que es deseable reducir el número de bits ancilla y basura tanto como sea posible[292] [154].

¿Podrían simplemente eliminarse estos bits adicionales? Actualmente no. La abrumadora mayoría de

los científicos consideran al trabajo fundamental de Landauer algo más que un trabajo seminal, una obra maestra de la ciencia que conecta información y termodinámica, a pesar de que, a poco más de 60 años de publicarse, se encuentra todavía acompañado de controversias. Desde un punto de vista teórico, algunos científicos han argumentado persistentemente que el principio de Landauer no es una forma pertinente de discutir la disipación en dispositivos informáticos. Desde el punto de vista experimental, aparte de los recientes experimentos exitosos que validan el principio de borrado de Landauer , existen pocos trabajos que sugieran algún tipo de inconvenientes[215]. Esperar a una prueba concluyente sobre la validez de la tesis de Landauer para decidir si la computación cuántica no requiere cómputo reversible podría retrasar considerablemente los avances en la investigación.

Es comprensible que se haya desarrollado un esquema que cubriera esa posibilidad, aún a costa de la cantidad de Qbits disponibles para cálculo. Dado que todo cálculo irreversible puede realizarse mediante cómputo reversible, cabe preguntarse cuales serían las implicaciones de que se demostrara falsa. Uno de los argumentos es que, en principio, incluso las operaciones lógicamente irreversibles pueden realizarse sin un aumento de entropía en el entorno. Esto indicaría que la preocupación por desarrollar computación lógicamente reversible está de mas: la computación podría hacerse físicamente reversible, aunque lógicamente no lo sea. De manera similar,si no existe relación entre pérdida de información e irreversibilidad física,la computación lógicamente reversible sobra. Esto en realidad sería beneficioso porque los Qbits que se reservan como ancilla o basura podrían emplearse en el cálculo directo. Por otra parte, si todas las operaciones de procesamiento de datos, sean lógicamente reversibles o no, requieren la disipación de al menos $k_B T Ln(2)$ de energía, la computación reversible resulta insuficiente. No importa que clase de lógica se use, se seguirá disipando energía al realizar operaciones. No habría forma de construir una compuerta cuántica ideal, porque todo cálculo implicaría pérdida de energía.

Aunque este último podría parecer el peor escenario para la computación cuántica, debe considerarse lo siguiente. La idea de que todos los procesos cuánticos son unitarios y reversibles solo es cierta en entornos idealistas. Si la situación es tal que existe un fuerte acoplamiento entre el sistema y el entorno, entonces se produce una pérdida de información y energía entre ambos de una manera nada despreciable. En la práctica, cualquier sistema cuántico es abierto, ya que un sistema cuántico nunca puede aislarse completamente de su entorno, y la interacción entre ambos resulta en la creación de correlaciones entre los estados del sistema y del entorno[46].

En resumen, la tesis de Landauer impone a la computación cuántica la necesidad de implementar lógica reversible, ya que las operaciones no reversibles producen pérdida de información y, a su vez, esta se traduce en pérdida de energía. Dotar a la computación cuántica de la lógica no reversible garantiza que las compuertas cuánticas sean unitarias en el extremo de que cualquier otro mecanismo de disipación de calor sea eliminado de una implementación real. En caso de que no haya costo por borrado, habría una mayor cantidad de Qbits disponibles para cálculo y las operaciones irreversibles podrían implementarse en una computadora cuántica. En caso de que toda computación sea, en cualquier forma, irreversible, el empleo de compuertas unitarias sigue siendo un modelo de primera aproximación, en la que herramientas como el control numérico óptimo podrían ayudar a corregir las desviaciones, por lo que es necesario conocer el comportamiento ideal de las compuertas en primer instancia.

¿Las computadoras cuánticas pueden sustituir a las clásicas?
Si las computadoras cuánticas parecían estar diseñadas para resolver problemas que las clásicas no, pero además deberían poder con el trabajo que si realizan. Tras revisar algunos algoritmos cuánticos, he observado que hay tareas que la computadora clásica realiza de manera eficiente y que no hay necesidad de sustituir, sino de complementar. Mientras que los sistemas clásicos seguirán siendo fundamentales para muchas tareas cotidianas y problemas bien estructurados, la computación cuántica ofrecerá soluciones únicas para desafíos específicos. En mi opinión, a medida que aumente la disponibilidad de sistemas cuánticos y la preparación de especialistas dedicados a ellos, se observará una evolución de conceptos que, sin abandonar los cimientos clásicos, los lleve hacia nuevos territorios en donde los límites de lo computable estén definidos en términos del entrelazamiento, la reversibilidad y la probabilidad.

Como se expuso en la sección dedicada al cómputo clásico, Turing definió "lo computable" mediante un dispositivo abstracto que opera con un conjunto finito de reglas y estados, capaz de generar un resultado en un número finito de pasos a partir de una entrada específica. Este modelo se fundamenta en un enfoque determinista, ya que cada paso de la máquina está completamente determinado por su estado actual y el símbolo que se lee en la cinta; secuencial, porque las operaciones se realizan una tras otra, sin un paralelismo intrínseco; y universal, dado que la Máquina de Turing Universal puede simular cualquier otra máquina de

Turing, estableciendo que todo lo computable en este marco puede ser calculado por un único modelo universal. De manera que aquello computable según Turing se refiere a funciones que pueden evaluarse o problemas que pueden resolverse mediante un procedimiento mecánico, finito y en un tiempo razonable. Turing demostró también la existencia de problemas no computables, fuera del alcance de cualquier máquina de Turing, incluyendo la versión cuántica. Una máquina de Turing Cuántica sigue estando limitada por los mismos fundamentos que su contraparte clásica y no expande el conjunto de funciones computables según Tuing. Lo que si puede hacer es cambiar los límites de lo computable en términos de la eficiencia y capacidades prácticas, como es el caso de problemas que son teóricamente computables bajo el modelo de Turing pero que requieren tiempos exponenciales para obtener la solución. Redefinir los límites de lo computable, en este sentido, se refiere a ampliar el espectro de problemas que se pueden abordar y a la transformación de las herramientas y métodos con los que se les hace frente, marcando una transición hacia un nuevo paradigma computacional.

Las implicaciones de esta transición apenas se han explorado y es muy posible que las mas profundas y sorprendentes estén aún por presenciarse, a medida que la computación cúantica supere la etapa de investigación y desarrollo y se consolide como una herramienta que trascienda el circulo académico, tecnológico y militar y encuentre su sitio en la vida diaria, como alguna vez lo hizo la computación clásica. No pretendiendo sustituirla, sino complementándola. En tanto que los sistemas computacionales clásicos seguirán resultando fundamentales para tareas elementales y cotidianas, así como para problemas bien estructurados, la computación cuántica ofrecerá soluciones para desafíos especificativos.

Referencias

[1] International Journal of Theoretical Physics | Volume 21, issue 3.

[2] 40 years of quantum computing. *Nature Reviews Physics*, 4(1):1–1, Enero 2022.

[3] M. Agrawal, N. Kayal, and N. Saxena. Primes is in P. *Annals of Mathematics*, 160, Septiembre 2002.

[4] N. Ahmed and K. R. Rao. *Orthogonal Transforms for Digital Signal Processing*. Springer Science & Business Media, Diciembre 2012.

[5] M. Al-Razouki and S. Smith. *Hybrid Healthcare*. Springer Nature, Julio 2022.

[6] C. E. Alchourrón. *Lógica*. Number 7 in Enciclopedia Iberoamericana de Filosofía. Editorial Trotta CSIC, 1995.

[7] M. Alonso and H. Valk. *Mecánica Cuántica fundamentos y aplicaciones*. Universidad de Salamanca, Junio 2009.

[8] N. Anderson. Conditional Erasure and the Landauer Limit: A Review of Landauer's Principle, Theory and Experiments. pages 65–100. Enero 2019.

[9] E. Andersson and P. Öhberg. *Quantum Information and Coherence*. Springer, Julio 2014.

[10] S. Arora and B. Barak. *Computational Complexity: A Modern Approach*. Cambridge University Press, Abril 2009.

[11] W. R. Ashby, J. T. Culbertson, M. D. Davis, S. C. Kleene, K. D. Leeuw, D. M. M. Kay, J. M. Carthy, M. L. Minsky, E. F. Moore, C. E. Shannon, N. Shapiro, A. M. Uttley, and J. V. Neumann. *Automata Studies. (AM-34)*. Princeton University Press, 1956.

[12] B. E. Baaquie and L.-C. Kwek. *Quantum Computers: Theory and Algorithms*. Springer Nature Singapore, Singapore, 2023.

[13] E. Bach and J. O. Shallit. *Algorithmic Number Theory: Efficient algorithms*. MIT Press, 1996.

[14] E. Bach and J. Sorenson. Sieve algorithms for perfect power testing. *Algorithmica*, 9(4):313–328, Abril 1993.

[15] E. Ballico, A. Bernardi, I. Carusotto, S. Mazzucchi, and V. Moretti. *Quantum Physics and Geometry*. Springer, Marzo 2019.

[16] A. Barenco, C. H. Bennett, R. Cleve, D. P. DiVincenzo, N. Margolus, P. Shor, T. Sleator, J. Smolin, and H. Weinfurter. Elementary gates for quantum computation. *Physical Review A*, 52(5):3457–3467, Noviembre 1995.

[17] J. Barzen and F. Leymann. Continued Fractions and Probability Estimations in Shor's Algorithm: A Detailed and Self-Contained Treatise. *AppliedMath*, 2(3):393–432, 2022.

[18] R. Bassoli, H. Boche, C. Deppe, R. Ferrara, F. H. P. Fitzek, G. Janssen, and S. Saeedinaeeni. *Quantum Communication Networks*. Springer Nature, Febrero 2021.

[19] D. C. Bastos and L. A. Brasil Kowada. How to detect whether Shor's algorithm succeeds against large integers without a quantum computer. *Procedia Computer Science*, 195:145–151, Enero 2021.

[20] W. C. Bauldry. *Computational Calculus: A Numerical Companion to Elementary Calculus*. Springer Nature, Junio 2023.

[21] G. Benenti, G. Casati, D. Rossini, and G. Strini. *Principles Of Quantum Computation And Information: A Comprehensive Textbook*. World Scientific, Diciembre 2018.

[22] P. Benioff. The computer as a physical system: A microscopic quantum mechanical Hamiltonian model of computers as represented by Turing machines. *Journal of Statistical Physics*, 22(5):563–591, Mayo 1980.

[23] P. Benioff. Quantum Mechanical Models of Turing Machines That Dissipate No Energy. *Physical Review Letters*, 48(23):1581–1585, Junio 1982.

[24] C. H. Bennett. Logical Reversibility of Computation. *IBM Journal of Research and Development*, 17(6): 525–532, 1973.

[25] C. H. Bennett. The thermodynamics of computation—a review. *International Journal of Theoretical Physics*, 21(12):905–940, Diciembre 1982.

[26] C. H. Bennett. Thermodynamically Reversible Computation. *Physical Review Letters*, 53(12):1202–1202, Septiembre 1984.

[27] C. H. Bennett. Notes on Landauer's principle, Reversible Computation and Maxwell's Demon, Enero 2003.

[28] R. K. Bera. *The Amazing World of Quantum Computing*. Springer Nature, Marzo 2020.

[29] J. Berkowitz. *Tales From the Word Guy: What Your English Teacher Never Taught You*. FriesenPress, Julio 2022.

[30] M. V. Berry. Quantal Phase Factors Accompanying Adiabatic Changes. *Proceedings of the Royal Society of London. Series A, Mathematical and Physical Sciences*, 392(1802):45–57, 1984.

[31] J. Birnbaum and R. S. Williams. Physics and the Information Revolution. *Physics Today*, 53(1):38–42, Enero 2000.

[32] E. K. Blum and A. V. Aho. *Computer Science: The Hardware, Software and Heart of It*. Springer Science & Business Media, Diciembre 2011.

[33] R. Blumel. *Foundations of Quantum Mechanics: From Photons to Quantum Computers*. Jones & Bartlett Learning, 2010.

[34] J. A. Bondy and U. S. R. Murty. *Graph Theory with Applications*. American Elsevier Publishing Company, 1976.

[35] G. Boole. *Investigación de las leyes del pensamiento sobre las que se fundamentan las teorías matemáticas de la lógica y las probabilidades: (Selección)*. Universidad del Zulia, Escuela de Filosofía, Facultad de Humanidades y Educación, 1972.

[36] G. Boolos and R. C. Jeffrey. *Computability and Logic*. CUP Archive, Julio 1974.

[37] M. Born and P. Jordan. Zur Quantenmechanik. *Z. Phys.*, 34(1):858–888, 1925.

[38] M. Born, W. Heisenberg, and P. Jordan. Zur Quantenmechanik. II. *Z. Phys.*, 35(8-9):557–615, 1926.

[39] M. Born. *My Life: Recollections of a Nobel Laureate*. Taylor & Francis, Junio 1978.

[40] F. Boudot, P. Gaudry, A. Guillevic, N. Heninger, E. Thomé, and P. Zimmermann. Comparing the difficulty of factorization and discrete logarithm: a 240-digit experiment, 2020.

[41] G. E. Bowman. *Essential Quantum Mechanics*. Oxford University Press, 2008.

[42] D. Braha, A. A. Minai, and Y. Bar-Yam. *Complex Engineered Systems: Science Meets Technology*. Springer, Junio 2007.

[43] G. Brassard. Searching a Quantum Phone Book. *Science*, 275(5300):627–628, 1997.

[44] G. Brassard, P. Hoyer, and A. Tapp. Quantum Algorithm for the Collision Problem. volume 1380, pages 163–169. 1998.

[45] M. P. Brenner and H. A. Stone. Modern Classical Physics Through the Work of G. I. Taylor. *Physics Today*, 53(5):30–35, Mayo 2000.

[46] H. P. Breuer and F. Petruccione. *The Theory of Open Quantum Systems*. Oxford University Press, 2002.

[47] L. Brillouin. Maxwell's Demon Cannot Operate: Information and Entropy. I. *Journal of Applied Physics*, 22:334–337, Abril 1951.

[48] J. G. Brookshear. *Computer Science: An Overview*. Pearson/Addison Wesley, 2007.

[49] J. Brookshear. *Introducción a las ciencias de la computación*. Addison Wesley. Addison-Wesley Iberoamericana, 1995.

[50] J. Buchmann. *Introduction to Cryptography*. Springer Science & Business Media, Diciembre 2013.

[51] A. W. Burks, H. H. Goldstine, and J. Neumann. Preliminary Discussion of the Logical Design of an Electronic Computing Instrument. In B. Randell, editor, *The Origins of Digital Computers*, pages 399–413. Springer Berlin Heidelberg, Berlin, Heidelberg, 1982.

[52] A. Campos. *Introducción a la historia y a la filosofía de la matemática*. Universidad Nacional de Colombia, Enero 2008.

[53] G. Cantor. Ueber eine elementare Frage der Mannigfaltigketislehre. *Jahresbericht der Deutschen Mathematiker-Vereinigung*, 1:72–78, 1890.

[54] G. Carcassi, L. Maccone, and C. A. Aidala. The four postulates of quantum mechanics are three. *Physical Review Letters*, 126(11):110402, Marzo 2021.

[55] A. Cevik. *Philosophy of Mathematics: Classic and Contemporary Studies*. CRC Press, Noviembre 2021.

[56] M. Chaichian and R. Hagedorn. *Symmetries in Quantum Mechanics: From Angular Momentum to Supersymmetry (PBK)*. CRC Press, Julio 2023.

[57] N. Champion and C. Babbage. *Charles Babbage*. Heinemann Library, 2001.

[58] L. Chaumont and M. Yor. *Exercises in Probability: A Guided Tour from Measure Theory to Random Processes, Via Conditioning*. Cambridge University Press, Noviembre 2003.

[59] G. Chen, D. A. Church, B.-G. Englert, C. Henkel, B. Rohwedder, M. O. Scully, and M. S. Zubairy. *Quantum Computing Devices: Principles, Designs, and Analysis*. CRC Press, Septiembre 2006.

[60] B. Chopard and M. Tomassini. *An Introduction to Metaheuristics for Optimization*. Springer, Noviembre 2018.

[61] J. Cilleruelo. *Los números*. Editorial CSIC - CSIC Press, Diciembre 2010.

[62] C. Cohen-Tannoudji, B. Diu, and F. Laloë. *Quantum Mechanics, Volume 1: Basic Concepts, Tools, and Applications*. John Wiley & Sons, Diciembre 2019.

[63] D. P. Collantes and L. A. . B. Vega. *Fundamentos de electrónica digital*. Universidad de Salamanca, 2006.

[64] E. Colomés, J. Mateos, T. González, and X. Oriols. Noise and charge discreteness as ultimate limit for the THz operation of ultra-small electronic devices. *Scientific Reports*, 10(1):15990, Octubre 2020.

[65] D. E. Comer, D. Gries, M. C. Mulder, A. Tucker, A. J. Turner, and P. R. Young. Computing as a discipline. *Communications of the ACM*, Enero 1989.

[66] E. Commins. Electron Spin and Its History. *Annual Review of Nuclear and Particle Science*, 62:133–157, Octubre 2012.

[67] B. J. Copeland, C. J. Posy, and O. Shagrir. *Computability: Turing, Gödel, Church, and Beyond*. MIT Press, Enero 2015.

[68] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction To Algorithms*. MIT Press, 2001.

[69] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms, fourth edition*. MIT Press, Abril 2022.

[70] M. Cozzens and S. J. Miller. *The Mathematics of Encryption*. American Mathematical Soc., Septiembre 2013.

[71] J. Crowe and B. Hayes-Gill. *Introduction to Digital Electronics*. Elsevier, Marzo 1998.

[72] C. Davisson. Are electrons waves? *Journal of the Franklin Institute*, 205(5):597–623, Mayo 1928.

[73] H. Delfs and H. Knebl. *Introduction to Cryptography: Principles and Applications*. Springer Science & Business Media, Diciembre 2012.

[74] D. Deutsch. *The Fabric of Reality*. Penguin UK, Abril 2011.

[75] W. Dietrich, H. Düker, and G. Möllenstedt. Ein Elektronen-Spektrograph für metallphysikalische Untersuchungen. *International Journal of Materials Research*, 47(4):240–242, Abril 1956.

[76] M. Dietzfelbinger. *Primality Testing in Polynomial Time: From Randomized Algorithms to "PRIMES Is in P"*. Springer Science & Business Media, Junio 2004.

[77] Y. Ding and F. T. Chong. *Quantum Computer Systems: Research for Noisy Intermediate-Scale Quantum Computers*. Springer Nature, Mayo 2022.

[78] P. A. M. Dirac. *The Principles of Quantum Mechanics*. Clarendon Press, 1981.

[79] D. P. Divincenzo. Two-Bit Gates are Universal for Quantum Computation. *Physical Review A*, 51(2): 1015–1022, Febrero 1995.

[80] J. D. Dixon. Factorization and primality tests. *The American Mathematical Monthly*, 91(6):333–352, 1984.

[81] I. B. Djordjevic. *Quantum Information Processing, Quantum Computing, and Quantum Error Correction: An Engineering Approach*. Academic Press, Febrero 2021.

[82] B. Duplantier and V. Rivasseau. *Information Theory: Poincaré Seminar 2018*. Springer Nature, Julio 2021.

[83] J. Earman and J. D. Norton. Exorcist XIV: The wrath of Maxwell's demon. Part II. from Szilard to Landauer and beyond. *Studies in History and Philosophy of Science Part B: Studies in History and Philosophy of Modern Physics*, 30(1):1–40, 1999.

[84] D. L. Eggleston. *Basic Electronics for Scientists and Engineers*. Cambridge University Press, Abril 2011.

[85] D. F. Elliott and K. R. Rao. *Fast Transforms Algorithms, Analyses, Applications*. Elsevier, Marzo 1983.

[86] B. Evangelidis. Quantum logic as reversible computing, Noviembre 2021.

[87] J. N. Faus. *El principio de incertidumbre de Heisenberg*. RBA Libros, Octubre 2017.

[88] B. M. Fernando. *Introducción a la teoría de grupos*. UAEH, 2004.

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

[90] R. P. Feynman. Quantum mechanical computers. *Foundations of Physics*, 16(6):507–531, Junio 1986.

[91] R. P. Feynman. *Conferencias sobre computación*. Grupo Planeta (GBS), Septiembre 2003.

[92] R. P. Feynman, R. B. Leighton, and Matthew Sands. *Lecciones de fisica de Feynman III. Mecánica cuantica*, volume 3. Fondo de Cultura Economica, México, Marzo 2023.

[93] D. D. Fitts. *Principles of Quantum Mechanics: As Applied to Chemistry and Chemical Physics*. Cambridge University Press, Agosto 1999.

[94] R. Fleischer, B. Moret, and E. M. Schmidt. *Experimental Algorithmics: From Algorithm Design to Robust and Efficient Software*. Springer, Julio 2003.

[95] J. B. Fraleigh. *Algebra abstracta: primer curso*. Addison-Wesley Iberoamericana, 1988.

[96] M. Frank. Approaching the physical limits of computing. In *Proceedings of The International Symposium on Multiple-Valued Logic*, pages 168 – 185, Junio 2005.

[97] M. P. Frank and T. F. K. Jr. Ultimate theoretical models of nanocomputers. *Nanotechnology*, 9(3):162, Septiembre 1998.

[98] M. P. Frank and K. Shukla. Quantum Foundations of Classical Reversible Computing. *Entropy*, 23(6): 701, Junio 2021.

[99] E. Fredkin, R. Landauer, and T. Toffoli. Physics of Computation. *International Journal of Theoretical Physics*, 21(12):903–903, Diciembre 1982.

[100] E. Fredkin and T. Toffoli. Conservative Logic. *International Journal of Theoretical Physics*, 21:219–253, Abril 1982.

[101] B. Friedrich and H. Schmidt-Böcking. *Molecular Beams in Physics and Chemistry: From Otto Stern's Pioneering Exploits to Present-Day Feats*. Springer Nature, Junio 2021.

[102] M. Fürer. Faster Integer Multiplication. *Proceedings of the Annual ACM Symposium on Theory of Computing*, 39, Enero 2009.

[103] L. Gammaitoni. *The Physics of Computing*. Springer Nature, Octubre 2021.

[104] N. Garcia, A. Damask, and S. Schwarz. *Physics for Computer Science Students: With Emphasis on Atomic and Semiconductor Physics*. Springer Science & Business Media, Diciembre 2012.

[105] T. Gaudin, M.-C. Maurel, and J.-C. Pomerol. *Chance, Calculation and Life*. John Wiley & Sons, Abril 2021.

[106] C. F. Gauss and W. C. Waterhouse. *Disquisitiones Arithmeticae*. Springer, Febrero 2018.

[107] M. Gavrilov and J. Bechhoefer. Erasure without work in an asymmetric, double-well potential. *Physical Review Letters*, 117(20):200601, Noviembre 2016.

[108] K. C. Gilston 1856-1922. *Napier Tercentenary Memorial Volume*. 2013.

[109] M. C. Ginzburg. *Técnicas Digitales con Circuitos Integrados*. Reverte, 2002.

[110] R. Golub and S. K. Lamoreaux. Further steps to quantum mechanics: the old quantum mechanics of Bohr and Sommerfeld. In R. Golub and S. Lamoreaux, editors, *The Historical and Physical Foundations of Quantum Mechanics*, pages 66–77. Oxford University Press, Febrero 2023.

[111] M. T. Goodrich and R. Tamassia. *Algorithm Design: Foundations, Analysis, and Internet Examples*. John Wiley & Sons, Octubre 2001.

[112] R. L. Graham. *Handbook of Combinatorics*. Elsevier, Diciembre 1995.

[113] A. Granville. It is easy to determine whether a given integer is prime. *Bulletin of the American Mathematical Society*, 42:3–38, 2004.

[114] D. Gries and F. B. Schneider. *A Logical Approach to Discrete Math*. Springer Science & Business Media, Marzo 2013.

[115] D. H. Griffel. *Applied Functional Analysis*. Courier Corporation, Abril 2012.

[116] L. K. Grover. Quantum Mechanics Helps in Searching for a Needle in a Haystack. *Physical Review Letters*, 79(2):325–328, Julio 1997.

[117] L. K. Grover. From Schrödinger's equation to the quantum search algorithm. *Pramana*, 56(2-3):333–348, Febrero 2001.

[118] J. Gutiérrez and J. G. T. Ayuso. *Protocolos criptográficos y seguridad de redes*. Ed. Universidad de Cantabria, 2003.

[119] B. C. Hall. *Quantum Theory for Mathematicians*. Springer Science & Business Media, Junio 2013.

[120] L. Hardy. Quantum Theory From Five Reasonable Axioms, Septiembre 2001.

[121] D. Harvey, J. van der Hoeven, and G. Lecerf. Even faster integer multiplication. *Journal of Complexity*, 36:1–30, Octubre 2016.

[122] N. Hatano and M. Suzuki. Finding Exponential Product Formulas of Higher Orders. In A. Das and B. K. Chakrabarti, editors, *Quantum Annealing and Other Optimization Methods*, pages 37–68. Springer Berlin Heidelberg, Berlin, Heidelberg, 2005.

[123] W. Heisenberg. Über quantentheoretische Umdeutung kinematischer und mechanischer Beziehungen. *Zeitschrift für Physik*, 33(1):879–893, Diciembre 1925.

[124] W. Heisenberg. Über den anschaulichen Inhalt der quantentheoretischen Kinematik und Mechanik. *Zeitschrift für Physik*, 43(3):172–198, Marzo 1927.

[125] L. A. Hemaspaandra and A. L. Selman. *Complexity Theory: Retrospective II*. Springer Science & Business Media, Junio 1997.

[126] K. Hepp. Quantum theory of measurement and macroscopic observables. *Helvetica Physica Acta*, 45: 237–248, 1972.

[127] M. Hirvensalo. *Quantum Computing*. Springer Science & Business Media, Marzo 2013.

[128] S. Homer and A. L. Selman. *Computability and Complexity Theory*. Springer Science & Business Media, Diciembre 2011.

[129] J. Hromkovič. *Algorithmics for Hard Problems: Introduction to Combinatorial Optimization, Randomization, Approximation, and Heuristics*. Springer Science & Business Media, Marzo 2013.

[130] J.-P. Hsu and Y.-L. Wu. *Frontiers Of Physics At The Millennium, The, Proceedings Of The Symposium*. World Scientific, Abril 2001.

[131] E. V. Huntington. Sets of Independent Postulates for the Algebra of Logic. *Transactions of the American Mathematical Society*, 5(3):288–309, 1904.

[132] V. G. Ivancevic and T. T. Ivancevic. *Quantum Neural Computation*. Springer Science & Business Media, Enero 2010.

[133] A. J., A. Adedoyin, J. Ambrosiano, P. Anisimov, W. Casper, G. Chennupati, C. Coffrin, H. Djidjev, D. Gunter, S. Karra, N. Lemons, S. Lin, A. Malyzhenkov, D. Mascarenas, S. Mniszewski, B. Nadiga, D. O'Malley, D. Oyen, S. Pakin, L. Prasad, R. Roberts, P. Romero, N. Santhi, N. Sinitsyn, P. J. Swart, J. G. Wendelberger, B. Yoon, R. Zamora, W. Zhu, S. Eidenbenz, A. Bärtschi, P. J. Coles, M. Vuffray, and A. Y. Lokhov. Quantum Algorithm Implementations for Beginners. *ACM Transactions on Quantum Computing*, 3(4):1–92, Diciembre 2022.

[134] M. Jammer. *The Conceptual Development of Quantum Mechanics*. McGraw-Hill, 1966.

[135] E. T. Jaynes. Information Theory and Statistical Mechanics. *Physical Review*, 106(4):620–630, Mayo 1957.

[136] S. R. Jena and D. S. K. Swain. *Theory of Computation and Application (2nd Revised Edition)- Automata, Formal Languages and Computational Complexity: Implementation through JFLAP Simulator*. University Science Press, Laxmi Publications, New Delhi, Marzo 2020.

[137] P. Johnson. *El nacimiento del mundo moderno*. Javier Vergara, 2000.

[138] T. H. Johnson, S. R. Clark, and D. Jaksch. What is a quantum simulator?, Mayo 2014.

[139] Joyrup Bhattacharya. *Rudiments of computar science: Part 1*. Academic Publishers, tercera edition, 2016.

[140] J. F. N. Jr and M. T. Rassias. *Open Problems in Mathematics*. Springer, Julio 2016.

[141] C. Jönsson. Elektroneninterferenzen an mehreren künstlich hergestellten Feinspalten. *Zeitschrift für Physik*, 161(4):454–474, Agosto 1961.

[142] B. M. Kapron. *Logic, Automata, and Computational Complexity: The Works of Stephen A. Cook*. Morgan & Claypool, Mayo 2023.

[143] A. Karatsuba and Y. Ofman. Multiplicación de números de muchos digitales mediante computadoras automáticas(en ruso). *Actas de la Academia de Ciencias de la URSS*, 1945(2):293–294, 1962.

[144] V. Kasirajan. *Fundamentals of Quantum Computing: Theory and Practice*. Springer Nature, Junio 2021.

[145] P. Kaye, R. Laflamme, and M. Mosca. *An Introduction to Quantum Computing*. OUP Oxford, 2007.

[146] A. Kent and J. G. Williams. *Encyclopedia of Microcomputers: Volume 24 - Supplement 3: Characterization Hierarchy Containing Augmented Characterizations to Video Compression*. CRC Press, Octubre 1999.

[147] A. I. Khinchin and H. Eagle. *Continued Fractions*. Courier Corporation, Mayo 1997.

[148] L. B. Kish, S. P. Khatri, C. G. Granqvist, and J. M. Smulko. Critical remarks on Landauer's principle of erasure-dissipation: Including notes on Maxwell demons and Szilard engines. In *2015 International Conference on Noise and Fluctuations (ICNF)*, pages 1–4, Xian, China, Junio 2015. IEEE.

[149] T. Kleinjung, K. Aoki, J. Franke, A. Lenstra, E. Thomé, J. Bos, P. Gaudry, A. Kruppa, P. Montgomery, D. A. Osvik, H. Riele, A. Timofeev, and P. Zimmermann. Factorization of a 768-Bit RSA Modulus. volume 6223, pages 333–350, Agosto 2010.

[150] D. E. Knuth. *The Art of Computer Programming*. Addison Wesley Professional, Marzo 2009.

[151] P. M. Kogge. *The Zen of Exotic Computing*. SIAM, Diciembre 2022.

[152] T. Koshy. *Elementary Number Theory with Applications*. Elsevier, Mayo 2007.

[153] J. V. Koski, V. F. Maisi, J. P. Pekola, and D. V. Averin. Experimental realization of a Szilard engine with a single electron. *Proceedings of the National Academy of Sciences*, 111(38):13786–13789, Septiembre 2014.

[154] S. Kotiyal, H. Thapliyal, and N. Ranganathan. Circuit for Reversible Quantum Multiplier Based on Binary Tree Optimizing Ancilla and Garbage Bits. In *2014 27th International Conference on VLSI Design and 2014 13th International Conference on Embedded Systems*, pages 545–550, Enero 2014.

[155] H. Kragh. *Generaciones cuánticas*. Ediciones AKAL, Febrero 2007.

[156] A. A. Kumar. *Fundamentals of Digital Circuits*. PHI Learning Pvt. Ltd., Julio 2016.

[157] J. Ladyman and K. Robertson. Landauer Defended: Reply to Norton. *Studies in History and Philosophy of Science Part B: Studies in History and Philosophy of Modern Physics*, 44:263–271, Agosto 2013.

[158] J. Ladyman and K. Robertson. Going Round in Circles: Landauer vs. Norton on the Thermodynamics of Computation. *Entropy*, 16(4):2278–2290, 2014.

[159] D. Lairez. Thermodynamical versus logical irreversibility: a concrete objection to Landauer's principle. *Entropy*, 25(8):1155, Agosto 2023.

[160] B. J. LaMeres. *Introduction to Logic Circuits & Logic Design with Verilog*. Springer Nature, Octubre 2023.

[161] R. Landauer. Irreversibility and Heat Generation in the Computing Process. *IBM Journal of Research and Development*, 5(3):183–191, 1961.

[162] R. Landauer. Response to Comment on Energy requirements in communication. *Applied Physics Letters*, 52(25):2191–2192, Junio 1988.

[163] R. Landauer. Information is Physical. *Physics Today*, 44(5):23–29, Mayo 1991.

[164] G. Langholz, A. Kandel, and J. L. Mott. *Foundations of Digital Logic Design*. World Scientific, 1998.

[165] R. LaPierre. *Introduction to Quantum Computing*. Springer Nature, Septiembre 2021.

[166] J.-S. Lee, Y. Chung, J. Kim, and S. Lee. A Practical Method of Constructing Quantum Combinational Logic Circuits, Noviembre 1999.

[167] Y. Lee. Symmetric Trotterization in digital quantum simulation of quantum spin dynamics. *Journal of the Korean Physical Society*, 82(5):479–485, Marzo 2023.

[168] H. S. Leff. *Energy and Entropy: A Dynamic Duo*. CRC Press, Agosto 2020.

[169] H. S. Leff and A. F. Rex. *Maxwell's Demon 2 Entropy, Classical and Quantum Information, Computing*. CRC Press, Diciembre 2002.

[170] C. S. Lent, A. O. Orlov, W. Porod, and G. L. Snider. *Energy Limits in Computation: A Review of Landauer's Principle, Theory and Experiments*. Springer, Agosto 2018.

[171] X. Li, J. Wu, and X. Li. *Theory of Practical Cellular Automaton*. Springer, Mayo 2018.

[172] R. J. Lipton and K. W. Regan. *Introduction to Quantum Algorithms via Linear Algebra, second edition*. MIT Press, Abril 2021.

[173] S. Lloyd. Universal Quantum Simulators. *Science*, 273(5278):1073–1078, 1996.

[174] E. Lutz and S. Ciliberto. Information: From Maxwell's demon to Landauer's eraser. *Physics Today*, 68: 30–35, Septiembre 2015.

[175] M. López-Suárez, I. Neri, and L. Gammaitoni. Sub-kBT micro-electromechanical irreversible logic gate. *Nature Communications*, 7(1):12068, Junio 2016.

[176] J. MacCormick. *What Can Be Computed?: A Practical Guide to the Theory of Computation*. Princeton University Press, Mayo 2018.

[177] R. Manenti and M. Motta. *Quantum Information Science*. Oxford University Press, 2023.

[178] Y. I. Manin. *Computable e incomputable*. Sovetskoye radio (Radio Soviética), 1980.

[179] Y. I. Manin. Classical computing, quantum computing, and Shor's factoring algorithm, 1999.

[180] M. M. Mano. *Digital Design*. Pearson Educación, 2002.

[181] D. C. Marinescu. *Classical and Quantum Information*. Academic Press, Enero 2011.

[182] F. d. L. Marquezino, R. Portugal, and C. Lavor. *A Primer on Quantum Computing*. Springer, Junio 2019.

[183] L. Marton. Electron Interferometer. *Physical Review*, 85(6):1057–1058, Marzo 1952.

[184] J. M. Martyn, Z. M. Rossi, A. K. Tan, and I. L. Chuang. Grand Unification of Quantum Algorithms. *PRX Quantum*, 2(4):040203, Diciembre 2021.

[185] A. Maruoka. *Concise Guide to Computation Theory*. Springer Science & Business Media, Abril 2011.

[186] L. Masanes, M. P. Müller, R. Augusiak, and D. Pérez-García. Existence of an information unit as a postulate of quantum theory. *Proceedings of the National Academy of Sciences of the United States of America*, 110(41):16373–16377, Octubre 2013.

[187] L. Masanes, T. D. Galley, and M. P. Müller. The measurement postulates of quantum mechanics are operationally redundant. *Nature Communications*, 10(1):1361, Marzo 2019.

[188] J. Mattingly. *The SAGE Encyclopedia of Theory in Science, Technology, Engineering, and Mathematics*. SAGE Publications, Octubre 2022.

[189] J. C. Maxwell. *Theory of Heat*. Longmans, 1871.

[190] P. G. Merli, G. F. Missiroli, and G. Pozzi. On the statistical aspect of electron interference phenomena. *American Journal of Physics*, 44(3):306–307, Marzo 1976.

[191] N. D. Mermin. *Quantum Computer Science: An Introduction*. Cambridge University Press, Agosto 2007.

[192] T. Metodi and A. I. Faruque. *Quantum Computing for Computer Architects, Second Edition*. Springer Nature, Junio 2022.

[193] G. L. Miller. Riemann's hypothesis and tests for primality. *Journal of Computer and System Sciences*, 13 (3):300–317, 1976.

[194] J. C. P. Miller. On Factorisation, with a Suggested New Approach. *Mathematics of Computation*, 29(129): 155–172, Enero 1975.

[195] S. J. Miller and R. Takloo-Bighash. *An Invitation to Modern Number Theory*. Princeton University Press, Julio 2020.

[196] B. I. Mills. *Theoretical Introduction to Programming*. Springer Science & Business Media, Diciembre 2005.

[197] C. Miquel, J. P. Paz, and R. Perazzo. Factoring in a dissipative quantum computer. *Physical Review A*, 54 (4):2605–2613, Octubre 1996.

[198] T. Monz, K. Kim, W. Hänsel, M. Riebe, A. Villar, P. Schindler, M. Chwalla, M. Hennrich, and R. Blatt. Realization of the quantum Toffoli gate with trapped ions. *Physical Review Letters*, 102(4):040501, Enero 2009.

[199] C. Moore and S. Mertens. *The Nature of Computation*. OUP Oxford, Agosto 2011.

[200] G. E. Moore. Excerpts from A Conversation with Gordon Moore: Moore's Law.

[201] G. E. Moore. Cramming more components onto integrated circuits. 38(8), 1965.

[202] S. E. D. Morgan and A. D. Morgan. *Memoir of Augustus De Morgan: With Selections from His Letters*. Cambridge University Press, Julio 2010.

[203] E. Nagel and J. R. Newman. *Godel's Proof*. NYU Press, Enero 2008.

[204] M. Nakahara and T. Ohmi. *Quantum Computing: From Linear Algebra to Physical Realizations*. CRC Press, Marzo 2008.

[205] M. Nakahara and Y. Sasaki. *Quantum Information and Quantum Computing*. World Scientific, Septiembre 2012.

[206] National Academies of Sciences, Engineering, and Medicine 2019. *Quantum Computing: Progress and Prospects*. National Academies Press, Abril 2019.

[207] Niels Bohr. I. On the constitution of atoms and molecules. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, Julio 1913.

[208] M. A. Nielsen and I. L. Chuang. *Quantum Computation and Quantum Information: 10th Anniversary Edition*. Cambridge University Press, Diciembre 2010.

[209] K. Nikolić, M. Forshaw, and R. Compañó. The Current Status of Nanoelectronic Devices. *International Journal of Nanoscience*, 02(01n02):7–29, Febrero 2003.

[210] I. Niven, H. S. Zuckerman, and H. L. Montgomery. *An Introduction to the Theory of Numbers*. Wiley, 1991.

[211] J. D. Norton. Eaters of the lotus: Landauer's principle and the return of Maxwell's demon. *Studies in History and Philosophy of Science Part B: Studies in History and Philosophy of Modern Physics*, 36(2): 375–411, Junio 2005.

[212] L. Nottale and M.-N. Célérier. Derivation of the postulates of quantum mechanics from the first principles of scale relativity. *Journal of Physics A: Mathematical and Theoretical*, 40(48):14471–14498, Noviembre 2007.

[213] A. R. Omondi. *Cryptography Arithmetic: Algorithms and Hardware Architectures*. Springer Nature, Enero 2020.

[214] G. O'Regan. *A Brief History of Computing*. Springer Science & Business Media, Marzo 2012.

[215] X. Oriols and H. Nikolić. Three types of Landauer's erasure principle: a microscopic view. *The European Physical Journal Plus*, 138(3):250, Marzo 2023.

[216] A. Orlov, I. Hänninen, C. Campos-Aguillón, R. Celis Cordova, M. McConnell, G. Szakmany, C. Thorpe, B. Appleton, G. Boechler, C. Lent, and G. Snider. Experimental Tests of the Landauer Principle in Electron Circuits, and Quasi-Adiabatic Computing Systems: A Review of Landauer's Principle, Theory and Experiments. In *Energy Limits in Computation: A Review of Landauer's Principle, Theory and Experiments*, pages 177–230. Enero 2019.

[217] C. H. Papadimitriou. *Computational Complexity*. Addison-Wesley, 1994.

[218] A. Parkin. *Digital Imaging Primer*. Springer Science & Business Media, Septiembre 2015.

[219] J. M. R. Parrondo, J. M. Horowitz, and T. Sagawa. Thermodynamics of information. *Nature Physics*, 11 (2):131–139, Febrero 2015.

[220] A. Pathak. *Elements of Quantum Computation and Quantum Communication*. Taylor & Francis, Junio 2013.

[221] D. Patranabis. *Principles of Electronic Instrumentation*. PHI Learning Pvt. Ltd., Febrero 2008.

[222] A. Peres. Reversible logic and quantum computers. *Physical Review A*, 32(6):3266–3276, Diciembre 1985.

[223] Peter Rodgers. The double-slit experiment, Septiembre 2002.

[224] J. Peter Toennies. Otto Stern and Wave-Particle Duality. In B. Friedrich and H. Schmidt-Böcking, editors, *Molecular Beams in Physics and Chemistry: From Otto Stern's Pioneering Exploits to Present-Day Feats*, pages 519–545. Springer International Publishing, Cham, 2021.

[225] A. Petrenko and S. Petrenko. Applied Quantum Cryptanalysis.

[226] R. P. Poplavskii. Modelos termodinámicos de procesos de información (en ruso). *Soviet Physics Uspekhi (Avances Soviéticos en Física)*, 115(3):465–501, 1975.

[227] W. Porod. Comment on Energy requirements in communication. *Applied Physics Letters*, 52(25):2191, Junio 1988.

[228] W. Porod. The Thermodynamics of Computation: A Contradiction: A Review of Landauer's Principle, Theory and Experiments. In *Energy Limits in Computation: A Review of Landauer's Principle, Theory and Experiments*, pages 141–154. Springer, Enero 2019.

[229] W. Porod, R. O. Grondin, D. K. Ferry, and G. Porod. Dissipation in Computation. *Physical Review Letters*, 52(3):232–235, Enero 1984.

[230] J. P. Portillo, Tello. *Introducción a las señales y sistemas*. Universidad del Norte, Enero 2018.

[231] R. Portugal. *Quantum Walks and Search Algorithms*. Springer Science & Business Media, Febrero 2013.

[232] M. O. Rabin. Probabilistic algorithm for testing primality. *Journal of Number Theory*, 12(1):128–138, Febrero 1980.

[233] V. Rajaraman and T. Radhakrishnan. *Digital Logic and Computar Organization*. PHI Learning Pvt. Ltd., Enero 2006.

[234] M. D. Reed, L. DiCarlo, S. E. Nigg, L. Sun, L. Frunzio, S. M. Girvin, and R. J. Schoelkopf. Realization of Three-Qubit Quantum Error Correction with Superconducting Circuits. *Nature*, 482(7385):382–385, Febrero 2012.

[235] M. Reiher, N. Wiebe, K. M. Svore, D. Wecker, and M. Troyer. Elucidating Reaction Mechanisms on Quantum Computers.

[236] E. D. Reilly. *Concise Encyclopedia of Computer Science*. John Wiley & Sons, Septiembre 2004.

[237] P. Rendell. *Turing Machine Universality of the Game of Life*. Springer, Julio 2015.

[238] J. Resag. *Feynman and His Physics: The Life and Science of an Extraordinary Man*. Springer, Diciembre 2018.

[239] E. Rieffel and W. Polak. An Introduction to Quantum Computing for Non-Physicists. *ACM Computing Surveys*, 32(3):300–335, Septiembre 2000.

[240] H. P. Robertson. The Uncertainty Principle. *Physical Review*, 34(1):163–164, Julio 1929.

[241] C. E. Roger. *Métodos de solución y análisis de programación lineal*. Céspedes Esteban, Roger, Enero 2021.

[242] R. Rosa. The Merli–Missiroli–Pozzi Two-Slit Electron-Interference Experiment. *Physics in Perspective*, 14(2):178–195, 2012.

[243] S. Rubinstein-Salzedo. *Cryptography*. Springer, Septiembre 2018.

[244] T. Ryhänen, M. A. Uusitalo, O. Ikkala, and A. Kärkkäinen. *Nanotechnologies for Future Mobile Devices*. Cambridge University Press, Febrero 2010.

[245] S. K. Sarkar. *A Textbook of Discrete Mathematics (LPSPE)*. S. Chand Publishing, 2016.

[246] E. Schrödinger. An Undulatory Theory of the Mechanics of Atoms and Molecules. *Physical Review*, 28 (6):1049–1070, Diciembre 1926.

[247] B. Schumacher. Quantum coding. *Physical Review A*, 51(4):2738–2747, Abril 1995.

[248] A. Schönhage and V. Strassen. Schnelle Multiplikation großer Zahlen. *Computing*, 7(3):281–292, Septiembre 1971.

[249] J. Seberry and M. Yamada. *Hadamard Matrices: Constructions using Number Theory and Linear Algebra*. John Wiley & Sons, Agosto 2020.

[250] C. E. Shannon and W. Weaver. *The Mathematical Theory of Communication*. University of Illinois Press, 1949.

[251] O. R. Shenker. Logic and Entropy, Junio 2000.

[252] V. P. Shmerko, S. N. Yanushkevich, and S. E. Lyshevski. *Computer Arithmetics for Nanoelectronics*. CRC Press, Febrero 2009.

[253] P. Shor. Algorithms for quantum computation: discrete logarithms and factoring. In *Proceedings 35th Annual Symposium on Foundations of Computer Science*, pages 124–134, 1994.

[254] D. R. Simon. On the Power of Quantum Computation. *SIAM Journal on Computing*, 26(5):1474–1483, 1997.

[255] S. Singh. *El enigma de Fermat*. Grupo Planeta Spain, Febrero 2015.

[256] M. Sipser. *Introduction to the Theory of Computation*. Cengage Learning, Junio 2012.

[257] A. Smith, M. Kim, F. Pollmann, and J. Knolle. Simulating quantum many-body dynamics on a current digital quantum computer. *Nature Partner Journals Quantum Information*, 5, Diciembre 2019.

[258] J. C. Solem and L. C. Biedenharn. Understanding geometrical phases in quantum mechanics: An elementary example. *Foundations of Physics*, 23(2):185–195, Febrero 1993.

[259] R. M. Solovay and V. Strassen. A Fast Monte-Carlo Test for Primality. *SIAM J. Comput.*, 6:84–85, 1977.

[260] S. Somaroo, C. H. Tseng, T. F. Havel, R. Laflamme, and D. G. Cory. Quantum Simulations on a Quantum Computer. *Physical Review Letters*, 82(26):5381–5384, Junio 1999.

[261] A. Sommerfeld. *Atombau und Spektrallinien*. F. Vieweg & sohn akt.-ges., 1924.

[262] G. Stefanucci and R. v. Leeuwen. *Nonequilibrium Many-Body Theory of Quantum Systems: A Modern Introduction*. Cambridge University Press, Marzo 2013.

[263] D. R. Stinson and M. Paterson. *Cryptography: Theory and Practice*. CRC Press, Agosto 2018.

[264] D. F. Styer, M. S. Balkin, K. M. Becker, M. R. Burns, C. E. Dudley, S. T. Forth, J. S. Gaumer, M. A. Kramer, D. C. Oertel, L. H. Park, M. T. Rinkoski, C. T. Smith, and T. D. Wotherspoon. Nine formulations of quantum mechanics. *American Journal of Physics*, 70(3):288–297, Marzo 2002.

[265] M. Swan, R. P. D. Santos, and F. Witte. *Quantum Computing: Physics, Blockchains, And Deep Learning Smart Networks*. World Scientific, Marzo 2020.

[266] A. Syropoulos. *Hypercomputation: Computing Beyond the Church-Turing Barrier*. Springer Science & Business Media, Diciembre 2008.

[267] A. Syropoulos. *Demystifying Computation: A Hands-on Introduction*. World Scientific Publishing Company, Abril 2017.

[268] L. Szilard. Über die Entropieverminderung in einem thermodynamischen System bei Eingriffen intelligenter Wesen. *Zeitschrift für Physik*, 53(11):840–856, Noviembre 1929.

[269] L. Szilard. On the decrease of entropy in a thermodynamic system by the intervention of intelligent beings. *Behavioral Science*, 9(4):301–310, 1964.

[270] F. Tacchino, A. Chiesa, S. Carretta, and D. Gerace. Quantum computers as universal quantum simulators: state-of-art and perspectives. *Advanced Quantum Technologies*, 3(3):1900052, Marzo 2020.

[271] H. A. Taha. *Investigacion de operaciones*. Pearson Educación, 2004.

[272] J. Talbot and D. J. A. Welsh. *Complexity and Cryptography: An Introduction*. Cambridge University Press, Enero 2006.

[273] A. S. Tanenbaum. *Sistemas operativos modernos*. Pearson Educación, 2003.

[274] G. I. Taylor. Interference fringes with feeble light. volume 15, pages 114–115, 1909.

[275] G. Tenenbaum. *Introduction to Analytic and Probabilistic Number Theory*. American Mathematical Soc., Julio 2015.

[276] H. Terashima and M. Ueda. Nonunitary quantum circuit. *International Journal of Quantum Information*, 03(04):633–647, Diciembre 2005.

[277] G. P. Thomson and A. Reid. Diffraction of Cathode Rays by a Thin Film. *Nature*, 119(3007):890–890, Junio 1927.

[278] R. J. Tocci and N. S. Widmer. *Sistemas digitales: principios y aplicaciones*. Pearson Educación, 2003.

[279] A. Tonomura, J. Endo, T. Matsuda, T. Kawasaki, and H. Ezawa. Demonstration of single-electron buildup of an interference pattern. *American Journal of Physics*, 57(2):117–120, Febrero 1989.

[280] C. Torres. La filosofía y el programa de Hilbert. *Mathesis*, 5(1):33–55, 1989.

[281] S. Toyabe, T. Sagawa, M. Ueda, E. Muneyuki, and M. Sano. Experimental demonstration of information-to-energy conversion and validation of the generalized Jarzynski equality. *Nature Physics*, 6(12):988–992, Diciembre 2010.

[282] B. A. Trakhtenbrot. A Survey of Russian Approaches to Perebor (Brute-Force Searches) Algorithms. *Annals of the History of Computing*, 6(4):384–400, 1984.

[283] A. M. Turing. I.—Computing Machinery and Intelligence. *Mind*, LIX(236):433–460, Octubre 1950.

[284] M. van den Nest. Classical simulation of quantum computation, the Gottesman-Knill theorem, and slightly beyond, Octubre 2009.

[285] B. L. van der Waerden. *Sources of Quantum Mechanics*. Courier Corporation, Enero 2007.

[286] E. Viso and C. Peláez. *Introducción a las Ciencias de la Computación con JAVA*. UNAM, México, 2007.

[287] J. Von Neumann. *Mathematische Grundlagen der Quantenmechanik*. Springer, Berlin, Heidelberg, 1996.

[288] J. Vos. *Quantum Computing in Action*. Simon and Schuster, Febrero 2022.

[289] J. F. Wakerly. *Diseño Digital*. Pearson Educación, 2001.

[290] F. Z. Wang. Breaking Landauer's bound in a spin-encoded quantum computer. *Quantum Information Processing*, 21(11):378, Noviembre 2022.

[291] I. Wegener. *Complexity Theory: Exploring the Limits of Efficient Algorithms*. Springer Science & Business Media, Abril 2005.

[292] J. Welch, D. Greenbaum, S. Mostame, and A. Aspuru-Guzik. Efficient quantum circuits for diagonal unitaries without ancillas. *New Journal of Physics*, 16(3):033040, Marzo 2014.

[293] A. Wigderson. *Mathematics and Computation: A Theory Revolutionizing Technology and Science*. Princeton University Press, Octubre 2019.

[294] S. R. Wigderson, Avi. *Computational Complexity Theory*. American Mathematical Soc., 2004.

[295] A. G. Williams. *Quantum Field Theory: Classical Mechanics to Gauge Field Theories*. Cambridge University Press, Agosto 2022.

[296] C. P. Williams. *Explorations in Quantum Computing*. Springer Science & Business Media, Diciembre 2010.

[297] F. Winkler. *Polynomial Algorithms in Computer Algebra*. Springer Science & Business Media, Diciembre 2012.

[298] F. Wojcieszyn. *Introduction to Quantum Computing with Q# and QDK*. Springer Nature, Mayo 2022.

[299] W. K. Wootters and W. H. Zurek. A single quantum cannot be cloned. *Nature*, 299(5886):802–803, Octubre 1982.

[300] Z. Xu and J. Zhang. *Computational Thinking: A Perspective on Computer Science*. Springer Nature, Enero 2022.

[301] Y. Yamamoto and K. Semba. *Principles and Methods of Quantum Information Technologies*. Springer, Diciembre 2015.

[302] S. Y. Yan. *Computational Number Theory and Modern Cryptography*. John Wiley & Sons, Noviembre 2012.

[303] N. S. Yanofsky and M. A. Mannucci. *Quantum Computing for Computer Scientists*. Cambridge University Press, Agosto 2008.

[304] M. Ying. *Foundations of Quantum Programming*. Morgan Kaufmann, Marzo 2016.

[305] N. M. Zakaria. Binary Subdivision for Quantum Search, Enero 2011.

[306] E. Zapusek, A. Javadi, and F. Reiter. Nonunitary gate operations by dissipation engineering. *Quantum Science and Technology*, 8(1):015001, Enero 2023.

[307] C. Zheng. Universal quantum simulation of single-qubit nonunitary operators using duality quantum algorithm. *Scientific Reports*, 11:3960, Febrero 2021.

[308] A. Zujev. Note on Non-Unitary Quantum Gates in Quantum Computing, Enero 2017.

Casa abierta al tiempo
UNIVERSIDAD AUTÓNOMA METROPOLITANA

ACTA DE EXAMEN DE GRADO

No. 00151
Matrícula: 2212801497

Algoritmos Cuánticos.

DIDIER GAMALIEL BUENDIA ORTIZ
ALUMNO

REVISÓ

MTRA. ROSALIA SERRANO DE LA PAZ
DIRECTORA DE SISTEMAS ESCOLARES

En la Ciudad de México, se presentaron a las 12:00 horas del día 22 del mes de abril del año 2025 en la Unidad Iztapalapa de la Universidad Autónoma Metropolitana, los suscritos miembros del jurado:

DR. MOISES MARTINEZ MARES
DR. CARLOS FRANCISCO PINEDA ZORRILLA
DR. NORBERTO AQUINO AQUINO

Bajo la Presidencia del primero y con carácter de Secretario el último, se reunieron para proceder al Examen de Grado cuya denominación aparece al margen, para la obtención del grado de:

MAESTRO EN CIENCIAS (FÍSICA)

DE: DIDIER GAMALIEL BUENDIA ORTIZ

y de acuerdo con el artículo 78 fracción III del Reglamento de Estudios Superiores de la Universidad Autónoma Metropolitana, los miembros del jurado resolvieron:

Aprobar

Acto continuo, el presidente del jurado comunicó al interesado el resultado de la evaluación y, en caso aprobatorio, le fue tomada la protesta.

DIRECTOR DE LA DIVISIÓN DE CBI

Roman Linares Romero

DR. ROMAN LINARES ROMERO

PRESIDENTE

DR. MOISES MARTINEZ MARES

VOCAL

DR. CARLOS FRANCISCO PINEDA ZORRILLA

SECRETARIO

DR. NORBERTO AQUINO AQUINO