Implementación de algoritmo de deutsch y deutsch-jozsa

Mayerlly Suárez Correa
Escuela Colombiana de Ingeniería Julio Gravito
mayerlly.suarez-c@mail.escuelaing.edu.co

2024-1

Este reporte se entrega para cumplir con los requisitos parciales del curso CNYT: Computación Cuántica- 2020-1

Tabla de contenidos

1 Introducción

La computación cuántica representa un campo emergente y prometedor en el ámbito de la informática, que se basa en los principios de la mecánica cuántica para realizar operaciones en un nivel fundamentalmente diferente al de los ordenadores clásicos. En lugar de utilizar bits clásicos que pueden representar un estado de 0 o 1, los ordenadores cuánticos emplean qubits, que pueden estar en una superposición de estados, permitiendo así un procesamiento de información mucho más potente en ciertas aplicaciones. IBM es una de las empresas líderes en este campo y ha desarrollado plataformas como IBM Quantum, que permite a los usuarios experimentar con algoritmos cuánticos en la nube utilizando sus computadoras cuánticas.

Este informe se enfoca en la implementación y verificación de varios algoritmos cuánticos utilizando el computador cuántico de IBM. En primer lugar, se implementarán y probarán las cuatro funciones posibles de una sola qubit a través de circuitos cuánticos, incluyendo su dibujo, la matriz correspondiente, el circuito y los resultados obtenidos de las pruebas. Luego, se verificará el algoritmo de Deutsch para determinar qué funciones son balanceadas o constantes. A continuación, se realizará la implementación de cuatro funciones con cuatro qubits para probar el algoritmo Deustch-Jozsa, con la misma estructura de análisis que en el caso anterior. Finalmente, se explicarán los resultados obtenidos en cada sección, utilizando tanto texto como ecuaciones para justificar y respaldar los hallazgos.

El informe constará de cuatro secciones principales:
1. Implementación y verificación de las funciones posibles de {0,1} a {0,1}.
2. Verificación del algoritmo de Deutsch.
3. Implementación y pruebas del algoritmo Deutsch-Jozsa con funciones de cuatro qubits.
4. Explicación de los resultados obtenidos en cada sección, destacando las observaciones clave y las implicaciones de los resultados en el contexto de la computación cuántica. Cada sección incluirá una introducción contextual, seguida de los detalles de la implementación y los resultados obtenidos, concluyendo con un análisis y discusión de los hallazgos.

2 Algoritmo de Deutsch

En este capítulo, nos enfocaremos en el algoritmo de Deutsch, un algoritmo cuántico que resuelve un problema específico relacionado con funciones booleanas. El objetivo del algoritmo de Deutsch es determinar si una función booleana dada es balanceada o constante, utilizando solo una consulta a la función. Este algoritmo ilustra una de las principales ventajas de la computación cuántica: la capacidad de realizar ciertas tareas de manera más eficiente que los algoritmos clásicos.
El problema que aborda el algoritmo de Deutsch es el siguiente: se nos da una función booleana $f: \{0,1\} \rightarrow \{0,1\}$, es decir, una función que toma como entrada un solo bit y

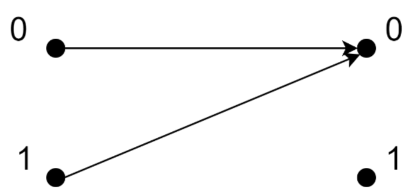
devuelve un solo bit. Esta función puede ser de dos tipos: balanceada o constante. Una función se considera constante si devuelve el mismo valor para todas las posibles entradas, es decir, $f(0) = f(1)$. Por otro lado, una función se considera balanceada si devuelve un valor diferente para cada posible entrada, es decir,
$f(0) \neq f(1)$.

2.1 Problema

El algoritmo de Deutsch nos permite determinar si una función es constante o balanceada con solo una consulta a la función, lo cual es significativamente más eficiente que los enfoques clásicos que requieren múltiples consultas. Este algoritmo demuestra el poder de la computación cuántica al ofrecer una ventaja significativa en la resolución de ciertos problemas booleanos. En este capítulo, implementaremos el algoritmo de Deutsch y lo aplicaremos a cuatro funciones booleanas específicas para demostrar su eficacia y comprender mejor su funcionamiento.
A continuación, se implementarán las cuatro posibles funciones de {0,1} a {0,1} utilizando el computador cuántico de IBM. Cada función toma un valor binario y devuelve otro valor binario. Se debe determinar si cada función es balanceada o constante. El problema que busca resolver el algoritmo de Deutsch es, dada una función, determinar si es balanceada o constante.

Función 1

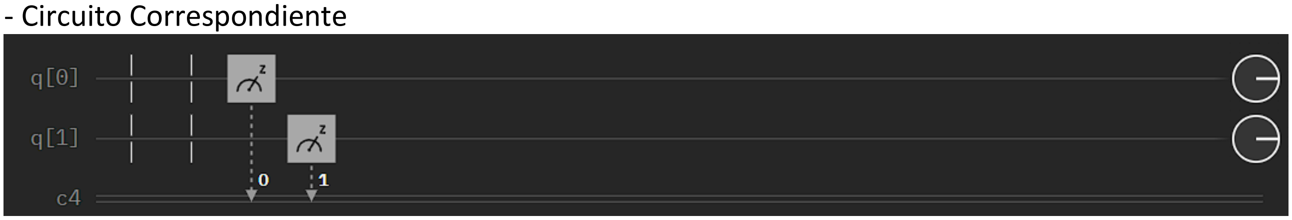
-Dibujo



-Matriz Asociada

```
[1, 0, 0, 0]
[0, 1, 0, 0]
[0, 0, 1, 0]
[0, 0, 0, 1]
```

- Circuito Correspondiente



-Resultados de las 4 pruebas

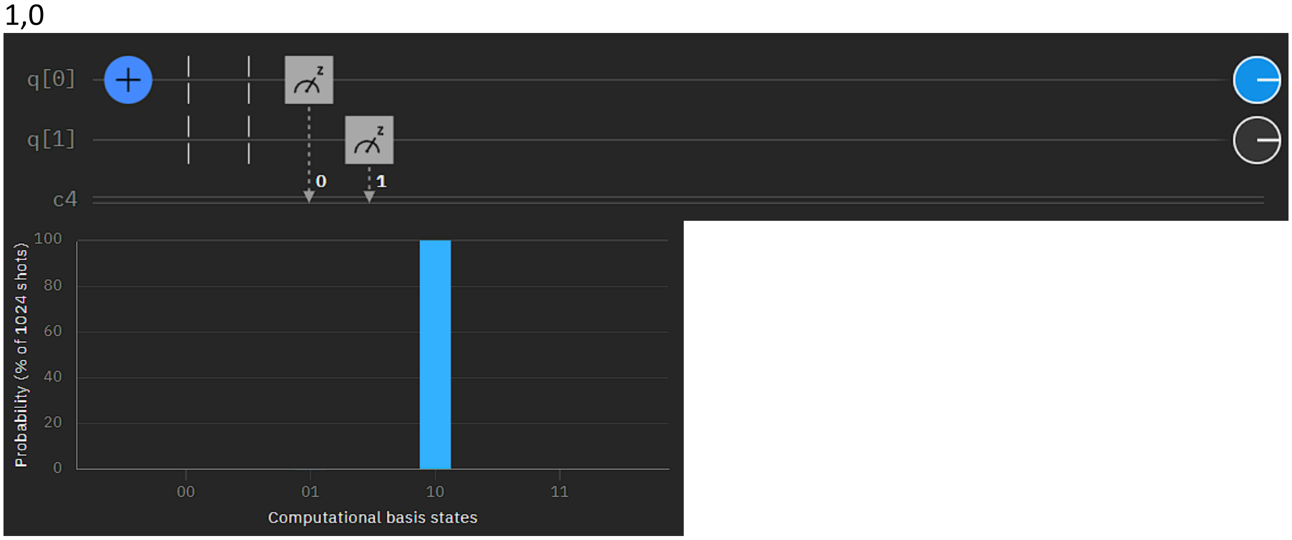
0,0


q[0]
q[1]
c4
0
1
Probability (% of 1024 shots)
100
80
60
40
20
0
00
01
10
11
Computational basis states

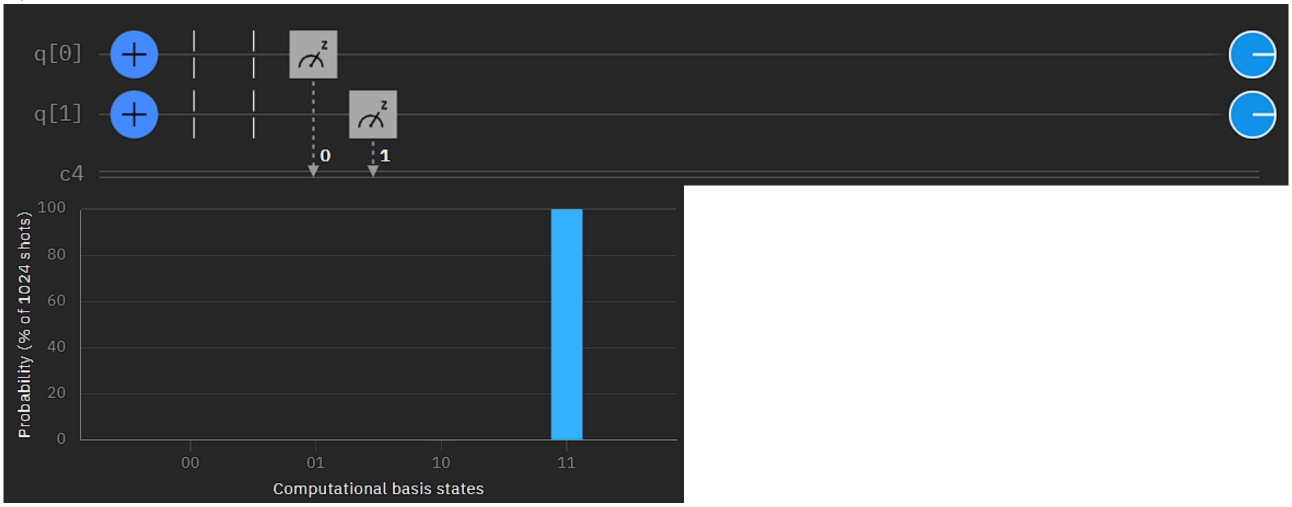
0,1


q[0]
q[1]
c4
0
1
Probability (% of 1024 shots)
100
80
60
40
20
0
00
01
10
11
Computational basis states

1,0

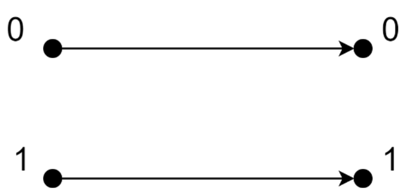

q[0]
q[1]
c4
0
1
Probability (% of 1024 shots)
100
80
60
40
20
0
00
01
10
11
Computational basis states

1,1



Función 2

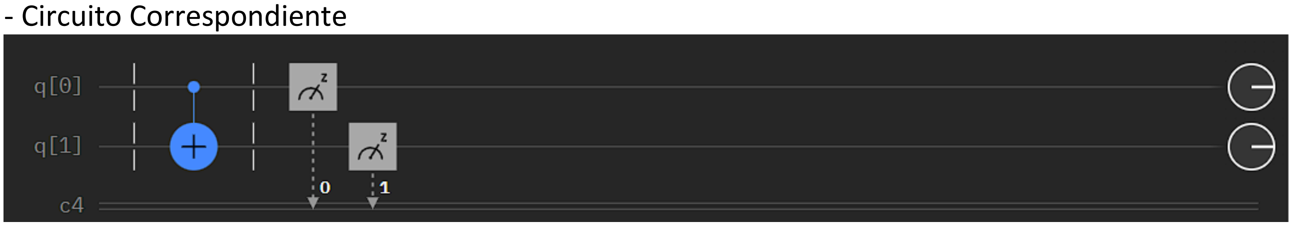
-Dibujo



-Matriz Asociada

```
[1, 0, 0, 0]
[0, 1, 0, 0]
[0, 0, 0, 1]
[0, 0, 1, 0]
```

- Circuito Correspondiente



-Resultados de las 4 pruebas

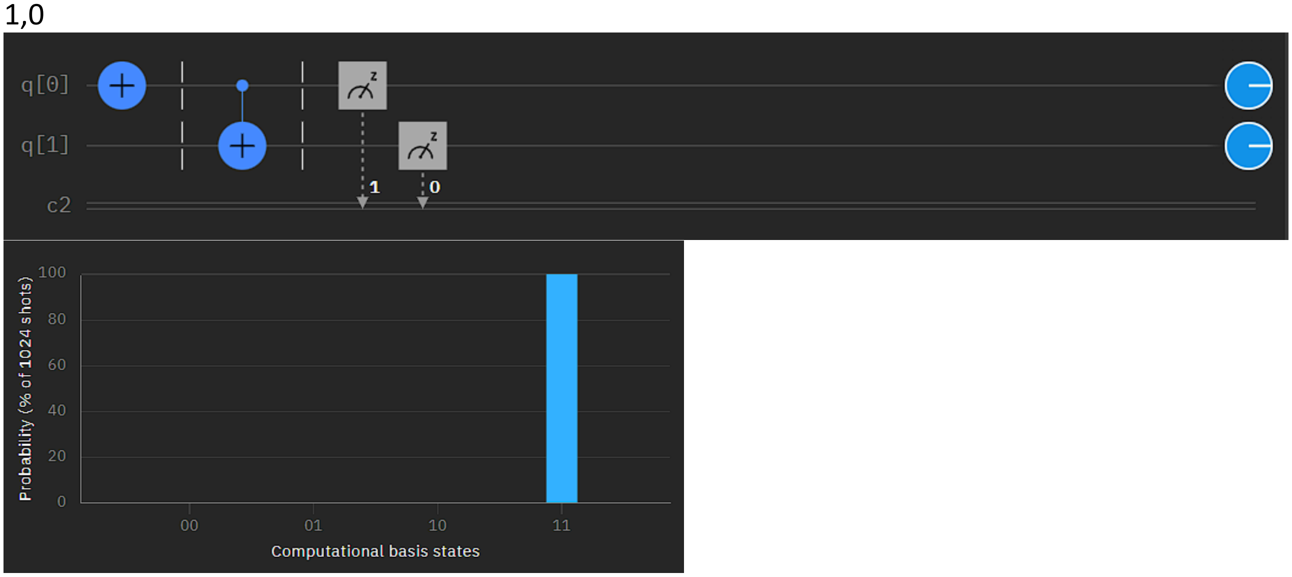
0,0


q[0]
q[1]
c2
1
0
Probability (% of 1024 shots)
100
80
60
40
20
0
00
01
10
11
Computational basis states

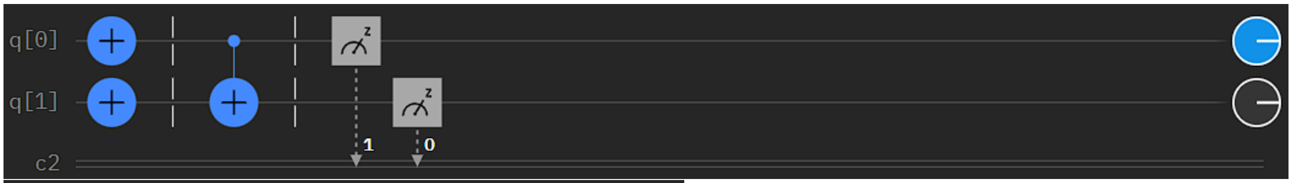
0,1

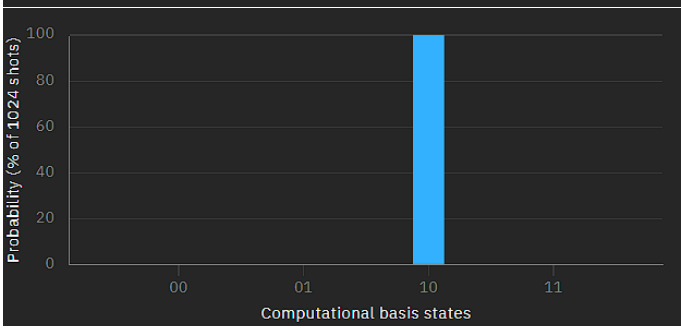

q[0]
q[1]
c2
1
0
Probability (% of 1024 shots)
100
80
60
40
20
0
00
01
10
11
Computational basis states

1,0


q[0]
q[1]
c2
1
0
Probability (% of 1024 shots)
100
80
60
40
20
0
00
01
10
11
Computational basis states

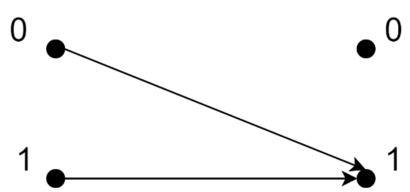
1,1





Función 3

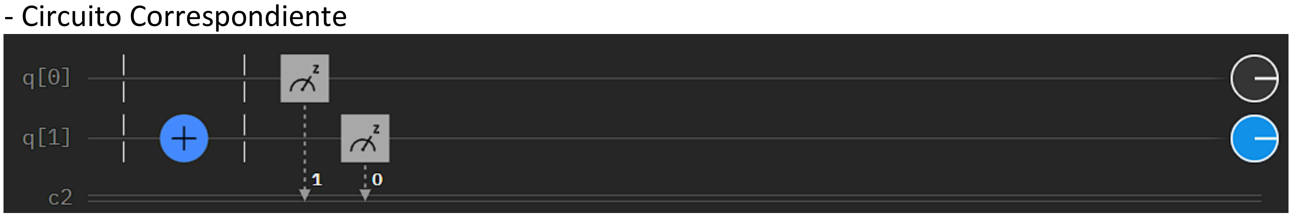
-Dibujo



-Matriz Asociada

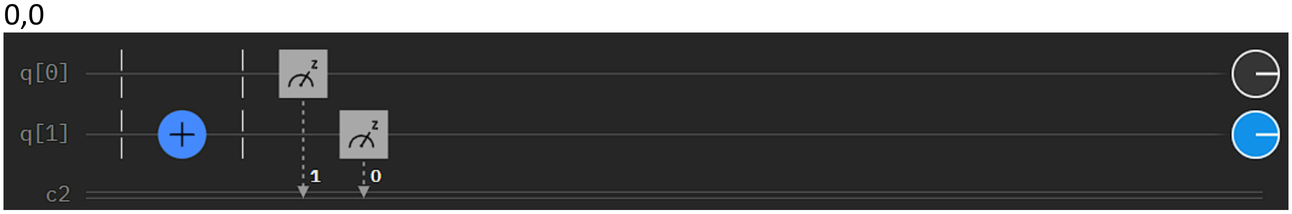
```
[0, 1, 0, 0]
[1, 0, 0, 0]
[0, 0, 0, 1]
[0, 0, 1, 0]
```

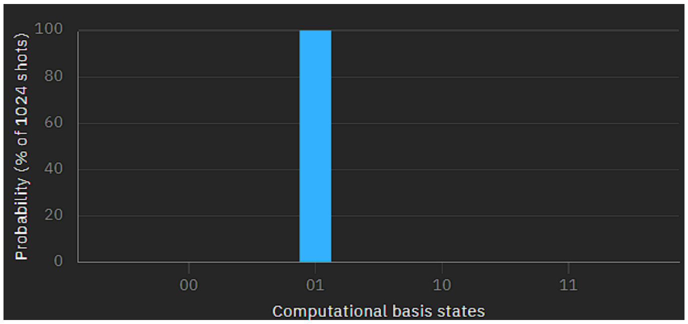
- Circuito Correspondiente



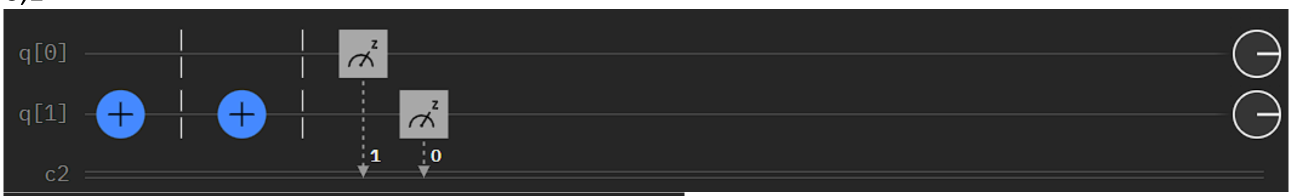
-Resultados de las 4 pruebas

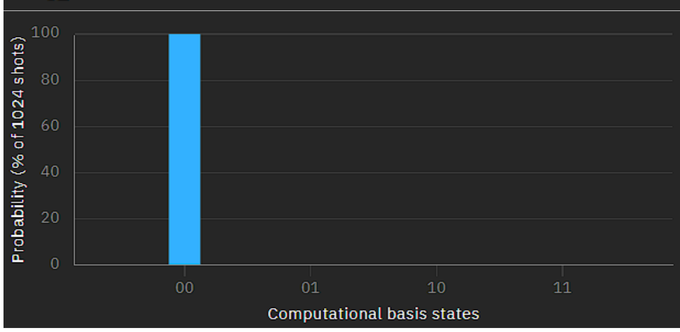
0,0



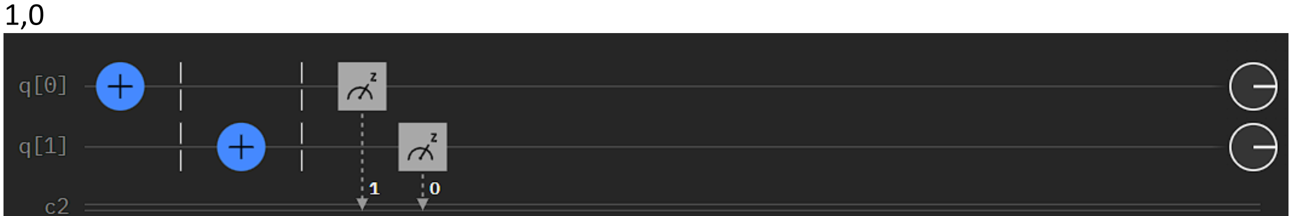

Probability (% of 1024 shots)
100
80
60
40
20
0
00
01
10
11
Computational basis states

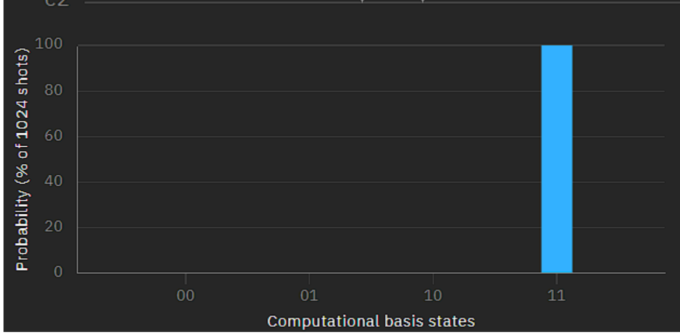
0,1


q[0]
q[1]
c2
z
z
1
0

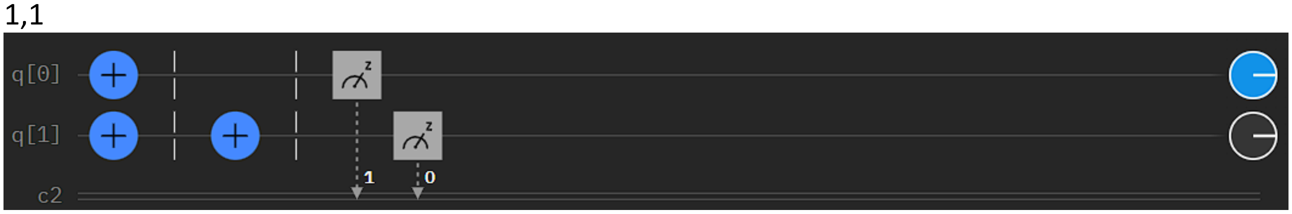

Probability (% of 1024 shots)
100
80
60
40
20
0
00
01
10
11
Computational basis states

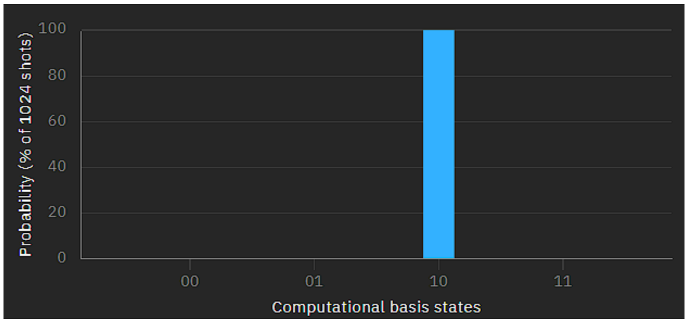
1,0


q[0]
q[1]
c2
z
z
1
0


Probability (% of 1024 shots)
100
80
60
40
20
0
00
01
10
11
Computational basis states

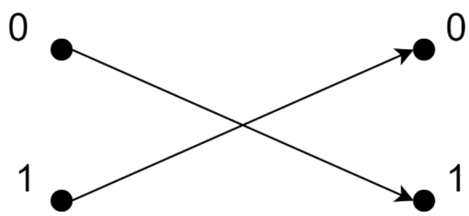
1,1


q[0]
q[1]
c2
z
z
1
0



Función 4

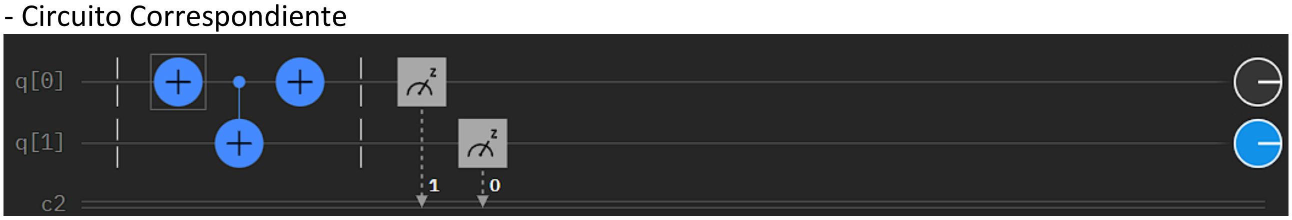
-Dibujo



-Matriz Asociada

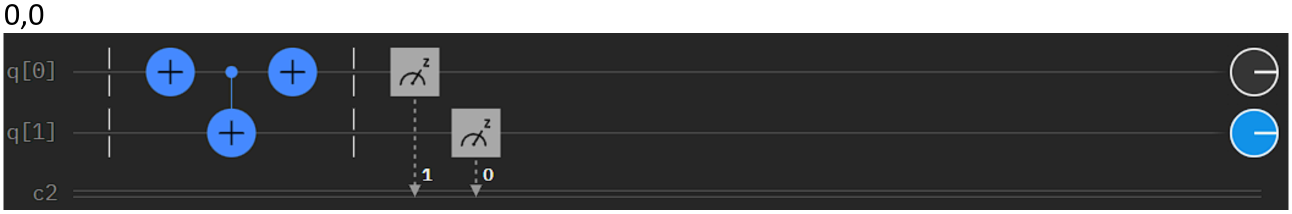
```
[0, 1, 0, 0]
[1, 0, 0, 0]
[0, 0, 0, 1]
[0, 0, 1, 0]
```

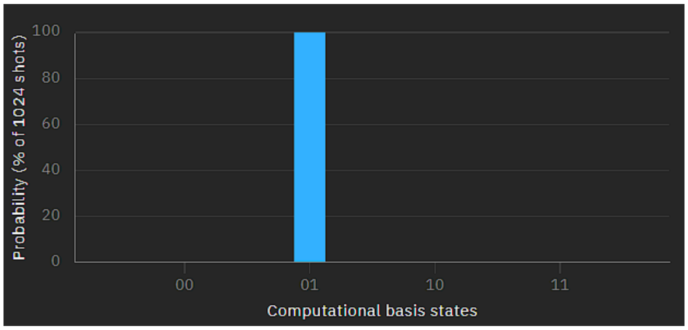
- Circuito Correspondiente



-Resultados de las 4 pruebas

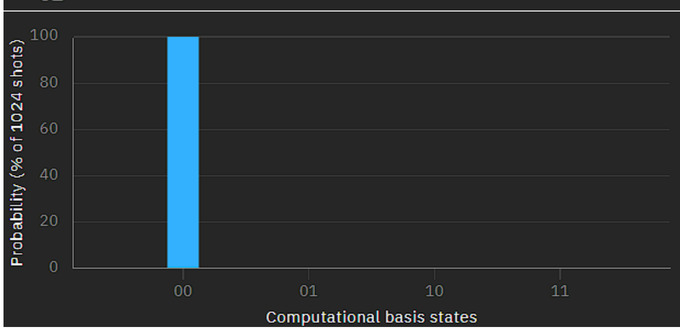
0,0



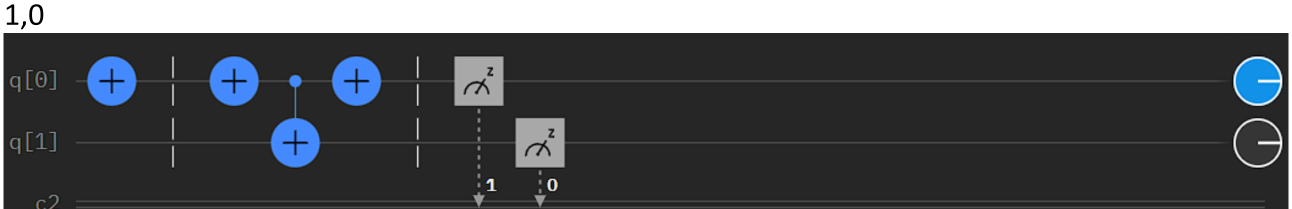

Probability (% of 1024 shots)
100
80
60
40
20
0
00
01
10
11
Computational basis states

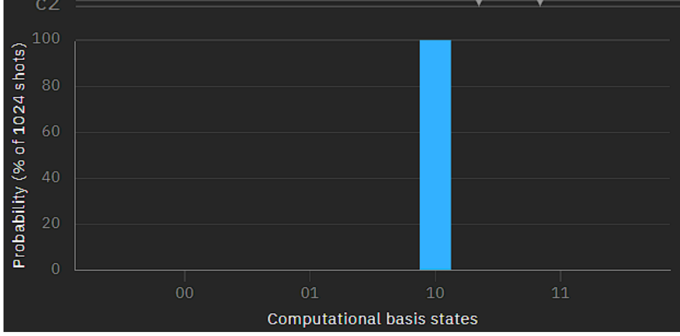
0,1


q[0]
q[1]
c2
1
0


Probability (% of 1024 shots)
100
80
60
40
20
0
00
01
10
11
Computational basis states

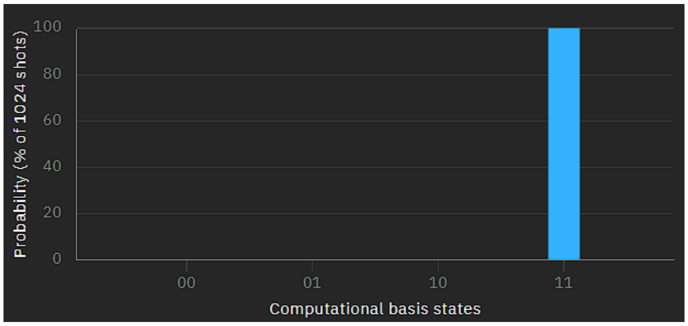
1,0


q[0]
q[1]
c2
1
0


Probability (% of 1024 shots)
100
80
60
40
20
0
00
01
10
11
Computational basis states

1,1


q[0]
q[1]
c2
1
0



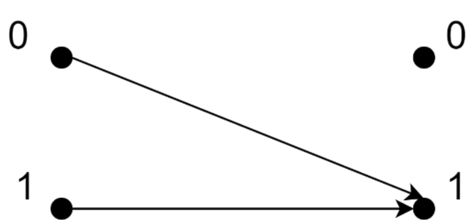
2.2 Implementando las funciones en el computador cuántico

Basándonos en lo anterior, podemos concluir que, para cada entrada en la función, la salida será la misma según la matriz correspondiente.
Para resolver este problema, el algoritmo de Deutsch utiliza circuitos cuánticos que involucran dos qubits. Cuando evaluamos el estado final del sistema, si el primer qubit es 1 y el segundo qubit es 1, obtenemos un resultado. Este resultado nos proporciona un valor binario que podemos interpretar: si el segundo qubit devuelve 0, significa que la función es constante; si devuelve 1, significa que la función está balanceada.

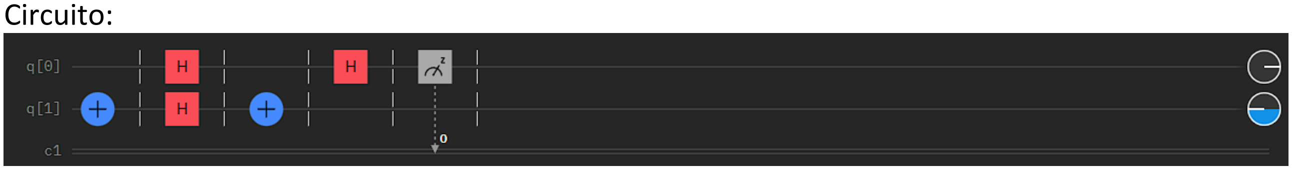
2.3 Implementando el algoritmo de Deutsch en un computador cuántico

Función 1

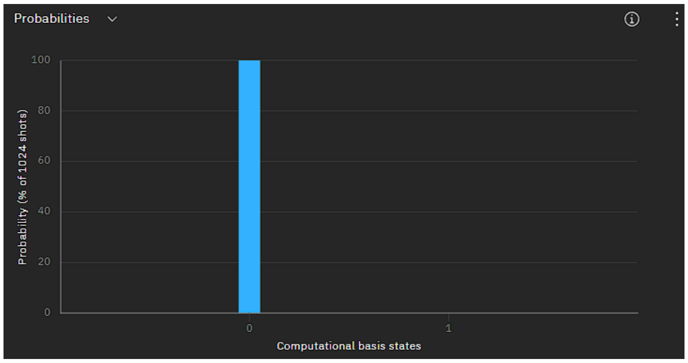
Dibujo:



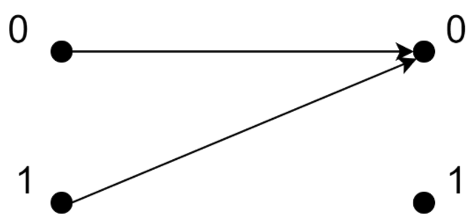
Circuito:



Probabilidad:

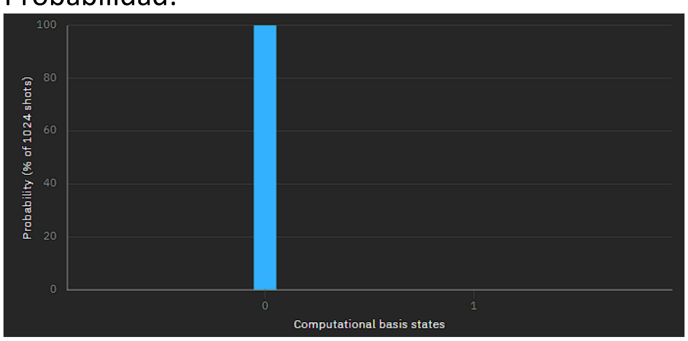


Función 2



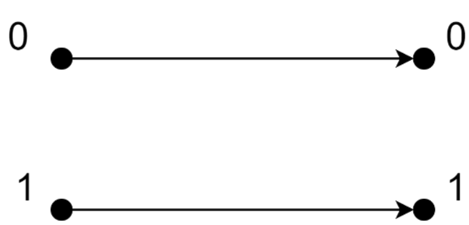
Circuito:

Probabilidad:

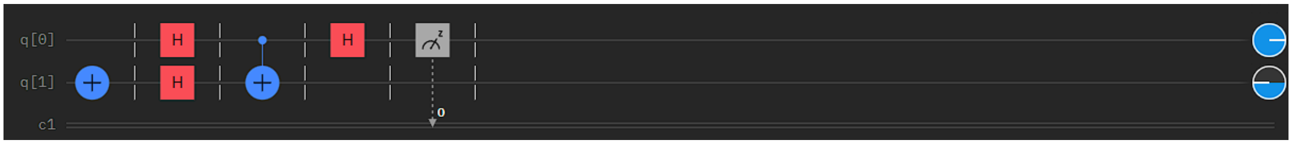


Función 3

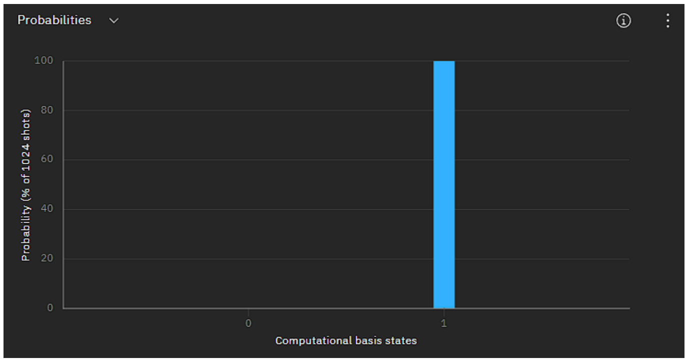
Dibujo



Circuito

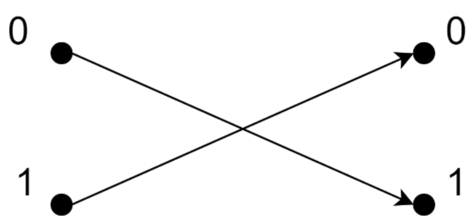


Probabilidad

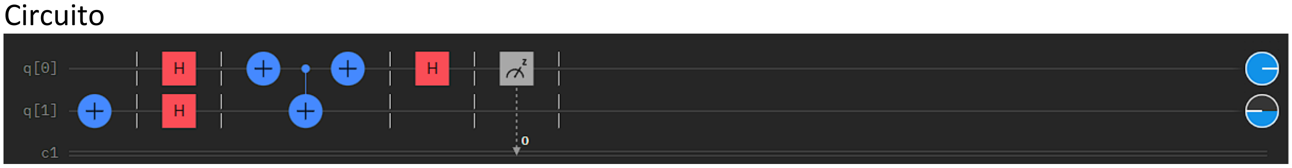


Función 4

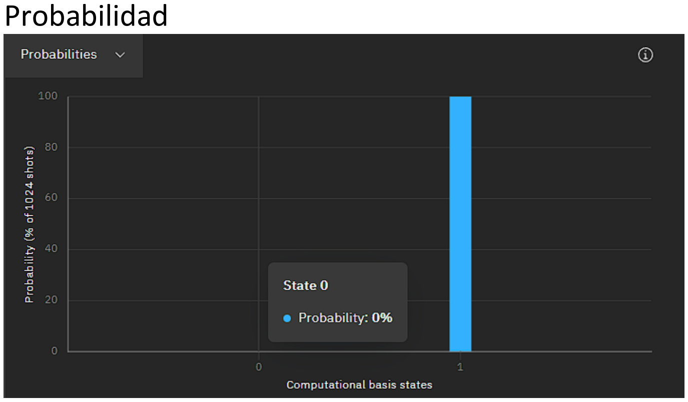
Dibujo



Circuito



Probabilidad



Al observar los resultados obtenidos luego de simular el circuito en el algoritmo de Deutsch, podemos encontrar que las funciones 1 y 2 son constantes, donde determinamos que, al simular el sistema en una computadora cuántica, el qubit superior es 0, lo que indica que las funciones simuladas son constantes. Y las funciones 3 y 4 son balanceadas porque el qubit superior se evalúa en 1, lo que teóricamente muestra que las funciones están balanceadas.

3 Algoritmo de Deutsch-Jozsa

En este capítulo, nos adentraremos en el algoritmo de Deutsch-Jozsa, una extensión del algoritmo de Deutsch que se aplica a funciones booleanas de mayor complejidad. El algoritmo de Deutsch-Jozsa resuelve el problema de determinar si una función booleana dada es balanceada o constante, pero en lugar de limitarse a funciones de un solo bit, puede manejar funciones de múltiples bits. Este algoritmo sigue demostrando la ventaja cuántica al resolver este problema de manera eficiente incluso para funciones más complejas.

El experimento que realizaremos consiste en implementar el algoritmo de Deutsch-Jozsa utilizando un computador cuántico, específicamente el proporcionado por IBM Quantum. Presentaremos una serie de funciones booleanas de \(n\) bits, donde algunas son balanceadas y otras son constantes. Luego, diseñaremos y ejecutaremos circuitos cuánticos para cada función, aplicando el algoritmo de Deutsch-Jozsa para determinar su naturaleza (balanceada o constante). Mostraremos los resultados obtenidos de estas ejecuciones y analizaremos cómo el algoritmo clasifica correctamente las funciones, demostrando así su eficacia y superioridad en comparación con los enfoques clásicos.

Mi comprensión del tema radica en entender que el algoritmo de Deutsch-Jozsa aprovecha la capacidad cuántica para evaluar múltiples entradas simultáneamente, lo que permite una eficiencia superior en comparación con los métodos clásicos. Además, reconozco que este algoritmo es fundamental para mostrar las ventajas prácticas de la computación cuántica en la resolución de problemas booleanos y sienta las bases para algoritmos más avanzados en este campo. En este capítulo, mi objetivo es demostrar cómo el algoritmo de Deutsch-Jozsa puede aplicarse con éxito en un entorno práctico utilizando tecnología cuántica accesible, como la ofrecida por IBM Quantum.

3.1 Problema

El problema que resuelve el algoritmo de Deutsch-Jozsa es determinar la naturaleza de una función booleana de múltiples variables. Específicamente, el algoritmo busca distinguir si una función dada es constante o balanceada.

Las funciones booleanas que aborda el algoritmo de Deutsch-Jozsa pueden tener un número arbitrario de variables, pero para simplificar el análisis, comúnmente se consideran funciones con un número par de variables. Estas funciones se expresan mediante tablas de verdad que muestran todas las posibles combinaciones de valores de entrada y el valor de salida correspondiente.

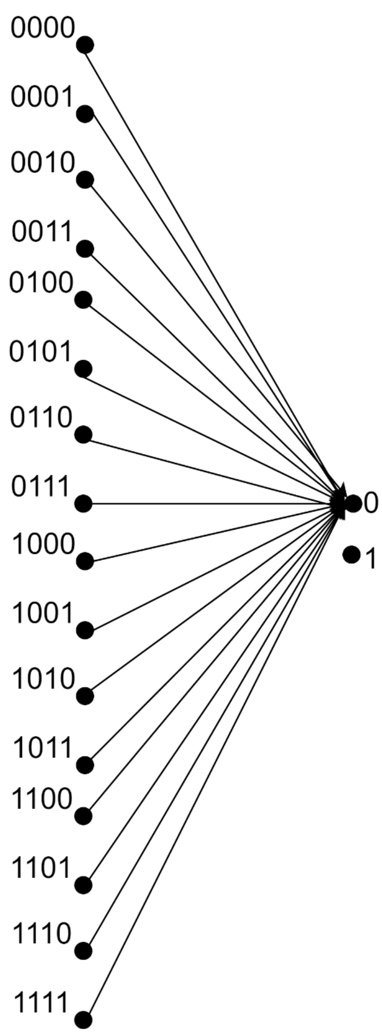
Por ejemplo, si consideramos una función booleana de dos variables, digamos $f(x,y)$, habría cuatro combinaciones posibles de entrada: $(0,0)$, $(0,1)$, $(1,0)$, y $(1,1)$.Para cada una de estas combinaciones, la función puede devolver un valor de 0 o 1. En total, habrá $2^2 = 4$posibles funciones booleanas de dos variables.

Estas funciones pueden visualizarse en forma de tabla de verdad, donde cada fila representa una combinación de entrada y la columna final muestra el valor de salida correspondiente. Por ejemplo:

x	y	$f(x, y)$
0	0	0
0	1	1
1	0	0
1	1	1

Esta tabla representa una función booleana de dos variables donde $f(x, y)$ es constante. El algoritmo de Deutsch-Jozsa se encarga de determinar si una función booleana de múltiples variables es constante (devuelve el mismo valor para todas las entradas) o balanceada (devuelve el mismo número de 0s y 1s para todas las entradas). Esto lo logra con una eficiencia cuántica significativa, lo que lo hace especialmente relevante en el campo de la computación cuántica.

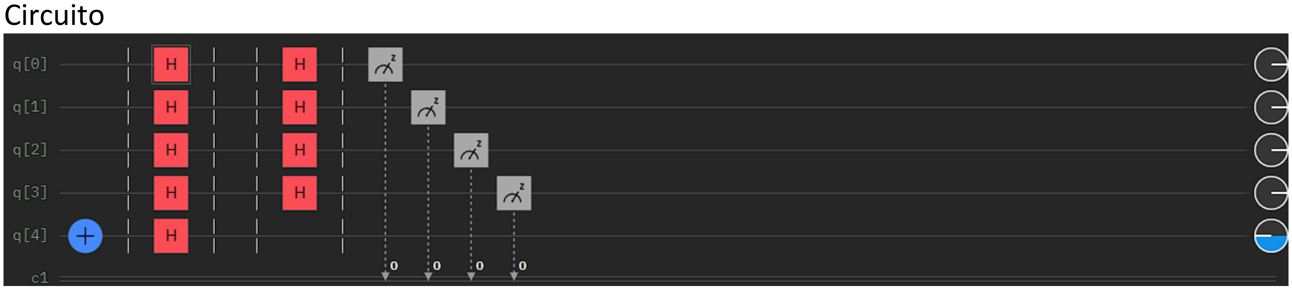
Función 1



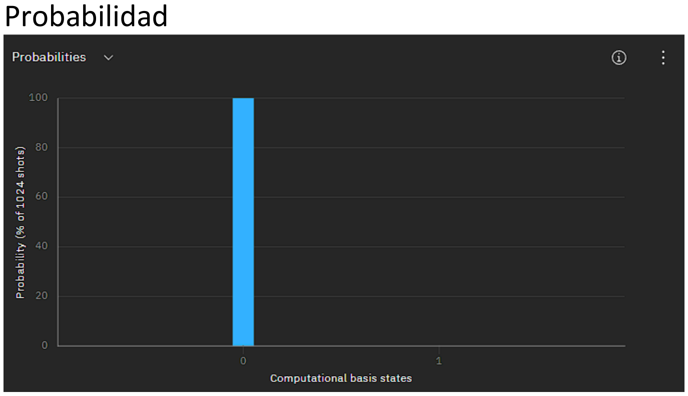
Matriz

```
[1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1]
```

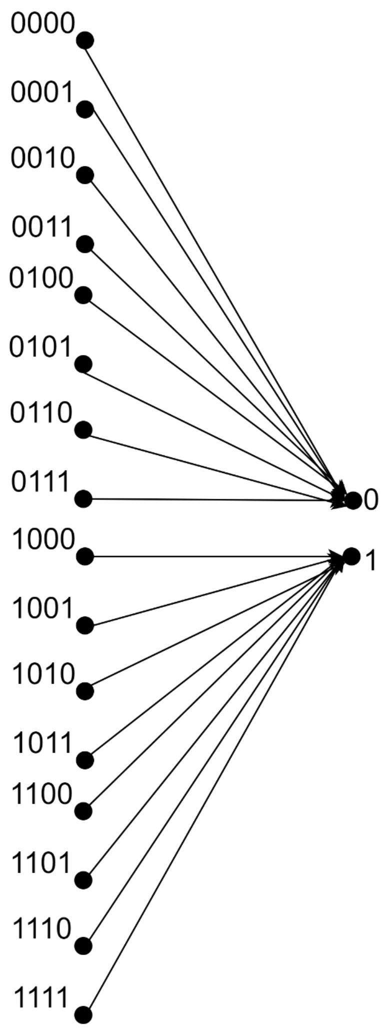
Circuito



Probabilidad



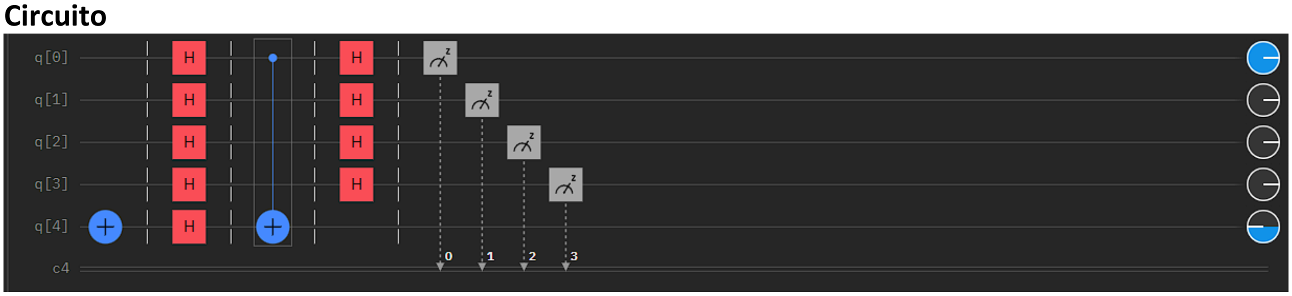
Función 2


0000
0001
0010
0011
0100
0101
0110
0111
1000
1001
1010
1011
1100
1101
1110
1111
0
1

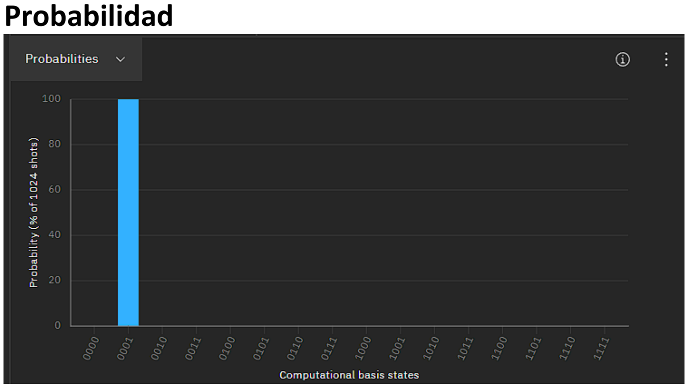
Matriz

```
[1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0]
```

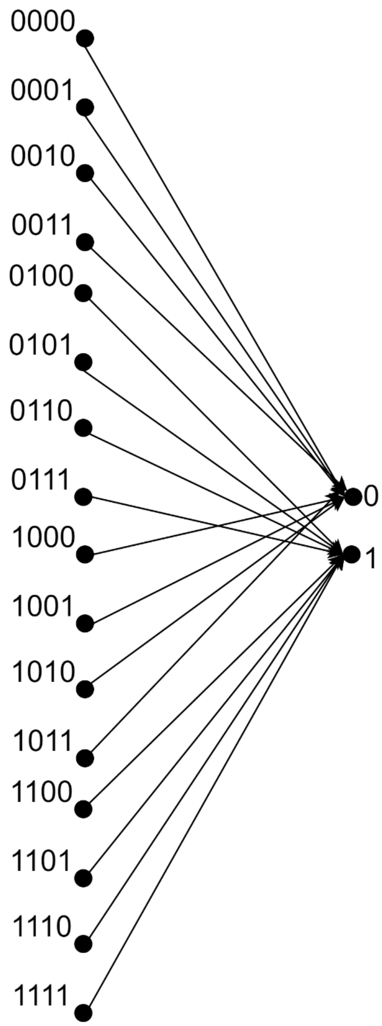
Circuito



Probabilidad



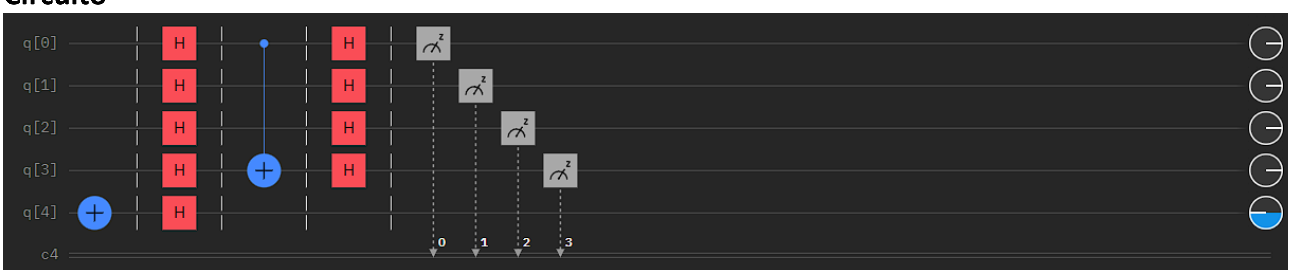
Función 3


0000
0001
0010
0011
0100
0101
0110
0111
1000
1001
1010
1011
1100
1101
1110
1111
0
1

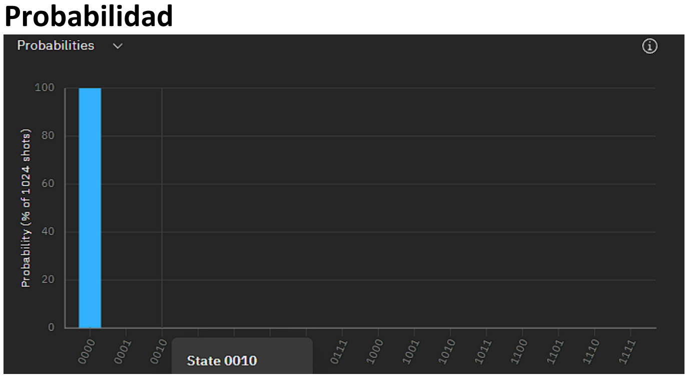
Matriz

```
[1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0]
```

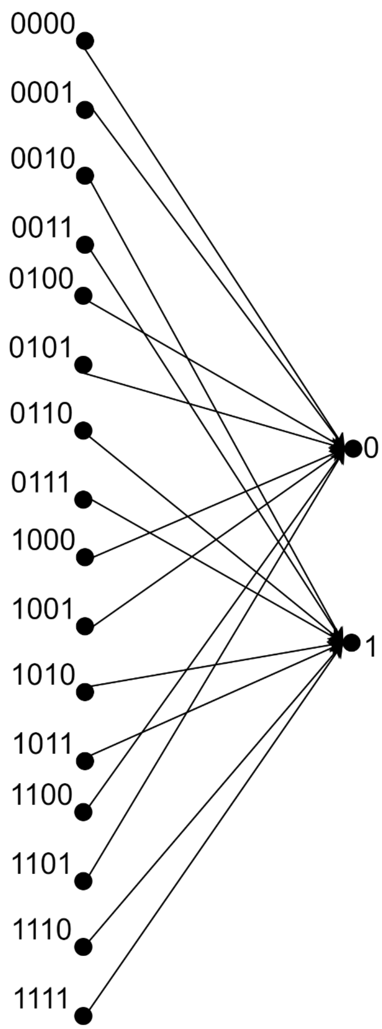
Circuito



Probabilidad



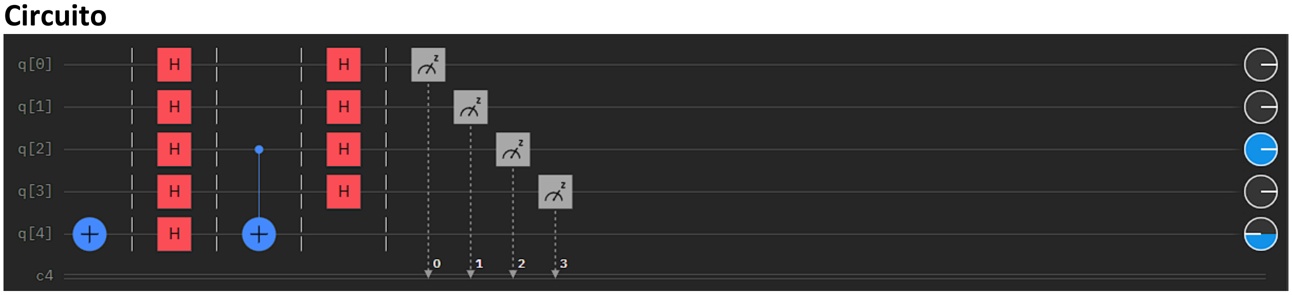
3.2 Funcion 4


0000
0001
0010
0011
0100
0101
0110
0111
1000
1001
1010
1011
1100
1101
1110
1111
0
1

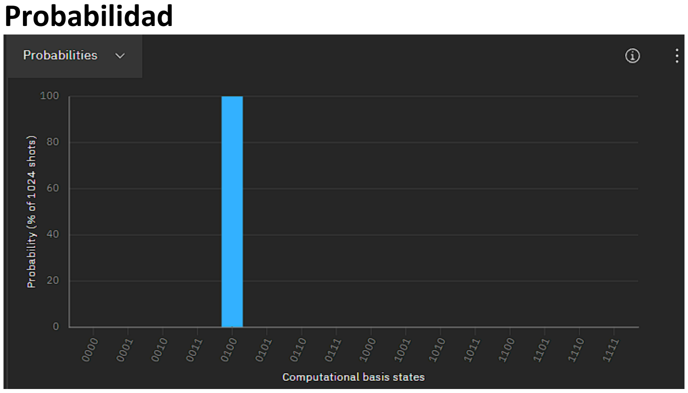
Matriz

```
[1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0]
```

Circuito



Probabilidad



3.3 Implementando las funciones en el computador cuántico

La implementación de funciones booleanas en un computador cuántico implica la representación de las variables y operaciones lógicas utilizando qubits y compuertas cuánticas, implementación:

- Representación de variables: Cada variable en la función booleana se representa utilizando un qubit. Un qubit puede estar en un estado de superposición de $|0\rangle$ y $|1\rangle$, lo que significa que puede contener simultáneamente ambos valores booleanos. Por ejemplo, si tenemos una función booleana de dos variables $f(x, y)$, necesitaremos dos qubits para representar las variables x y y.

- Aplicación de operaciones lógicas: Las operaciones lógicas como AND, OR, NOT, etc., se implementan utilizando compuertas cuánticas. Por ejemplo, la compuerta cuántica CNOT (Controlled-NOT) puede usarse para representar la operación lógica AND en un sistema cuántico. Otras compuertas como la compuerta cuántica Toffoli se pueden utilizar para implementar operaciones lógicas más complejas.

- Construcción del circuito cuántico: Se construye un circuito cuántico que representa la función booleana deseada. Cada compuerta cuántica en el circuito corresponde a una operación lógica en la función booleana. Por ejemplo, si queremos implementar la función $f(x, y) = x$ AND y., podemos construir un circuito que aplique la compuerta CNOT al qubit que representa \(y\) controlado por el qubit que representa x.

- Medición de resultados: Una vez que se aplica el circuito cuántico a los qubits que representan las variables de entrada, se realiza una medición para obtener el resultado de la función booleana. La medición colapsa los qubits a un estado clásico, lo que nos da un resultado booleano que podemos interpretar.

En resumen, la implementación de funciones booleanas en un computador cuántico implica la representación de variables y operaciones lógicas utilizando qubits y compuertas cuánticas, y la construcción de un circuito cuántico que represente la función booleana deseada. Este enfoque aprovecha las propiedades únicas de la computación cuántica para resolver problemas booleanos de manera eficiente en el computador cuántico.

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

De los resultados de las primeras 4 funciones, podemos concluir que las primeras 3 funciones están balanceadas. Esto lo podemos probar por el número de Qbits procesados, ya que cuanto más aumenta el número de qbits utilizados, más probable es que falle, por otro lado, podemos estar absolutamente seguros de que la función 4 es constante.

- El algoritmo de Deutsch es 100% efectivo para demostrar si una funcion es

constante o balanceada teniendo en cuenta sus parámetros y este en buena disposición.
- El computador IBM es actualmente el más capacitado en hacer dichos cálculos, con minúsculos desvíos o percances, pero no afectan en su efectividad, debido a que la tecnología ha hecho todo lo posible con dichos computadores cuánticos.
- El algoritmo Deutsch-Jozsa será efectivo sin importar el dominio de la funcion o su Dificultad,

4 Conclusiones

En este reporte, se exploraron dos algoritmos cuánticos fundamentales: el Algoritmo de Deutsch y el Algoritmo de Deutsch-Jozsa. A través de la descripción y comprensión de estos algoritmos, se reveló cómo la computación cuántica puede resolver problemas booleanos de manera eficiente y escalable.

Lo que entendí es que el algoritmo de Deutsch-Jozsa aprovecha las propiedades de la computación cuántica, como la superposición y la interferencia cuántica, para resolver eficientemente este problema booleano de manera más rápida que los algoritmos clásicos. Aprendí cómo la representación y manipulación de variables booleanas en un entorno cuántico pueden conducir a una solución más eficiente de problemas booleanos complejos.

La proyección que veo a partir de lo aprendido es que el campo de la computación cuántica tiene un potencial significativo para revolucionar la forma en que resolvemos problemas computacionales, especialmente aquellos que involucran cálculos intensivos. Entender y dominar algoritmos como el de Deutsch-Jozsa es crucial para aprovechar al máximo las capacidades de los computadores cuánticos en el futuro.

5 Bibliografía

- Nielsen, M. A., & Chuang, I. L. (2010). Quantum Computation and Quantum Information: 10th Anniversary Edition. Cambridge University Press.
- Preskill, J. (2018). Quantum Computing in the NISQ era and beyond. Quantum, 2, 79.
- Noson S. Yanofsky, Mirco A. Mannucci. Quantum Computing for Computer Scientists.
- Cambridge University Press. 2013 (First published 2008).
- Artículos y documentación proporcionados por IBM Quantum.