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

**Camilo Andrés Quintero Rodriguez**

**Escuela Colombiana de ingeniería Julio Garavito**

**camilo.quintero-r@mail.escuelaing.edu.co**

**21/11/2024**

*Este reporte se entrega para cumplir con los requisitos parciales del curso CNYT: Computación Cuántica- 2024-2*

# Tabla de contenidos

[**Tabla de contenidos 1**](#_heading=h.gjdgxs)

[**1**](#_heading=h.30j0zll) **Introducción** 2

[**2**](#_heading=h.1fob9te) **Algoritmo de Deutsch 2**

[2.1](#_heading=h.3znysh7) Problema 3

[2.2](#_heading=h.2et92p0) Implementando las funciones en el computador cuántico 5

[2.3](#_heading=h.tyjcwt) Implementando el algoritmo de Deutsch en un computador cuántico 10

[**3**](#_heading=h.3dy6vkm) **Algoritmo de Deutsch-Jozsa** 12

[3.1](#_heading=h.1t3h5sf) Problema 13

[3.2](#_heading=h.4d34og8) Implementando las funciones en el computador cuántico 17

[3.3](#_heading=h.2s8eyo1) Implementando el algoritmo de Deutsch-Josza en un computador cuántico 18

[**4**](#_heading=h.17dp8vu) **Conclusiones** 20

[**5**](#_heading=h.3rdcrjn) **Bibliografía** 22

# Introducción

En el siguiente informe se presentarán varios temas interesantes relacionados a la computación cuántica, los temas que explicaremos son el Algoritmo de Deutsch-Jozsa y el Algoritmo De Deutsch, sin embargo para esto, debemos aclarar que la computación cuántica es un campo multidisciplinario que utiliza mecánicas cuánticas para solucionar problemas complejos con una mayor facilidad a comparación de una computadora clásica. Actualmente la computación cuántica es usada para diferentes procesos en organizaciones, tales como el análisis de grandes cantidades de datos, optimización de cadenas de producción y suministros, simulaciones de sistemas precisos, entre otros.

En este reporte, como ya se dijo anteriormente, se presentará la implementación de los algoritmos de Deutsch y Deutsch-Jozsa con el fin de explicar su funcionamiento en el mundo de la computación cuántica, todo esto, mediante la solución y análisis de algunos problemas y formulación de diversas situaciones que nos ayudarán a comprender el funcionamiento de cada uno de estos algoritmos.

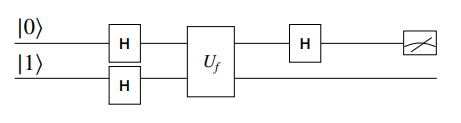
Este reporte se compone por la parte teórica de cada algoritmo donde se explican algunas características de cada uno seguido de las explicaciones correspondientes para cada experimento, posteriormente se explicará el problema a tratar con respecto a cada algoritmo, se explicarán las diferentes implementaciones de cada Deutsch y Deutsch-Jozsa mediante gráficas y resultados obtenidos con ayuda del computador de ibm. Finalmente, se presentarán las conclusiones y resultados finales que alcanzamos durante el desarrollo de los experimentos y la solución de cada problema con respecto a la implementación de los diferentes algoritmos.

# Algoritmo de Deutsch

El Algoritmo de Deutsch es un algoritmo de la computación cuántica que fue diseñado con el fin de resolver un problema de esta ciencia, determinar si una función f(x) es constante o balanceada, el problema a tratar se va a explicar más adelante. Sin embargo, vale la pena aclarar que para dar solución a dicho problema, se implementará el Algoritmo de Deutsch con ayuda del computador de IBM, para esto, después de ya tener el algoritmo desarrollado, se realizarán varias pruebas con funciones constante y balanceadas para probar la eficacia y funcionamiento del algoritmo.

En esta sección expondrá el problema principal a tratar con el Algoritmo de Deutch y así mismo, se explicará la solución a dicho problema, es decir, se presentará la implementación del algoritmo en el computador cuántico además de la implementación de las diferentes funciones f(x) que como ya se dijo anteriormente, pueden ser identificadas como balanceadas o constantes.

La estructura del algoritmo de Deutsch se muestra a continuación:



Podemos ver que el primer canal debe entrar un qubit |0> y en el segundo canal un qubit |1>, posteriormente en cada canal actúa una matriz de Hadamard, luego actúa la matriz que representa la función y finalmente sobre el primer canal actúa una matriz de Hadamard. Si realizamos la medida sobre el primer qubit, podemos ver que el resultado puede ser estrictamente 0 o 1, en este caso el algoritmo nos indica que si la salida o bien, el qubit de estado superior está en estado |0>, implica que la función es constante, de lo contrario, si es |1>, la función es balanceada.

## Problema

El algoritmo de Deutsch nos ayuda a determinar si una función f(x) es de tipo constante o de tipo balanceada, sea cual sea el caso, la función f(x) recibe como parámetro un valor, para este caso, el parámetro que recibe es el valor de un qubit, como ya sabemos puede ser 0 o 1. Es importante a aclarar que implica que una función sea balanceada o constante:

Decimos que una función es balanceada si f(1) es diferente de f(0), y a su vez, decimos que una función es constante si f(1) = f(0).

A continuación se presentarán las funciones que será tratadas en el desarrollo de este capítulo:

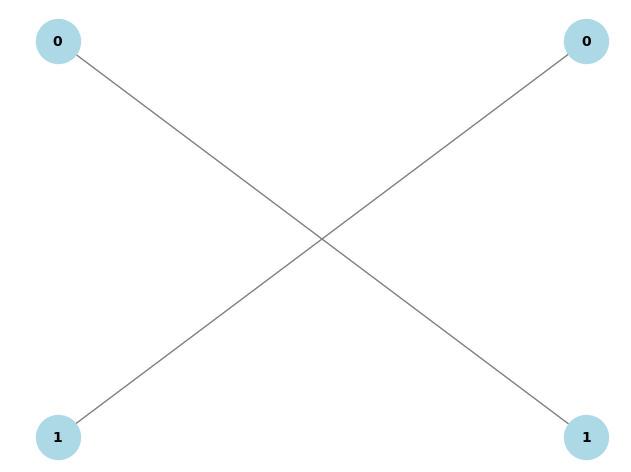
1. f(x) = x

Donde f(0) = 0 y f(1) = 1



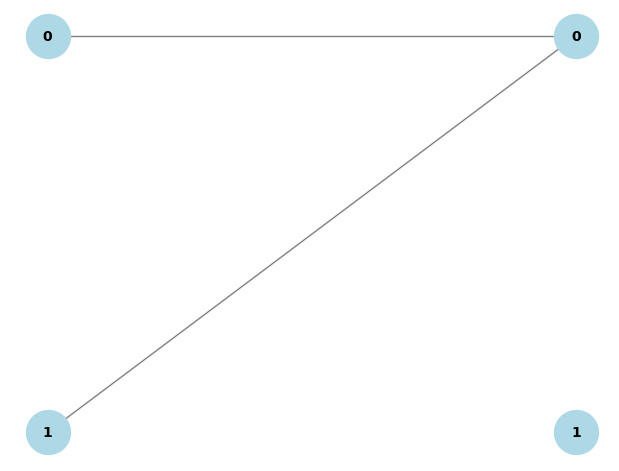
1. f(x) = x XSOR 1

Donde f(0) = 1 y f(1) = 0



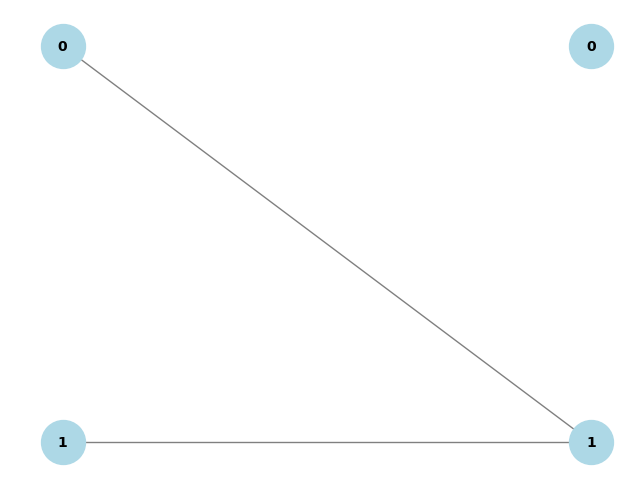
1. f(x) = 0

Donde f(0) = 0 y f(1) = 0



1. f(x) = 1

Donde f(0) = 1 y f(1) = 1



Como se puede observar, tanto la primera como la segunda función se pueden identificar como funciones balanceadas, pues cumplen que f(1) es diferente de f(0). Ahora, la tercera y cuarta función son funciones constantes, ya que cumplen que f(1) = f(0).

## Implementando las funciones en el computador cuántico

Las funciones fueron graficadas con ayuda de una librería que ofrece python, esta librería llamada networkx está diseñada para el modelado de grafos, en este caso, nos ayuda a ver el comportamiento de cada una de las diferentes funciones, así mismo, mediante este diseño, se logró desarrollar la matriz que representa cada una de las diferentes funciones, a continuación se explica cada una de las matrices y a su vez, se presentará la matriz controlada correspondiente para cada función:

1. f(x) = x

Donde f(0) = 0 y f(1) = 1

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

Matriz :

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

1. f(x) = x XSOR 1

Donde f(0) = 1 y f(1) = 0

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

Matriz :

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

1. f(x) = 0

Donde f(0) = 0 y f(1) = 0

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

Matriz :

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

1. f(x) = 1

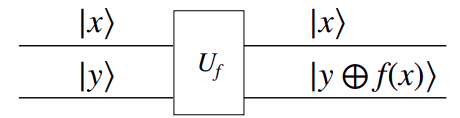
Donde f(0) = 1 y f(1) = 1

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

Matriz :

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

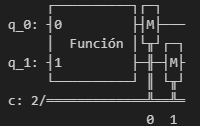
Es importante aclarar que la matriz fue calculada mediante el siguiente circuito:



Ahora bien, con respecto a la implementación de los circuitos correspondientes a cada una de las anteriores funciones, para eso, se tuvo en cuenta la matriz resultante y con ayuda de qiskit se logró representar el circuito y posteriormente realizar las diferentes pruebas para cada función, a continuación se presentan los diferentes circuitos con una entrada :

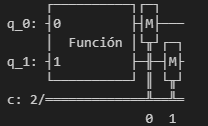
1. f(x) = x

Donde f(0) = 0 y f(1) = 1



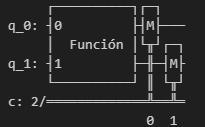
1. f(x) = x XSOR 1

Donde f(0) = 1 y f(1) = 0



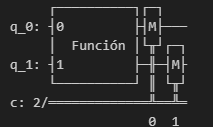
1. f(x) = 0

Donde f(0) = 0 y f(1) = 0



1. f(x) = 1

Donde f(0) = 1 y f(1) = 1



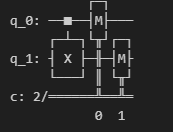
Podemos ver que para cada función el circuito es el mismo, lo unico que varia es que Función es según corresponda, esto hace que cuando se evalúe el circuito en los diferentes estados 00, 01, 10 y 11, los resultados correspondan a la estructura de la matriz .

Ahora bien, si representamos los circuitos mediante compuertas lógicas podemos ver que lo circuitos son diferentes:

1. f(x) = x

Donde f(0) = 0 y f(1) = 1

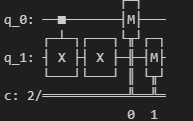
Usando la compuerta Not controlada teniendo a q0 como qubit de control:



1. f(x) = x XSOR 1

Donde f(0) = 1 y f(1) = 0

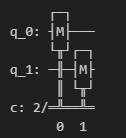
Usando la compuerta Not controlada teniendo a q0 como qubit de control seguida de la compuerta Not en el segundo canal:



1. f(x) = 0

Donde f(0) = 0 y f(1) = 0

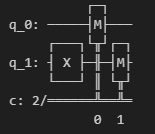
Teniendo como compuerta a la identidad, en este caso simplemente no se coloca compuerta:



1. f(x) = 1

Donde f(0) = 1 y f(1) = 1

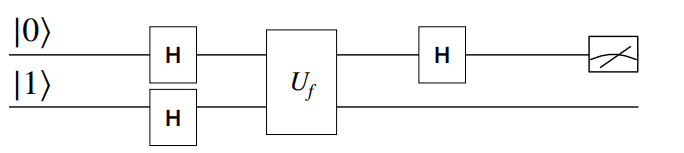
Teniendo como compuerta a Not en el segundo canal:



Si se quiere ver cómo se llegó a la implementación de cada compuerta o diferentes combinaciones , se puede recurrir al archivo .ipynb donde se desarrolla todo el procedimiento correspondiente.

## Implementando el algoritmo de Deutsch en un computador cuántico

Para la implementación del algoritmo se tuvo en cuenta la estructura del mismo, dicha estructura se muestra a continuación:

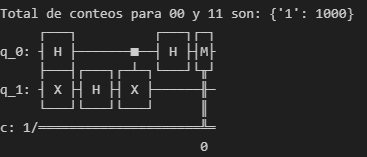


Para esto, con ayuda de qiskit, se implementó el circuito correspondiente, como podemos observar, los qubit de entrada en este caso los llamaremos q0 y q1, son 0 y 1 respectivamente, posteriormente, una matriz de Hadamard actúa sobre 0 y otra sobre 1, luego, podemos analizar que actúa sobre el estado la matriz que representa la función , en este caso, para cada función se tuvo que varias , pues cada función anterior tiene una asociada, posteriormente, sobre el canal 0 actúa una matriz de Hadamard y finalmente, realizamos la medición sobre el primer qubit para determinar el comportamiento de la función y poder concluir su naturaleza, como ya dijimos anteriormente, 0 si es constante y 1 si es balanceada. Este mismo ejercicio se realizó con las compuertas reversibles y las combinaciones de compuertas para definir .

A continuación se presentan los resultados, como veremos, tanto la primera como la segunda función, después de someterlas al algoritmo, el primer qubit resultante es 1, lo que indica que son funciones balanceadas, mientras para la tercera y cuarta función, el primer qubit resulta siendo 0, lo que india que son funciones constantes según la definición.

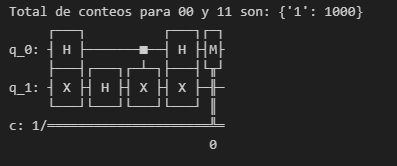
1. f(x) = x

Donde f(0) = 0 y f(1) = 1



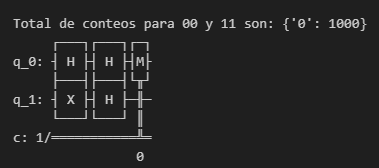
1. f(x) = x XSOR 1

Donde f(0) = 1 y f(1) = 0



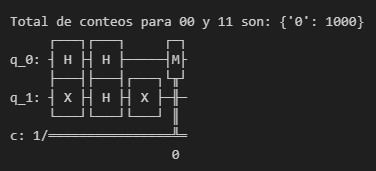
1. f(x) = 0

Donde f(0) = 0 y f(1) = 0



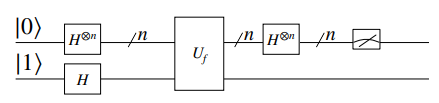
1. f(x) = 1

Donde f(0) = 1 y f(1) = 1



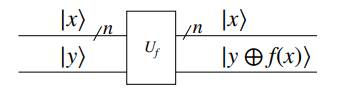
# Algoritmo de Deutsch-Jozsa

El algoritmo de Deutsch-Jozsa es una extensión del algoritmo de Deutsch, pues determina la naturaleza de una función, en este caso se presentará el diseño del algoritmo para dar claridad a su explicación:



Como se puede evidenciar, su estructura es muy similar a la del algoritmo Deutsch, sin embargo tenemos algunas diferencias importantes, empezando por el término , este término indica la cantidad de qubits de entrada para el primer componente, lo que implica que para el caso del primer componente donde su entrada es o bien, y así con cualquier . Para el segundo comente, es una única entrada, para este caso . Posteriormente podemos ver que la matriz de Hadamard actúa sobre cada componente, luego la matriz que representa la función y finalmente, actúa sobre los componentes de primer canal una matriz de Hadamard. Es importante aclarar que la matriz de Hadamard que actúa sobre el primer componente, representa el producto tensor veces de la matriz de Hadamard.

Con respecto a la función , su construcción la realizamos igual que en el primer algoritmos, sin embargo en este caso hay que tener en cuenta que para el componente en el primer canal, entra una cadena de qubits, así que el circuito para determinar la matriz de la función resulta como se muestra a continuación:



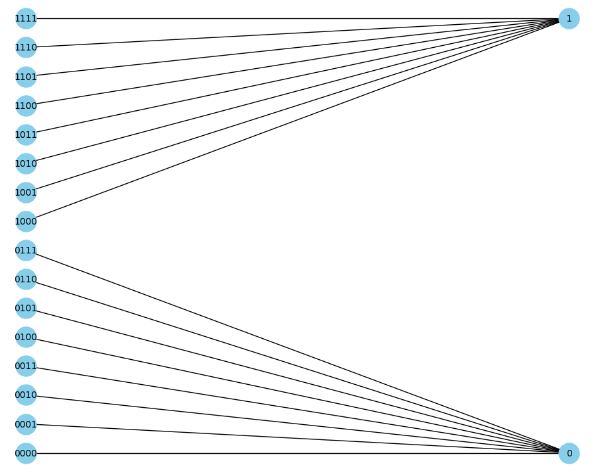
Ahora bien, cómo podemos saber si una función es de tipo constante o balanceada, para esto el algoritmo nos indica que la función es balanceada si exactamente la mitad de las entradas van a 0 y la otra mitad a 1 y es constante si todas las entradas van a 0 o todas van a 1.

## Problema

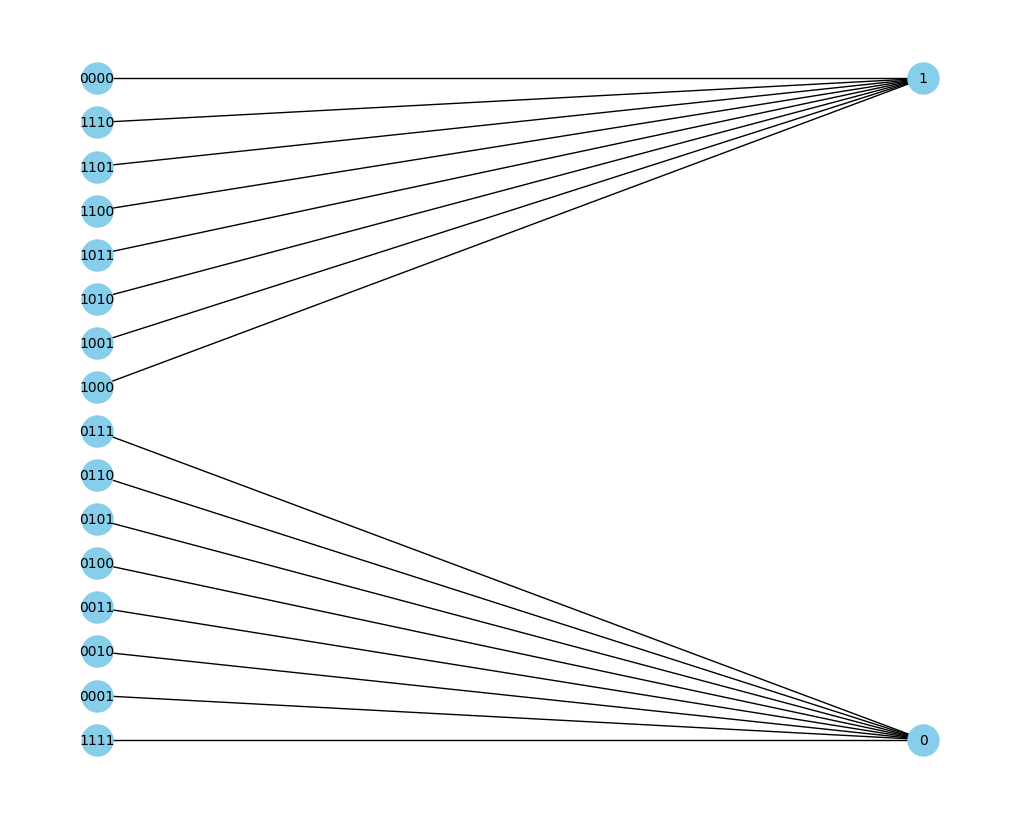
En este caso, al igual que el problema tratado en el primer algoritmo, se busca determinar si la función es balanceada o constante, en este caso sabemos que la función de entrada es estrictamente balanceada o constante, nunca la función será de otro tipo

El algoritmos será evaluado con 4 funciones, para esto, se plantearon cuatro funciones con n = 4, tres balanceadas y una constante, para realizar la representación gráfica de estas funciones se usó la misma librería descrita anteriormente (networkx), a continuación se se presentan los gráficos correspondientes:

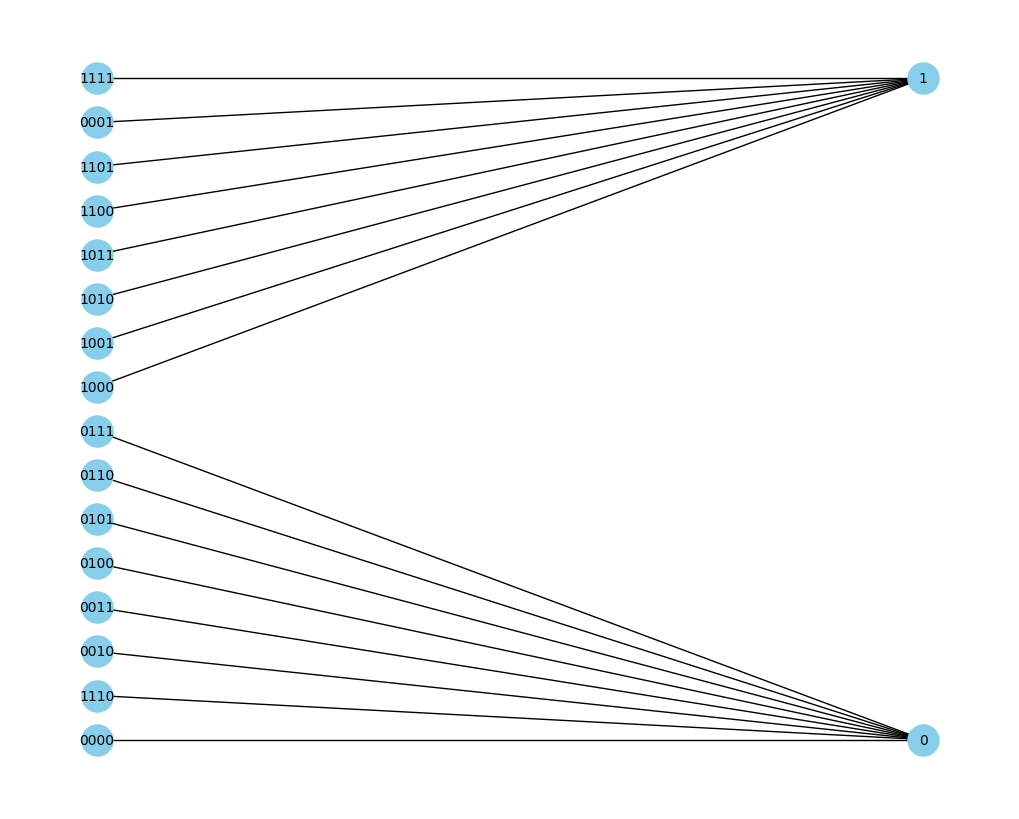
1. Función 1 (balanceada)



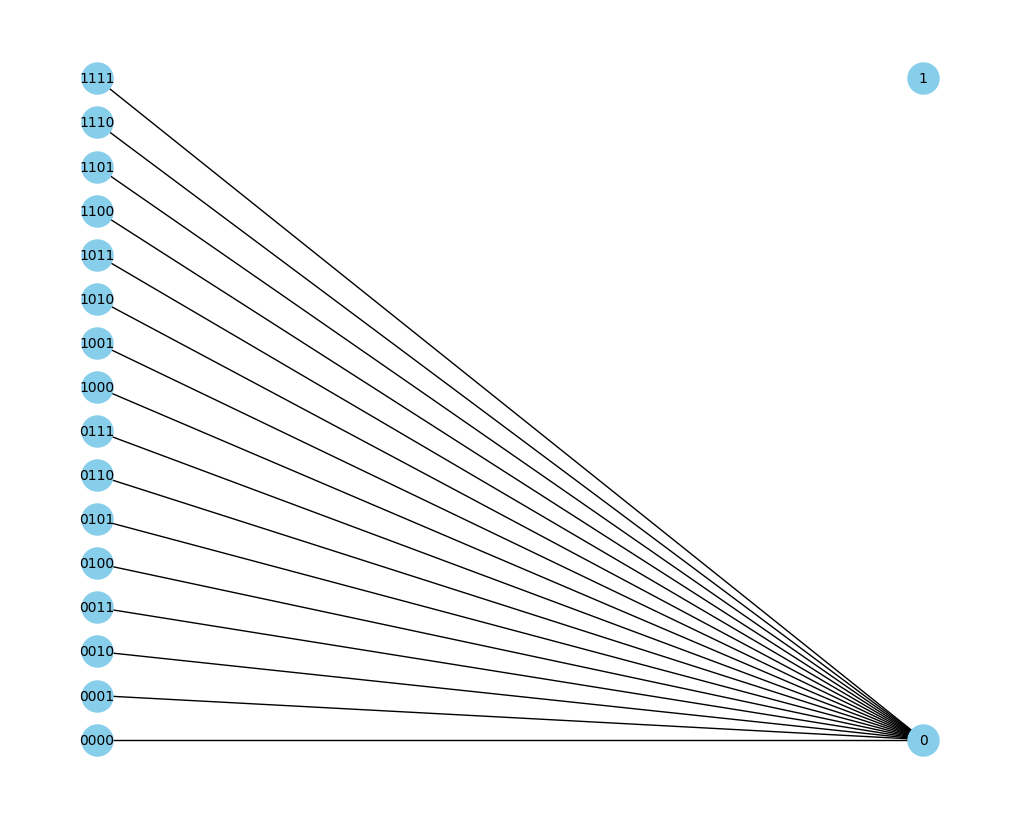
1. Función 2 (balanceada)



1. Función 3 (balanceada)



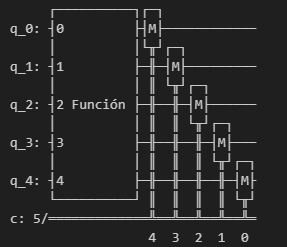
1. Función 4 (constante)



## Implementando las funciones en el computador cuántico

Para la implementación de las funciones anteriores, se usó la función , para esto se generaron las matrices asociadas a la función mediante un pequeño programa que desarrolla el circuito para el cálculo de dicha matriz descrito en la teoría. Posteriormente, para cada función, tanto constantes como balanceadas, representamos el circuito correspondiente, así mismo, se evaluó cada circuito con las diferentes entradas posibles, tras el desarrollo de estas pruebas, se pudo concluir que los circuitos son correctos.

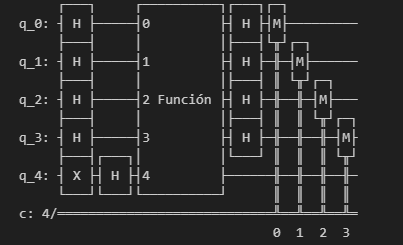
La estructura general de los circuitos para las diferentes funciones es la siguiente:



Es importante aclarar que la única diferencia entre circuitos correspondientes para cada una de las funciones es la Función, pues esta función corresponde la asociada a cada función.

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

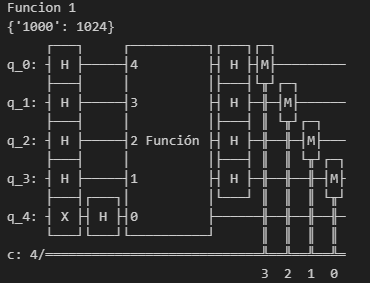
Para la implementación del algoritmo de Deutsch-Josza se tuvo en cuenta la estructura descrita en la teoría del algoritmo (comienzo del capítulo), de esta forma, con ayuda de qiskit, se implementó el circuito que representa el algoritmo A continuación se muestra la estructura general del circuito:



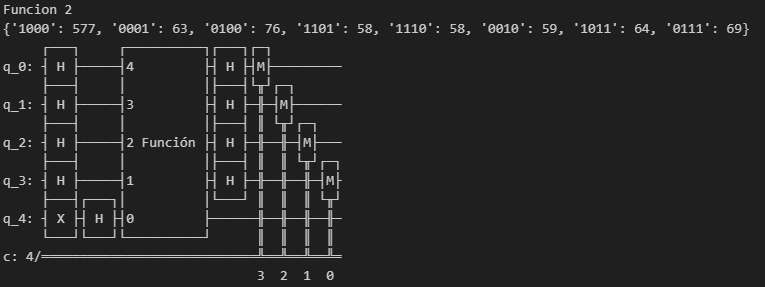
En este caso, al igual que para la representación de las funciones, la compuerta Función presente en el circuito varía dependiendo la función a evaluar, pues corresponde a la matriz que representa la función . Como se puede aparecer, los tres primero qubits tiene estado |0> mientras el ultimo qubit tiene estado |1>, tal cual lo indica el algoritmo de Deutsch-Josza.

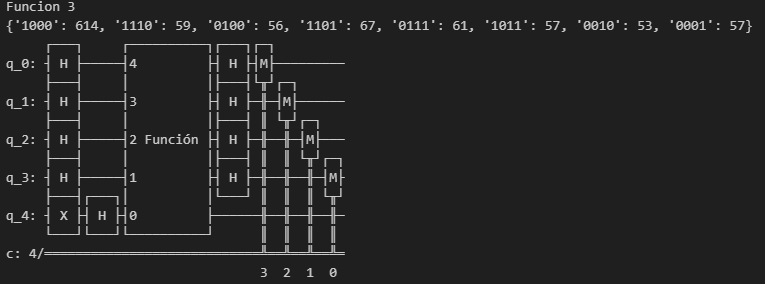
Para cada función, se realizaron las correspondientes pruebas que nos ayudan a verificar el funcionamiento del circuito y el comportamiento de cada una de las cuatro funciones, como se dijo anteriormente, tres balanceadas y una constante.

A continuación se muestran los resultados:

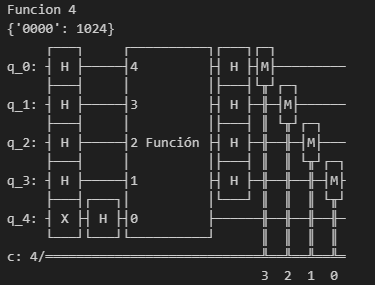


Podemos ver que el resultado es diferente a 0000, por lo tanto es otro tipo de función que no podemos identificar como constante, en este caso esta función es balanceada. Esto mismo sucede con las siguientes funciones:





En el siguiente caso podemos ver que el resultado es 0000, lo que nos indica que la función es constante, esto nos muestra el correcto funcionamiento del algoritmo.



# Conclusiones

Para definir la conclusiones de este informe, daremos respuesta a las pregunta propuesta:

* ¿Qué presentó?

Se presentó la implementación de funciones balanceadas y constantes, posteriormente se desarrolló cada algoritmo, en el primer caso, la implementación del Algoritmo de Deutsch nos ayudó a identificar la naturaleza de cada función definida, de esta forma logramos determinar que función era constante y balanceada. Esta misma metodología fue aplicada en la implementación del algoritmo de Deutsch-Jozsa, el cual nos ayudó a identificar el comportamiento de 4 funciones previamente implementadas, 3 balanceadas y una constante. Tanto para el primer algoritmo, como para el segundo, las pruebas fueron exitosas, se logró determinar de forma correcta el comportamiento de cada función, esto se debe la correcta implementación de los algoritmos y la formulación de la función relacionada a cada una de las funciones analizadas.

* ¿Qué entendió?

Pude comprender y entender a profundidad las diferentes implementaciones de los algoritmos y el uso de la computadora de IBM para la implementación de las funciones y como ya se dijo anteriormente, la implementación de los algoritmos estudiados. Esto me ayudó a entender el comportamiento de las funciones implementadas, en este caso, funciones balanceadas y funciones constantes, así mismo, comprendí cómo implementar compuertas lógicas para representar los circuitos relacionados a cada función y a la vez, como si usamos la matriz completa, en este caso , o bien, combinaciones de compuertas lógicas para simular la acción de , obtenemos los mismos resultados.

* ¿Qué aprendió?

Aprendí a implementar los dos algoritmos estudiados, así mismo aprendí a simular los posibles resultados de los algoritmos con ayuda del computador de IBM. Además de esto, como ya dije en la conclusión anterior, aprendí a representar funciones complejas, en este caso mediante combinaciones de compuertas lógicas más simples.

* ¿Qué proyección le ve a lo aprendido?

Aprender a implementar los algoritmos de Deutsch y Deutsch-Jozsa me puede ayuda a obtener mejores bases con respecto a temas relacionados a la computación básica, un estudio interesante que aun con limitaciones tecnológicas se encuentra en un amplio desarrollo y crecimiento. Esto a su vez, nos ofrece una amplia gama de oportunidades en temas de estudio como investigaciones de optimización y diseño de algoritmos cuánticos un poco más complejos a los que estudiamos en este informe.

* ¿Qué trabajos futuros propone?

Sería interesante tener laboratorio prácticos con más simulaciones a partir del computador IBM con herramientas diferentes que nos pueda ofrecer este simulador, herramientas que no fueron implementadas en este trabajo. Así mismo, un trabajo futuro interesante puede ser el talleres relacionados al desarrollo, optimización y extensión de estos algoritmos, pues sería de provecho analizar implementaciones mejoradas y a su vez diseñar nuevas soluciones a problemas no antes vistos.

# Bibliografía

<https://aws.amazon.com/es/what-is/quantum-computing/>

<https://www.docirs.cl/algoritmo_de_deutsch.asp>

<https://es.wikipedia.org/wiki/Algoritmo_de_Deutsch-Jozsa>