**Implementación de Algoritmo de** **Deutsch y Deutsch-Jozsa**

**Mateo Sebastián Forero Fuentes**

**Escuela Colombiana de ingeniería Julio Garavito**

**Mateo.forero-f@mail.escuelaing.edu.co**

**27/11/2022**

*Este reporte se entrega para cumplir con los requisitos parciales del curso CNYT: Computación Cuántica- 2022-2*

# Tabla de contenidos

[Tabla de contenidos 1](#_Toc39509128)

[1 Introducción 2](#_Toc39509129)

[2 Algoritmo de Deutsch 2](#_Toc39509130)

[2.1 Problema 3](#_Toc39509131)

[2.2 Implementando las funciones en el computador cuántico 5](#_Toc39509132)

[2.3 Implementando el algoritmo de Deutsch en un computador cuántico 5](#_Toc39509133)

[3 Algoritmo de Deutsch-Jozsa 7](#_Toc39509134)

[3.1 Problema 8](#_Toc39509135)

[3.2 Implementando las funciones en el computador cuántico 12](#_Toc39509136)

[3.3 Implementando el algoritmo de Deutsch-Josza en un computador cuántico 13](#_Toc39509137)

[4 Conclusiones 16](#_Toc39509138)

# Introducción

Nuestra sociedad actualmente presenta una creciente demanda tecnológica, la cual de manera constante a buscado una mayor velocidad de procesamiento, proceso que ha a implicado inherentemente una disminución del tamaño de los transistores y microchips, este proceso se ha enfrentado a una gran barrera la cual a sido el comportamiento de los electrones como ondas, lo cual implica que ante barreras muy delgadas los electrones son capaz de atravesarlas este comportamiento a implicado la incapacidad de hacer microchips que manejen escalas de nanómetro, que es cuando se produce el efecto túnel que es el mencionado anteriormente.

Ante este problema en 1980 Paul Benioff propuso un modelo cuántico mecánico de la maquina de Turing, la cual consiste en un manejo especial de la información basándose en el uso de los qbits, que indican la cantidad de bits en superposición, esto produce que ante una cantidad mayor de qbits se permite una mayor cantidad de operaciones realizadas, tomando el ejemplo de 3 bits cuyos valores pueden ser 1 y 0 es correcto afirma que existen ocho valores posibles que se toman 1 de estos valores secuencialmente, por otro lado los qbits por su capacidad de aprovechar la superposición cuántica permite la ejecución de 8 estados de manera paralela para 3 qbits.

En la actualidad la computación cuántica se puede usar para el desarrollo de problemas simples de manera más eficaz de cómo se realizaría en un computador clásico, en este trabajo veremos dos de las aplicaciones más básicas que se pueden tener, las cuales se trabajaran mediante el algoritmo de Deutsch y Deutsch-Jozsa.

# Algoritmo de Deutsch

El algoritmo de Deutsch es un algoritmo cuántico el cual busca usar el comportamiento de un computador cuántico aprovechando el paralelismo de los estados de superposición cuántica, para dar solución más eficaz a un problema que como se haría en un computador clásico.

Para realizar el experimento se tendrá en cuenta el conjunto de los binarios 0 y 1, y a su vez las posibles funciones de , las cuales después de un breve análisis podemos determinar que son 4, a partir de estas 4 funciones vamos a realizar un análisis usando el algoritmo de Deutsch para comprobar su correcto funcionamiento a la hora de simularlo en un computador cuántico.

Para la simulación de este algoritmo vamos a usar la librería de Python qiskit la cual nos ofrece la posibilidad de simular un circuito cuántico e interpretar los resultados que el circuito nos arroje, para el uso de esta librería debemos tener en cuanta las diversas compuertas cuánticas y lógicas que nos ayudaran a la construcción del circuito cuántico.

## Problema

El problema que busca resolver el algoritmo de Deutsch es dada una función que toma un valor binario y devuelve otro valor binario averiguar si esta es una función balanceada (Devuelve 1 para un valor binario y 0 para el otro) o es una función constante(Devuelve 0 para los dos valores binarios o devuelve 1 para los dos valores binarios).

Para la solución de este problema, el algoritmo se basa en el uso de un circuito cuántico, que dada dos qbit el superior teniendo un 0 y el inferior teniendo un 1 cuando se obtiene el estado final del sistema evaluado en el primer qbit de la parte superior, este resultado nos va a arrojar un valor binario el cual podemos interpretar como, cuando el qbit arroja 0 significa que la función es constante, cuando el qbit superior arroja 1 significa que la función es balanceada.

Como establecimos anteriormente existen 4 funciones las cuales cumplen la condición de tener un dominio y rango de la forma , las cuales son las siguientes:

***Función 1:***

Dibujo correspondiente a la función:

1. 0
2. 1

Matriz correspondiente a la función:

|  |  |  |  |
| --- | --- | --- | --- |
| 0 | 1 | 0 | 0 |
| 1 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 |

***Función 2:***

Dibujo correspondiente a la función:

1. 0
2. 1

Matriz correspondiente a la función:

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 |
| 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 |

***Función 3:***

Dibujo correspondiente a la función:

1. 0
2. 1

Matriz correspondiente a la función:

|  |  |  |  |
| --- | --- | --- | --- |
| 0 | 1 | 0 | 0 |
| 1 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 |
| 0 | 0 | 0 | 1 |

***Función 4:***

Dibujo correspondiente a la función:

1. 0
2. 1

Matriz correspondiente a la función:

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 |
| 0 | 0 | 1 | 0 |
| 0 | 0 | 0 | 1 |

## Implementando las funciones en el computador cuántico

Después de obtener la matriz correspondiente a cada función necesitamos encontrar una Uf que es una caja negra, que nos permite modificar nuestros qbits de entrada para que estos correspondan a los qbits de salida que tenemos en las matrices halladas, esto lo haremos mediante el uso de compuertas básicas que nos permiten manejar la información dependiendo de la entrada que poseamos. Después de hallar la información correspondiente a Uf si simulamos el sistema aplicando Uf vamos a obtener la misma matriz hallada en el punto anterior, esto se puede comprobar ejecutando el archivo ”ConstrucciónMatrizD.py”.

SI simulamos la función dada todos los posibles qbits obtendremos todas las salidas según la matriz que determina el sistema, este proceso lo podemos simular al ejecutar el archivo “Implementación FuncionesD.py” a su vez este archivo nos mostrara el circuito correspondiente a la función y el resultado de las 4 pruebas.

## Implementando el algoritmo de Deutsch en un computador cuántico

Después de hallar el circuito relacionado a las funciones requeridas vemos a realizar la simulación de las funciones evaluadas en el algoritmo de Deutsch, comprobando así su correcto funcionamiento. Para esto vamos a mostrar a continuación los circuitos de cada función en orden como se estableció en el inciso 2.1 y los resultados arrojados, se recomienda ejecutar el archivo “Algoritmo Deutsch.py” para ver el proceso de construcción de los circuitos, y a su vez ver la información mostrada a continuación de manera más comoda.

***Función 1:***

Circuito:

Imagen que contiene Forma

Descripción generada automáticamente

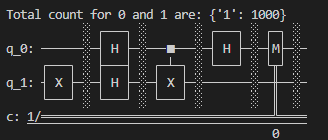
Resultado:

Gráfico, Gráfico de rectángulos

Descripción generada automáticamente

***Función 2***

Circuito:



Resultado:

Gráfico, Gráfico de rectángulos

Descripción generada automáticamente

***Función 3:***

Circuito:

Imagen que contiene Diagrama

Descripción generada automáticamente

Resultado:

Gráfico, Forma, Gráfico de rectángulos, Cuadrado

Descripción generada automáticamente

***Función 4:***

Circuito:

Imagen que contiene Diagrama

Descripción generada automáticamente

Resultado:

Gráfico, Gráfico de rectángulos

Descripción generada automáticamente

Tras observar los resultados obtenidos después de la simulación de los circuitos en el algoritmo de Deutsch podemos darnos cuenta de que las funciones 1 y 4 son constantes debido a lo establecido en la teoría en el inciso 2.1 en la cual establecemos que cuando se simula el sistema en un computador cuántico y se interpreta el qbit de arriba como 0 significa que la función simulada es constante. En consecuencia y basados en los resultados podemos darnos cuenta de que las funciones 2 y 3 son funciones balanceadas debido a que el qbit superior se evalúa en 1 indicando por la teoría que la función es balanceada.

# Algoritmo de Deutsch-Jozsa

El algoritmo de Deutsch-Jozsa es un algoritmo cuántico el cual toma como base de su funcionamiento el algoritmo de Deutsch y busca de igual manera dar de manera eficaz la solución a un problema que de manera clásica seria solucionado con mayor lentitud comparado a la solución que se le puede dar al problema en un computador cuántico.

Para la realización de este experimento y a diferencia del algoritmo de Deutsch tenemos un dominio más amplio el cual se representa mediante y representa todos los numero binarios con longitud n, de manera semejante el rango es el mismo que en el algoritmo de Deutsch y es de , por lo tanto, el algoritmo de Deutsch toma como parámetro las funciones de las funciones que cumplen tal condición son en total y de manera consecuente las funciones que son constantes son 2, la cantidad de balanceadas son en total .

Debido a la magnitud que abarca el algoritmo de Deutsch-Jozsa no podemos abarcar en completitud todos sus casos por lo tanto vamos a trabajar con un n=4 y tomaremos 4 funciones en las cuales 3 serán balanceadas y 1 será constante, y se simulara de la misma manera que se simulo el algoritmo de Deutsch mediante el uso de Python y la librería qiskit.

## Problema

El problema que busca resolver el algoritmo de Deutsch-Jozsa es dada una función que toma un valor binario de n de longitud devuelve otro valor binario y averigua si esta es una función balanceada (Devuelve 1 para la mitad de los valores binarios) o es una función constante(Devuelve 0 para todos los valores binarios o devuelve 1 para todos los valores binarios), en esta se van a considerar solo las funciones balanceadas o constantes para su correcto funcionamiento, en las otras funciones al algoritmo falla.

Para la solución de este problema, el algoritmo se basa en el uso de un circuito cuántico, que dado n + 1 qbits los n qbits superiores teniendo un 0 y el qbit inferior teniendo un 1 cuando se obtiene el estado final del sistema evaluado en los n qbits de la parte superior, este resultado nos va a arrojar un valor binario de n de longitud el cual podemos interpretar como, cuando los n qbits superiores arrojan todos 0 significa que la función es constante, cuando existe uno de los n qbits superiores distinto a 1 se interpreta que la función es balanceada, teniendo en cuenta que las funciones dadas son estrictamente balanceadas o constantes.

Como establecimos anteriormente vamos a usar 4 funciones las cuales cumplen la condición de tener un dominio y rango de la forma y son balanceadas o constantes, las funciones con las que trabajaremos son las siguientes:

***Función 1:***

Dibujo correspondiente a la función:

0000

0001

0010

0011

0100

0101

0110

0111 0

1000 1

1001

1010

1011

1100

1101

1110

1111

Matriz correspondiente a la función:

Patrón de fondo

Descripción generada automáticamente

***Función 2:***

Dibujo correspondiente a la función:

0000

0001

0010

0011

0100

0101

0110

0111 0

1000 1

1001

1010

1011

1100

1101

1110

1111

Matriz correspondiente a la función:

Patrón de fondo

Descripción generada automáticamente

***Función 3:***

Dibujo correspondiente a la función:

0000

0001

0010

0011

0100

0101

0110

0111 0

1000 1

1001

1010

1011

1100

1101

1110

1111

Matriz correspondiente a la función:

Patrón de fondo

Descripción generada automáticamente

***Función 4:***

Dibujo correspondiente a la función:

0000

0001

0010

0011

0100

0101

0110

0111 0

1000 1

1001

1010

1011

1100

1101

1110

1111

Matriz correspondiente a la función:

Patrón de fondo

Descripción generada automáticamente

## Implementando las funciones en el computador cuántico

Después de obtener la matriz correspondiente a cada función necesitamos encontrar una Uf que es una caja negra, que nos permite modificar nuestros qbits de entrada para que estos correspondan a los qbits de salida que tenemos en las matrices halladas, esto lo haremos mediante el uso de compuertas básicas que nos permiten manejar la información dependiendo de la entrada que poseamos. Después de hallar la información correspondiente a Uf si simulamos el sistema aplicando Uf vamos a obtener la misma matriz hallada en el punto anterior, esto se puede comprobar ejecutando el archivo ”ConstrucciónMatrizD-J.py”.

SI simulamos la función dada todos los posibles qbits obtendremos todas las salidas según la matriz que determina el sistema, este proceso lo podemos simular al ejecutar el archivo “Implementación FuncionesD-J.py” a su vez este archivo nos mostrara el circuito correspondiente a la función y el resultado de las 4 pruebas.

## Implementando el algoritmo de Deutsch-Josza en un computador cuántico

Después de hallar el circuito relacionado a las funciones requeridas vemos a realizar la simulación de las funciones evaluadas en el algoritmo de Deutsch-Jozsa, comprobando así su correcto funcionamiento. Para esto vamos a mostrar a continuación los circuitos de cada función en orden como se estableció en el inciso 3.1 y los resultados arrojados, se recomienda ejecutar el archivo “Algoritmo Deutsch-Jozsa.py” para ver el proceso de construcción de los circuitos, y a su vez ver la información mostrada a continuación de manera más comoda.

***Función 1:***

Circuito:

Diagrama

Descripción generada automáticamente

Resultado:

Gráfico, Gráfico de rectángulos

Descripción generada automáticamente

***Función 2***

Circuito:

Diagrama

Descripción generada automáticamente

Resultado:

Gráfico, Gráfico de rectángulos

Descripción generada automáticamente

***Función 3:***

Circuito:

Imagen que contiene circuito

Descripción generada automáticamente

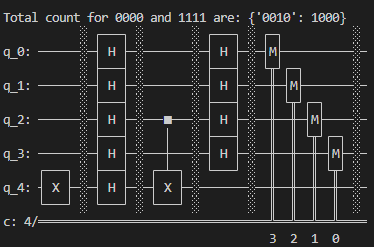
Resultado:

Gráfico, Gráfico de rectángulos

Descripción generada automáticamente

***Función 4:***

Circuito:



Resultado:

Gráfico de rectángulos

Descripción generada automáticamente

Tras observar los resultados obtenidos después de la simulación de los circuitos en el algoritmo de Deutsch-Jozsa podemos darnos cuenta de que la función 1 es constante debido a lo establecido en la teoría en el inciso 3.1 en la cual establecemos que cuando se simula el sistema en un computador cuántico y se interpreta los qbits de arriba como 0 significa que la función simulada es constante. En consecuencia y basados en los resultados podemos darnos cuenta de que las funciones 2, 3 y 4 son funciones balanceadas debido a que algunos de los qbits superiores se evalúa en 1 indicando por la teoría que la función es balanceada.

# Conclusiones

A partir del proyecto realizado nos pudimos dar cuenta que la computación cuántica es una alternativa a los computadores clásicos y esta comenzado a tomar relevancia en los últimos tiempos, en el proyecto realizado dimos solución a un problema que tenía una complejidad en el computador clásico de debido a que en el computador clásico la forma mas eficaz de resolver el problema es en la que llamamos a la función la mitad de todas las posibles combinaciones de 1 y 0 , de forma que si en la mitad de veces el resultado de la función es una secuencia de 1 y 0 únicamente, entonces la función es balanceada, de otra forma significa que es constante. Por otra parte, la complejidad de los algoritmos trabajados funcionaba a partir de n teniendo una complejidad de siendo una complejidad lineal a diferencia de la del computador cuántico que era exponencial.