Universidad Nacional de Educación a Distancia
Facultad de Ciencias
Departamento de Física


Memoria del Trabajo de Fin de Grado

# ESTUDIO DE ALGORITMOS CUÁNTICOS Y RELACIÓN DE LOS MISMOS CON LA FÍSICA

Raúl Osuna Sánchez-Infante
Tutor: Víctor Alberto Fairén Le Lay
Curso 2020/21

# Índice

**1. Resumen o abstract** **3**

**2. Introducción** **3**
2.1. Fundamentos matemáticos y lógicos . . . 3
2.1.1. Números complejos . . . 3
2.1.2. Vectores . . . 3
2.1.3. Álgebra de Boole y puertas lógicas clásicas . . . 3
2.2. Bases físicas y particularidades de la física cuántica . . . 3
2.2.1. Mecánica cuántica y aplicación a algoritmos . . . 3
2.3. Fundamentos computacionales . . . 3
2.3.1. Algoritmos . . . 3

**3. Objetivos** **4**

**4. Métodos y material** **4**

**5. Resultados** **4**
5.1. Algoritmo de Deutsch-Josza . . . 4
5.1.1. Caso constante . . . 4
5.1.2. Caso balanceado . . . 4
5.1.3. Definición del oráculo . . . 5
5.2. Algoritmo de Grover . . . 6
5.2.1. Aplicación a dos qubits [1] . . . 7
5.2.2. Aplicación a tres qubits [2] . . . 8

**6. Discusión** **10**

**7. Conclusiones** **10**

**8. Anexos** **10**
8.1. Código fuente . . . 10

**9. Bibliografía** **11**

# 1. Resumen o abstract

Esta parte queda mejor si se rellena la última.

# 2. Introducción

## 2.1. Fundamentos matemáticos y lógicos

En principio, simplemente enunciarlas y quizás incluir una simple información al respecto de las herramientas que se emplearán, ya que ya se han estudiado en los cursos anteriores del grado y no es el objetivo de este TFG. Conviene considerar si se debería mover todo este punto a la sección

### 2.1.1. Números complejos

### 2.1.2. Vectores

### 2.1.3. Álgebra de Boole y puertas lógicas clásicas

## 2.2. Bases físicas y particularidades de la física cuántica

Para no caer en el error de hacer un TFG con poca relación con la física, extenderse aquí.

### 2.2.1. Mecánica cuántica y aplicación a algoritmos

- Dualidad onda-partícula
- Superposición
- Entrelazamiento
- Interferencia
- Qubits y esfera de Bloch. Notación bra-ket. Probabilidades de un estado
- Postulados cuánticos
- Puertas cuánticas
- Representación matemática de los circuitos cuánticos
- Supremacía cuántica
- Codificación superdensa
- Criptografía cuántica (BB84)
- Teleportación cuántica

## 2.3. Fundamentos computacionales

### 2.3.1. Algoritmos

Algoritmos cuánticos. Oráculos. Descripción de algoritmos (Deutsch-Josza, Grover, etc...)

# 3. Objetivos

En este trabajo de fin de grado se buscan diversos objetivos, tales como:

- Estudio de diversos ejemplos de algoritmos cuánticos
- Comparación de los mismos con los algoritmos clásicos equivalentes, en términos como velocidad y eficiencia
- Implementación práctica personal de un algoritmo ya existente (Deutsch-Josza)
- Implementación personal y ampliación de un algoritmo ya existente (Grover para 3 qubits, implementando la búsqueda de cualquier posible solución dentre de las $2^3 = 8$ posibles, incluyendo soluciones dobles)
- Ejecución de los algoritmos tanto en simuladores cuánticos, como en ordenadores cuánticos reales.

# 4. Métodos y material

- Lenguaje Python para la programación de los algoritmos.
- Librería Qiskit para, entre otras cosas, la implementación de puertas cuánticas de una manera simple mediante código.
- Cuenta gratuita de IBM Quantum Experience para acceder a la ejecución del código en hardware cuántico real.

# 5. Resultados

## 5.1. Algoritmo de Deutsch-Josza

El algoritmo de Deutsch-Josza resuelve un problema que no tiene ningún objetivo práctico, salvo demostrar la existencia de enunciados cuya resolución a partir de un algoritmo cuántico es más eficiente que el método análogo correspondiente mediante computación tradicional.

El algoritmo se basa en hallar si una función cualquiera f(x) es constante o balanceada. Esto quiere decir, para una entrada de un único bit, se quiere hallar si su salida depende de la entrada o no.

La forma más fácil y gráfica de visualizar cada uno de los casos es mediante una tabla de la verdad, habitual en la electrónica digital. Dicha tabla podría tomar dos formas en cada uno de los dos casos, como se muestra en los siguientes apartados

### 5.1.1. Caso constante

Como el propio nombre indica, la salida será siempre 0 ó 1, independientemente de la entrada

| x | f(x) |
|---|---|
| 0 | 0 |
| 1 | 0 |

| x | f(x) |
|---|---|
| 0 | 1 |
| 1 | 1 |

Cuadro 1: Tablas de la verdad para el caso constante

### 5.1.2. Caso balanceado

La salida se alterna según sea la entrada 0 ó 1:

| x | f(x) |
|---|---|
| 0 | 0 |
| 1 | 1 |

| x | f(x) |
|---|---|
| 0 | 0 |
| 1 | 1 |

Cuadro 2: Tablas de la verdad para el caso balanceado

Se puede observar para ambos casos, que se necesitan 2 evaluaciones de la función para saber si la misma es constante o balanceada. Generalizando, para n bits, serían necesarias $2^n$ evaluaciones (o visto de otra manera, analizar todas las combinaciones posibles de bits de entrada). La complejidad del algoritmo clásico sería por tanto de $O(2^n)$.

Para la evaluación de la función mediante un algoritmo cuántico, que se espera que sea más eficiente, se conoce que todas las puertas cuánticas empleadas han de ser reversibles. Analizando f(x), se concluye rápidamente que ésta no es reversible. Se debe evitar este problema mediante la introducción de un oráculo (vocablo inglés, en español se podría denominar también como «caja negra»). Dicho oráculo tendrá como entradas dos qubits y como salidas otros dos qubits. Las entradas serán denominadas como q0 y q1, y las salidas serán simplemente q0 y la operación XOR de q1 con f(x). En el mundo cuántico, la puerta XOR no existe, sin embargo sí que hay un equivalente, que no es otro que la puerta CNOT.

El algoritmo emplea dos de las propiedades del mundo cuántico antes introducidas y que tanto se emplean para esta clase de algoritmos: la superposición y la interferencia (en este caso no se hace uso de la tercera propiedad, el entrelazamiento cuántico.

Esquemáticamente, el algoritmo para construcción del circuito cuántico y que se ha codificado en python como se puede ver en el código anexo, es el siguiente:

1. Definir dos qubits, «x» e «y», o «q0» y «q1».
2. Invertir «y» o «q1» para tener un valor de 1 en dicho qubit (los qubit se inicializan por defecto a 0).
3. Aplicar superposición mediante la puerta H de Hadamard a ambos qubits.
4. Aplicar el oráculo.
5. Aplicar de nuevo puertas H a ambos qubits, mediante lo cual se estará aplicando interferencia para poder medir el resultado.
6. Medir la salida. El valor de q0 codifica el tipo de función, significando 0 que dicha función es constante y 1 que es balanceada.

Se puede observar como el oráculo sólo se aplica una vez. Las puertas Hadamard tienen una complejidad computacional despreciable respecto al oráculo, una vez que se escala el algoritmo. Se ha pasado de una complejidad exponencial, a una constante $O(1)$, siendo por tanto la mejora exponencial.

### 5.1.3. Definición del oráculo

La definición detallada del oráculo daría para otro estudio completo por separado. Tanto en este caso del algoritmo de Deutsch-Josza como en el siguiente de Grover, no se detallarán en demasía estas «cajas negras» y simplemente se hará uso de ellas, suponiendo conocida su forma. A la hora de codificar, la forma del oráculo se implementa conocido el tipo de función, balanceada o constante (esto es en realidad la solución del problema). Sin embargo, no se vuelve a consultar esta solución para obtener la respuesta. Esto permite centrarse en el algoritmo, tema principal de estudio de este trabajo. Para este caso de Deutsch-Josza, se conoce que para la entrada de dos qubits, se deja el primero de ellos igual y para el segundo se efectúa una XOR del mismo con la función a evaluar f(x). Se analizará si hay que hacer algún procesado adicional para cada uno de los cuatro casos enunciados en el punto anterior, los dos de la función constante y los dos de la función balanceada.

Con esta explicación, se procede a analizar las tablas de la verdad para los cuatro casos. Recuérdese que una de las salidas del oráculo es inmediata y coincide con el valor del primer qubit, por lo que no se muestra en las tablas

para simplificar. Se estudia primero la construcción del oráculo para el caso de f(x) balanceada, como se muestra en el cuadro 3:

| x(q0) | y(q1) | f(x) | y XOR f(x) |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 |
| 1 | 0 | 0 | 0 |
| 1 | 1 | 0 | 1 |

| x(q0) | y(q1) | f(x) | y XOR f(x) |
|---|---|---|---|
| 0 | 0 | 1 | 1 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 1 | 0 |

Cuadro 3: Tablas de la verdad del oráculo para el caso constante

Como se puede ver, en el primero de los casos la salida corresponde con y(q0), y en el segundo de ellos, su valor opuesto. La implementación mediante puertas cuánticas sería la que se puede observar en la figura 1. Por simplicidad, se muestra todo el circuito y no sólo el oráculo, que correspondería a lo que hay entre los separadores (barreras) segundo y tercero:

Figura 1: Circuitos cuánticos para el caso de una función constante


Desarrollando el caso análogo para una función balanceada, se llega a los valores mostrados en el cuadro 4:

| x(q0) | y(q1) | f(x) | y XOR f(x) |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 1 | 0 |

| x(q0) | y(q1) | f(x) | y XOR f(x) |
|---|---|---|---|
| 0 | 0 | 1 | 1 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 |
| 1 | 1 | 0 | 1 |

Cuadro 4: Tablas de la verdad del oráculo para el caso balanceado

La implementación utilizará las mencionadas puertas cuánticas CNOT, como se puede mostrar en la figura 2:

Figura 2: Circuitos cuánticos para el caso de una función balanceada


## 5.2. Algoritmo de Grover

El algoritmo fue propuesto por Lov Grover en 1996. Se trata de un algoritmo para hallar un elemento en una lista, problema muy habitual en computación y cuya resolución clásica se estudia en cualquier curso básico de programación. Esto mismo se puede aplicar no sólo para búsquedas de una base de datos, sino para problemas de optimización (que también se basan en búsquedas de posibles soluciones), o estádisticas tales como la media, mediana, etc... Resulta intuitivo pensar que la complejidad de un algoritmo clásico de búsqueda en una lista,

buscando elemento por elemento de la misma, es de $O(n)$. Se asume que la lista no tiene estructura y se ha tomado el peor caso posible (worst case scenario).

El algoritmo cuántico a estudiar reduce la complejidad al $O(\sqrt{N})$, produciendo una mejora cuadrática. Análogamente al caso estudiado anteriormente de Deutsch-Josza, el algoritmo se vuelve a basar en los principios de superposición e interferencia para acelerar la búsqueda. Se necesitarán $log_2 N$ qubits. Esquemáticamente, los pasos del algoritmo serían los siguientes:

1. Inicialización de los qubits a 0.
2. Aplicar una puerta H de Hadamard a cada uno de los qubits (superposición).
3. Aplicación de un oráculo a los qubits, y aplicación del algoritmo de difusión de Grover (interferencia).
4. Medición.

Los puntos 2 y 3 se pueden repetir, para los casos en los que el algoritmo precise de más de una iteración. Esto se verá con más claridad en alguno de los casos de la implementación.

### 5.2.1. Aplicación a dos qubits [1]

En primer lugar, se asigna igual probabilidad a cualquiera de los elementos de la lista, ya que no se presupone ninguna solución más probable que otra, lo cual se consigue mediante la superposición (segundo punto).

La mayor complejidad del algoritmo estará por tanto en el punto tercero. Respecto al oráculo («caja negra»), para este caso lo que hará será simplemente multiplicar la amplitud de la probabilidad (que no es más que la raíz cuadrada de la probabilidad) de la solución por un valor de (-1). Recuérdese que el oráculo conoce la solución (como se introdujo anteriormente, la construcción real del oráculo es un campo complicado en el que se están realizando muchos estudios al respecto). Analizando matemáticamente, y para un ejemplo de cuatro elementos en la lista (necesitándose por tanto dos qubits), con el elemento a buscar en la cuarta posición, se necesitaría pasar del vector de amplitudes de probabilidad: $\frac{1}{2}\begin{pmatrix}1\\1\\1\\1\end{pmatrix}$, a este otro vector: $\frac{1}{2}\begin{pmatrix}1\\1\\1\\-1\end{pmatrix}$

Este se conseguiría aplicando una puerta cuya matriz fuera: $\begin{pmatrix}1&0&0&0\\0&1&0&0\\0&0&1&0\\0&0&0&-1\end{pmatrix}$, matriz que coincide con la puerta Z controlada. Para el caso genérico de una posición distinta a la cuarta, se deberán manipular debidamente los qubits para obtener una matriz análoga con el -1 en otro punto de la diagonal principal, sin más que aplicar puertas X de Pauli (equivalente a la puerta NOT de los ordenadores clásicos) antes y después de la puerta Z controlada. Esta implementación se puede ver en el código.

Operando en el ejemplo anterior:

$$\begin{pmatrix}1&0&0&0\\0&1&0&0\\0&0&1&0\\0&0&0&-1\end{pmatrix}\cdot\frac{1}{2}\begin{pmatrix}1\\1\\1\\1\end{pmatrix}=\frac{1}{2}\begin{pmatrix}1\\1\\1\\-1\end{pmatrix}$$

A partir de aquí, sólo faltaría el operador de difusión de Grover, que es el que se encarga de reflejar las amplitudes respecto de la media. La media de las amplitudes ahora es de 1. Si se refleja la amplitud respecto a esa nueva media, las tres primeras componentes del vector van a resultar nulas y la última la unidad, es decir, se pasa del vector $\frac{1}{2}\begin{pmatrix}1\\1\\1\\-1\end{pmatrix}$, al vector: $\begin{pmatrix}0\\0\\0\\1\end{pmatrix}$.

En este caso se puede ver como la medición nos daría la solución con una probabilidad del 100 % tras una única ejecución del algoritmo (recuérdese que $\sqrt{N}$ es un valor esperado para N suficientemente grande). En la figura (3) se puede ver el circuito cuántico correspondiente. Se han separado mediante barreras o separadores cada uno de los pasos del algoritmo: paso 2 al principio (los qubits ya están inicializados a cero), primera parte del paso 3 (oráculo) tras la siguiente barrera, algoritmo de difusión después de la siguiente barrera y por último la medición tras la última barrera:

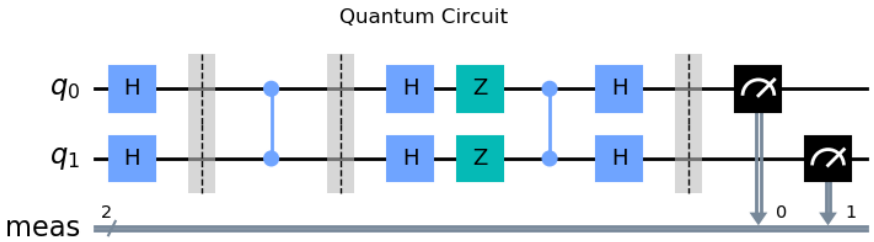


Figura 3: Circuito para el algoritmo cuántico de Grover con 2 qubits y solución «11»

Esta particularización para únicamente dos qubits se puede encontrar en muchos estudios. Por ello, y queriendo ampliar el algoritmo a un caso más complicado, se procede a implementar el mismo para tres qubits, lo que equivaldría a la búsqueda en una lista de ocho elementos.

### 5.2.2. Aplicación a tres qubits [2]

La base es la misma que en el caso anterior. Se estudirán dos casos distintos, el primero de ellos con una única solución (un único elemento a buscar), y el segundo con dos elementos distintos de la lista a buscar.

Para el primero de los casos, en la bibliografía se puede ver como la probabilidad de hallar el valor adecuado es de:

$$p = \left( \left[ \frac{N-2t}{N} + \frac{2(N-t)}{N} \right] \frac{1}{\sqrt{N}} \right)^2 = \left( \frac{5}{4\sqrt{2}} \right)^2 = 0,78125 \tag{1}$$

En donde el parámetro «t» corresponde al número de soluciones a buscar. Comparado con el caso clásico, el mismo valor de probabilidad sería:

$$p' = \frac{t}{N} + \frac{N-t}{N} \cdot \frac{t}{N-1} = \frac{1}{8} + \frac{7}{8} \cdot \frac{1}{7} = 0,25$$

Se adjunta el circuito resultante para la medición del valor «000» en la figura 4:

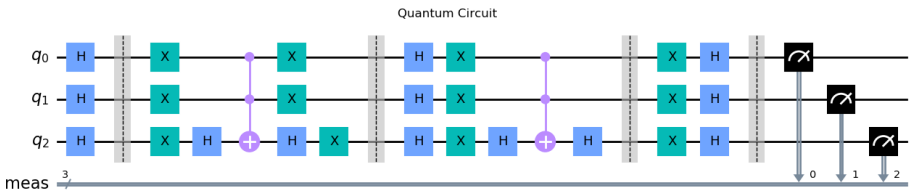


Figura 4: Circuito para el algoritmo cuántico de Grover con 3 qubits y solución «000»

Para este caso también se adjunta la probabilidad medida, como se puede ver en la figura 5:

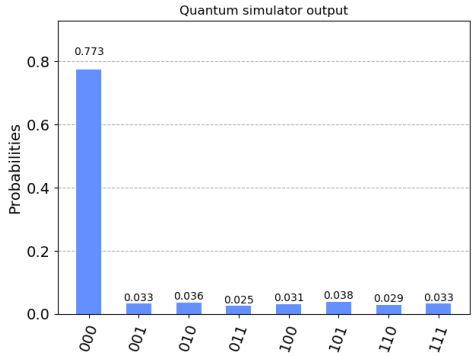


Figura 5: Probabilidad medida para el algoritmo cuántico de Grover con 3 qubits y solución «000»

Obviamente la probabilidad para el caso cuántico repitiendo el algoritmo una segunda vez mejorará, al tener $2 < \sqrt{N} < 3$. En el circuito cuántico correspondiente de la figura 6 se puede apreciar como hay una parte duplicada que corresponde a la segunda iteración. En este caso se está tratando de hallar la solución «111»:

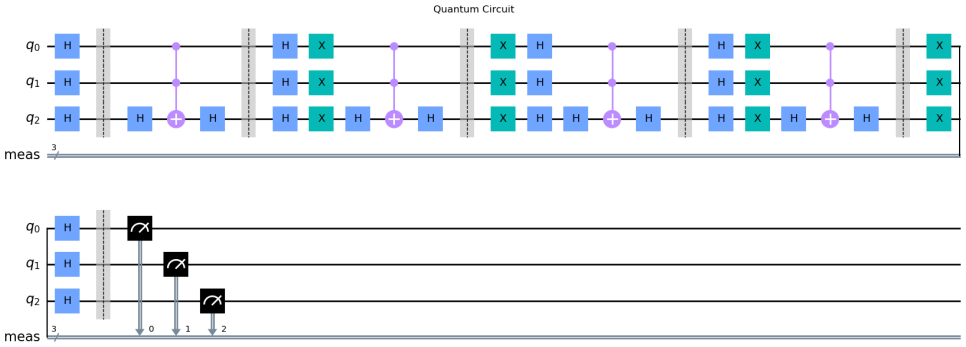


Figura 6: Circuito para el algoritmo cuántico de Grover con 3 qubits, doble iteración y solución «111»

Y efectivamente, la probabilidad aumenta en este caso:

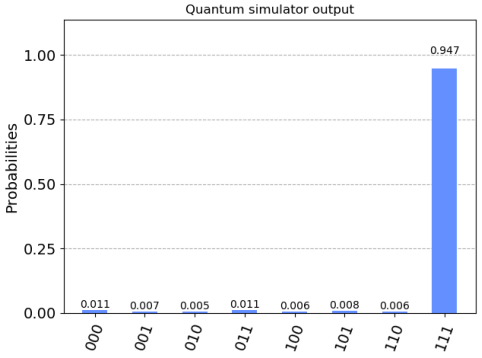


Figura 7: Probabilidad medida para el algoritmo cuántico de Grover con 3 qubits, doble iteración y solución «111»

Si se repitiera una tercera vez, el valor pasaría a empeorar (lo cual no parece intuitivo a primera vista, pero se basa en la parte del algoritmo de difusión de Grover, que refleja las amplitudes de probabilidad respecto de la media tras invertir previamente la del valor a buscar, tal y como se comentó en el punto anterior).

Para el caso de dos soluciones a buscar, $t = 2$, sustituyendo en (1), se obtiene $p = 1$ (probabilidad de encontrar al menos una de las dos soluciones), mientras que el mismo caso para el algoritmo clásico obtendría $p' = 0,464$. Por ello, en este caso no se ha permitido al algoritmo ejecutarse más de una vez.

Se puede ver un ejemplo de la ejecución del algoritmo en la figura 8:

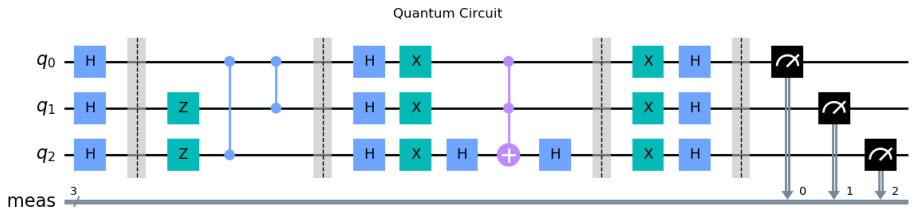


Figura 8: Circuito para el algoritmo cuántico de Grover con 3 qubits y soluciones «010» y «100»

Y las probabilidades asociadas:

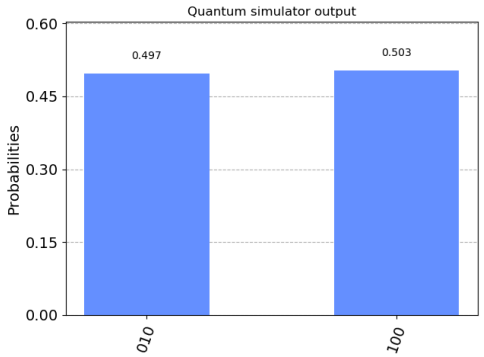


Figura 9: Probabilidad medida para el algoritmo cuántico de Grover con 3 qubits y soluciones «010» y «100»

# 6. Discusión

- Discusión de los errores en el hardware real. Explicaciones al respecto.
- Limitaciones de los algoritmos. Hardware. Estado del arte para la generación de qubits. Caracterización de los circuitos y puertas cuánticas.

# 7. Conclusiones

# 8. Anexos

## 8.1. Código fuente

Disponible en Github: https://github.com/raulillo82/TFG-Fisica-2021
Listado de archivos:

```
license.txt
d-j.py
grover.py
```

# 9. Bibliografía

# Referencias


[1] Qiskit reference: Grover's Algorithm (https://qiskit.org/textbook/ch-algorithms/grover.html)

[2] Nature: Complete 3-Qubit Grover search on a programmable quantum computer (https://www.nature.com/articles/s41467-017-01904-7)