Apuntes de las clases de Física Cuántica
Juan Pablo Paz 2020

8

16 de mayo de 2020

Computación Cuántica

Veremos aquí una aplicación sencilla de los postulados de la Mecánica Cuántica que está en la base de una de las tecnologías más modernas en desarrollo. En efecto, describiremos el funcionamiento básico de las computadoras cuánticas y veremos que, para comprender sus aspectos esenciales es suficiente con utilizar lo que hasta aquí hemos aprendido. Cabe mencionar que las computadoras cuánticas fueron concebidas a mediados de la década de 1980 a partir del genio de Richard Feynman y de otros pocos pioneros (como David Deutsch, Charles Bennett, entre otros). Sin embargo, como mencionamos, su funcionamiento se basa en leyes que existían desde mucho antes. Por eso, nuevamente, cabe preguntarse el motivo por el cual no fueron concebidas con anterioridad a esa fecha. La respuesta, tal como sucedió con la Distribución Cuántica de Claves, se basa en el hecho de que, como veremos, para concebir una computadora cuántica es necesario aceptar la idea de que es posible controlar en forma muy precisa a un sistema cuántico individual (en realidad, a muchos de ellos). La computación cuántica requiere un control exquisito sobre el estado de un conjunto de sistemas físicos (átomos, por ejemplo), el control de su evolución temporal y la capacidad de medir su estado individualmente. Hasta que Feynman insistió en la necesidad de explorar esa frontera, los requerimientos de tal empresa parecían tan extremos que nadie se había animado a aventurarse en ese territorio.

8.1. Física y Computación

La motivación de Feynman para desarrollar el concepto en el que se funda la Computación Cuántica no estaba basada en su interés por resolver algún problema matemático o computacional abstracto. Por el contrario, Feynman propuso a las computadoras cuánticas como una necesidad para la física basándose en una

observación muy sencilla sobre la complejidad involucrada en la solución numérica de las ecuaciones de la Mecánica Cuántica. Vale la pena revisar el argumento de Feynman. Para mediados de la década de 1980 las computadoras estaban revolucionando el mundo y los científicos las utilizaban para múltiples propósitos. Para un físico, la computadora es algo así como un laboratorio en el cual es posible poner a prueba modelos basados en ecuaciones que no admiten una solución analítica exacta. El laboratorio numérico en realidad permite extraer predicciones del modelo matemático que luego deben ser contrastadas con la experiencia, lo que sirve para falsificar el modelo o, caso contrario, reafirmarlo. Pero para caracterizar los problemas numéricos que es posible resolver con una computadora, es fundamental evaluar la cantidad de recursos necesarios para hacerlo (cantidad de memoria en la computadora, cantidad de operaciones elementales necesarias, etc). Más en particular, lo esencial es analizar cómo depende el costo del cómputo en función del tamaño del problema a resolver.

Feynman notó que la dinámica de sistemas cuánticos es exponencialmente difícil de simular utilizando una computadora ordinaria (que definiremos más abajo, pero que aquí llamaremos "computadora clásica"). La observación de Feynman se basa en un hecho muy simple: Supongamos que queremos analizar la evolución de un sistema formado por N spines 1/2 (N sistemas de dos niveles) y supongamos que conocemos las interacciones entre sus componentes (o sea, conocemos el Hamiltoniano del sistema). Si nuestro objetivo es estudiar la dinámica del sistema (o sea, dado un estado inicial, queremos conocer el estado del sistema transcurrido un cierto tiempo T) es fácil ver que surgen dos graves inconvenientes. Por un lado, para guardar la información acerca del estado del sistema en la memoria de nuestra computadora se necesitan almacenar del orden de 2^N números complejos (ya que la dimensión del espacio de estados toma ese valor). Por otra parte, la dinámica involucra resolver numéricamente ese mismo número de ecuaciones acopladas, lo que en cierto modo implica que es necesario utilizar matrices de evolución cuya dimensión es $2^N \times 2^N$. Es decir, la simulación numérica de este problema requiere recursos que aumentan exponencialmente con el tamaño del sistema, en este sentido es exponencialmente difícil. Sin embargo, ¡en la naturaleza hay muchos sistemas físicos como este que no necesitan de una computadora tan grande para saber cómo deben evolucionar! Es decir, la naturaleza resuelve automáticamente la ecuación de Schrödinger (en un sentido metafórico) y encuentra su destino sin invertir recursos exponenciales.

Feynman vio esta dificultad como algo fundamental: si no es posible resolver numéricamente las ecuaciones fundamentales de la física (que son cuánticas) entonces nuestra capacidad de predecir está seriamente limitada. Para tener una idea de la gravedad del problema, basta decir que la solución numérica de la dinámica de un sistema de 50 spines acoplados entre sí en forma arbitraria, requeriría utilizar 2^{50} números complejos, para lo cual se requerirían cerca de más de 10^{15} números complejos en la memoria, una cantidad inalcanzable en sistemas modernos. Por lo demás, el escaleo exponencial muestra que agregando unos pocos espines al sistema, la situación empeora tan rápido que este problema no puede ser resuelto en la práctica. En los hechos, hasta el presente, el máximo número de

espines que se han incluido en una solución de este tipo no supera el medio centenar. El propio Feynman propuso una solución a este dilema: según dijo (durante una célebre conferencia que dictó durante una cena en un congreso en 1984) el problema es que estamos utilizando computadoras clásicas, que son objetos que evolucionan de acuerdo a reglas que son muy distintas de aquellas que rigen a los sistemas que queremos simular. Para resolver eficientemente las ecuaciones fundamentales de la física debemos entonces utilizar otro tipo de computadoras: las computadoras cuánticas.

Después del trabajo de Feynmann, las computadoras cuánticas comenzaron a ser estudiadas por un reducido grupo de científicos, sin duda este era un tema marginal dentro de la física. Una pregunta que comenzó a ser explorada era si, en el caso de contar algún día con computadoras cuánticas, sería posible utilizarlas para resolver algún problema matemático de aquellos considerados "duros" (en los que su solución también requiere recursos que escalan exponencialmente con el tamaño del mismo). Recién en 1994 un notable matemático llamado Peter Shor demostró que esto era posible: en efecto, Shor demostró que una computadora cuántica, de existir, podría resolver el problema de la factorización de los números enteros, invirtiendo un tiempo que depende polinomialmente del número de dígitos de la incógnita. Mientras que todos los algoritmos clásicos conocidos hasta ahora necesitan recursos que aumentan exponencialmente con la cantidad de dígitos. La relevancia del problema para la criptografía moderna, generó un interés enorme por este resultado y desató la carrera por construir una computadora cuántica. Esta carrera, pese a los grandes recursos que se han invertido en ella, todavía no ha concluido. En lo que sigue, haremos una breve introducción a la computación clásica y luego nos adentraremos en la definición y estudio de sus contrapartes cuánticas.

8.1.1. Computadoras clásicas

Una computadora es un objeto físico que sirve para almacenar y procesar información. Cabe notar que esta primera definición es natural para un físico, que no concibe a una computadora como algo abstracto sino como un objeto material en el que la información está representada en el estado físico de objetos (ver más abajo) y que cambia de acuerdo a reglas que, sin duda, deben ser compatibles con las leyes de la física. El modelo matemático en el que se funda la ciencia de la computación fue formulado por Alan Turing, que creó una representación abstracta para este tipo de dispositivo, que se conoce con el nombre de máquina de Turing. No describiremos aquí este enfoque sino que nos limitaremos a describir a las computadoras clásicas de otro modo, que puede demostrarse equivalente al introducido por Turing.

Pensaremos a la computadora como un dispositivo que consta de "cables" en los que la información está representada. Cada cable es portador de una unidad binaria de información: un bit, un objeto que sólo puede existir en dos estados que, arbitrariamente, denominamos "0" y "1". Los cables representan a los bits almacenados en la computadora (podemos imaginar un dispositivo real en el cual

la información está almacenada, por ejemplo, en un disco magnético y en ese caso los bits no están escritos en el estado de verdaderos cables sino que corresponden al estado de los granitos de material magnético que constituyen el disco, cada uno de los cuales representa un 0 o un 1 según sea la orientación del dipolo de cada dominio en alguna dirección convenientemente elegida). El proceso de cómputo puede pensarse entonces como un "circuito" tal como ilustra la Figura Fig. 8.1 en la cual el tiempo fluye de izquierda a derecha: El estado inicial de los cables es transformado de alguna manera, dependiendo del programa que la computadora ejecuta. La pregunta del problema a resolver debe escribirse en el estado inicial de los cables y, tras ejecutarse el programa, en los mismos cables puede leerse la respuesta a dicho problema. El programa, si bien puede pensarse en forma abstracta, tiene una representación física muy concreta: consiste en una secuencia de acciones físicas de modo tal que la entrada se modifica de acuerdo a ciertas reglas, elegidas para resolver el problema en cuestión.

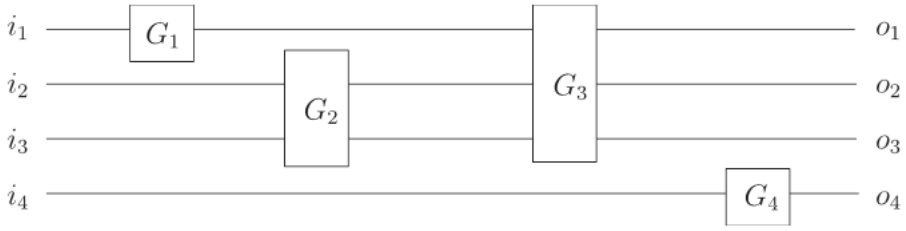

Figura 8.1: Modelo de circuitos, el tiempo fluye de izquierda a derecha. El estado inicial de cada uno de los cables lo llamamos i_j y el final o_j. Las cajas representan operaciones entre cables y definen el programa.

Para diseñar el programa es necesario abrir cada caja que aparece en la Figura 8.1 y escribir el circuito asociado a una dada tarea. Por ejemplo, si quisiéramos utilizar nuestra computadora para sumar dos números, deberíamos tener tantos cables como para poder escribir la representación binaria de estos números y luego diseñar un circuito que vaya ejecutando una serie de acciones sobre estos cables transformando su estado, de modo tal que al final del cómputo podamos leer el resultado correspondiente a la suma entre los números mirando algunos de los cables de salida. Teniendo en cuenta que el estado de los cables de entrada puede representarse como una K-upla binaria $\vec{n}_{in}$ (donde K es el número de cables, o número de bits de la memoria) y que el estado de la salida puede representarse como otra K-upla binaria $\vec{n}_{out}$, el proceso de cómputo siempre se corresponde con la evaluación de una función Booleana $f(\vec{n})$ cuyo dominio y su imágen son el conjunto de las K-uplas binarias. O sea, la computación mapea el estado de entrada $\vec{n}_{in}$ en el estado de salida $\vec{n}_{out}$ de modo tal que $\vec{n}_{out} = F(\vec{n}_{in})$. Construir un circuito asociado a la evaluación de una dada función Booleana es una tarea que puede ser complicada, pero siempre puede realizarse. Es más, hay un teorema importante que afirma que cualquier función Booleana $F(\vec{n})$ puede obtenerse a partir de la aplicación sucesiva de ciertas operaciones "elementales" entre bits tomados de a

uno o de a parres. Es decir, es posible demostrar que cualquier cómputo puede descomponerse en la aplicación de un cierto número de "compuertas universales" que, combinadas, siempre darán el resultado deseado. Estas compuertas universales pueden ser elegidas de varias maneras pero una opción conveniente es usar las compuertas "COPY" y "NAND". Es decir, si sabemos copiar bits (copiar el estado de un cable a otro) y sabemos ejecutar la operación que es la negación de la conjunción de dos bits (escribiendo la respuesta en un tercero) podemos combinar estas acciones para ejecutar cualquier cómputo.

En resumen, el modelo de computación clásica que brevemente mencionamos aquí, y que resulta conveniente para compararlo con su análogo cuántico, es el de la computadora como un circuito que es equivalente al célebre modelo de Alan Turing. Cabe aclarar que todos estos modelos de computación tienen algo en común, una hipótesis subyacente: son modelos clásicos ya que los bits siempre están en alguno de sus dos estados (0 ó 1) y nunca, como admite la cuántica, en estados que son superposiciones de ellos. Asimismo, la computadora evoluciona paso a paso recorriendo un conjunto de estados computacionales. Cada uno de los bits de una computadora cuántica tiene un estado bien definido en cada instante y la computadora recorre una trayectoria en el espacio abstracto de los estados computacionales. Dicho esto, resulta casi evidente cómo generalizar este modelo al caso cuántico definiendo entonces a las computadoras cuánticas.

8.1.2. Computadoras cuánticas

Una computadora cuántica, en general, podrá concebirse como un sistema cuántico capaz de almacenar información y procesarla. Si bien es posible generalizar los conceptos de máquina de Turing al caso cuántico, respetando el efoque que presentamos en el párrafo anterior, presentaremos a las computadoras cuánticas usando el denominado "modelo de circuitos". Al igual que en el caso clásico, podemos pensar que tenemos un conjunto de K sistemas de spín 1/2 y que podemos imaginar que cada uno de ellos se representa como un cable en un diagrama como el de la Figura anterior donde, recordemos, el tiempo fluye de izquierda a derecha. Cada cable es un sistema cuántico de dos niveles, al que denominaremos *qubit* (palabra que se utiliza para nombrar a un "bit cuántico"). Un qubit tendrá dos estados ortogonales a los que denominamos $|0\rangle$ y $|1\rangle$, estos estados forman lo que llamaremos la *base computacional* de dos qubits $\{|0\rangle, |1\rangle\}$ (para fijar ideas podemos pensar a estos estados asociados a $|0\rangle \equiv |0_z\rangle$ y $|1\rangle \equiv |1_z\rangle$ para una partícula de spin $1/2$). Pero además, a diferencia del caso clásico, un qubit no solamente podrá encontrarse en alguno de esos dos estados sino que su estado más general será una superposición de la forma $|\phi\rangle = \alpha|0\rangle + \beta|1\rangle$. Para una computadora cuántica con K qubits, el espacio de estados tiene dimensión 2^K y el estado más general puede escribirse usando una notación compacta como

$$|\Psi\rangle = \sum_{\vec{n}} c_{\vec{n}} \, |\vec{n}\rangle, \tag{8.1}$$

donde la sumatoria es sobre todas las K-uplas binarias en las que $\vec{n} = (n_1, \ldots, n_K)$ donde cada $n_j = 0, 1$.

En consecuencia, el primer ingrediente de las computadoras cuánticas son los qubits y la primera diferencia notable entre una computadora cuántica y una clásica es que la primera puede existir en infinitos estados, todas las superposiciones de estados $|\vec{n}\rangle$ a los que podemos denominar *estados computacionales* (notación que se origina en el hecho que estos son los estados en los que pueden existir las computadoras clásicas).

Para utilizar nuestra computadora cuántica lo primero que debemos hacer es preparar eficientemente algún estado computacional sencillo, por ejemplo aquel en el que todos los qubits están en el estado $|0\rangle$, que denominamos naturalmente $|\vec{0}\rangle$.

El segundo ingrediente que caracteriza a las computadoras cuánticas es su evolución. En ese sentido, una computadora cuántica universal es un objeto capaz de evolucionar con cualquier operador de evolución temporal U. Este operador es, tal como sucede en el caso clásico, lo que representa al programa ejecutado por la computadora. En consecuencia, una computadora cuántica para poder ser programable, debe poder ser manipulada por su operador de modo tal que el operador de evolución temporal sea arbitrario.

Como físicos, no estamos acostumbrados a la idea de poder controlar la evolución de un sistema cuántico obligándolo a que cambie su estado con un operador unitario U elegido por nosotros. Por el contrario, siempre pensamos que el operador evolución deviene de un dado Hamiltoniano que caracteriza al sistema y sus interacciones con el resto del universo. Sin embargo, la idea de la computación cuántica es precisamente explorar las posibilidades que se abren si fuéramos capaces de controlar la evolución e imponerla desde afuera. Para esto, como veremos, necesitaremos actuar sobre nuestro sistema "prendiendo y apagando" interacciones entre sus componentes o entre ellas y campos externos. Antes de explicar brevemente cómo podemos lograr esto, completemos la descripción de la computadora cuántica diciendo que debemos ser capaces de medir el estado final del conjunto de qubits en una cierta base. En efecto, supondremos que somos capaces de medir un observable que tiene a la base computacional como su base de autoestados. De la medición final, como veremos, extraemos información que puede servir para resolver algún problema matemático.

En síntesis, una computadora cuántica es un conjunto de qubits que satisface que:

i. podemos prepararlos en algún estado computacional,

ii. podemos obligarlos a evolucionar con un operador de evolución temporal U arbitrario,

iii. podemos medir al sistema en el estado final (proyectándolo sobre la base computacional).

Algunas aclaraciones son pertinentes en este momento. En primer lugar es notable una aparente diferencia entre la versión que expusimos de la computadora

cuántica y de su contraparte clásica. Por cierto, la evolución cuántica esta gobernada por un operador U que es unitario y, por lo tanto, reversible. En efecto, la computación cuántica es "reversible". En cambio, las computadoras clásicas que utilizamos no son reversibles, ni en el sentido físico (termodinámico) ni en el sentido lógico. La irreversibilidad lógica se manifiesta en el hecho de que, típicamente, cuando resolvemos un problema computacional, calculamos la salida (el resultado) del cálculo pero en el camino perdemos información sobre la entrada (la información superflua no es almacenada). La irreversibilidad física tiene que ver con el hecho de que las computadoras ordinarias disipan calor y, por consiguiente, no son reversibles desde el punto de vista termodinámico (y requieren energía externa para poder funcionar). Es interesante notar que, precisamente en la década de 1980, se desarrolló el campo de la computación clásica reversible. La motivación para estos trabajos, gestados por Rolf Landauer, Charles Bennett, Tommaso Toffoli, Edward Fredkin y otros, se basaba en el interés por determinar si era posible reducir el consumo de energía involucrado en un dado cómputo. Notablemente, estos autores demostraron que la respuesta a esta pregunta es afirmativa: no existe un límite inferior al consumo de energía, que puede ser tan bajo como uno quiera. Para que esto ocurra, demostraron que también es necesario que el cómputo sea lógicamente reversible (y que, por lo tanto, la entrada pueda recuperarse de la salida). La existencia de una relación íntima entre reversibilidad lógica y reversibilidad física fue una notable sorpresa que influyó mucho en el surgimiento de una rama de la física que se dió en llamar "la física de la información", que tuvo a Rolf Landauer (uno de los jefes científicos de los laboratorios de la empresa IBM) como uno de sus principales impulsores. Landauer demostró la validez de una conjetura formulada por él mismo: que todo cómputo puede realizarse en forma reversible a menos que se borre algún bit de la memoria y que, por cada bit borrado, se genera una entropía que es igual a $k_B \ln(2)$. No hablaremos aquí sobre este notable resultado pero queremos enfatizar que la computación clásica puede ser totalmente reversible y que esa propiedad no es, por lo tanto, una curiosidad de la computación cuántica.

Otra diferencia entre los modelos cuánticos y clásicos de la computación que acabamos de describir tiene que ver con el azar y el determinismo. La computación clásica, tal como la presentamos, es determinista: dada una entrada obtendremos una única salida. En cambio, el modelo de la computación cuántica es intrínsecamente no determinista: dado un estado de entrada, en el caso más general obtendremos varios estados a la salida, distribuidos con cierta probabilidad. Si bien esto aparece como una diferencia esencial, no lo es. Por un lado, hay modelos de computación clásica no-determinista, donde las salidas se encuentra distribuidas con cierta probabilidad. En estos casos, lo importante es que a pesar de ello, el cómputo nos ayude a resolver un problema específico. Lo cual puede suceder de dos formas: que la respuesta correcta sea significativamente más probable que las otras o que la respuesta esté escondida en alguna propiedad de la distribución de probabilidades de la salida. En ese caso, la computadora es un instrumento para "muestrear" una distribución de probabilidades. En el caso de las computadoras cuánticas esa es la situación típica: salvo en algunos casos excepcionales (algorit-

mos cuánticos que, como veremos más adelante, conducen a una única salida) la computadora cuántica funciona como un dispositivo para muestrear eficientemente una distribución de probabilidad a partir de la cual podemos descubrir la clave para resolver un problema matemático (o físico) relevante.

8.1.3. Evolución de una computadora cuántica: compuertas lógicas universales

Como mencionamos, para ejecutar un programa cuántico tenemos que lograr que la computadora evolucione con algún operador de evolución temporal particular. Así, como en el caso clásico existe un teorema que nos muestra que cualquier circuito puede construirse a partir de compuertas universales, existe una contraparte cuántica para este resultado. En efecto, es posible demostrar que cualquier operador unitario que actúa sobre un conjunto de K qubits puede obtenerse a partir de un número pequeño de operaciones elementales: el *conjunto de compuertas cuánticas universales*. Existen muchas elecciones posibles para este conjunto de compuertas pero una de las más convenientes es la siguiente. Para construir cualquier operador unitario se deben combinar adecuadamente:

1. *Operaciones unitarias arbitrarias que afecten a un único qubit a la vez*. Las denominaremos U_i, donde el índice i denota al qubit que es afectado por la operación. Más abajo mostraremos cómo es posible implementarla.

2. *Operaciones unitarias que afectan a dos qubits a la vez* (y que, por lo tanto, requieren la interacción entre ambos). Pese a que casi cualquier operación de dos qubits sirve para alcanzar la universalidad, es conveniente restringirse a utilizar una operación particular, que denominamos "Control-not" (C-NOT) y que denotamos $U_{CN,ij}$ donde los índices i y j corresponden a los qubits sobre los que U_{CN} actúa. Esta compuerta opera de modo tal que transforma los estados de la base computacional del siguiente modo:

$$\begin{aligned} U_{CN}|00\rangle &= |00\rangle, \quad U_{CN}|01\rangle = |01\rangle \\ U_{CN}|10\rangle &= |11\rangle, \quad U_{CN}|11\rangle = |10\rangle. \end{aligned} \tag{8.2}$$

   Es decir, el C-NOT nunca altera el estado del primer qubit, que actúa como un qubit de control. Si el control está en el estado $|0\rangle$ el segundo qubit no sufre ninguna modificación, mientras que si el estado del control es $|1\rangle$ podemos decir que sobre el segundo qubit aplica el operador representado por la matriz σ_x en la base $\{|0\rangle, |1\rangle\}$.

Combinando unitarias arbitrarias en qubits individuales con interacciones de a pares (del tipo C-NOT) podemos construir cualquier operación unitaria actuando sobre un conjunto arbitrario de qubits. Cabe aclarar que este conjunto de operaciones elementales no es el más práctico. Si bien, como veremos, es fácil concebir la forma en la cual implementar una unitaria general sobre un qubit, dado que

tenemos un continuo infinito de tales operadores resulta difícil aplicarlos con precisión adecuada. Es mucho más conveniente, en cambio, tener un conjunto finito y pequeño de compuertas elementales en términos de las cuales sea posible construir al resto. En efecto, esto es factible pero debe hacerse con cuidado. Se puede demostrar que es posible "acercarse tanto como uno quiera" a cuaquier operación unitaria sobre K qubits aplicando C-NOT entre cualquier par de ellos y un conjunto de dos operaciones unitarias sobre cada qubit. Estas operaciones son las siguientes:

(i) la transformación de Hadamard (U_H) que en la base computacional actúa de la siguiente manera:

$$U_H|0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle), \quad \text{y} \quad U_H|1\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle). \tag{8.3}$$

(ii) la compuerta $\pi/8$ (que llamaremos T) realiza una rotación en un ángulo de $\pi/4$ alrededor del eje $\vec{e}_z$.

$$T|0\rangle = |0\rangle, \quad \text{y} \quad T|1\rangle = e^{i\pi/4}|1\rangle. \tag{8.4}$$

El nombre compuerta $\pi/8$, se debe a una cuestión histórica, por lo que simplemente podemos hablar de compuerta T.

Estas tres compuertas $\{U_H, T, \text{C}-\text{NOT}\}$ forma un conjunto universal de compuertas en el sentido mencionado. Es decir, se puede mostrar que si quisiéramos aplicar una compuerta U pero aplicamos una secuencia de operaciones de este conjunto que nos da V como aproximación, y definimos el error en la aproximación como $E(U,V) \equiv \text{máx}_{|\Psi\rangle} \|(U-V)|\psi\rangle\|$, donde la maximización es sobre todo estado normalizado. Entonces, dado $\epsilon > 0$, es posible construir una operación unitaria V en términos de una secuencia de compuertas del conjunto $\{U_H, T, \text{C}-\text{NOT}\}$ tal que $E(U,V) < \epsilon$.

8.1.4. Hamiltoniano para un Control-not

Veremos aquí cómo podemos implementar físicamente las operaciones elementales necesarias para construir cualquier programa cuántico. En primer término, es fácil demostrar que cualquier operador unitario actuando sobre un qubit (que por ahora podemos pensarlo como una partícula de spin 1/2) puede ser generado aplicando un campo magnético apropiadamente elegido y que sólo afecte a ese qubit. En efecto, cualquier operador unitario que actúa sobre un único qubit se puede escribir de la forma $U = e^{-i\hat{H}t/\hbar}$ para algún operador $\hat{H}$ que siempre puede escribirse como combinación lineal de la identidad y las matrices de Pauli. Expresando este operador de ese modo y dejando de lado el término proporcional a la identidad (que actúa trivialmente sobre todos los estados), concluimos que un operador unitario arbitrario sobre un qubit siempre es de la forma

$$U = e^{-i(\Omega\vec{n}\cdot\vec{\sigma})t}$$

para algún versor $\vec{n}$. En consecuencia, esta observación nos conduce a la conclusión de que la operación unitaria más general para un qubit siempre puede implementarse "prendiendo" un campo magnético en alguna dirección apropiada, durante un tiempo convenientemente elegido. Entonces, para poder implementar cualquier operador de evolución temporal necesitamos ser capaces de prender y apagar campos magnéticos que afecten individualmente a los qubits (y estos campos deben tener direcciones e intensidades controlables). En particular, la compuerta de Hadamard es la composición de dos rotaciones $H = e^{i\frac{\pi}{2}\sigma_x}\, e^{i\frac{\pi}{4}\sigma_y}$.

La implementación de la compuerta C-NOT es un poco más complicada ya que requiere de la interacción de dos qubits. Nos limitaremos a describirla suponiendo que nuestro sistema consta solamente de estos dos qubits (omitiremos al resto de ellos que, durante la ejecución de esta operación, se mantienen inalterados). Por conveniencia, y para simplificar la presentación, en lugar de describir la forma de implementar al C-NOT, explicaremos cómo implementar un pariente cercano de esta compuerta que podríamos denominar "C-(i NOT)" (junto a las operaciones de un qubit también forma parte de un conjunto universal). Esta compuerta, cuyo operador unitario denominaremos como U_{C-iX}, actúa sobre la base computacional de la siguiente manera:

$$U_{C-(iX)}|00\rangle = |00\rangle, \qquad U_{C-(iX)}|01\rangle = |01\rangle, \tag{8.5}$$

$$U_{C-(iX)}|10\rangle = i|11\rangle, \qquad U_{C-(iX)}|11\rangle = i|10\rangle. \tag{8.6}$$

Es decir, aplica el operador representado por la matriz $i\sigma_x$ al segundo qubit si el estado del primero es $|1\rangle$ y la identidad cuando el estado de este qubit de control es $|0\rangle$. Es simple escribir este operador unitario explícitamente como:

$$U_{C-(iX)} = |0\rangle\langle 0| \otimes \mathbb{1} + i|1\rangle\langle 1| \otimes \sigma_x. \tag{8.7}$$

Asimismo, usando el formalismo que hemos visto hasta ahora, podemos demostrar que este operador puede escribirse como la siguiente exponencial:

$$U_{C-(iX)} = \exp\left(-i\frac{3\pi}{2}\,|1\rangle\langle 1| \otimes \sigma_x\right). \tag{8.8}$$

Para demostrar la validez de esta identidad se puede desarrollar la exponencial como suma de potencias del exponente y luego separar las potencias pares de las impares. Parte del orden cero $\mathbb{1} \otimes \mathbb{1} = |0\rangle\langle 0| \otimes \mathbb{1} + |1\rangle\langle 1| \otimes \mathbb{1}$ contribuye al primer término de la Ec. (8.7) y el resto a las potencias pares que resulta ser proporcional al $\cos(3\pi/2)$, y se anula. En cambio, la contribución de las potencias impares tiene la forma requerida proporcional a $-i\sin(3\pi/2)$ (la demostración de esta identidad es un ejercicio recomendable). Lo anterior, demuestra que la operación $U_{C-(iX)}$ puede derivarse de un Hamiltoniano, que simplemente puede leerse a partir de la expresión de esta compuerta como una exponencial. En efecto, tenemos que

$$itH_{C-(iX)}/\hbar = \frac{3\pi}{2}|0\rangle\langle 0| \otimes \sigma_x. \tag{8.9}$$

Teniendo en cuenta que $|0\rangle\langle 0| = (\mathbb{1} - \sigma_z)/2$ el Hamiltoniano anterior puede concebirse como representando la interacción entre dos espines de la siguiente forma

$$itH_{C-(iX)}/\hbar = \frac{3\pi}{4}(\mathbb{1} \otimes \sigma_x - \sigma_z \otimes \sigma_x) \tag{8.10}$$

Es decir, para ejecutar una compuerta del tipo C-(iX) debemos lograr que los dos qubits interactúen durante un tiempo con el Hamiltoniano anterior.

Esto obviamente no es trivial pero no es imposible de concebir. La computación cuántica requiere que aceptemos esta idea: supondremos que podemos prender y apagar interacciones entre los qubits y campos externos así como también interacciones entre los qubits que son descriptas por los Hamiltonianos anteriores (naturalmente, para apagar las interacciones podríamos alejar los qubits convenientemente pero para prenderlas no solamente debemos acercarlos sino que también debemos lograr que la interacción efectiva está descripta por el Hamiltoniano anterior). La forma de lograr esto dependerá del sustrato que utilicemos para construir nuestra computadora. En particular, podemos pensar Hamiltonianos más sencillos que nos permitan generar compuertas análogas al CNOT, por ejemplo: una compuerta que se utiliza mucho es la control-Z definida por el operador U_{CZ} tal que: $U_{CZ}|00\rangle = |00\rangle$, $U_{CZ}|01\rangle = |01\rangle$, $U_{CZ}|10\rangle = |10\rangle$, $U_{CZ}|11\rangle = -|11\rangle$ (aplica el operador σ_z sobre el segundo qubit si el primero se encuentra en 1). Es fácil mostrar que podemos reemplazar esta compuerta por el CNOT en el conjunto universal ya que se encuentran relacionadas de la siguiente manera: $U_{CNOT} = (1 \otimes H)U_{CZ}(1 \otimes H)$. En este caso, dicha compuerta puede ser implementada (a menos de una fase global) mediante la combinación de diferentes evoluciones de un qubit más una interacción entre los qubits generada por un Hamiltoniano bien sencillo tipo Ising: $U_{CZ} = \sqrt{i}\ e^{i\frac{\pi}{4}\sigma_z\otimes\sigma_z}e^{-i\frac{\pi}{4}\sigma_z\otimes\mathbb{1}}e^{-i\frac{\pi}{4}\mathbb{1}\otimes\sigma_z}$, queda como ejercicio demostrar la igualdad anterior.

Notablemente, en la actualidad existen muchas propuestas para lograr esto (o alguna forma de interacción, en algunos casos mediada por un tercer sistema), que permite implementar la compuerta CNOT o alguna similar y que, por lo tanto, permite la construcción de un operador unitario genérico.

8.2. Algoritmos Cuánticos

Luego de las ideas de Feynman, parte de la comunidad científica comenzó a estudiar las consecuencias de pensar a una computadora como un dispositivo que siguiera las leyes de la mecánica cuántica. Si además de permitir simular la dinámica de sistemas físicos interesantes, era posible encontrar alguna ventaja fundamental para el cálculo computacional. En esta búsqueda, se extendieron conceptos de teoría de computación clásica a este nuevo paradigma. En 1985, David Deutsch introdujo la noción de máquina de Turing cuántica y estudió cuáles eran las generalizaciones de las compuertas lógicas necesarias para construir computadoras cuánticas universales. Sin embargo, llevo algunos años más encontrar tareas en las cuales una computadora cuántica permitiera mejorar los algoritmos clásicos conocidos hasta ese momento. También David Deutsch, en 1992, presentó una serie

de problemas (que discutiremos a continuación) en los cuales existía una ventaja cuántica. Estos problemas no era de gran utilidad, pero permitían avizorar efectivamente el poder de estos dispositivos. Unos años más tarde, en 1994, el área sufrió un cambio radical cuando Peter Shor mostró que el problema de factorizar números enteros podría ser resuelto eficientemente por una computadora cuántica. A primera vista parece un problema matemático abstracto, sin embargo tiene aplicaciones importantes. En efecto, el sistema criptográfico RSA es el sistema más usado en la industria y entes gubernamentales para encriptar información sensible. La seguridad del sistema se basa en asumir que para una computadora es difícil factorizar números grandes. Esto es, no se conocen algoritmos que encuentren los factores de un número de n dígitos en un tiempo que sea polinomial en n. Si esto fuera posible comprometería la seguridad de la mayoría de las comunicaciones, y es por esta razón que la aparición del algoritmo de Shor produjo un cambio radical en el área. Esto no sólo ocurrió desde el punto de vista teórico, sino estratégico, este resultado despertó el interés de diferentes entes estatales y privados en el área que comenzaron a financiar la carrera por construir una computadora cuántica (carrera que sigue en marcha). A partir de allí se fue incrementando el interés en la computación cuántica y nuevos algoritmos se fueron desarrollando, tales como el algoritmo de Grover en 1996 para la búsqueda en una base de datos desordenada que provee una reducción polinomial en la complejidad. Sin embargo, encontrar algoritmos que tengan mejoras sustanciales sobre los clásicos no es una tarea sencilla. En la actualidad existen varios algoritmos cuánticos que proveen mejoras respecto de sus contrapartes clásicas, y una descripción de ellos puede encontrarse en http://quantumalgorithmzoo.org.

A continuación describiremos los primeros algoritmos cuánticos sencillos que dieron dieron comienzo al estudio de las computadoras cuánticas como dispositivos efectivamente capaces de mejorar la capacidad de cómputo conocida hasta el momento. Los algoritmos que presentaremos pueden no tener una aplicación útil inmediata pero nos servirán para ilustrar el potencial de estos dispositivos para el proceso de la información. Estos algoritmos buscan resolver problemas basados en oráculos o cajas negras, son dispositivos capaces de evaluar funciones, pero su funcionamiento se encuentra oculto para el ente que lo ejecuta. Simplemente uno puede operar con ellos proveyendo un input y recibiendo un output como respuesta. De esta manera, los problemas propuestos buscan responder preguntas acerca de determinadas propiedades de la función que se evalúa "consultando" al oráculo. En este caso se definirá a la complejidad del algoritmo como la mínima cantidad de veces que es necesario recurrir al oráculo para responder a la pregunta en cuestión. Antes de ir a los ejemplos, dedicaremos algunas líneas a la notación que adoptaremos en este contexto.

En primer lugar, utilizaremos la notación binaria para representar inputs y outputs clásicos. La representación de n bits o variables lógicas será por medio de una n-upla de 0's y 1's:

$$\mathbf{x} = (x_1, \ldots, x_n), \quad \text{con} \quad x_i \in \{0, 1\}. \tag{8.11}$$

Podremos también asociar a cada una de ellas un número natural $x = x_n 2^{n-1} + \cdots +$

$x_1 2^0$. Con n bits entonces podremos representar 2^n números enteros, que van desde $x = 0$ hasta $x = 2^n - 1$. Definiremos ahora la operación suma lógica de cadenas, dadas $\mathbf{x}$ y $\mathbf{y}$:

$$\mathbf{x} \oplus \mathbf{y} = (x_1 \oplus y_1, \ldots, x_n \oplus y_n) = \mathbf{z}, \tag{8.12}$$

que es simplemente la suma binaria o la operación XOR de cada par de elementos ($0 \oplus 0 = 1 \oplus 1 = 0$ y $0 \oplus 1 = 1 \oplus 0 = 1$), y además podemos ver que la cadena resultante es la asociada a la suma módulo 2^n de esos elementos, $z = x + y \pmod{2^n}$. Los algoritmos que veremos se encuentran relacionados con el estudio de propiedades de funciones entre cadenas binarias:

$$f : \{0,1\}^n \to \{0,1\}^m \tag{8.13}$$

que van de n bits a m bits en general.

Esta misma notación es la que utilizaremos para identificar a cada elemento de la base computacional de n qubits:

$$|\mathbf{x}\rangle \equiv |x_1 \ldots x_n\rangle, \;\; \text{con} \;\; x_i \in \{0,1\}. \tag{8.14}$$

Otra notación que se utiliza frecuentemente para describir a los elementos de la base computacional de n qubits se encuentra relacionada con el número natural que define cada cadena, donde por ejemplo $|0\rangle \equiv |0\ldots00\rangle$, $|1\rangle \equiv |0\ldots01\rangle$, $|2\rangle \equiv |0\ldots10\rangle$, etc.

8.2.1. Paralelismo cuántico

Introduciremos aquí una propiedad que tienen los dispositivos cuánticos y que será explotada por los algoritmos que presentaremos. Con este fin, es conveniente introducir en primer lugar el concepto de oráculo o caja negra.

Como anticipamos, un *oráculo* o caja negra es un dispositivo que permite evaluar una función desconocida. Si consideramos un oráculo clásico reversible, podemos definir su acción en forma sencilla: dada una función $f : \{0,1\}^n \to \{0,1\}^m$ y dos cadenas binarias como entradas $(\mathbf{x}, \mathbf{y}) \to (\mathbf{x}, \mathbf{y} \oplus f(\mathbf{x}))$, es decir, el oráculo recibe la entrada y en la salida utiliza el segundo registro para escribir el valor de la función $f(\mathbf{x})$. (sumando la entrada $\mathbf{x}$ al segundo registro obtenemos $\mathbf{y} \oplus (\mathbf{y} \oplus f(\mathbf{x})) = f(\mathbf{x})$). El tipo de problemas que se definen utilizando este dispositivo asumen que no es posible abrir dispositivo para obtener información acerca de su funcionamiento y determinar por ejemplo cuál es la función, por esa razón se adopta el nombre de oráculo o caja negra.

Un *oráculo cuántico* es un dispositivo similar, sólo que admite estados cuánticos como entradas y salidas. Dado que los oráculos realizan cómputos reversibles, podemos reemplazar cada compuerta lógica reversible del oráculo clásico por su análogo unitario. De esta forma, la versión cuántica del oráculo será una operación unitaria U_f sobre estados cuánticos de n qubits de entrada $|\mathbf{x}\rangle|\mathbf{y}\rangle$ tal que:

$$U_f : |\mathbf{x}\rangle|\mathbf{y}\rangle \to |\mathbf{x}\rangle|\mathbf{y} \oplus f(\mathbf{x})\rangle. \tag{8.15}$$

La acción de los oráculos clásicos es sencilla, para cadena binaria de entrada se corresponde con un único punto donde el oráculo evalúa la función. Veamos qué es lo que sucede con oráculos cuánticos. El análogo de una entrada definida para un oráculo cuántico es un estado puro. Si ingresamos estados de la base computacional, tendremos una respuesta que coincide exactamente con el caso clásico. Sin embargo, podemos pensar en estados más generales. Una manera de generar estados más generales de la base computacional es, como vimos, mediante la operación unitaria Hadamard U_H Ec. (8.3). Por ejemplo, consideremos un estado arbitrario de la base computacional al que le aplicamos un Hadamard sobre cada qubit:

$$\begin{aligned} U_H^{\otimes n}|\mathbf{x}\rangle &= U_H^{\otimes n}|x_1 x_2 \dots x_n\rangle = \left(\frac{1}{\sqrt{2}}\right)^n (|0\rangle + (-1)^{x_1}|1\rangle)(|0\rangle + (-1)^{x_2}|1\rangle)\cdots(|0\rangle + (-1)^{x_n}|1\rangle) \\ &= \frac{1}{\sqrt{2^n}} \sum_{\mathbf{z}\in\{0,1\}^n} (-1)^{\mathbf{x}\cdot\mathbf{z}}|\mathbf{z}\rangle. \end{aligned} \tag{8.16}$$

Esta última igualdad la obtuvimos expandiendo la expresión anterior, y además definiendo el producto módulo 2 entre cadenas como $\mathbf{x}\cdot\mathbf{z} = x_1 z_1 \oplus x_2 z_2 \oplus \cdots \oplus x_n z_n$ para simplificar la notación. El resultado entonces es un estado *superposición* lineal de *todos* los elementos de la base computacional con fases que dependen del estado inicial. De especial interés, como veremos, será ver qué sucede cuando todos los qubits se encuentran en $|0\rangle$. En este caso:

$$U_H^{\otimes n}|\mathbf{0}\rangle = U_H^{\otimes n}|00\dots0\rangle = \frac{1}{\sqrt{2^n}} \sum_{\mathbf{x}\in\{0,1\}^n} |\mathbf{x}\rangle. \tag{8.17}$$

Tenemos entonces un estado que es la superposición lineal de todos los valores donde la función se encuentra definida y todos con la misma fase relativa. Si este estado es la entrada de un oráculo cuántico, su salida será:

$$U_f\left[(U_H^{\otimes n})|\mathbf{0}\rangle|\mathbf{0}\rangle\right] = \frac{1}{\sqrt{2^n}} \sum_{\mathbf{x}\in\{0,1\}^n} |\mathbf{x}\rangle|f(\mathbf{x})\rangle.$$

En general, dependiendo de la función, este es un estado entrelazado entre los dos registros, y existe una correlación perfecta entre un dado $\mathbf{x}$ y el valor de la función en la mismo. A esta característica se la denomina *paralelismo cuántico*. Es decir, con una única consulta al oráculo, obtenemos un estado que tiene información completa acerca de la función. Este hecho resulta sorprendente ya que para adquirir esta información clásicamente requeriríamos una cantidad de consultas exponencial en la cantidad de bits de la cadena. Sin embargo, como dijimos, el estado es el que contiene o codifica la información acerca de la función, y extraer esa información es una tarea que no es trivial. La manera que conocemos de obtener la información codificada en un estado es mediante una medición. Si realizamos una medición del primer registro y, por ejemplo, obtenemos $\mathbf{x}_0$, el estado luego de la medición será

$$|\mathbf{x}_0\rangle|f(\mathbf{x}_0)\rangle,$$

luego, midiendo el segundo registro, sólo podremos determinar el valor de la función en dicho punto. En efecto, luego de la medición habremos destruido las correlaciones que habíamos generado en el estado. Es decir, mediante esta estrategia ¡estaremos efectuando un sampleo de la función de manera completamente aleatoria!, y no obtendremos ninguna ventaja respecto de su estrategia clásica. La mayoría de los algoritmos cuánticos explotan esta propiedad pero, como veremos a continuación, el éxito de los mismos reside en diseñar la forma de extraer información de una manera más eficiente.

8.2.2. Algoritmo de Deutsch

Comenzaremos describiendo un algoritmo bien sencillo que nos permitirá apreciar las propiedades que recientemente discutimos. Consideremos un oráculo clásico que permite evaluar una función desconocida de 1 bit, $f : \{0,1\} \to \{0,1\}$. Es decir dicha función: dada la entrada de dos registros binarios (x, y), el oráculo actúa como: $y \oplus (y \oplus f(x)) = f(x)$). Como dijimos, de acuerdo a las reglas del juego, no está permitido abrir dispositivo para obtener información acerca de su funcionamiento. Podemos notar, entonces, que existen sólo cuatro funciones de este estilo (ya que $f(0)$ puede tomar dos valores, y lo mismo para $f(1)$). En particular, es fácil darse cuenta que esas cuatro funciones pueden agruparse en dos conjuntos, que llamamos funciones *constantes* (las salidas de los dos inputs son iguales) y *balanceadas* (las salidas son distintas). El algoritmo de Deutsch resuelve el siguiente problema:

- Dada una función desconocida $f : \{0,1\} \to \{0,1\}$ y un oráculo que nos permite evaluarla, determinar si esa función es constante o balanceada. Esto equivale simplemente a evaluar $f(0) \oplus f(1)$, ya que si el resultado es 0 entonces f es constante, y en el otro caso f es balanceada.

Entonces, ¿cuántas veces debemos recurrir al oráculo para responder la pregunta? Clásicamente, la respuesta es inmediata, necesitamos hacerlo 2 veces. Con una vez es imposible, ya que sabiendo el valor de la función para 0 o 1 no podremos evaluar $f(0) \oplus f(1)$. A continuación, mostraremos que el algoritmo de Deutsch permite resolver el problema con una *única* evaluación.

Ahora consideremos que tenemos un oráculo cuántico U_f que admite estados de dos qubits $|x\rangle|y\rangle$ tal que:

$$U_f : |x\rangle|y\rangle \to |x\rangle|y \oplus f(x)\rangle. \tag{8.18}$$

Donde $|x\rangle$ e $|y\rangle$ son estados de la base computacional del primer y segundo qubit respectivamente. Notemos que si como entradas elegimos a los estados $|0\rangle$ o $|1\rangle$ para cualquiera de los registros, estaremos utilizando la estrategia clásica para resolver el problema. Se deja como ejercicio mostrar los circuitos cuánticos asociados al oráculo de cada una de las 4 funciones combinando compuertas σ_x y $CNOT$.

Veamos ahora qué es lo que sucede si en el segundo registro comenzamos con una superposición de estados, en particular donde el input es un estado de la forma

$|x\rangle(|0\rangle - |1\rangle)/\sqrt{2}$. En este caso podemos ver que dado (8.18):

$$\begin{aligned} U_f\left(|x\rangle\,\frac{(|0\rangle-|1\rangle)}{\sqrt{2}}\right) &= U_f\left(\frac{|x\rangle|0\rangle-|x\rangle|1\rangle}{\sqrt{2}}\right) = \frac{|x\rangle|0\oplus f(x)\rangle-|x\rangle|1\oplus f(x)\rangle}{\sqrt{2}} \\ &= (-1)^{f(x)}\,|x\rangle\left(\frac{|0\rangle-|1\rangle}{\sqrt{2}}\right), \end{aligned} \tag{8.19}$$

En el último paso usamos que, si $f(x) = 0$ la transformación actúa sobre el estado como la identidad, mientras que si $f(x) = 1$ agrega una fase global -1. En resumen, sobre este estado inicial, la transformación aporta una fase global igual a $(-1)^{f(x)}$, pero por ahora no es posible apreciar alguna ventaja respecto de la estrategia clásica.

El algoritmo de Deutsch se basará en considerar una entrada levemente diferente y su diseño puede resumirse en el circuito de la Fig. 8.2. Veamos entonces cómo construir el estado de entrada al oráculo $|\psi_1\rangle$ y cómo es la evolución paso a paso del algoritmo:

0. Comenzaremos con un estado producto:

$$|\psi_0\rangle = |0\rangle|1\rangle.$$

1. Preparamos el estado de entrada a partir de la aplicación de una compuerta Hadamard (8.3) sobre cada subsistema:

$$\begin{aligned} |\psi_1\rangle &= \left[\frac{|0\rangle+|1\rangle}{\sqrt{2}}\right]\left[\frac{|0\rangle-|1\rangle}{\sqrt{2}}\right] \\ &= \frac{|0\rangle}{\sqrt{2}}\left(\frac{|0\rangle-|1\rangle}{\sqrt{2}}\right)+\frac{|1\rangle}{\sqrt{2}}\left(\frac{|0\rangle-|1\rangle}{\sqrt{2}}\right). \end{aligned}$$

   Notemos que en este paso el estado de entrada es una combinación lineal de todas las entradas clásicas posibles.

2. En este paso $|\psi_2\rangle = U_f|\psi_1\rangle$ que, a partir de la expresión de la Ec. (8.19), resulta ser:

$$\begin{aligned} |\psi_2\rangle &= (-1)^{f(0)}\,\frac{|0\rangle}{\sqrt{2}}\left(\frac{|0\rangle-|1\rangle}{\sqrt{2}}\right)+(-1)^{f(1)}\,\frac{|1\rangle}{\sqrt{2}}\left(\frac{|0\rangle-|1\rangle}{\sqrt{2}}\right) \\ &= (-1)^{f(0)}\left(\frac{|0\rangle+(-1)^{f(0)\oplus f(1)}|1\rangle}{\sqrt{2}}\right)\left(\frac{|0\rangle-|1\rangle}{\sqrt{2}}\right). \end{aligned} \tag{8.20}$$

   Para escribir la diferencia de fase en el primer qubit utilizamos que $(-1)^{f(0)}(-1)^{f(1)} = 1$, si $f(0)\oplus f(1) = 0$ y (-1) si $f(0)\oplus f(1) = 1$. Notemos que a la salida del oráculo tenemos un único estado que almacena coherentemente información del valor de la función para todo punto. El algoritmo cuántico "evalúa" en un único paso la función en todos los puntos donde se encuentra definida. En nuestro caso, la información de interés se encuentra codificada en la diferencia de fase, que podemos extraer mediante la *interferencia* de estas dos alternativas en el próximo paso.

3. Esto se realiza, en forma similar a lo que vimos en el Capítulo 1, mediante la aplicación de una compuerta Hadamard sobre el primer qubit. Este qubit se encontrará en $(|0\rangle+|1\rangle)/\sqrt{2}$ para una función es constante, y $(|0\rangle-|1\rangle)/\sqrt{2}$ para una función balanceada. Por lo tanto el Hadamard sobre este qubit dará $|0\rangle$ como resultado en la primer situación y $|1\rangle$ en la segunda. Esta conclusión puede resumirse de la siguiente manera:

$$|\psi_3\rangle = |f(0)\oplus f(1)\rangle\left(\frac{|0\rangle-|1\rangle}{\sqrt{2}}\right).$$

4. Finalmente, una medición del primer registro en la base computacional revela si la función es constante (resultado 0) o balanceada (resultado 1).

Mostramos entonces cómo es posible averiguar si una función desconocida es constante o balanceada mediante una única llamada al oráculo, mientras que clásicamente puedo hacerlo dos veces. A pesar que el algoritmo no tiene utilidad y, la ventaja en este caso es marginal, en su momento sirvió para motivar un estudio más profundo de estos dispositivos. Esto fue posible ya que, como vimos, un dispositivo cuántico no se encuentra limitado a evaluar un único valor de la función en casa consulta al oráculo.

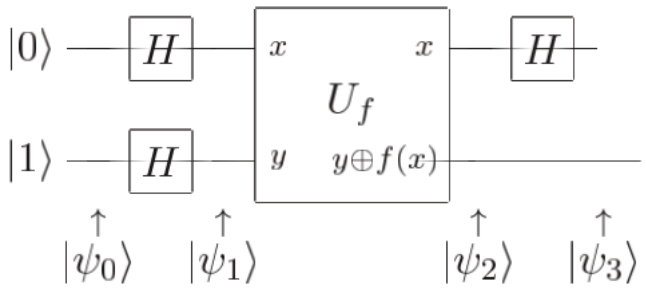

Figura 8.2: Circuito asociado al algoritmo de Deutsch. La caja asociada al oráculo implementa la transformación unitaria U_f y H es la compuerta de Hadamard que en el texto llamamos U_H.

8.2.3. Algoritmo de Deutsch-Jozsa

El algoritmo de Deusch-Jozsa resuelve la generalización del problema anterior. Consideremos ahora una función

$$f : \{0,1\}^n \to \{0,1\}, \tag{8.21}$$

que nos aseguran que es constante (vale lo mismo para toda cadena de n bits $\mathbf{x}$) o es balanceada (vale lo 1 para mitad de los posibles $\mathbf{x}$ y 0 para el resto). En este caso, existen 2^{2^n} funciones de este tipo, y el oráculo es tal que dado un input $\mathbf{x}$ y otro registro de 1 bit y: $(\mathbf{x}, y) \to (\mathbf{x}, y \oplus f(\mathbf{x}))$. De esta manera, podemos enunciar el problema de la siguiente manera:

- Dada una función desconocida $f : \{0,1\}^n \to \{0,1\}$ y un oráculo que nos permite evaluarla, determinar si esa función es constante o balanceada.

Es posible recurrir las veces que sean necesarias al oráculo y, nuevamente, la pregunta es cuál es la mínima cantidad de veces que será necesario hacerlo. Clásicamente, en el peor de los casos será necesario evaluar la mitad más uno de los inputs que, para cadenas de n bits, equivale a $2^{n-1}+1$ evaluaciones (ya que podría suceder que los inputs elegidos coincidan con la mitad que tiene el mismo valor). Es decir, clásicamente será necesario recurrir al oráculo una cantidad de veces exponencial en el número de bits. En lo que sigue mostraremos que cuánticamente es suficiente con hacerlo una *única* vez. El algoritmo de Deutsch-Jozsa se encuentra resumido en la Fig. 8.3. A continuación analizaremos paso a paso la evolución del algoritmo:

0. Comenzamos con dos registros en los cuales ingresan los estados $|\mathbf{0}\rangle|1\rangle \equiv |0\ldots0\rangle|1\rangle$ de n y 1 qubits respectivamente

$$|\psi_0\rangle = |\mathbf{0}\rangle|1\rangle.$$

1. La preparación del estado de input en el oráculo se realiza aplicando la compuerta Hadamard sobre todos los registros. Utilizando la expresión (8.16) podemos ver que:

$$|\psi_1\rangle \quad = \quad U_H^{\otimes n}|\mathbf{0}\rangle\ U_H|1\rangle = \frac{1}{\sqrt{2^n}} \sum_{\mathbf{x}\in\{0,1\}^n} |\mathbf{x}\rangle \left(\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right).$$

   Es decir, el oráculo recibe como entrada un estado superposición lineal de todos las cadenas para las cuales la función se encuentra definida.

2. En este punto actúa el oráculo unitario U_f y, en forma similar a lo que ocurría con el algoritmo de Deutsch, es fácil darse cuenta que:

$$|\psi_2\rangle = \frac{1}{\sqrt{2^n}} \sum_{\mathbf{x}\in\{0,1\}^n} (-1)^{f(\mathbf{x})}\ |\mathbf{x}\rangle \left(\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right).$$

3. Luego, se aplica la compuerta Hadamard sobre todos los qubits del registro superior:

$$\begin{aligned} |\psi_3\rangle &= \left(\frac{1}{\sqrt{2^n}} \sum_{\mathbf{x}\in\{0,1\}^n} (-1)^{f(\mathbf{x})} \frac{1}{\sqrt{2^n}} \sum_{\mathbf{z}\in\{0,1\}^n} (-1)^{\mathbf{x}\cdot\mathbf{z}}|z\rangle\right)\left(\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right) \\ &= \frac{1}{2^n} \sum_{\mathbf{z}\in\{0,1\}^n} \left(\sum_{\mathbf{x}\in\{0,1\}^n} (-1)^{f(\mathbf{x})+\mathbf{x}\cdot\mathbf{z}}\right)|\mathbf{z}\rangle \left(\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right) \end{aligned} \tag{8.22}$$

4. Finalmente se realiza una medición sobre los n qubits del registro superior en la base computacional. Evaluemos entonces sólo la probabilidad de medir $|\mathbf{0}\rangle$.

Dado que $|\psi_3\rangle$ es un estado producto entre los primeros n qubits y el último podemos escribirlo como $|\psi_3\rangle = |\psi_3^{(n)}\rangle\left(\frac{|0\rangle-|1\rangle}{\sqrt{2}}\right)$. Entonces la probabilidad de medir $\mathbf{0}$ se calcula como $\text{Prob}(\mathbf{0}) = |\langle\mathbf{0}|\psi_3^{(n)}\rangle|^2$, por lo tanto:

$$\text{Prob}(\mathbf{0}) = \frac{1}{2^{2n}}\left(\sum_{\mathbf{z}\in\{0,1\}^n}(-1)^{f(\mathbf{z})}\right)^2.$$

De manera que si la función es *constante* todos los 2^n términos de la suma tienen el mismo signo y por lo tanto $\text{Prob}(\mathbf{0}) = 1$. Mientras que si la función es *balanceada*, la mitad de los términos serán +1 y la otra mitad será -1, es decir, para estas funciones $\text{Prob}(\mathbf{0}) = 0$. En conclusión, si medimos $\mathbf{0}$ la función es constante, y en cualquier otro caso es balanceada.

Mostramos entonces que, a diferencia de lo que sucede con la estrategia clásica, con una única llamada al oráculo es posible resolver el problema con probabilidad 1. Sin embargo, vale la pena señalar que si bien clásicamente es necesario evaluar la función una cantidad exponencial en n para resolver el problema *determinísticamente*, cuando se permite una determinada probabilidad de error en la respuesta esta cantidad puede reducirse drásticamente.

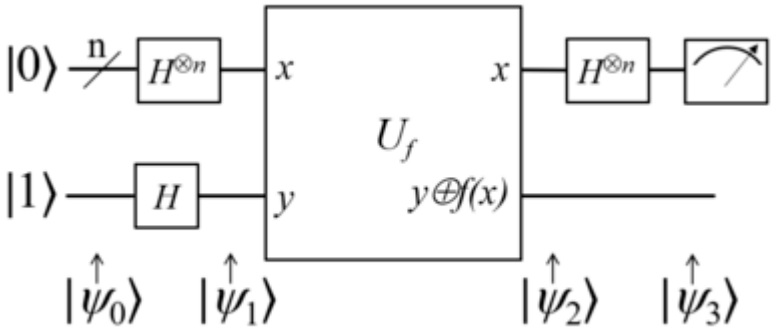

Figura 8.3: Algoritmo de Deutsch-Jozsa. El registro superior está formado por n qubits, al final del algoritmo se realiza una medición de ese registro en la base computacional. H es la compuerta de Hadamard que en el texto llamamos U_H.

8.2.4. Algoritmo de Simon

Concluiremos esta breve introducción a la computación cuántica con el algoritmo de Simon. Este fue el primero en mostrar una diferencia sustancial entre los cómputos clásicos y cuánticos, y además fue el que inspiró el célebre algoritmo de Shor. En efecto, Shor arribó a su resultado generalizando el algoritmo de Simon y aplicando la transformada de Fourier cuántica en lugar de las transformaciones Hadamard que veremos a continuación.

Consideremos entonces una función:

$$f : \{0,1\}^n \to \{0,1\}^n,$$

y la promesa de que existe una cadena $\mathbf{a} = (a_1, a_2, \ldots, a_n)$ tal que

$$\forall \mathbf{x}, \mathbf{y} \in \{0,1\}^n \quad f(\mathbf{x}) = f(\mathbf{y}) \iff \mathbf{y} = \mathbf{x} \oplus \mathbf{a}.$$

Es decir, f es una función periódica de período $\mathbf{a}$. Entonces el problema en cuestión se enuncia de la siguiente manera:

- Dada una función desconocida $f : \{0,1\}^n \to \{0,1\}^n$ y un oráculo que nos permite evaluarla, existe una cadena $\mathbf{a} = (a_1, a_2, \ldots, a_n)$ tal que $f(\mathbf{x}) = f(\mathbf{y})$ si y sólo si $\mathbf{y} = \mathbf{x} \oplus \mathbf{a}$. Determinar el período $\mathbf{a}$ de la función f.

Se puede demostrar que este problema no puede ser resuelto eficientemente clásicamente, ni si quiera en forma probabilística, es necesario evaluar función una cantidad exponencial de veces en n. Intuitivamente, podemos pensar que necesitamos conocer dos puntos donde la función valga lo mismo, pero no tenemos ninguna información acerca de la misma. O sea, que en general deberemos samplear la mitad de las entradas, una cantidad de cadenas que es exponencial en la cantidad de bits.

En contraposición, el algoritmo de Simon permite realizar la tarea utilizando el oráculo cuántico una cantidad de veces que es polinomial en n. El algoritmo se encuentra resumido en el circuito de la Fig. 8.4, donde la transformación unitaria asociada al oráculo U_f actúa de la siguiente forma en la base computacional: $U_f : |\mathbf{x}\rangle|\mathbf{y}\rangle \to |\mathbf{x}\rangle|\mathbf{y} \oplus f(\mathbf{x})\rangle$. A continuación explicaremos su funcionamiento:

1. Se inicializan todos los qubits en $|0\rangle$. En este caso, el registro superior estará compuesto por n qubits, y el inferior por otros n qubits. Se aplica la compuerta Hadamard sobre el primer registro para generar la entrada al oráculo, entonces:
$$|\psi_1\rangle = \frac{1}{\sqrt{2^n}} \sum_{\mathbf{x} \in \{0,1\}^n} |\mathbf{x}\rangle|\mathbf{0}\rangle.$$

2. Luego de la aplicar U_f, es fácil ver que el registro superior finaliza entrelazado con el registro inferior. El estado resultante es una superposición lineal de todas las entradas perfectamente correlacionados con el valor de la función en esos puntos:
$$|\psi_2\rangle \quad = \quad \frac{1}{\sqrt{2^n}} \sum_{\mathbf{x} \in \{0,1\}^n} |\mathbf{x}\rangle|f(\mathbf{x})\rangle.$$
Como la función f tiene período $\mathbf{a}$, podemos dividir a las n-uplas en dos conjuntos del mismo tamaño, X e Y, tal que para $\mathbf{x} \in X$ entonces $\mathbf{y} = \mathbf{x} \oplus \mathbf{a} \in Y$. De esta manera, la expresión anterior puede reescribirse convenientemente como:
$$|\psi_2\rangle = \frac{1}{\sqrt{2^n}} \sum_{\mathbf{x} \in X} (|\mathbf{x}\rangle + |\mathbf{x} \oplus \mathbf{a}\rangle)\,|f(\mathbf{x})\rangle.$$

3. Luego se mide el registro inferior. En realidad este paso no es necesario, pero simplifica bastante el análisis del algoritmo. Aquí supongamos que el resultado fue $f(\mathbf{x}_0)$, entonces el estado posterior a la medición es:

$$|\psi_3\rangle = \frac{1}{\sqrt{2}}\left(|\mathbf{x_0}\rangle + |\mathbf{x_0} \oplus \mathbf{a}\rangle\right)|f(\mathbf{x_0})\rangle.$$

4. A continuación se aplica Hadamard sobre los qubits del registro superior, de forma tal que:

$$|\psi_4\rangle = \frac{1}{\sqrt{2^{n+1}}} \sum_{\mathbf{z}\in\{0,1\}^n} \left((-1)^{\mathbf{x_0}\cdot\mathbf{z}} + (-1)^{(\mathbf{x_0}\oplus\mathbf{a})\cdot\mathbf{z}}\right)|\mathbf{z}\rangle|f(\mathbf{x_0})\rangle$$

5. Finalmente, se realiza una medición del registro superior. El cálculo de la probabilidad se realiza en forma similar al que hicimos para el algoritmo anterior, ya que aquí también el estado es producto. Es decir si $|\psi_4\rangle = |\psi_4^{(n)}\rangle|f(\mathbf{x_0})\rangle$, entonces $\text{Prob}(\mathbf{z}) = |\langle\mathbf{z}|\psi_4^{(n)}\rangle|^2$, por lo tanto:

$$\begin{aligned}
\text{Prob}(\mathbf{z}) &= \frac{1}{2^{n+1}}\left|(-1)^{\mathbf{x_0}\cdot\mathbf{z}} + (-1)^{(\mathbf{x_0}\oplus\mathbf{a})\cdot\mathbf{z}}\right|^2 \\
&= \frac{1}{2^{n+1}}|1 + (-1)^{\mathbf{a}\cdot\mathbf{z}}|^2 \\
&= \begin{cases} \frac{1}{2^{n-1}} & \text{si} \quad \mathbf{a}\cdot\mathbf{z} = 0 \\ 0 & \text{si} \quad \mathbf{a}\cdot\mathbf{z} = 1 \end{cases}
\end{aligned}$$

Sólo será posible obtener como resultado de la medición los $\mathbf{z}$ que satisfagan $\mathbf{a}\cdot\mathbf{z} = 0$, y esos resultados serán sampleados con una probabilidad uniforme sobre ese conjunto. En cada llamada al oráculo obtendremos los $\mathbf{z}$ para construir un sistema de ecuaciones lineales que permitirá determinar $\mathbf{a}$. Si se logra obtener las $n-1$ ecuaciones linealmente independientes el problema se resuelve en forma determinista.

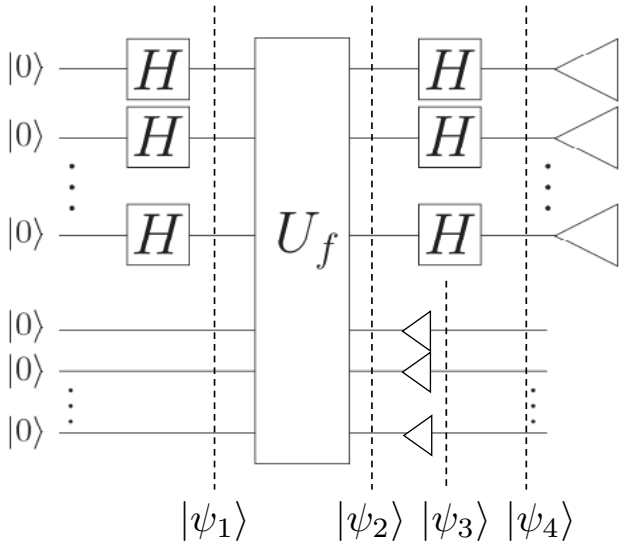

Figura 8.4: Circuito asociado al algoritmo de Simon. Los triángulos representan mediciones en la base computacional. H es la compuerta de Hadamard que en el texto llamamos U_H.

Como vimos es necesario realizar más de una llamada al oráculo para encontrar $\mathbf{a}$. Veamos esto entonces. Supongamos que luego de $n-1$ repeticiones obtenemos $\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_{n-1}$ cadenas, entonces tendremos $n-1$ ecuaciones $\mathbf{z}_1 \cdot a = 0, \mathbf{z}_2 \cdot a = 0 \ldots \mathbf{z}_{n-1} \cdot a = 0$. Si estas son linealmente independientes, es posible resolver el problema eficientemente, por ejemplo por eliminación Gaussiana. Si las ecuaciones no son linealmente independientes podemos resolver el sistema, pero no tendrá solución única. Sin embargo, podremos verificar si los candidatos son soluciones simplemente evaluando $f(\mathbf{0})$ y $f(\mathbf{a})$ y viendo si son iguales o no. Es posible mostrar que con una cantidad $O(n)$ de repeticiones es posible resolver el problema con una probabilidad exponencialmente cercana a uno.

Como dijimos al principio, este algoritmo fue el que inspiró el resultado de Shor. En efecto, la factorización de números enteros en una computadora cuántica a grandes rasgos se divide en dos partes. En la primera parte del algoritmo convierte dicho problema en el problema de encontrar el período de una función, y esa parte es puramente clásica. En la segunda parte encuentra el período usando una variación del algoritmo Simon utilizando la transformada de Fourier cuántica, y esta parte es la responsable de la aceleración cuántica donde el paralelismo cuántico y el entrelazamiento resultan ser escenciales. Sin embargo, tal como vimos con el algoritmo de Simon, la información acerca de la solución se encuentra en la distribución de probabilidad de los resultados de la medición y no ofrece la solución luego de una única iteración. Esto sucede con la gran mayoría de los algoritmos cuánticos: son *probabilísticos*, entregan la respuesta correcta con alta probabilidad, y dicha probabilidad de error puede disminuirse repitiendo el algoritmo. Es decir, podemos pensar a las computadoras cuánticas simplemente como dispositivos que generan salidas con una determinada distribución de probabi-

lidad determinada por el circuito. Entonces, si fuera cierto que las computadoras cuánticas permiten resolver una clase de problemas que no es posible hacerlo con una computadora clásica, indicaría que no es posible simular distribuciones de probabilidad cuánticas con computadoras clásicas. Es sabido que la Mecánica Cuántica efectivamente produce distribuciones de probabilidad que no pueden ser reproducidas por sistemas clásicos pero cuando donde existen otras restricciones físicas, como lo es la localidad en escenarios de Bell. La pregunta entonces en este contexto es si la estadística generada por las computadoras cuánticas, cuando se imponen restricciones en la complejidad por ejemplo, difiere de la que pueden generar computadoras clásicas. La respuesta a esta pregunta no se conoce, pero la evidencia indica que esto es así.