UNIVERSIDAD COMPLUTENSE DE MADRID

FACULTAD DE CIENCIAS FÍSICAS

DEPARTAMENTO DE FÍSICA TEÓRICA

TRABAJO DE FIN DE GRADO

Código de TFG: FT12

Introducción a la Información Cuántica y Computación Cuántica

Introduction to Quantum Information and Quantum Computation

Supervisor/es: Miguel A. Martín-Delgado

Diego Ximénez de Embún Lucía

Grado en Física

Curso académico 2019-20

Convocatoria PRIMERA

Resumen:

El presente texto es una introducción a la información cuántica y la computación cuántica. Se hace un estudio bibliográfico de la computación clásica, los qubits, las puertas lógicas cuánticas y los resultados de universalidad. Se expone la simulación de sistemas físico-cuánticos para ilustrar las ventajas de la computación cuántica, y se explica brevemente el software cuántico como herramienta para llevar a cabo los circuitos estudiados. Por último, se pone en contexto el resultado de supremacía cuántica de Google, explicando los circuitos cuánticos aleatorios, la *cross-entropy* y su realización experimental. Se realiza una simulación de circuitos cuánticos aleatorios en redes 4x4, y los resultados obtenidos se comparan con la teoría y con resultados anteriores de la literatura.

Abstract:

The present text is an introduction to quantum information and quantum computation. Classical computation, qubits, quantum logic gates and universality results are studied from literature. Simulation of quantum physical systems is discussed to ilustrate the advantages of quantum computation, and quantum software is briefly explained as a tool to perform the studied circuits. Finally, Google's quantum supremacy result is put into context, by explaining random quantum circuits, cross-entropy and its experimental realization. Random quantum circuit simulations in a 4x4 lattice are performed, and the obtained results are compared with theory and previous results found in literature.

Índice

1. Introducción

La computación cuántica es la disciplina que estudia el procesamiento y transmisión de información mediante el uso de sistemas fisico-cuánticos [1], es decir, aquellos sistemas que se describen con las leyes de la física cuántica. La física cuántica es por tanto la base fundamental de la computación cuántica.

Como teoría científica, la fisica cuántica surgió a principios del siglo XX en un intento por explicar algunas de las cuestiones que parecían no poder ser resueltas con la física clásica. Desde entonces se ha utilizado para describir un gran abanico de sistemas físicos y es una de las teorías científicas más exitosas y verificadas.

Medio siglo después de su concepción, en la década de los setenta, algunos físicos se preguntaron si podía ser utilizada no solo para describir sistemas físico-cuánticos, sino para obtener control absoluto sobre ellos. Hasta el momento todos los experimentos de la disciplina trabajaban con muestras que contenían una enorme cantidad de sistemas cuánticos, ninguno de los cuales era accesible de forma directa [1]. Surgió por tanto la idea de que los sistemas cuánticos podían ser accesibles directamente, y por lo tanto, podían utilizarse para procesar información.

El campo de la computación cuántica ha evolucionado mucho desde la década de los setenta, tanto teórica como experimentalmente. Desde que se vio que los ordenadores cuánticos podían realizar operaciones exponencialmente más deprisa que los clásicos ha habido un creciente interés en conseguir realizarlos de forma experimental. Esta realización experimental se ha conseguido de muchas formas, sin embargo nunca con la escala requerida para superar a ordenadores clásicos. La superación de la barrera clásica se denomina comunmente *supremacía cuántica*.

La carrera por la supremacía cuántica se ha avivado en la década de los 2010, a medida que las realizaciones experimentales se han hecho cada vez más potentes. El creciente interés no ha aparecido solamente en la comunidad científica, sino también en agencias estatales y compañías privadas. A finales de 2019, Google [2] afirmó que había logrado obtener la supremacía cuántica utilizando un ordenador superconductor. Aunque no hay un consenso sobre si este experimento constituye realmente la supremacía cuántica, sin duda la computación cuántica real esta cada vez más cerca.

2. Teoría clásica de la computación

Los algoritmos son los componentes fundamentales de la computación clásica, y por tanto juegan también un papel fundamental en la computación cuántica. El término algoritmo se puede entender como una serie de instrucciones que se aplican sobre un conjunto de valores y los transforman en otro conjunto de valores. En otras palabras, un algoritmo es una herramienta que permite resolver un problema computacional particular [1].

Históricamente, el concepto de algoritmo ha existido desde hace siglos. Sin embargo su definición moderna fue dada por separado en el año 1936 por Alan Turing [3] y Alonzo Church. Church y Turing intentaban resolver un problema que había sido planteado años atrás por el matemático David Hilbert: el *Entscheidungsproblem* (problema de decisión) [1].

Hilbert se había preguntado si existía o no un algoritmo que podría ser usado, en principio, para resolver todos los problemas de las matemáticas. Él esperaba que la respuesta fuese positiva. Sin embargo, tanto Church como Turing llegaron a la misma conclusión: la respuesta a la pregunta de Hilbert era un *no* rotundo [1]. Para resolver la cuestión, Church y Turing tuvieron que definir de forma rigurosa el concepto de algoritmo y al hacerlo sentaron los cimientos de la computación moderna.

2.1. Máquinas de Turing

El modelo que creó Turing para poder definir un algoritmo de forma matemática se denomina *Máquina de Turing* y se trata de la forma más fundamental en la que se puede describir una computación. Una máquina de Turing tiene los siguientes componentes [3, 4]

- Un conjunto finito de m estados $q_1, ..., q_m$
- Un conjunto finito Γ de símbolos
- Una serie infinita ordenada de cajas, denominada cinta, tal que cada una contiene un símbolo $x \in \Gamma$

Como puede verse en la Fig. 1a, la máquina además tiene dos variables: el estado actual y el lugar de la cinta que se esta leyendo en el momento. Estas variables cambian mientras se ejecuta el algoritmo correspondiente. Además hay dos estados especiales, q_s y q_h, que son los estados que tiene la máquina al comenzar y al acabar la ejecución respectivamente.

El algoritmo o programa es por tanto un conjunto de instrucciones con la siguiente forma

$$\langle q, x, q', x', s \rangle$$

donde q, q' son dos estados cualesquiera, $x, x' \in \Gamma$ y $s \in \{+1, -1, 0\}$.

En cada paso de la computación la máquina recorre todas las instrucciones hasta que encuentra una con el mismo q que el estado interno y con el mismo x que hay en el lugar de la cinta que está leyendo en ese momento. Cuando encuentra tal instrucción entonces se cambia el estado interno a q' y el símbolo de la caja a x'. Por último la se cinta se mueve a la derecha, a la izquierda o se queda estacionaria, según el valor de s.

Escribir programas para máquinas de Turing es complicado incluso para los algoritmos más sencillos. Sin embargo, cualquier algoritmo (sin importar su complejidad) puede ser computado por una. Es más, la clase de funciones computables por una máquina de Turing corresponde exactamente con la clase de funciones computables por un algoritmo cualquiera.

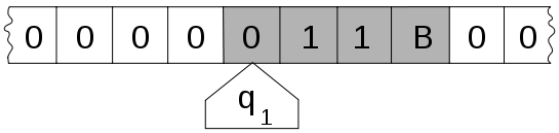
Por tanto el concepto de máquina de Turing encapsula completamente la noción de computar una función por un algoritmo [1]. Esto se conoce como la *tesis de Church-Turing*. La tesis de Church-Turing establece un marco con el que estudiar los algoritmos de una forma más rigurosa.

Los ordenadores cuánticos también obedecen la tesis de Church-Turing, porque computan la misma clase de funciones que son computables por una máquina de Turing [1]. La ventaja de los ordenadores cuánticos es entonces la *eficiencia* con la que pueden realizar ciertas operaciones en comparación con sus equivalentes clásicos. Para cuantificiar esta eficiencia se utilizan las teorías de complejidad computacional.

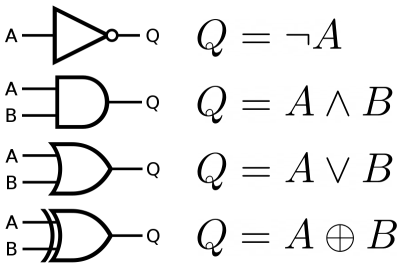
2.2. Complejidad computacional

En la vida real, los algoritmos no se ejecutan en máquinas de Turing idealizadas sino en sistemas físicos reales, con memoria y tiempo límitados. Por ello, cuando se diseñan algoritmos, es importante asegurarse de que los recursos que utilizan se mantienen dentro de los límities físicos que esten disponibles en el momento.

Para poder cuantificar estos recursos se utiliza principalmente la notación asintótica. Esta notación se define de la siguiente forma. Sean $f(n)$ y $g(n)$ dos funciones cualesquiera, entonces $f(n)$ es $\mathcal{O}(g(n))$ si existen enteros positivos c y n_0 tales que para todo $n \geq n_0$ se tiene $f(n) \leq cg(n)$. En otras palabras $f(n)$ es $\mathcal{O}(g(n))$ si $g(n)$ es una



(a) Ejemplo de una máquina de Turing. En este caso el alfabeto es $\Gamma = \{0, 1, B\}$, la máquina se encuentra en el estado q_1 y además está leyendo el lugar de la cinta que indica el marcador.



(b) Algunas puertas lógicas fundamentales junto a su operación equivalente en el álgebra de Boole (*NOT*, *AND*, *OR*, *XOR*, en orden).

Figura 1

cota superior de $f(n)$ [5].

En general $f(n)$ se refiere a un algoritmo que utiliza un número de valores de entrada proporcional a n y $g(n)$ suele ser una función simple polinómica o exponencial. De esta forma la notación asintótica es útil para estudiar el comportamiento de algoritmos en el peor caso posible. Un problema se dice que es *sencillo* si existe un algoritmo para resolverlo que sea $\mathcal{O}(n^k)$. Por otra parte, un problema es *complicado* si el mejor algoritmo posible para resolverlo es $\mathcal{O}(k^n)$.

Además, en máquinas de Turing, la notación asintótica puede referirse tanto a número de operaciones requeridas (**TIME**) como a la cantidad de memoria utilizada (**SPACE**) [5]. Por ejemplo, si se tiene un algoritmo que actúa sobre n elementos y este algoritmo es **TIME**(n^3), entonces en el peor de los casos el algoritmo necesitara aproximadamente n^3 operaciones.

Una de las dificultades del estudio de la complejidad computacional es que es posible que diferentes modelos de computación utilicen diferentes recursos para resolver el mismo problema [1]. Sin embargo se sabe que si una función puede computarse utilizando k pasos en un modelo cualquiera, entonces esta misma función puede computarse en una máquina de Turing utilizando como mucho $p(k)$ pasos, donde p es una función polinómica. Esta proposición se conoce como la *tesis de Church-Turing fuerte* [1].

La tesis de Church-Turing fuerte garantiza que se puede reducir todo el estudio de algoritmos al estudio de máquinas de Turing, ya que (salvo un incremento polinómico) el estudio no depende del modelo utilizado. Es decir, un problema *sencillo* será sencillo independientemente del modelo utilizado. Y lo mismo ocurre si el problema es *complicado*.

Dependiendo de su complejidad, los problemas pueden clasificarse en *clases de complejidad*, que contienen todos los problemas que pueden ser resueltos por algoritmos con la misma cota superior. Se sabe que la relación entre las clases más importantes es

$$\mathbf{L} \subseteq \mathbf{P} \subseteq \mathbf{NP} \subseteq \mathbf{EXP}$$

donde **L** son problemas que pueden ser resueltos en **SPACE**$(\log n)$ y los problemas de **EXP** pueden resolverse en **TIME**(k^n). Por otra parte, la clase **P** contiene los problemas que pueden ser resueltos en tiempo polinómico, mientras que **NP** contiene los problemas cuyas soluciones pueden ser *verificadas* en tiempo polinómico [4, 5].

En general, aunque se sabe que estas relaciones se cumplen, no se sabe cuales de ellas son estrictas, es decir, no se sabe cuales de los conjuntos anteriores son iguales [5]. El caso de **P** y **NP** es particularmente importante, porque ambas clases contienen problemas muy comunes en criptografía y computación. De hecho, la cuestión de si ambos son iguales o no está considerada el problema más importante de la computación en la actualidad [1].

El ejemplo más relevante para ilustrar esta cuestión es el problema de, dado un número p cualquiera, determinar sus factores no triviales. Este problema está en **EXP**, ya que obtener los factores de un número está considerado en gene-

ral un problema complicado. Sin embargo también pertenece a **NP**, ya que verificar si un número dado es factor de p es tan sencillo como dividir ambos números.

Podría parecer entonces que **P** y **NP** son distintos, ya que el problema de factorización mencionado pertenece a **NP** pero no a **P**. Sin embargo podría existir un algoritmo (aún no descubierto) que permita factorizar un número en tiempo polinómico. Para saber con certeza si ambos conjuntos son distintos o no habría que demostrar estrictamente que tal algoritmo no puede existir, y habría que hacer lo mismo para todos los problemas de **NP**. La cuestión, por tanto, continua abierta.

2.3. Modelo de circuitos

Las máquinas de Turing son un modelo idealizado de computación, en el sentido de que utilizan un espacio infinito que no se puede realizar de forma física [3]. Existe otro modelo, que es equivalente a una máquina de Turing en términos de poder de computación, y que sin embargo es posible de realizar físicamente. Este modelo es el modelo de circuitos.

En el modelo de circuitos la computadora está compuesta de *cables* y *puertas lógicas*. Los cables representan la unidad básica de computación, el *bit*, que puede ser 0 (no corriente) ó 1 (corriente). Por otra parte una puerta lógica es una función $f : \{0,1\}^k \to \{0,1\}^l$ que va desde k bits de entrada hasta l bits de salida [1, 4].

Una computación en este modelo comienza con unos bits de entrada en un estado inicial particular y sobre ellos se aplican las puertas lógicas correspondientes de izquierda a derecha hasta llegar a los bits de salida, que contienen el resultado de la computación. Esto no es más que la composición de las funciones que corresponden a las puertas que hay en el circuito.

Algunas puertas lógicas básicas se muestran en la Fig. 1b, junto a las operaciones lógicas correspondientes. Estas operaciones lógicas corresponden con operaciones en álgebras de Boole, que son álgebras que tratan con tan solo dos variables: verdadero y falso.

Cualquier función puede ser computada por un conjunto discreto de puertas lógicas, que se denomina *conjunto de puertas universales* [1]. Existen múltiples conjuntos universales (como por ejemplo $\{AND, NOT\}$ [4]), pero todos ellos pueden evaluar cualquier función. La realización física es entonces muy sencilla, porque solo hace falta crear un número finito de puertas que actúan en un número finito de cables y con ello se puede computar cualquier algoritmo.

3. Principios de la computación cuántica

El modelo comunmente utilizado en la computación cuántica es similar al modelo de circuitos clásico. En este sentido, el bit clásico se sustituye por un equivalente cuántico, el *qubit*, mientras que las puertas lógicas se sustituyen por *puertas lógicas cuánticas*.

3.1. El qubit

El estado de una computadora de m qubits viene descrito por un vector Ψ normalizado perteneciente al espacio vectorial $\mathbb{C}^N$, donde $N = 2^m$. El vector tiene la siguiente forma [6]

$$\Psi = \sum_{x \in \{0,1\}^m} \alpha_x \left| x_1, ..., x_m \right\rangle \tag{1}$$

Nótese que si $x_i \in \{0,1\}$ es un bit, se define un bitstring $x \in \{0,1\}^m$ como la serie de bits $x = x_1...x_m$. Por lo tanto el vector que describe m qubits es una combinación lineal de todas las posibles 2^m permutaciones de m bits clásicos.

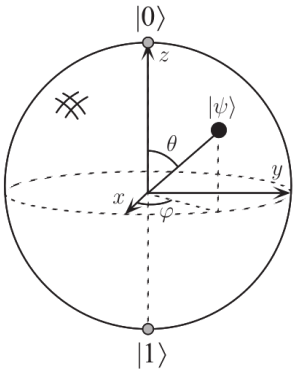
El sistema más fundamental es entonces el sistema de un qubit, y se representa como

$$\left|\psi\right\rangle = \alpha \left|0\right\rangle + \beta \left|1\right\rangle \tag{2}$$

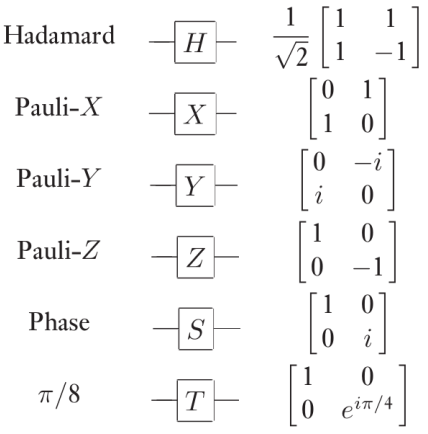
donde como se ha mencionado se cumple $|\alpha|^2 + |\beta|^2 = 1$, de tal forma que $|\alpha|^2$ y $|\beta|^2$ son las probabilidades de que el qubit se encuentre en los estados $|0\rangle$ y $|1\rangle$ respectivamente.

Otra forma de representar un solo qubit que resulta muy útil para visualizarlo es la esfera de Bloch, que puede verse en la Fig. 2a. La ecuación de normalización de α y β es la ecuación de una esfera, ya que α y β son números complejos. Entonces se puede escribir la Ec. (2) como [1]

$$\left|\psi\right\rangle = \cos\frac{\theta}{2}\left|0\right\rangle + e^{i\varphi}\sin\frac{\theta}{2}\left|1\right\rangle$$



(a) La esfera de Bloch, una herramienta útil para visualizar sistemas cuánticos de dos niveles (figura 1.3 de [1]).



(b) Las puertas lógicas cuánticas de un qubit más importantes, junto a su símbolo y representación matricial (figura 4.2 de [1]).

Figura 2

donde θ y φ son los ángulos que se indican en la Fig. 2a.

A primera instancia, uno podría pensar que, como un qubit pertenece a $\mathbb{C}^2$, entonces puede almacenar una cantidad infinita de información en las amplitudes α y β. Sin embargo aunque técnicamente el valor de las amplitudes sí puede contener infinita información, esta información no es accesible. Esto se debe a que, para poder obtener información de un sistema cuántico, hay que realizar una medición y al hacerlo el sistema colapsa a 0 ó 1 y la información de las amplitudes se pierde por completo [1]. Por ello aunque en principio un qubit almacene infinitamente más información que un bit clásico, el acceso a esta información es muy limitado.

Sin embargo, aunque todos los qubits de un circuito acaban siempre siendo colapsados en bits clásicos, los pasos intermedios pueden encerrar un gran poder computacional. Los primeros en demostrar que tal poder podía sobrepasar en una tarea concreta a un ordenador clásico fueron Shor, en 1994, y Grover, en 1995 [1].

Hay además otra diferencia fundamental entre un bit clásico y un qubit. Se trata de la puerta lógica *FANOUT*, una de las puertas lógicas más importantes en computación clásica. Esta puerta copia la información contenida en un bit de entrada a dos bits de salida. La habilidad de poder copiar información de forma indefinida y fiable simplifica mucho la creación de algoritmos y es uno de los pilares fundamentales del diseño de circuitos clásicos.

En computación cuántica, sin embargo, tal puerta no puede existir. El primero en demostrar este hecho fue Dieks en 1982, y desde entonces se ha demostrado en múltiples contextos y de forma más general [1]. El teorema se conoce como *teorema de la no clonación* y se puede resumir de la siguiente forma. Se define una clonación como la transformación

$$|\psi\rangle \otimes |s\rangle \rightarrow U(|\psi\rangle \otimes |s\rangle) = |\psi\rangle \otimes |\psi\rangle$$

donde $|\psi\rangle$ es un estado puro cualquiera y $|s\rangle$ es el estado en el que se va a realizar la copia [1]. Tal transformación U solo existe para estados ortogonales entre sí, es decir, es imposible diseñar una transformación que pueda copiar un estado desconocido cualquiera.

El hecho de que no se puedan copiar qubits de forma arbitraria, a diferencia de como se hacía de forma clásica, es también uno de los desafíos añadidos a la hora de diseñar circuitos cuánticos.

3.2. Puertas cuánticas de un qubit

En el caso de las puertas lógicas cuánticas, no es suficiente con que actuen de forma individual sobre $|0\rangle$ y $|1\rangle$, sino que además deben actuar sobre una superposición $|\psi\rangle$ de la forma definida

por la Ec. (2). Una puerta lógica cuántica de un qubit es por tanto una transformación que actúa sobre $|\psi\rangle$ de la siguiente forma

$$|\psi\rangle \to U|\psi\rangle = \alpha'|0\rangle + \beta'|1\rangle \tag{3}$$

Como $|\psi\rangle$ está normalizado también debe estarlo el estado final, es decir, $|\alpha'|^2 + |\beta'|^2 = 1$. Esto implica que U debe conservar la norma y por tanto U es unitaria ($U^\dagger U = UU^\dagger = I$). Esta condición es la *única* condición que deben cumplir las puertas lógicas cuánticas [1]. Como ya se ha visto, en computación clásica solo existe una puerta lógica de un bit no trivial, la puerta *NOT* (el resto de puertas de la Fig. 1b actúan sobre dos bits). En computación cuántica, sin embargo, hay infinitas puertas diferentes que actúan sobre un solo qubit, puesto que hay infinitas matrices unitarias de dimensión 2. El grupo que forman estás matrices se denomina $U(2)$.

Las puertas más importantes pueden verse en la Fig. 2b. La puerta X es el equivalente cuántico de *NOT*, porque su operación sobre el estado de la Ec. (2) es la siguiente [1]

$$|\psi\rangle \to X|\psi\rangle = \beta|0\rangle + \alpha|1\rangle$$

Es decir, al igual que hacía la puerta *NOT*, esta operación cambia los 0 por 1 y viceversa. La diferencia con la operación clásica es que, como ya se ha mencionado, la transformación cuántica actúa sobre una superposición y cambia ambos valores a la vez.

El resto de las puertas que aparecen en la Fig. 2b no tienen equivalente clásico. Entre estas las más importantes son la puerta Z y la puerta Hadamard, H, que actúan de la siguiente forma [1] sobre un qubit

$$|\psi\rangle \to Z|\psi\rangle = \alpha|0\rangle - \beta|1\rangle$$
$$|\psi\rangle \to H|\psi\rangle = \alpha\frac{|0\rangle + |1\rangle}{\sqrt{2}} + \beta\frac{|0\rangle - |1\rangle}{\sqrt{2}}$$

Cabe mencionar que las matrices que aparecen en la Fig. 2b están expresadas en la base

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

Volvamos ahora a la representación de la esfera de Bloch que se muestra en la Fig. 2a. Una rotación genérica en esta esfera se puede definir como [1]

$$\begin{aligned} R_{\hat{n}}(\theta) &= \exp(-i\theta\hat{n}\cdot\vec{\sigma}/2) \\ &= \cos\left(\frac{\theta}{2}\right)I - i\sin\left(\frac{\theta}{2}\right)\hat{n}\cdot\vec{\sigma} \end{aligned} \tag{4}$$

donde θ es el ángulo de la rotación alrededor del vector $\hat{n}$ y $\vec{\sigma}$ es el vector de tres componentes de las matrices de Pauli X, Y y Z (Fig. 2b). Este operador de rotación actúa entonces sobre el estado $|\psi\rangle$ de la Ec. (2) rotándolo alrededor de $\vec{n}$ en la representación de la esfera de Bloch.

Cualquier rotación genérica $R_{\hat{n}}(\theta)$ conserva la norma y por tanto se trata de un operador unitario. De hecho, se puede ver [1] que un operador unitario cualquiera U se puede escribir en función de una rotación de la siguiente forma

$$U = \exp(i\alpha)R_{\hat{n}}(\theta)$$

Por tanto la única diferencia entre las matrices de $U(2)$ y las rotaciones en la esfera de Bloch es una fase arbitraria α.

De la Ec. (4) se puede obtener la forma matricial para rotaciones alrededor de los ejes x, y y z, que es

$$R_x(\theta) = \begin{bmatrix}\cos\frac{\theta}{2} & -i\sin\frac{\theta}{2}\\ -i\sin\frac{\theta}{2} & \cos\frac{\theta}{2}\end{bmatrix}$$
$$R_y(\theta) = \begin{bmatrix}\cos\frac{\theta}{2} & -\sin\frac{\theta}{2}\\ \sin\frac{\theta}{2} & \cos\frac{\theta}{2}\end{bmatrix}$$
$$R_z(\theta) = \begin{bmatrix}e^{-i\theta/2} & 0\\ 0 & e^{i\theta/2}\end{bmatrix}$$

Además, de Barenco *et al.* [6], se sabe que cualquier matriz unitaria puede ser expresada en una descomposición de rotaciones alrededor de los ejes Z-Y. Es decir, para cualquier matriz U unitaria existen números reales α, β, γ y δ tal que

$$U = e^{i\alpha}R_z(\beta)R_y(\gamma)R_z(\delta) \tag{5}$$

Este resultado puede extenderse para dos direcciones $\hat{n}$ y $\hat{m}$ arbitrarias no paralelas de la siguiente forma

$$U = e^{i\alpha}R_{\hat{n}}(\beta)R_{\hat{m}}(\gamma)R_{\hat{n}}(\delta) \tag{6}$$

En el mismo artículo [6] se expone el siguiente lema, que resulta ser muy útil en la construcción de circuitos cúanticos más complejos.

Lema 1: Para cualquier matriz unitaria de determinante unidad W ($W \in SU(2)$), existen matrices A, B y $C \in SU(2)$ tal que $ABC = I$ y $AXBXC = W$, donde X es la matriz de Pauli (Fig. 2b). Este lema se puede extender para matrices de $U(2)$ si se añade una fase arbitraria $\exp(i\alpha)$ a la descomposición de W.

3.3. Puertas cuánticas de múltiples qubits

Las puertas lógicas cuánticas se pueden generalizar a estados como el de la Ec. (1), es decir, es posible definir puertas cuánticas que actuen sobre múltiples qubits a la vez. Una puerta cuántica W que actúa sobre m qubits es por tanto una matriz cuadrada de 2^m filas y columnas, que además debe ser unitaria $W \in U(2^m)$. Las operaciones de múltiples qubits más importantes son las denominadas *operaciones controladas*.

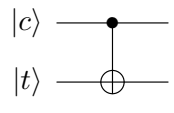
Las operaciones controladas son también muy importantes en computación clásica, y son aquellas que actúan de una forma u otra dependiendo del valor de uno o varios bits de control. En los lenguajes de programación modernos se utilizan cláusulas como *if*, *else*, etc., aunque también se pueden expresar como puertas lógicas. Un ejemplo de una operación controlada en un circuito clásico es la puerta *XOR*, que puede verse en la Fig. 1b. Esta puerta cambia el segundo bit si el primero es 1, y si el primero es 0 deja ambos como estaban.

El equivalente cuántico de la puerta *XOR* es la operación que se denomina *controlled-NOT*. Al igual que *XOR*, esta operación cambia el segundo qubit (aplica la puerta X) si el primero es $|1\rangle$. Por lo tanto se puede escribir como [1]

$$|c\rangle\,|t\rangle \rightarrow |c\rangle\,|t \oplus c\rangle$$

donde $\oplus$ es la suma módulo 2. La diferencia fundamental con la puerta clásica es que la puerta cuántica es reversible y por tanto el número de qubits inicial y final debe ser siempre el mismo. La forma matricial de esta transformación y el esquema de circuito son los siguientes

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

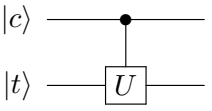


donde el primer qubit es el qubit de control y el segundo el qubit objetivo.

De forma más general, supongase una operación arbitaria sobre un qubit U. Una operación *controlled-U* es una operación de dos qubits que aplica U sobre el segundo qubit siempre que el primero sea $|1\rangle$. Es decir [1]

$$|c\rangle\,|t\rangle \rightarrow |c\rangle\,U^c\,|t\rangle$$

El esquema de circuito es el siguiente

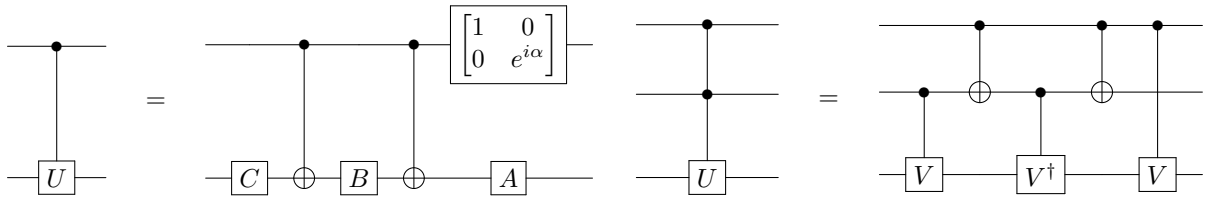


Como se ha mencionado, las operaciones controladas genéricas como esta son de extrema importancia para el diseño de algoritmos cuánticos. Sin embargo, su realización física es mucho más compleja que la de las puertas de un qubit o la puerta *CNOT*, por lo que es importante encontrar una forma de poder implementar una *controlled-U* genérica en función de estas puertas más sencillas.

Para llevar a cabo esta construcción se utiliza el Lema 1 como puede verse en la Fig. 3a. Esta construcción fue propuesta por Barenco *et al.* [6] De esta forma por tanto se puede descomponer una operación controlada arbitraria en operaciones de un solo qubit y puertas *CNOT*. La elección de las puertas A, B y C se hace de tal forma que se cumplan las propiedades que se indican en el lema.

En el mismo artículo [6] se propone también la siguiente notación para caracterizar operaciones controladas. Se utiliza $\wedge_k(U)$ para denotar una operación U controlada por k qubits. Por ejemplo la puerta *CNOT* se denota $\wedge_1(X)$, mientras que las que aparecen en las Figs. 3a y 3b serían $\wedge_1(U)$ y $\wedge_2(U)$ respectivamente. La construcción de la Fig. 3b fue propuesta también en este artículo.

La puerta lógica $\wedge_2(X)$, es decir, la puerta *CNOT* con dos qubits de control, es un caso especial de $\wedge_2(U)$ y se denomina la puerta Toffoli. Toffoli y Fredkin propusieron la puerta Toffoli clásica en 1982 y demostraron que es suficiente para realizar cualquier operación reversible, esto es, es una puerta universal clásica [1]. Más



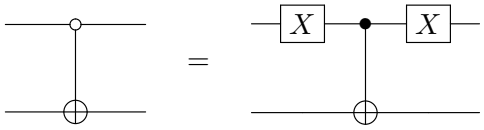
(a) Circuito que implementa la operación controlada arbitraria $\wedge_1(U)$ utilizando el Lema 1.

(b) Circuito que implementa la operación controlada arbitraria $\wedge_2(U)$, donde se cumple $V^2 = U$.

Figura 3

adelante, Barenco *et al.* [6] mostraron que se podía realizar cualquier operación $\wedge_k(X)$ utilizando $4(k-2)$ puertas Toffoli, y además propusieron un método para realizar operaciones controladas generales $\wedge_k(U)$.

Por último cabe mencionar que en las operaciones controladas hay ocasiones en las que la operación debe realizarse cuando el qubit es 0 y no 1. En este caso simplemente se puede aplicar la puerta X al qubit de control antes y después de la operación. Es decir



3.4. Puertas cuánticas universales

Como se mencionó en la Sec. 2.3, existen conjuntos discretos de pocas puertas clásicas que pueden ser usados para computar cualquier algoritmo. Un conjunto universal es por ejemplo $\{AND, NOT\}$. Otro conjunto es la puerta Toffoli, que es universal por sí misma. Como la puerta Toffoli es también una puerta cuántica, esto implica que un circuito cuántico puede realizar cualquier operación posible para uno clásico [7], utilizando puertas Toffoli. Por supuesto lo mismo no sucede al contrario, y la puerta Toffoli no es suficiente para realizar cualquier circuito cuántico.

Un resultado similar de universalidad existe también para circuitos cuánticos. El proceso es sin embargo un poco más complicado y contiene tres construcciones distintas que culminan en la prueba de que cualquier operación unitaria puede ser aproximada con precisón arbitraria por un conjunto discreto de puertas cuánticas [1]. Las tres construcciones son las siguientes.

3.4.1. Las puertas unitarias de dos niveles son universales

Supongase que se tiene una matriz unitaria U de dimensión N. Se denota esta matriz por $U(N)$. Se define además una matriz unitaria *de dos niveles* como una matriz que realiza una operación unitaria a un subespacio de dos dimensiones del espacio de Hilbert N-dimensional, dejando el resto del espacio, de dimensión $N-2$, tal y como estaba.

Las matrices de dos niveles pueden ser usadas para hacer todos los elementos no diagonales de $U(N)$ nulos mediante un proceso similar a la eliminiación Gaussiana. La matriz $U(N)$ se multiplica por matrices de dos niveles T_{Nq}. Se eligen estas matrices de tal forma que al multiplicar sucesivamente con $q = N-1, ..., 1$ la última fila y la última columna se hagan nulas excepto por el valor de la diagonal.

$$U(N) \cdot T_{N,N-1} \cdots T_{N,1} = \begin{bmatrix} U(N-1) & 0 \\ 0 & e^{i\alpha} \end{bmatrix}$$

Las transformaciones T_{Nq} pueden seguir aplicandose de forma recursiva hasta que se ha reducido la matriz original $U(N)$ a la identidad I_N de dimensión N. Es decir

$$U(N) \cdot T_{N,N-1} \cdot T_{N,N-2} \cdots T_{2,1} \cdot D = I_N$$

donde D es una matriz diagonal de elementos con módulo 1 que se deshace de las fases que van resultando. Despejando de esta expresión se llega a

$$U(N) = (T_{N,N-1} \cdot T_{N,N-2} \cdots T_{2,1} \cdot D)^{-1}$$

que, como todas las transformaciones son unitarias, puede escribirse como

$$U(N) = D^* \cdot T_{2,1}^\dagger \cdots T_{N,N-2}^\dagger \cdot T_{N,N-1}^\dagger$$

Por lo tanto, cualquier matriz unitaria de dimensión N puede escribirse como producto de matrices unitarias de dos niveles. Este producto tiene, como mucho, $N(N-1)/2$ operaciones T_{Nq}. Sin embargo, es posible que algunas matrices se puedan descomponer utilizando menos operaciones.

Esta demostración fue realizada por primera vez por Reck *et al.* [8], quienes además idearon un método para levar a cabo las operaciones T_{Nq} de forma experimental utilizando divisores de haz. En [1] se propone un método alternativo para obtener la descomposición de forma teórica, y se demuestra que tal descomposición siempre existe.

3.4.2. Las puertas de un qubit y CNOT son universales

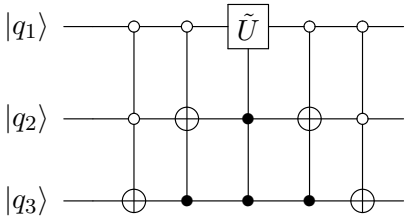
La universalidad de *CNOT* y operaciones unitarias de un qubit fue demostrada por primera vez por DiVincenzo [7]. Sin embargo, la prueba presentada aquí estaba basada en la presentada en [1] que es conceptualmente mucho más sencilla.

Se parte de una matriz U de dos niveles como se ha definido en la sección anterior. Como es una matriz de dos niveles solo actúa de forma no trivial en un subespacio de dimensión 2, generado por los estados denominados $|s\rangle$ y $|t\rangle$. Además se define $\tilde{U}$ como la submatriz 2x2 de U no trivial. $\tilde{U}$ puede entenderse como un operador unitario que actúa sobre un solo qubit.

Para poder construir U utilizando tan solo puertas de un qubit y *CNOT* se utilizan los *códigos de Gray*. Un código de Gray es una secuencia de números binarios de tal forma que los elementos adyacentes en la secuencia difieran solo en un bit. Por ejemplo, el código de Gray entre los números 0001 y 1100 es la secuencia 0001, 0000, 1000, 1100. Nótese que el código de Gray entre dos números representados por n bits puede tener como mucho $n+1$ elementos, ya que como mucho estos números difieren en n posiciones.

El procedimiento utilizado para construir U es entonces el siguiente. Se tiene el código de Gray $g_1, ..., g_m$ que conecta s y t con $g_1 = s$ y $g_m = t$. Se hacen actuar una serie de puertas cuánticas que lleven a cabo los cambios $|g_1\rangle \rightarrow |g_2\rangle \rightarrow ... \rightarrow |g_{m-1}\rangle$. Como los estados adyacentes difieren como mucho en un bit, estos cambios se pueden realizar trivialmente con una sola puerta $\wedge_k(X)$ con qubits de control los que son diferentes entre ambos estados. Después se realiza la operación $\tilde{U}$ controlada, con el qubit objetivo siendo el qubit que es diferente entre $|g_{m-1}\rangle$ y $|g_m\rangle$, y el resto de qubits siendo de control. Por último se deshacen los cambios hechos en el primer paso, es decir, $|g_{m-1}\rangle \rightarrow ... \rightarrow |g_1\rangle$, simplemente aplicando en orden inverso las mismas puertas que se aplicaron al principio.

Por ejemplo, el código de Gray 000, 001, 011, 111, que une los estados $|000\rangle$ y $|111\rangle$, da lugar al siguiente circuito



donde la base está expresada como $|q_1 q_2 q_3\rangle$. Como puede observarse, $\tilde{U}$ actúa sobre el primer qubit tan solo cuando el estado es $|000\rangle$ o $|111\rangle$. Este circuito implementa por tanto una operación de dos niveles que solo actúa de forma no trivial sobre los estados $|000\rangle$ y $|111\rangle$.

Se puede ver por tanto que implementar una operación unitaria de dos niveles requiere como mucho $2(n-1)$ operaciones controladas para realizar el cambio de $|g_1\rangle$ a $|g_{m-1}\rangle$ y deshacerlo después, donde n es el número de qubits. Cada una de estas operaciones controladas son del tipo $\wedge_k(X)$ y por tanto requieren $\mathcal{O}(n)$ puertas de un solo qubit y *CNOT*, como se ha visto en la Sec. 3.3. Por lo tanto la implementación de una operación de dos niveles require $\mathcal{O}(n^2)$ puertas de un qubit y *CNOT*. Como se ha visto en la sección anterior, una matriz unitaria de dimensión N se puede escribir como producto de, como mucho, $\mathcal{O}(N^2) = \mathcal{O}(4^n)$ operaciones de dos niveles.

Combinando ambos resultados se llega a que una operación unitaria cualquiera que actúa so-

bre n qubits puede ser implementada utilizando como mucho $\mathcal{O}(n^2 4^n)$ puertas de un qubit y *CNOT*. Es decir, estas son universales para la computación cuántica.

3.4.3. Un conjunto discreto de puertas universales

Falta por último utilizar los resultados anteriores para obtener un conjunto *discreto* de puertas que sean suficientes para la computación cuántica universal. Obviamente un conjunto discreto nunca va a poder implementar una operación unitaria arbitraria de forma exacta, porque el conjunto de matrices unitarias es continuo. Por lo tanto se va a buscar un conjunto discreto que sea capaz de *aproximar* cualquier matriz unitaria con una cierta precisión.

La siguiente demostración fue propuesta por Boykin *et al.* [9]. En este artículo se considera el conjunto discreto de puertas H, T (Fig. 2b) y *CNOT*, que es el conjunto considerado *estándar* de puertas universales y se denota con la letra ζ. Existen otros conjuntos discretos universales, sin embargo, su universalidad es más complicada de demostrar [1].

Es bastante útil definir las siguientes identidades.

$$\begin{aligned} \sigma_x^\alpha &= H\sigma_z^\alpha H \\ \sigma_y^\alpha &= \sigma_z^{1/2}\sigma_x^\alpha\sigma_z^{-1/2} \qquad \sigma_z^\alpha = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\pi\alpha} \end{bmatrix} \\ H^\alpha &= \sigma_y^{1/4}\sigma_z^\alpha\sigma_y^{-1/4} \end{aligned}$$

donde σ_i son las matrices de Pauli. Nótese que la puerta T perteneciente al conjunto universal es simplemente $\sigma_z^{1/4}$.

La demostración es la siguiente. Considerense la rotaciones que resultan de los siguientes productos de matrices unitarias del conjunto discreto ζ

$$\begin{aligned} e^{i\lambda\pi\hat{n}_1\cdot\vec{\sigma}} &\equiv \sigma_z^{-1/4}\sigma_x^{1/4} \\ e^{i\lambda\pi\hat{n}_2\cdot\vec{\sigma}} &\equiv H^{-1/2}\sigma_z^{-1/4}\sigma_x^{1/4}H^{1/2} \end{aligned} \tag{7}$$

de tal forma que ambas rotaciones son de $2\pi\lambda$ radianes pero alrededor de direcciones diferentes, $\hat{n}_1$ y $\hat{n}_2$. Ambas ecuaciones se pueden combinar para obtener λ, que resulta ser

$$\cos\lambda\pi = \cos^2\frac{\pi}{8} \tag{8}$$

Además al combinarlas se pueden obtener $\hat{n}_1$ y $\hat{n}_2$ de forma explícita, y se puede ver que son perpendiculares [9]. Se puede demostrar, partiendo de la Ec. (8), que λ es un número irracional [9]. Al ser irracional puede por tanto usarse para aproximar cualquier fase

$$e^{in\lambda\pi} \approx e^{i\phi}$$

para un $n \in \mathbb{N}$. Por lo tanto $\exists j \in \mathbb{N}$ tal que $\lambda\pi j \approx \alpha$ y $\exists k \in \mathbb{N}$ tal que $\lambda\pi k \approx \beta$. Entonces se tiene

$$\begin{aligned} \left(e^{i\lambda\pi\hat{n}_1\cdot\vec{\sigma}}\right)^j &\approx e^{i\alpha\hat{n}_1\cdot\vec{\sigma}} \\ \left(e^{i\lambda\pi\hat{n}_2\cdot\vec{\sigma}}\right)^k &\approx e^{i\beta\hat{n}_2\cdot\vec{\sigma}} \end{aligned}$$

Con esto ya es suficiente. Las rotaciones de la parte derecha de la ecuación anterior son suficientes para realizar cualquier matriz unitaria U, ya que al ser $\hat{n}_1$ y $\hat{n}_2$ perpendiculares se puede realizar una construcción como la de la Ec. (6). Como las expresiones de la Ec. (7) pueden escribirse únicamente en función de puertas H y T, entonces estas son suficientes para *aproximar* cualquier matriz de $U(2)$.

Este resultado no dice nada de la precisión con la que H y T pueden realizar esta aproximación. Sin embargo es fácil ver [9] que para un $\epsilon > 0$ cualquiera se pueden realizar un número polinómico en $1/\epsilon$ de iteraciones de $e^{i\pi\lambda}$ tal que se llege a $e^{i\phi}$ con una precisión de ϵ. Esta aproximación se puede realizar por tanto con una precisión ϵ arbitraria. Sin embargo, se sabe que en general aproximar operaciones unitarias es un problema complicado, en el sentido de que se requiere un número de puertas exponencial en el número de qubits [1].

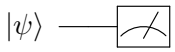
En la Sec. 3.4.2 se demostró que las puertas de un qubit junto con *CNOT* son universales. En esta sección se ha visto que H y T son suficientes para aproximar cualquier puerta de un qubit. Por lo tanto combinando ambos resultados se llega a que el conjunto $\{H,\ T,\ CNOT\}$ es universal para la computación cuántica con precisión arbitraria.

3.5. Medición

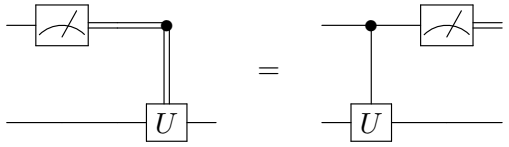
El elemento final de los circuitos cuánticos es la medición de los qubits. Medir un qubit con-

siste en colapasar el estado cuántico asociado en una base correspondiente (en general la base de computación $\{|0\rangle, |1\rangle\}$). Al colapsar los estados se transforma la información cuántica contenida en información clásica, es decir, los qubits pasan a ser bits. En general una medición es una operación irreversible, ya que a partir de un bit clásico no se puede averiguar las amplitudes que tenía el qubit original. Para que una medición sea reversible no debe revelar información sobre el estado cuántico que está siendo medido [1].

Para designar una medición se utiliza el siguiente símbolo



Las mediciones se utilizan a menudo como operación intermedia para controlar otras operaciones más adelante en el circuito. Sin embargo estas mediciones *siempre* pueden moverse al final del circuito [1]. Esta equivalencia se puede ilustrar de la siguiente forma, donde las líneas dobles denotan bits clásicos



3.6. Simulación de sistemas cuánticos

Una de las potenciales aplicaciones más importantes de los ordenadores cuánticos es la simulación de sistemas cuánticos reales. En general para simular clásicamente un sistema cuántico (como una molécula, por ejemplo) se requiere una cantidad de información que es exponencial con el tamaño del sistema a simular [1]. Parece evidente, entonces, que ordenadores que funcionan basados en las leyes de la mecánica cuántica puedan simular esta de forma más eficiente. El primero en proponer tal idea fue Feynman, en 1982 [1].

En general, las simulaciones de sistemas clásicos se basan en la resolución de las ecuaciones diferenciales que caractericen las leyes del sistema que se quiere simular. El objetivo es obtener el estado del sistema en un tiempo t dado un estado inicial particular. Para llevar a cabo este proceso se aproxima el estado con una representación digital y se discretiza la ecuación en el tiempo y el espacio, de tal forma que al iterar la evolución se avance desde el estado inicial hasta el estado final [1].

La ecuación que gobierna el comportamiento de la mayoría de sistemas cuánticos es la ecuación de Schrödinger, que se puede expresar de la siguiente forma [1]

$$i\frac{\partial}{\partial t}\psi = \left[-\frac{1}{2m}\frac{\partial^2}{\partial x^2} + V(x;t)\right]\psi \tag{9}$$

Esta ecuación no es particularmente más complicada que otras ecuaciones de sistemas clásicos, como puede ser la ecuación de Poisson. La dificultad añadida aparece, como ya se ha mencionado, en que se necesita resolver un número exponencial de ecuaciones diferenciales.

En el caso de hamiltonianos independientes del tiempo, la Ec. (9) se reduce a [1]

$$|\psi(t)\rangle = e^{-iHt}|\psi(0)\rangle \tag{10}$$

En general un hamiltoniano H cualquiera es complicado de exponenciar. Por suerte, en muchos sistemas físicos de múltiples partículas estas solo interactuan de forma local, por lo que se puede descomponer el hamiltoniano en

$$H = \sum_{k=1}^{L} H_k \tag{11}$$

donde cada H_k actúa como mucho sobre un número constante de partículas y L es polinómico con respecto al número de particulas total [1]. Esta aproximación de localidad es bastante razonable, porque la mayoría de las interacciones presentes en la naturaleza se debilitan rápidamente al aumentar la distancia.

La descomposición de la Ec. (11) se apoya además en la siguiente ecuación

$$\lim_{n\to\infty}\left(e^{iAt/n}e^{iBt/n}\right)^n = e^{i(A+B)t} \tag{12}$$

conocida como la fórmula de Trotter [1]. Nótese que A y B no tienen por qué conmutar. Modificaciones de esta fórmula, como la Ec. (13), permiten aproximar el hamiltoniano de la Ec. (11) como un producto de factores $e^{-iH_k\Delta t}$. Siempre y cuando cada uno de estos factores pueda ser

aproximado por un circuito cuántico, entonces también podrá serlo el hamiltoniano (ya que el circuito de un producto de operadores es simplemente la combinación de los circuitos de cada operador).

$$e^{i(A+B)\Delta t} = e^{iA\Delta t}e^{iB\Delta t} + \mathcal{O}(\Delta t^2) \qquad (13)$$

Algoritmo de simulación cuántica [1]

Requisitos: Se parte de un hamiltoniano de la forma dada por la Ec. (11), un estado inicial $|\psi_0\rangle$ y un tiempo t_f al que se desee evolucionar el sistema.

Procedimiento: Se elige una representación de tal forma que el estado $|\tilde{\psi}\rangle$ compuesto por n qubits aproxime el sistema físico, y de tal forma que cada operador $e^{-iH_k\Delta t}$ se pueda aproximar de forma eficiente con un circuito cuántico. Se construye el circuito cuántico correspondiente, $U_{\Delta t}$, utilizando una modificación adecuada de la fórmula de Trotter (como por ejemplo la Ec. (13)) y una discretización Δt.

1. $|\tilde{\psi}_0\rangle \leftarrow |\psi_0\rangle\,; \quad j \leftarrow 0$
2. $|\tilde{\psi}_{j+1}\rangle \leftarrow U_{\Delta t}\,|\tilde{\psi}_j\rangle$
3. $j \leftarrow j+1$; ir a 2 hasta que $j\Delta t \geq t_f$
4. $|\tilde{\psi}(t_f)\rangle \leftarrow |\tilde{\psi}_j\rangle$

Resultado: El resultado obtenido es el estado final $|\tilde{\psi}(t_f)\rangle$, que es el estado en el tiempo t_f tras la evolución del hamiltoniano.

Este algoritmo computa la evolución temporal del sistema (Ec. (10)) y por tanto es una simulación de un sistema cuántico. Hay una principal diferencia, sin embargo, con una simulación clásica: cada iteración del algoritmo cuántico reemplaza completamente el estado del sistema. Por lo tanto no hay forma de obtener información de un estado intermedio sin alterar significativamente el algoritmo. Además, el estado final debe ser medido con cautela para garantizar que se obtiene la información que se desea, ya que una vez medido no se puede obtener más información de él.

Por último hay que mencionar que existen muchos hamiltonianos que no pueden ser simulados por ningún circuito cuántico. Esto se debe a que, como se ha mencionado en la Sec. 3.4.3, existen transformaciones unitarias que no pueden ser aproximadas de forma eficiente por un ordenador cuántico. Estos algoritmos, sin embargo, no aparecen en la naturaleza, porque si lo hiciesen podrían utilizarse para realizar computaciones más allá del modelo de circuitos cuántico [1].

4. Software cuántico

El software cuántico es la descripción de algoritmos cuánticos utilizando lenguajes de programación. Utilizar lenguajes de programación es útil porque permite tratar ideas algorítmicas (como por ejemplo el algoritmo de simulación de la sección anterior) y transformarlas en programas prácticos para estudiar su utilidad, eficiencia y mejorar su diseño más allá de un simple trato teórico. Sin embargo, como la realización experimental de los ordenadores cuánticos es tan complicada, a menudo el software cuántico se ejecuta en ordenadores clásicos, que, si bien en principio no son los suficientemente rápidos como para simular muchos qubits, pueden ser útiles para estudiar programas pequeños.

Los lenguajes de programación cuánticos y sus arquitecturas y compiladores han sido un gran objeto de estudio en la literatura [10]. Sin embargo esta sección se centra en *OpenQASM*, que es el lenguaje de ensamblador cuántico diseñado por IBM y descrito en [10]. OpenQASM fue diseñado específicamente para ser utilizado en los ordenadores cuánticos de pocos qubits de IBM, aunque también puede ejecutarse en sistemas clásicos y esta es su principal ventaja frente a otros lenguajes.

La sintaxis de OpenQASM tiene elementos de C y lenguajes ensamblador. Las instrucciones se separan con ; y la primera línea debe ser siempre `OPENQASM M.m`; donde `M.m` indica la versión (por ejemplo 2.0). Los únicos tipos de variables disponibles son `qreg` y `creg`, y con ellos se pueden crear registros de bits cuánticos y clásicos respectivamente, de la siguiente forma

```
qreg rcuant[M];
creg rclas[N];
```

donde `rcuant` y `rclas` son los nombres de las variables y `M` y `N` el número de qubits y bits, respectivamente. Nótese que, como en la mayoría de lenguajes de programación, los nombres de las variables son completamente arbitrarios y se dejan a elección del programador.

Sobre los registros cuánticos se pueden aplicar puertas lógicas cuánticas, haciendo

```
puerta rcuant[i];
```

donde `i` indica el qubit del registro que se está modificando y `puerta` es el nombre (arbitrario) de la puerta lógica cuántica. Para puertas de más de un qubit se indican los diferentes qubits separados por comas. OpenQASM tiene por defecto definidas cualquier puerta de un qubit y la puerta *CNOT*, que, como se vio en la Sec. 3.4.2, son universales para la computación cuántica. Su sintaxis es

```
U(gamma, beta, delta) rcuant[i];
CX rcuant[i], rcuant[j];
```

donde `gamma`, `beta` y `delta` son los ángulos que permiten definir cualquier rotación de un qubit, tal y como se indica en la Ec. (5) (salvo fase arbitraria). En el caso de la puerta *CNOT*, `i` es el qubit de control y `j` el objetivo.

Se pueden definir además puertas propias, de forma similar a las funciones en lenguajes de programación clásicos

```
gate puerta(params) qargs {...}
```

donde `params` son los argumentos y `qargs` son los qubits sobre los que actúa la puerta. Múltiples argumentos y qubits se separan con comas. Dentro de los corchetes deben ir las instrucciones necesarias para definir la puerta, haciendo uso de los argumentos y los qubits.

Las mediciones se pueden realizar de la siguiente forma

```
measure rcuant[i] -> rclas[j];
```

donde `measure` colapsa el qubit `i` de `rcuant` y su información clásica se guarda en el bit clásico `j` del registro clásico `rclas`.

Como en muchos lenguajes de programación tradicionales, se pueden escribir comentarios utilizando `//`. Además se puede incluir código definido en otros archivos con la etiqueta `include "archivo"`. El resto de elementos de sintaxis son menos importantes, y están definidos de forma extensa en [10].

Para ilustrar la sintaxis se presenta el siguiente ejemplo, que pone en práctica la mayoría de las reglas anteriores para ejecutar el circuito ya visto de la Fig. 3a (salvo fase arbitraria).

```
OPENQASM 2.0;
qreg rcuant[2];
gate A q {...}
gate B q {...}
gate C q {...}
C rcuant[1];
CX rcuant[0], rcuant[1];
B rcuant[1];
CX rcuant[0], rcuant[1];
U(0, 0, 2*alpha) rcuant[1];
A rcuant[1];
```

Las puertas `A`, `B` y `C` y el parámetro `alpha` dependen del operador U de la Fig. 3a.

OpenQASM es un lenguaje de bajo nivel (como cualquier lenguaje ensamblador), en el sentido de que las instrucciones que ejecuta y los registros de memoria que maneja son muy cercanos a la realización física. Por ello puede ser complicado integrarlo con otras herramientas y por tanto hacer análisis estadístico, dibujar gráficas, etc. Para solucionar este problema IBM creó *qiskit*, una librería que permite ejecutar instrucciones de OpenQASM directamente en Python, de tal forma que pueden combinarse con el resto de funcionalidad de este lenguaje.

5. Supremacía cuántica

Si bien todos los resultados anteriores son de tremendo interés, sin una realización experimental la computación cuántica no sería más que una curiosidad teórica. Sin embargo la realización de circuitos, algoritmos y sistemas de comunicación cuánticos es extremadamente complicada [1] y por tanto hay múltiples paradigmas de construcción muy distintos entre sí (qubits superconductores, iones atrapados, óptica cuántica, etc).

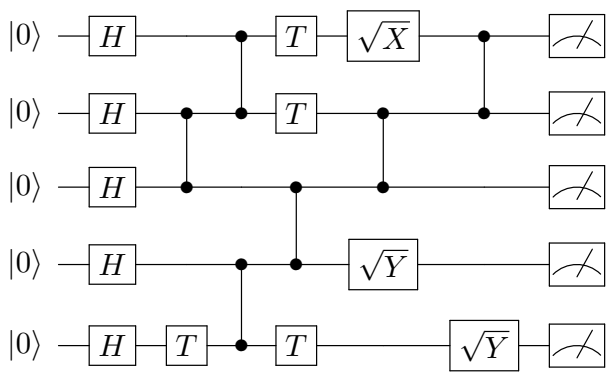


Figura 4: Ejemplo de un circuito cuántico aleatorio (de profundidad 6) en una configuración unidimensional de qubits. El conjunto discreto de puertas en este caso es $\xi = \{H, T, CZ, \sqrt{X}, \sqrt{Y}\}$.

Como no se pueden cubrir todos, esta sección se va a centrar en concreto en la realización física que ha llevado a Google a su resultado de supremacía cuántica [2], que, como se dijo en la introducción, supone la primera realización de un ordenador cuántico que ha superado a uno clásico en alguna tarea. Para ello se va a explicar primero el desarrollo teórico que hay tras la prueba de supremacía (expuesto en [11]) y después se relacionará con la realización experimental de [2].

5.1. Circuitos cuánticos aleatorios

Los circuitos cuánticos aleatorios son aquellos circuitos en los que las puertas lógicas cuánticas que los componen han sido elegidas de forma aleatoria. Constituyen un ejemplo de *caos cuántico*, ya que son sistemas en los que un pequeño cambio en las condiciones iniciales provoca una gran divergencia en el estado final. Más formalmente, la cantidad $|\langle\psi_t|\psi_t^\epsilon\rangle|^2$ decrece exponencialmente con el tiempo, donde $|\psi_t\rangle$ es el estado correspondiente al circuito y $|\psi_t^\epsilon\rangle$ el estado que resulta de una pequeña perturbación ϵ al hamiltoniano de evolución. Los estados resultantes no poseen simetrías y están extendidos por todo el espacio de Hilbert.

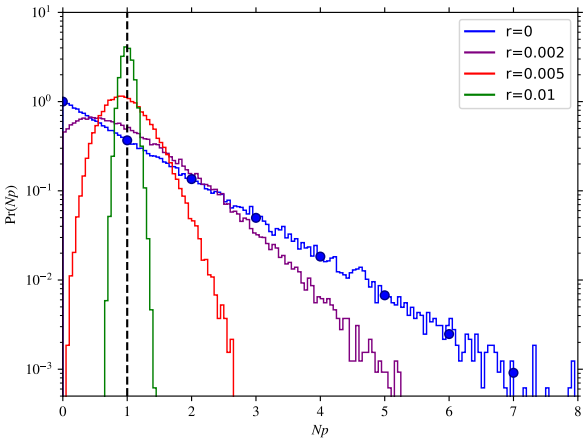
Las puertas utilizadas en los circuitos cuánticos aleatorios pertenecen a un conjunto discreto ξ, que es en general universal, de tal forma que la totalidad de los circuitos que pueden ser generados cubre $U(2)$ completamente. En la Fig. 4 puede verse un circuito cuántico aleatorio. En este caso, y en el resto de esta sección, el conjunto discreto de puertas es $\xi = \{H, T, CZ, \sqrt{X}, \sqrt{Y}\}$. La profundidad es el número de etapas que tiene el circuito, donde en cada etapa se le aplica a cada qubit una puerta de ξ, o ninguna. El circuito de la figura tiene una profundidad de 6.

En la práctica, los circuitos cuánticos aleatorios no se crean puramente al azar, sino que están sujetos a unas ciertas restricciones. La mayoría de estas restricciones son debidas a limitaciones de las realizaciones físicas de ordenadores cuánticos, aunque también se pueden imponer de forma arbitraria para dotar al circuito de ciertas características. Por ejemplo, en el caso de la Fig. 4, como la disposición real de los qubits es de una dimensión, solo puede aplicarse la puerta CZ entre qubits adyacentes. Además, como regla arbitraria, la puerta H se aplica solo a todos los qubits del primer ciclo.

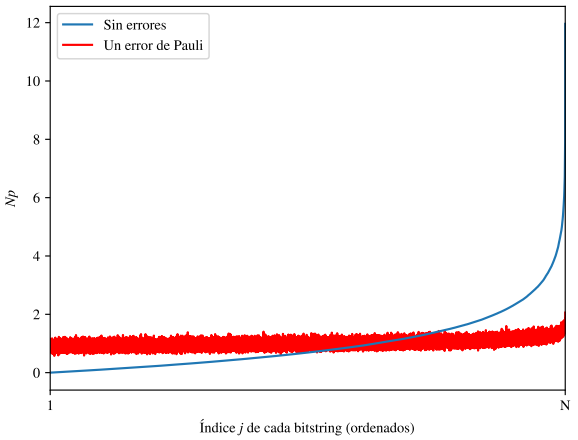
Un circuito cuántico aleatorio es por tanto una transformación U sobre n qubits de tal forma que el estado final sea $|\psi\rangle = U|\psi_0\rangle$ y el inicial $|\psi_0\rangle = |0\rangle^n$. La probabilidad de que el estado final sea el bitstring $x_i \in \{0,1\}^n$ es $p_U(x_i) = |\langle x_i|\psi\rangle|^2$. La distribución de estas probabilidades se aproxima a la forma exponencial Ne^{-Np}, conocida como la distribución de Porter-Thomas, a una velocidad que crece con la profundidad del circuito y con la dimensión de la red en la que están situados los qubits.

Es interesante estudiar no solo un resultado individual, sino una muestra de bitstrings obtenidos por múltiples ejecuciones del circuito. Se denomina muestra S al conjunto $S = \{x_0, ..., x_m\}$ de bitstrings obtenidos de m medidas del circuito en la base computacional. Siempre que m sea polinómico en n, cada bitstring de la muestra será único, ya que cada una de las probabilidades $p_U(x_i)$ es del orden $1/N = 2^{-n}$. Esto implica que el resultado de un circuito cuántico aleatorio no puede distinguirse del resultado de un muestreo uniforme, a no ser que se calcule la distribución de probabilidad del circuito $p_U(x_i)$.

Si se calcula la distribución de probabilidad, entonces, como la distribución de Porter-Thomas tiene un dominio sustancial en $Np < 1$, se puede



(a) Distribuciones de probabilidad reescalada. Los puntos azules son la distribución de Porter-Thomas y la línea negra discontinua la distribución uniforme. Las líneas de distintos colores son las distribuciones del circuito cuántico aleatorio simulado, con distintos ratios de error de Pauli r.



(b) Distribución de probabilidades $p_U(x_j)$ de los bitstrings x_j ordenada de forma creciente (línea azul). Misma distribución tras añadir una puerta Z en un solo lugar en el circuito y realizar la media de las probabilidades para todos los lugares posibles (línea roja).

Figura 5

distinguir completamente de la distribución uniforme (cuya forma viene dada por $\delta(p - 1/N)$). Ambas distribuciones pueden observarse en la Fig. 5a, donde además puede apreciarse su clara diferencia.

La probabilidad conjunta de todos los elementos de la muestra S es

$$Pr_U(S) = \prod_{x_i \in S} p_U(x_i)$$

Por el teorema del límite central se tiene entonces

$$\begin{aligned} \log Pr_U(S) &= \sum_{x_i \in S} \log p_U(x_i) \\ &= -mH(p_U) + \mathcal{O}(m^{1/2}) \end{aligned} \tag{14}$$

donde

$$H(p_U) = -\sum_{i=1}^{N} p_U(x_i) \log p_U(x_i) \tag{15}$$

es la entropía de la distribución. Como la distribución se aproxima a la de Porter-Thomas entonces su entropía se puede escribir como

$$\begin{aligned} H(p_U) &= -\int_0^\infty pN^2 e^{-Np} \log p dp \\ &= \log N - 1 + \gamma \end{aligned} \tag{16}$$

donde $\gamma \approx 0{,}577$ es la constante de Euler.

5.2. Cross-entropy

Sea $A_{cl}(U)$ un algoritmo clásico con coste computacional polinómico en n, que toma como entrada un circuito cuántico aleatorio U y devuelve como salida un bitstring x con distribución de probabilidad $p_{cl}(x|U)$. Considerese entonces una muestra de múltiples resultados de este algoritmo $S_{cl} = \{x_1^{cl}, ..., x_m^{cl}\}$. La probabilidad conjunta de que esta muestra sea observada por el circuito es

$$Pr_U(S_{cl}) = \prod_{x_i^{cl} \in S_{cl}} p_U(x_i^{cl})$$

El teorema del límite central implica que

$$\log Pr_U(S_{cl}) = -mH(p_{cl}, p_U) + \mathcal{O}(m^{1/2}) \tag{17}$$

donde

$$H(p_{cl}, p_U) = -\sum_{i=1}^{N} p_{cl}(x_i|U) \log p_U(x_i)$$

se denomina *cross-entropy* entre $p_U(x)$ y $p_{cl}(x|U)$. Esta ecuación relaciona las distribuciones de probabilidad teórica $p_U(x_i)$ con las resultantes de

un algoritmo cualquiera $A_{cl}(U)$ y sirve por tanto para juzgar si este algoritmo esta simulando de forma correcta el circuito cuántico aleatorio. Por ejemplo, si la *cross-entropy* es mayor que la entropía $H(p)$ eso implica que la distribución $p_{cl}(x|U)$ esta muestreando bitstrings que tienen una menor probabilidad de ser observados por el circuito U.

Cuando $A_{cl}(U)$ es un algoritmo ideal, entonces su distribución de probabilidad es igual a la teórica y por tanto la *cross-entropy* es simplemente a la entropía de la distribución (Ec. (15)). La *cross-entropy* de un algoritmo aleatorio uniforme es

$$H_0 = \log N + \gamma$$

donde γ es la constante de Euler.

Nótese que, para conocer el estado del circuito tras un tiempo finito se requiere una simulación directa de este. Por lo tanto, como tal simulación directa es necesariamente exponencial en n, la salida de cualquier algoritmo clásico que no sea exponencial en n está prácticamente no relacionada con la distribución de $p_U(x_i)$.

Se propone entonces la siguiente forma de comprobar si se ha llegado a la supremacia cuántica. Se mide la calidad de un algoritmo A como la diferencia entre su *cross-entropy* y la *cross-entropy* de una distribución clásica uniforme. Esta cantidad se denomina diferencia de *cross-entropy* y es

$$\begin{aligned}\Delta H(p_A) &= H_0 - H(p_A, p_U) \\ &= \sum_i \left(\frac{1}{N} - p_A(x_i|U)\right) \log \frac{1}{p_U(x_i)}\end{aligned}$$

Nótese que en este caso A puede ser un experimento cuántico real, o un algoritmo clásico exponencial o polinómico en n, y la diferencia de *cross-entropy* mide lo bien que A puede predecir la salida de un circuito cuántico aleatorio U típico. Esta cantidad es igual a la unidad para un circuito cuántico aleatorio ideal, y nula para una distribución uniforme.

La diferencia de *cross-entropy* en un ordenador cuántico experimental se denomina

$$\alpha = \mathbb{E}_U \left[\Delta H(p_{\text{exp}})\right] \tag{18}$$

donde $\mathbb{E}_U$ es la media de un conjunto $\{U\}$ de circuitos. La supremacía cuántica se consigue, en la práctica, cuando

$$C < \alpha \leq 1 \tag{19}$$

donde C es la diferencia de *cross-entropy* del mejor algoritmo clásico conocido.

Como la diferencia de *cross-entropy* mide la calidad de predicción de U, si se cumple la Ec. (19) esto implica que un ordenador cuántico ha superado al mejor algoritmo clásico conocido C en la tarea del muestreo de distribuciones de un circuito cuántico aleatorio. Nótese que para circuitos de pocos qubits, que se pueden simular completamente en ordenadores clásicos, $C = 1$ y la supremacía es imposible.

5.3. Realización experimental

Para poder demostrar la supremacia cuántica se requiere calcular la diferencia de *cross-entropy* para un ordenador experimental. Sin embargo, al tratarse de un sistema real, es imposible saber de forma exacta la distribución de probabilidad del circuito, por lo que es imposible obtener $p_A(x_i|U)$. Por suerte, con el teorema del límite central, la diferencia de *cross-entropy* puede aproximarse como

$$\alpha \simeq H_0 - \frac{1}{m}\sum_{i=1}^{m} \log \frac{1}{p_U(x_i^{\text{exp}})} \tag{20}$$

De tal forma que ya no es necesario conocer la distribución de probabilidad del circuito. El proceso de estimación experimental es entonces el siguiente

- Se selecciona un circuito aleatorio U con puertas de un conjunto universal.
- Se realizan $m \sim 10^3 - 10^6$ medidas, que resultan en los bitstrings $S_{\text{exp}} = \{x_1^{\text{exp}}, ..., x_m^{\text{exp}}\}$.
- Se calcula $\log 1/p_U(x_i^{\text{exp}})$ para todos los bitstrings x_i obtenidos, con la ayuda de un ordenador clásico.
- Se estima α con la Ec. (20).

Cuando el circuito es lo suficientemente grande las cantidades $p_U(x_i^{\text{exp}})$ no se pueden obtener

de forma numérica. Entonces $C \simeq 0$ y la supremacía puede llevarse a cabo. Sin embargo esto implica también que α no puede calcularse directamente.

¿Cómo puede demostrarse entonces que la realización experimental del circuito es más eficiente que el mejor algoritmo clásico C? En [11] se proponen dos métodos distintos para calcular α, de tal forma que si los dos valores obtenidos están en acuerdo, entonces α es válido y la Ec. (19) puede cumplirse. En primer lugar, α se puede extrapolar a partir de su valor en circuitos que sí pueden simularse clásicamente, ya sea porque tienen menos qubits o porque tienen menor profundidad. En segundo lugar, α se puede aproximar a partir de estimaciones experimentales de la fidelidad del circuito.

Todo el marco teórico presentado en esta sección fue el utilizado en el experimento de supremacía cuántica [2]. El circuito se llevó a cabo en un chip de 54 qubits superconductores, que son los suficientes como para que la simulación clásica sea imposible y por tanto $C \simeq 0$. En este chip, denominado *Sycamore*, los qubits están organizados en una red bidimensional, y cada qubit puede interactuar tan solo con sus cuatro vecinos próximos. El avance más grande de este procesador frente a sus antecesores es que se consigue una alta fidelidad en operaciones de uno y dos qubits, no solo de forma aislada sino al ejecutar computaciones realistas en las que intervienen todos los qubits [2].

Sycamore se calibra con la ayuda de algoritmos clásicos y simulaciones de circuitos de menos qubits. Después se estima α de las dos formas indicadas anteriormente (extrapolación y fidelidad del circuito), y como los valores están en acuerdo y $\alpha > C$ se puede afirmar que se ha conseguido la supremacía cuántica. *Sycamore* tarda tan solo 200 segundos en obtener un millón de muestras del circuito, mientras que un muestreo equivalente y de la misma fidelidad tardaría 10.000 años en un millón de núcleos clásicos [2].

5.4. Simulación para pocos qubits

En esta sección se va a realizar una simulación clásica de circuitos cuánticos aleatorios, con el objetivo de reproducir los resultados más relevantes de [11]. Para obtener la distribución de probabilidad final de un circuito cuántico aleatorio son necesarias todas las amplitudes del estado resultante, por lo que la memoria RAM necesaria en función de los qubits es

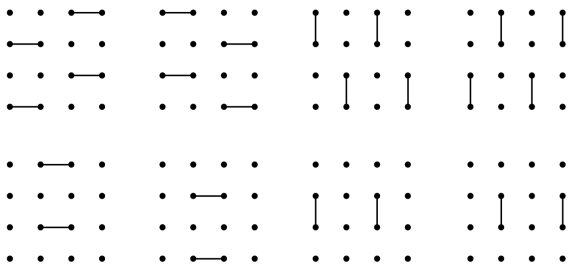


Figura 6: Distribución de las puertas CZ para una red de qubits 4x4. Estas distribuciones se iteran de forma secuencial de izquierda a derecha y arriba a abajo.

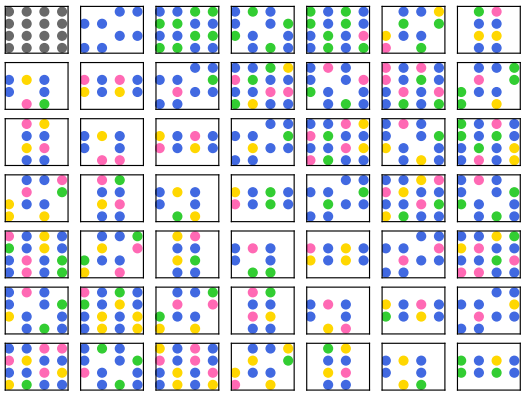
$$M(n) = 16 \cdot 10^{-9} 2^n \text{ GB}$$

ya que cada amplitud es un número complejo y por tanto ocupa 128 bits. Para llevar a cabo las simulaciones realizadas en la sección 4 de [11], por ejemplo, fueron necesarios $M(36) \approx 1100$ GB y $M(42) \approx 70400$ GB de memoria RAM.

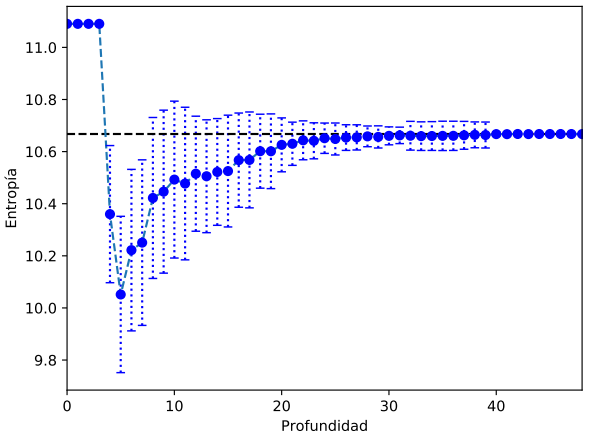
Esta cantidad de memoria es completamente inaccesible hoy en día salvo para grandes superordenadores. Por lo tanto en esta sección se van a exponer los resultados de una simulación equivalente en una red de 4x4 qubits. Esta disposición requiere tan solo $M(16) \approx 1$ MB, lo que permite usar circuitos de mayor profundidad y realizar todas las medidas necesarias en cualquier ordenador personal.

En [11] se exponen una serie de reglas que limitan la forma en la que se colocan las puertas de forma aleatoria. Estas reglas han sido elegidas utilizando optimizaciones numéricas para minimizar la profundidad requerida para llegar a la convergencia a Porter-Thomas. Además las simulaciones se limitan para que los circuitos puedan ser ejecutados por qubits superconductores reales. Es decir, no se pueden aplicar dos puertas CZ de forma simultánea entre vecinos, ya que esto es actualmente imposible de hacer con qubits superconductores [11].

El conjunto de puertas cuánticas utilizado es el ξ definido anteriormente, que es universal. Las



(a) Disposición de las puertas a distintas profundidades para uno de los circuitos cuánticos aleatorios generados (de profundidad 49). Cada color corresponde a una puerta distinta.



(b) Entropía media a distintas profundidades de un conjunto de circuitos cuánticos aleatorios. Las barras de error son la desviación estándar. La línea discontinua negra es la entropía de la distribución de Porter-Thomas.

Figura 7

reglas son por tanto las siguientes. En el primer ciclo siempre se aplica la puerta H a todos los qubits. En el resto de ciclos se hace

1. Se aplican puertas CZ de forma alternante entre las combinaciones de la Fig. 6.

2. Se aplican las puertas $\{\sqrt{X}, \sqrt{Y}, T\}$, a todos los qubits no ocupados por CZ, de forma aleatoria pero con las restricciones siguientes:

- Solo se pueden colocar puertas en el qubit q si este estaba ocupado por una puerta CZ en el ciclo anterior.

- Si no hay puertas de un qubit en el qubit q en ciclos previos (excepto la H inicial), entonces se debe colocar una puerta T.

- La puerta que se coloque en el qubit q debe ser distinta que la que había en ese mismo qubit en el ciclo anterior.

En la Fig. 7a se puede ver un ejemplo de uno de los circuitos aleatorios que se han generado utilizando estas reglas. Cada una de las cajas representa un ciclo de la ejecución, y cada círculo la puerta que se aplica al qubit correspondiente en ese ciclo. En azul puede verse la distribución de las puertas CZ tal y como se describe en la Fig. 6. Tras haber generado el circuito se realiza una simulación numérica para obtener las amplitudes de todos los qubits en el último ciclo.

Una vez se han obtenido las amplitudes es trivial calcular la distribuciónd de probabilidad, que puede verse en la Fig. 5a. Además están representadas la distribución de Porter-Thomas y la distribución uniforme. Las otras líneas son las distribuciones de probabilidad del mismo circuito, pero con distintos ratios de error de Pauli r, que es el número de errores de Pauli por puerta del circuito. Se llama error de Pauli a una puerta X (ó Z) que se añade en un qubit y a una profundidad en concreto. Añadir esta puerta provoca que el qubit "falle" ya que su valor (o su fase) queda invertido. Como puede verse, cuanto más defectuoso es el circuito más se acerca su distribución a la uniforme.

En la Fig. 5b esto se hace más evidente. Añadiendo tan solo una puerta Z en todo el circuito, y tras calcular la media resultante de introducir este error en todos los lugares posibles, la distribución pasa a ser prácticamente la uniforme. Este resultado demuestra que la distribución de probabilidad de los circuitos cuánticos aleatorios

es, en general, muy sensible incluso a errores muy pequeños.

Por último, resulta muy interesante simular la entropía del circuito a distintas profundidades, para estudiar como converge a la de Porter-Thomas y a que profundidad. Hasta el momento solo ha hecho falta calcular las amplitudes de los qubits al final del circuito, pero ahora hace falta obtenerlas tras cada ciclo. Una vez se tiene la distribución de probabilidad se puede obtener la entropía utilizando la Ec. (15). Los resultados pueden verse en la Fig. 7b.

La entropía de Porter-Thomas (línea discontinua) ha sido calculada con la Ec. (16) y es $H \approx 10{,}67$, ya que $N = 2^{16}$. Para observar mejor la convergencia a Porter-Thomas se ha calculado la media (además de la desviación estándar) de las entropías de múltiples circuitos cuánticos aleatorios distintos y como puede verse es muy clara a partir del ciclo 20. La entropía de los cuatro primeros ciclos es constante y corresponde con la de la distribución uniforme debido a la primera capa de puertas H. Las puertas en los siguientes tres ciclos son siempre diagonales y por tanto no modifican la entropía.

Si uno compara los resultados de la Fig. 7b con los de la sección 4 de [11], puede verse que la convergencia es claramente más lenta en el caso de la red 4x4 que en las otras de mayor tamaño. En principio se sabe que los circuitos cuánticos más pequeños alcanzan la distribución de Porter-Thomas más rápidamente [11]. Sin embargo, las reglas de aleatoriedad utilizadas han sido obtenidas de forma numérica para minimizar el tiempo de convergencia de las redes de [11], por lo que es posible que no sean tan efectivas para redes de menor tamaño.

Las simulaciones númericas de los circuitos se han llevado a cabo en un ordenador personal con las herramientas de software cuántico explicadas en la Sec. 4.

6. Conclusión

En el presente texto se ha hecho una introducción a la computación cuántica, y se ha puesto en contexto el primer resultado de su supremacía. La estructura de la mayoría de los textos introductorios a la computación cuántica está basada en el libro de Nielsen y Chuang [1], y la de este trabajo también. Sin embargo, las Secs. 4 y 5 ofrecen una perspectiva más moderna, y son una pequeña muestra de algunos de los campos de estudio en la actualidad.

En la Sec. 2 se explicaron los distintos componentes que forman la computación clásica. En las secciones siguientes se hizo lo propio para la computación cuántica, y se pudo ver las principales diferencias entre ambas: la computación cuántica es capaz de procesar más información, y en ocasiones muchísimo más deprisa; sin embargo, es extremadamente complicada de realizar físicamente y propensa a errores, y además tiene ciertas limitaciones que dificultan el diseño de algoritmos.

Como se ha visto, las puertas lógicas cuánticas son equivalentes a matrices unitarias, y por tanto demostrar que se puede realizar cualquier operación utilizando un conjunto discreto de estas puertas es mucho más complicado que en el caso clásico. Por suerte tal demostración existe y se vio en la Sec. 3.4, aunque, como podría esperarse, el conjunto discreto solo es capaz de *aproximar* cualquier operación unitaria.

La simulación de sistemas cuánticos es una de las aplicaciones de la computación cuántica con más potencial, ya que los sistemas cuánticos están entre los sistemas más complicados de simular en física. En la Sec. 3.6 se explicaron sus principios básicos y se expuso un algoritmo genérico que puede utilizarse para llevar a cabo cualquier simulación.

En la Sec. 5 se hizo un estudio del marco teórico tras el resultado de supremacía cuántica de Google, y se explicó brevemente su realización experimental. Aunque el muestreo de probabilidades de circuitos cuánticos aleatorios no tiene ninguna aplicación práctica, su estudio puede resultar muy útil para demostrar la fidelidad de realizaciones experimentales.

Por último, en la Sec. 5.4, se vieron los resultados de una simulación de circuitos cuánticos aleatorios 4x4, y se compararon con las predicciones teóricas y los resultados de [11]. Los resultados obtenidos permiten demostrar las propiedades más importantes de los circuitos cuánticos aleatorios, así como su relación con el caos

cuántico. La simulación se llevó a cabo en un ordenador clásico con el lenguaje de programación descrito en la Sec. 4.

La computación cuántica es una disciplina en auge y, aunque relativamente nueva, es probable que en los próximos años avance hasta convertirse en el nuevo paradigma de la computación contemporánea. Ya sea para simular sistemas cuánticos, o para llevar a cabo ciertas operaciones imposibles en ordenadores tradicionales, la computación cuántica encierra un poder computacional sin precedentes.

Referencias

[1] M. A. Nielsen, I. L. Chuang, "Quantum computation and quantum information", Cambridge Univ. Press, 2000

[2] Arute, F., Arya, K., Babbush, R. *et al.*, "Quantum supremacy using a programmable superconducting processor", Nature 574, 505–510, 2019

[3] A. M. Turing, "On computable numbers, with an aplication to the Entscheidungsproblem", Proc. Lond. Math. Soc., 42:230, 1936

[4] A. Galindo, M. A. Martín-Delgado, "Information and computation: Classical and quantum aspects", Rev. Mod. Phys. **74**, 347-423, 2002

[5] C. M. Papadimitriou, "Computational Complexity", Addison-Wesley, Reading, Massachusetts, 1994

[6] Barenco, A., C.H. Bennett, R. Cleve, D.P. DiVincenzo, N. Margolus, P. Shor, T. Sleator, J.A. Smolin, H. Weinfurter, "Elementary gates for quantum computation", Phys. Rev. A **52**, 3457-3467, 1995

[7] D. P. DiVincenzo, "Two-qubit gates are universal for quantum computation", Phys. Rev. A, 51(2):1015-1022, 1995

[8] M. Reck, A. Zeilinger, H. J. Bernstein, P. Bertani, "Experimental realization of any discrete unitary operator", Phys. Rev. Lett., 73(1):58-61, 1994

[9] P. O. Boykin, T. Mor, M. Pulver, V. Roychowdhury, F. Vatan, "On universal and fault-tolerant quantum computing", arXiv:quant-ph/9906054, 1999

[10] A. W. Cross, L. S. Bishop, J. A. Smolin, J. M. Gambetta, "Open Quantum Assembly Language", arXiv:1707.03429, 2017

[11] Boixo, S., Isakov, S.V., Smelyanskiy, V.N. *et al.* "Characterizing quantum supremacy in near-term devices", Nat. Phys. 14, 595–600, arXiv:1608.00263, 2018