UNIVERSIDAD
NACIONAL
DE COLOMBIA
SEDE MEDELLÍN

Análisis del algoritmo Deutsch-Jozsa y Bernstein-Vazirani utilizando la caminata cuántica de tiempo discreto con una única partícula

Sebastian Salazar Perez
Emmanuel Londoño Madrid

Universidad Nacional de Colombia
Facultad de Ciencias, Departamento de Física.
Computación Cuántica
Medellín, Colombia
Noviembre, 2025

Índice

1. Introducción

Este trabajo tiene como objetivo ofrecer un análisis de los fundamentos teóricos y la implementación práctica presentados en el artículo .ªlgoritmo Deutsch-Jozsa y Bernstein-Vazirani utilizando un paseo cuántico de tiempo discreto de una sola partícula". El trabajo original, escrito por Ravi Sangwan, Vikas Ramaswamy, Henry Sukumar y Gudapati Naresh Raghava, investiga los algoritmos Deutsch-Jozsa y Bernstein-Vazirani utilizando un enfoque de paseo cuántico de tiempo discreto de una sola partícula. Un hallazgo clave de la investigación original es la demostración de que se puede mantener la misma velocidad de procesamiento, optimizando así la asignación de recursos. Esta eficiencia se debe a la eliminación del requisito de un cúbit auxiliar, una ventaja notable, especialmente para sistemas de computación cuántica con recursos limitados, como los que utilizan paseos cuánticos fotónicos.

En este estudio también se aborda la implementación de los algoritmos Deutsch-Jozsa y Bernstein-Vazirani empleando la plataforma de simulación híbrida cuántico–clásica NVIDIA CUDA-Q, la cual posibilita la representación eficiente de dinámicas cuánticas complejas mediante arquitecturas aceleradas con GPU. Esta tecnología ofrece un entorno versátil para modelar el paseo cuántico discreto de una sola partícula, combinando operaciones cuánticas con un control clásico de alto rendimiento. Gracias a su habilidad para paralelizar tareas y simular sistemas de gran tamaño, CUDA-Q simplifica la validación, el ajuste y el estudio de los circuitos involucrados en estos algoritmos, permitiendo analizar configuraciones que serían difíciles de probar en dispositivos cuánticos reales. En conjunto, la plataforma fortalece la exploración práctica del enfoque basado en paseos cuánticos, proporcionando un medio eficaz y accesible para corroborar los resultados teóricos del trabajo original.

Para comprender a cabalidad el artículo, es esencial recordar algunos conceptos clave que servirán como base teórica. En esta sección introductoria abordaremos el modelo de consulta cuántica, las compuertas cuánticas de consulta y los fundamentos de la caminata cuántica discreta con una sola partícula. La revisión de estos conceptos nos preparará para analizar la implementación eficiente de los algoritmos de Deutsch-Jozsa y Bernstein-Vazirani utilizando este poderoso marco basado en caminatas cuánticas.

1.1. ¿Qué es un algoritmo cuántico de consulta?

Un **algoritmo cuántico de consulta** es un concepto central dentro del modelo de computación por consultas en la computación cuántica. En este modelo, la información de entrada no se entrega completamente al algoritmo, sino que se accede a ella a través de una función desconocida, conocida como **.ºráculo"** o **çaja negra"**. El algoritmo puede "consultar" a este *.ºráculo"* para obtener información parcial sobre la entrada, de manera similar a cómo se harían preguntas específicas sin conocer todo el contenido de antemano.

En la computación clásica, una consulta corresponde a evaluar la función en un punto específico x, obteniendo como resultado $f(x)$. Sin embargo, en el contexto cuántico, el proceso es mucho más poderoso: gracias al principio de superposición, un algoritmo cuántico puede realizar consultas sobre muchos valores de entrada simultáneamente. Esto permite obtener información global sobre la función con un número significativamente menor de consultas que en el caso clásico.

La eficiencia de un algoritmo cuántico en este modelo se mide, precisamente, por el número de consultas que requiere para resolver un problema. Este enfoque permite abstraer los detalles internos de la función y centrarse únicamente en cuántas veces debe ser consultada para obtener la información necesaria. De este modo, los algoritmos cuánticos de consulta proporcionan una forma práctica de analizar y comparar la eficiencia de los algoritmos cuánticos, especialmente en problemas donde la reducción del número de consultas implica una ventaja exponencial frente a los algoritmos clásicos.

1.2. Ejemplos de algoritmos cuánticos de consulta

- **.°R".** La función de entrada tiene la forma $f : \Sigma^n \to \Sigma$ (por lo que $m = 1$ para este problema). La tarea consiste en imprimir 1 si existe una cadena $x \in \Sigma^n$ para la cual $f(x) = 1$, e imprimir 0 si no existe tal cadena. Si consideramos la función f como una secuencia de 2^n bits a la que tenemos acceso aleatorio, el problema consiste en calcular la operación **.°R"** de estos bits.
- **"Minimum".** La función de entrada toma la forma $f : \Sigma^n \to \Sigma^m$ para cualquier elección de enteros positivos n y m. La salida requerida es la cadena $y \in \{f(x) : x \in \Sigma^n\}$ que aparece primero en el orden lexicográfico (es decir, de diccionario) de Σ^m. Si consideramos la función f como una secuencia de 2^n enteros codificados como cadenas de longitud m en notación binaria a las que tenemos acceso aleatorio, el problema consiste en calcular el mínimo de estos enteros.

A veces también se consideran problemas de consulta en los que se tiene una *"promesa"* sobre la entrada. Esto significa que se nos da algún tipo de garantía sobre la entrada y no somos responsables de lo que suceda cuando no se cumpla dicha garantía. Otra forma de describir este tipo de problema es decir que algunas funciones de entrada (aquellas para las que no se cumple la promesa) se consideran entradas *ïndiferentes"*. No se impone ningún requisito a los algoritmos cuando se les proporcionan entradas *ïndiferentes"*.

1.3. Compuertas cuánticas de consulta

Una **compuerta cuántcia de consulta** es una puerta unitaria que permite al algoritmo interactuar con el oráculo de forma coherente y reversible, respetando las reglas de la mecánica cuántica. Su función es incorporar la información de la función $f(x)$ al estado cuántico del sistema, sin destruir la superposición de los posibles valores de entrada.

Generalmente, una puerta de consulta se representa como una operación unitaria U_f que actúa de la siguiente manera:

$$U_f|x, y\rangle = |x,\ y \oplus f(x)\rangle \tag{1}$$

Intuitivamente, lo que hace la puerta U_f (para cualquier función f elegida) es repetir la cadena de entrada superior $|x\rangle$ y aplicar XOR al valor de la función $|f(x)\rangle$ sobre la cadena

de entrada inferior $|y\rangle$, lo cual es una operación unitaria para cualquier función f elegida. De hecho, es una operación determinista.

La siguiente figura ilustra la acción de la *compuerta cuántica de consulta*.

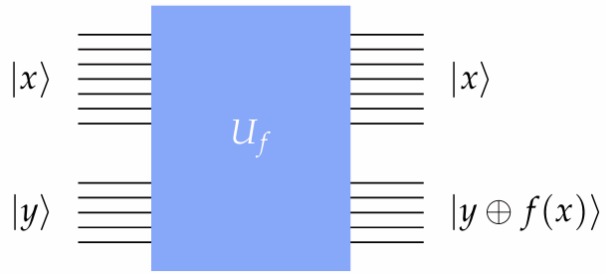


Figura 1: La acción de una puerta de consulta unitaria U_f sobre entradas base estándar.

1.4. Caminatas cuánticas

Las caminatas cuánticas son los análogos cuánticos mecánicos de los paseos aleatorios clásicos influenciados por los principios de la mecánica cuántica, lo que da lugar a un comportamiento y propiedades fundamentalmente diferentes en comparación con sus homólogos clásicos. Las caminatas cuánticas se pueden clasificar en dos tipos: caminatas cuánticas en tiempo discreto y caminatas cuánticas en tiempo continuo. Las caminatas cuánticas en tiempo discreto describen la evolución de un caminante cuántico en pasos de tiempo discretos. Por el contrario, las caminatas cuánticas en tiempo continuo describen la evolución continua de un estado cuántico bajo un hamiltoniano que dicta su evolución dinámica. En el contexto de las caminatas cuánticas en tiempo discreto, se distinguen dos categorías principales: modelos basados en monedas y modelos sin monedas. Los modelos basados en monedas se denominan comúnmente paseos cuánticos en tiempo discreto.

1.5. Caminatas cuánticas en tiempo discreto

La dinámica del paseo cuántico unidimensional en tiempo discreto se describe mediante una partícula con dos grados de libertad internos, que se define en un espacio de Hilbert combinado $\mathcal{H}_w = \mathcal{H}_c \otimes \mathcal{H}_p$. El espacio de Hilbert de moneda, $\mathcal{H}_c = gen\{|0\rangle, |1\rangle\}$ representa los estados internos de la moneda, y el espacio de Hilbert de la posición, $\mathcal{H}_p = gen\{|l\rangle, l \in \mathbb{Z}\}$ representa el número de estados de posición disponibles para la partícula.

La evolución de cada paso de la caminata se define mediante la acción de un operador cuántico unitario de moneda seguido del operador de desplazamiento. La forma general del operador cuántico de moneda toma la forma de una matriz unitaria que actúa solo sobre el espacio de la moneda y viene dada por:

$$\hat{C}(\tau, \xi, \zeta, \theta) = e^{i\tau} \begin{bmatrix} e^{i\xi}\cos(\theta) & e^{i\zeta}\sin(\theta) \\ -e^{-i\zeta}\sin(\theta) & e^{-i\xi}\cos(\theta) \end{bmatrix}. \tag{2}$$

El operador de desplazamiento rige el movimiento de la partícula en función del estado de la moneda, influyendo así en su evolución en el espacio de posiciones. Los operadores de desplazamiento correspondientes se definen de la siguiente manera:

$$\hat{S}_{-}^{k} = \sum_{\substack{l\in\mathbb{Z} \\ j}} \left[|k\rangle\langle k| \otimes |l-1\rangle\langle l| + |j \neq k\rangle\langle j \neq k| \otimes |l\rangle\langle l| \right] \tag{3}$$

$$\hat{S}_{+}^{j} = \sum_{\substack{l\in\mathbb{Z} \\ k}} \left[|k \neq j\rangle\langle k \neq j| \otimes |l\rangle\langle l| + |j\rangle\langle j| \otimes |l+1\rangle\langle l| \right]. \tag{4}$$

El conjunto de operadores $\left\{ \hat{S}_{\pm}^{0}, \hat{S}_{\pm}^{1}, \hat{C}(\tau, \xi, \zeta, \theta) \right\}$ junto con el operador identidad $\hat{S} = \mathbb{I}$ puede considerarse un conjunto genérico de operadores que describe la caminata cuántica en tiempo discreto cuántico. En el artículo utilizan este conjunto de operadores para la realización de compuertas cuánticas universales en un sistema de dos y tres qubits mediante el mapeo del espacio de posición a la base computacional.

2. Algoritmos de Deutsch-Jozsa y Bernstein-Vazirani usando la caminata cuántica en tiempo discreto con una sola partícula

2.1. Algoritmo de Deutsch-Jozsa

El primer paso consiste en examinar el problema para el que se creó el algoritmo. Este problema utiliza una función de entrada, denotada como f, que se define como $f : \Sigma^n \to \Sigma$ para cualquier entero positivo n. El objetivo es determinar una característica específica de la función; es decir, si f es constante o balanceada. El algoritmo está diseñado para generar un 0 si la función f es constante (lo que significa que produce siempre el mismo valor para todas las entradas) y un 1 si la función f está equilibrada (lo que indica que el número de entradas que dan como resultado 0 es exactamente igual al número de entradas que dan como resultado 1).

El algoritmo Deutsch-Jozsa, cuando se aplica a una función con una cadena de entrada de n bits, puede ejecutarse utilizando $n + 1$ qubits (el enfoque estándar, que incluye un qubit auxiliar) o n qubits (sin un qubit auxiliar). El método estándar, que utiliza un qubit auxiliar, se ajusta al diseño típico del oráculo, en el que el qubit auxiliar codifica la salida de la función como una fase. El enfoque de n qubits, que no utiliza un qubit auxiliar, es más eficiente en cuanto a recursos, ya que elimina la necesidad de un qubit auxiliar adicional. Sin embargo, este enfoque requiere una construcción de oráculo más compleja que aplica directamente la transformación de fase.

Presentamos el esquema del algoritmo, analizando la versión sin y con qubit auxiliar. Además, detallamos sus implementaciones cuánticas basadas en el modelo de la caminata cuántica discreta, y finalizamos con la descripción de sus correspondientes implementaciones físicas.

2.1.1. Esquema estándar usando el qubit auxiliar

1. El sistema cuántico se prepara en un estado de $n+1$ qubits, donde n qubits representan el registro de entrada y un qubit auxiliar adicional sirve como registro de salida de la función:
$$|\phi_0\rangle = |0\rangle^{\otimes n} \otimes |1\rangle \tag{5}$$

2. Cada qubit se somete a una transformación de Hadamard:
$$\begin{aligned} |\phi_1\rangle &= H^{\otimes(n+1)} |\phi_0\rangle \\ &= \left(\frac{1}{\sqrt{2^n}} \sum_{x=0}^{2^n-1} |x\rangle \right) \otimes \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) \end{aligned} \tag{6}$$

3. La función f se codifica en un oráculo cuántico, que aplica un cambio de fase condicionado por el resultado de la función.
$$U_f |x\rangle |y\rangle = |x\rangle |y \oplus f(x)\rangle \tag{7}$$

Como el qubit auxiliar está en el estado $(|0\rangle - |1\rangle)$, el oráculo introduce un cambio de fase $(-1)^{f(x)}$:

$$U_f |\phi_1\rangle = \frac{1}{\sqrt{2^n}} \sum_{x=0}^{2^n-1} (-1)^{f(x)} |x\rangle \otimes \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) \tag{8}$$

4. Se aplica una segunda transformación de Hadamard al registro de salida.

$$\begin{aligned} |\phi_2\rangle &= H^{\otimes n} |\phi_1\rangle \\ &= \frac{1}{2^n} \sum_{z=0}^{2^n-1} \left(\sum_{x=0}^{2^n-1} (-1)^{x \cdot z + f(x)} \right) |z\rangle \otimes \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) \end{aligned} \tag{9}$$

La probabilidad de medir $z = 0$, correspondiente a $|0\rangle^{\otimes n}$, es

$$\left| \frac{1}{2^n} \sum_{x=0}^{2^n-1} (-1)^{f(x)} \right|^2$$

que es igual a 1 si $f(x)$ es constante y 0 si $f(x)$ es equilibrada.

5. La medición de los primeros n qubits da como resultado $|0\rangle^{\otimes n}$ con probabilidad 1 si $f(x)$ es constante; si $f(x)$ es equilibrada, la probabilidad es 0.

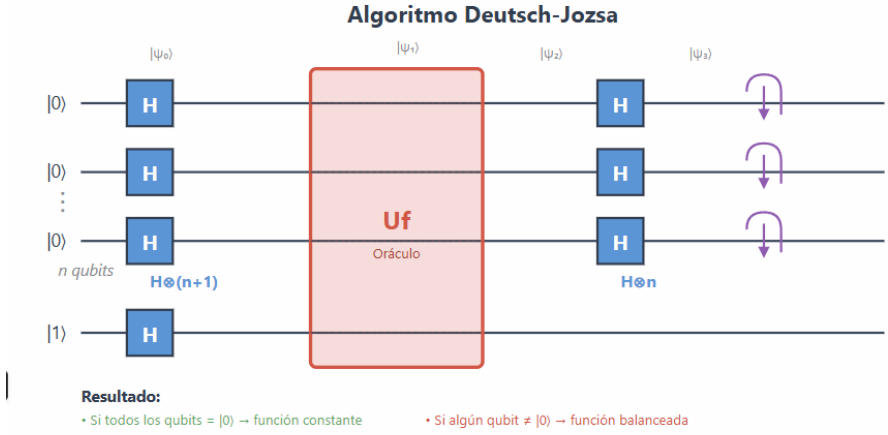


Figura 2: Diagrama del algoritmo Deutsch-Jozsa con qubit auxiliar

2.1.2. Esquema sin usar el qubit auxiliar

1. El sistema cuántico se prepara en un estado de n qubits:

$$|\psi_0\rangle = |0\rangle^{\otimes n} \tag{10}$$

2. Cada qubit se somete a una transformación de Hadamard:

$$\begin{aligned} |\psi_1\rangle &= H^{\otimes n} |\psi_0\rangle \\ &= \left(\frac{1}{\sqrt{2^n}} \sum_{x=0}^{2^n-1} |x\rangle \right) \end{aligned} \tag{11}$$

3. La función f se codifica en un oráculo cuántico, que aplica un cambio de fase condicionado por el resultado de la función.

$$U_f |x\rangle = (-1)^{f(x)} |x\rangle \tag{12}$$

De esa forma,

$$U_f |\psi_1\rangle = \frac{1}{\sqrt{2^n}} \sum_{x=0}^{2^n-1} (-1)^{f(x)} |x\rangle \tag{13}$$

4. Las puertas Hadamard se aplican a todos los qubits:

$$\begin{aligned} |\psi_2\rangle &= H^{\otimes n} |\psi_1\rangle \\ &= \frac{1}{2^n} \sum_{z=0}^{2^n-1} \left(\sum_{x=0}^{2^n-1} (-1)^{x \cdot z + f(x)} \right) |z\rangle \end{aligned} \tag{14}$$

La probabilidad de medir $z = 0$, correspondiente a $|0\rangle^{\otimes n}$, es

$$\left| \frac{1}{2^n} \sum_{x=0}^{2^n-1} (-1)^{f(x)} \right|^2$$

que es igual a 1 si $f(x)$ es constante y 0 si $f(x)$ es equilibrada.

5. La medición de los qubits da como resultado $|0\rangle^{\otimes n}$ con probabilidad 1 si $f(x)$ es constante; si $f(x)$ es equilibrada, la probabilidad es 0.

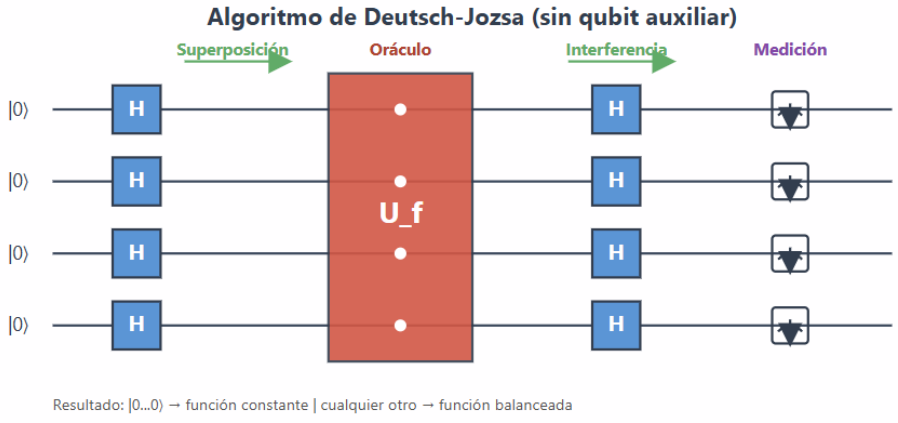


Figura 3: Diagrama del algoritmo Deutsch-Jozsa sin qubit auxiliar

2.1.3. Esquema de la caminata cuántica para algoritmo Deutsch-Jozsa de dos qubits con qubit auxiliar

El esquema de computación cuántica basado en la caminata cuántica que se propone en artículo utiliza una operador de desplazamiento con un operador de moneda dependiente de la posición para realizar las transformaciones correspondientes a cada compuerta cuántica. En nuestro caso, para realizar las operaciones del conjunto universal de compuertas cuánticas en un sistema de tres qubits, la partícula ejecutará un paseo cuántico de modo que la propia partícula actuará como primer qubit con dos grados de libertad internos, $gen\{|0\rangle, |1\rangle\}$, que representan el estado del primer qubit. Finalmente, los estados de los dos qubits restantes se mapean en el espacio de posición. El espacio de posición es un grafo cerrado bidimensional con cuatro vértices y cuatro aristas, $gen\{|00\rangle, |01\rangle\, |10\rangle, |11\rangle\}$.

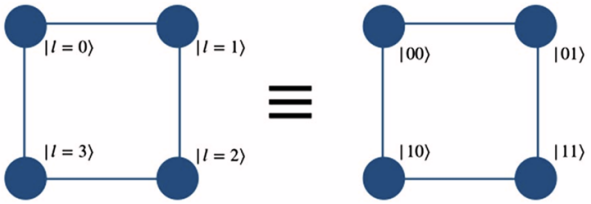


Figura 4: Correspondencia entre el espacio de Hilbert de posición en el estado de paseo cuántico cerrado unidimensional y la base computacional del segundo y tercer qubit en el sistema de tres qubits.

1. Inicialmente, la partícula comienza con el estado de moneda $|0\rangle$ y el estado de posición $|00\rangle$, después de lo cual se aplica la operación de moneda $\hat{C}(\frac{-\pi}{2}, 0, \frac{\pi}{2}, \frac{\pi}{4}) = \sigma_x$ (Que corresponde a la compuerta Pauli-X) sobre el estado de moneda (partícula), seguida de la operación de identidad en el espacio de posiciones para obtener el estado de moneda $|1\rangle$.

2. la puerta de Hadamard sobre el qubit auxiliar (estado de moneda) se implementa mediante la operación de moneda $\hat{C}(\frac{-\pi}{2}, \frac{-\pi}{2}, \frac{\pi}{2}, \frac{\pi}{4}) = H$ aplicada al estado de moneda (partícula), seguida de una operación de identidad en el espacio de posiciones.

3. Las puertas de Hadamard que actúan sobre los qubits de trabajo en el espacio de posición se implementan evolucionando el estado de la moneda a una superposición en todo el espacio de posición. Para ello tengamos en cuenta las siguientes operaciones:

$$\begin{aligned} W_+^0 |k\rangle \otimes |m\rangle &= \left[\sigma_x^m S_+^k (\sigma_x \otimes \mathbb{I})\right] \\ W_+^1 |k\rangle \otimes |m\rangle &= \left[\sigma_x^m S_+^k (\sigma_z \otimes \mathbb{I})\right] \\ W_-^0 |j\rangle \otimes |n\rangle &= \left[\sigma_x^n S_-^j (\sigma_x \otimes \mathbb{I})\right] \\ W_-^1 |j\rangle \otimes |n\rangle &= \left[\sigma_x^n S_-^j (\sigma_z \otimes \mathbb{I})\right], \end{aligned} \tag{15}$$

donde, $\sigma_x^m = \sigma_x \otimes |m\rangle\langle m| + \mathbb{I}_c \otimes \sum_{l \neq m} |l\rangle\langle l|$ y $|m\rangle$ es el estado de posición inicial de la partícula.

Teniendo en cuenta estos operadores, la implementacion de las compuertas Hadarmard en el segundo y tercer qubit se hacen de la siguiente manera:

Implementación de la compuerta Hadarmard en el segundo qubit.

$$\begin{aligned} H_2 |k00\rangle &\to W_+^{(k \text{ mód } 2)} (\hat{H} \otimes \mathbb{I}) |k, l = 0\rangle , \\ H_2 |k01\rangle &\to W_+^{(k \text{ mód } 2)} (\hat{H} \otimes \mathbb{I}) |k, l = 1\rangle , \\ H_2 |k11\rangle &\to W_+^{((k+1) \text{ mód } 2)} (\hat{H} \otimes \mathbb{I}) |k, l = 2\rangle , \\ H_2 |k10\rangle &\to W_+^{((k+1) \text{ mód } 2)} (\hat{H} \otimes \mathbb{I}) |k, l = 3\rangle \end{aligned} \tag{16}$$

Y

Implementación de la compuerta Hadarmard en el tercer qubit.

$$\begin{aligned} H_3 |k00\rangle &\to W_+^{(k \text{ mód } 2)} (\hat{H} \otimes \mathbb{I}) |k, l = 0\rangle , \\ H_3 |k01\rangle &\to W_+^{((k+1) \text{ mód } 2)} (\hat{H} \otimes \mathbb{I}) |k, l = 1\rangle , \\ H_3 |k11\rangle &\to W_+^{((k+1) \text{ mód } 2)} (\hat{H} \otimes \mathbb{I}) |k, l = 2\rangle , \\ H_3 |k10\rangle &\to W_-^{(k \text{ mód } 2)} (\hat{H} \otimes \mathbb{I}) |k, l = 3\rangle , \end{aligned} \tag{17}$$

Donde l son los nombres de las etiquetas de los puntos en el espacio de posición en sentido horario, así como se ilustra en la siguiente figura.

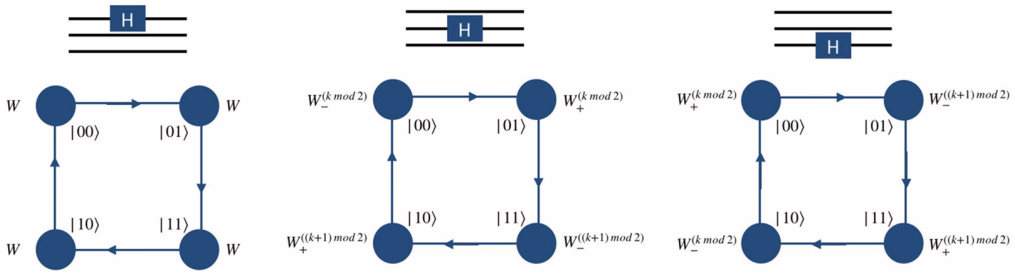


Figura 5: Ilustración esquemática de la operación de Hadamard sobre la base computacional del sistema de tres qubits utilizando operadores de paseo cuántico dependientes de la posición.

4. Las implementaciones de oráculo basadas en paseos cuánticos para todas las funciones booleanas de dos bits que son constantes o equilibradas con qubit auxiliar se pueden lograr utilizando una operación de moneda dependiente de la posición $\hat{C}(\frac{-\pi}{2}, 0, \frac{\pi}{2}, \frac{\pi}{4}) = \sigma_x$ junto con un operador de desplazamiento de identidad.

Para la ejecución del oráculo tenemos que tener en cuenta las diferentes funciones f porque a cada una corresponde una implementacion diferente del oráculo.

Inputs	**Functions** $f(x_1, x_2)$							
(x_1, x_2)	(i)	(ii)	(iii)	(iv)	(v)	(vi)	(vii)	$(viii)$
$(0, 0)$	0	1	0	0	1	1	0	1
$(0, 1)$	0	1	0	1	1	0	1	0
$(1, 0)$	0	1	1	0	0	1	1	0
$(1, 1)$	0	1	1	1	0	0	0	1

Figura 6: Ilustración de las entradas y salidas de todas las funciones booleanas de dos bits posibles que son constantes o equilibradas.

En la siguiente figura se muestra las implementaciones del oráculo para todas las funciones booleanas de dos bits que son constantes o equilibradas, utilizando las compuertas Pauli X y $CNOT$.

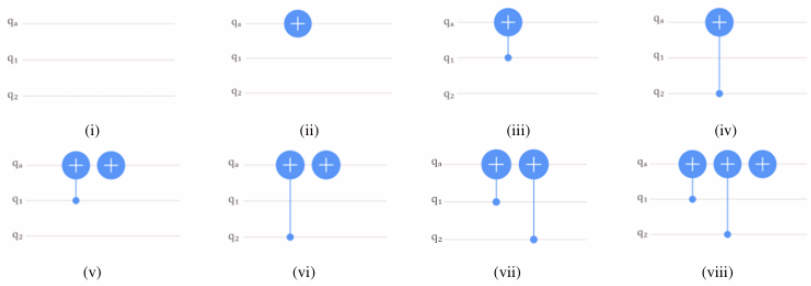


Figura 7: implementaciones del oráculo para todas las funciones booleanas de dos bits que son constantes o equilibradas, en base a la Figura 6.

Teniendo en cuenta el diagrama anterior, la implementación del oráculo para cada función, teniendo en cuenta el esquema de la caminata cuántica se encuentra en la siguiente figura:

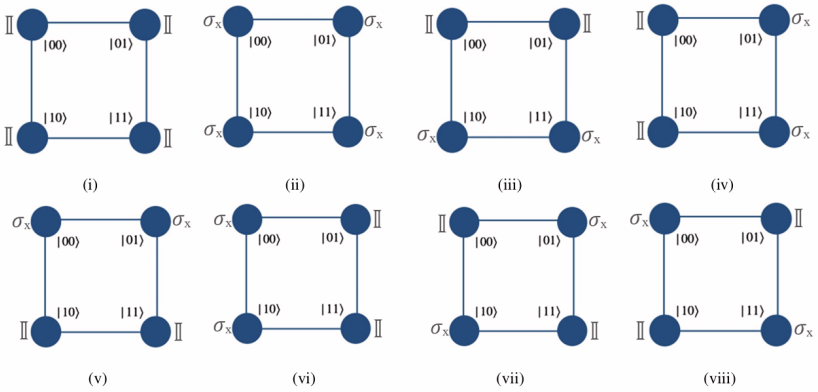


Figura 8: Ilustración esquemática de las implementaciones de oráculo basadas en paseo cuántico con qubit auxiliar para todas las funciones booleanas de dos bits que son constantes o balanceadas.

2.1.4. Esquema de la caminata cuántica para algoritmo Deutsch-Jozsa de dos qubits sin qubit auxiliar

En este caso, la caminata cuántica se describe en un grafo simple de dos vértices. El sistema está definido por dos qubits: el primero representa el estado de la "moneda"(que dirige el paseo) y el segundo representa la posición de la partícula, limitada a los estados $|0\rangle$ o $|1\rangle$.

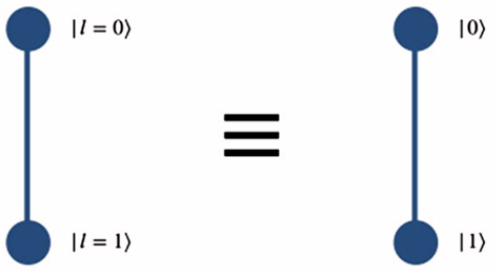


Figura 9: Correspondencia entre los dos estados del espacio de Hilbert de posición en una dimensión y la base computacional del segundo qubit en un sistema de dos qubits.

La caminata comienza en la posición $|0\rangle$ y con la moneda en el estado $|0\rangle$. Las transformaciones de Hadamard, cruciales para la caminata, se logran aplicando una serie específica de operaciones de desplazamiento y de moneda. Las transformaciones Hadamard son implementadas de la siguiente manera

Transformación Hadamard en el primer qubit:

$$H_1 \left|kl\right\rangle \rightarrow \mathbb{I}\left(\hat{C}(\frac{-\pi}{2}, \frac{-\pi}{2}, \frac{\pi}{2}, \frac{\pi}{4}) \otimes I_p\right) \left|k, l\right\rangle \tag{18}$$

Transformación Hadamard en el segundo qubit:

$$\begin{aligned} H_2 \left|k0\right\rangle &\rightarrow W_+^{(k \text{ mód } 2)} (\hat{H} \otimes \mathbb{I}) \left|k, l = 0\right\rangle, \\ H_2 \left|k1\right\rangle &\rightarrow W_-^{((k+1) \text{ mód } 2)} (\hat{H} \otimes \mathbb{I}) \left|k, l = 1\right\rangle \end{aligned} \tag{19}$$

Donde los operadores $W_+^{(k \text{ mód } 2)}, W_+^{(k+1 \text{ mód } 2)}, W_-^{(k \text{ mód } 2)}$ y $W_-^{(k+1 \text{ mód } 2)}$ están dados por (15).

Para cualquier función booleana de dos bits que sea constante o balanceada, y que no requiera un qubit auxiliar, es posible implementar su oráculo usando caminatas cuánticas. Esto se logra aplicando la operación de identidad ($\mathbb{I}$) junto con tres operadores de evolución que dependen de la posición ($\hat{O}_1, \hat{O}_2, \hat{O}_3$).

$$\hat{O}_1 = \hat{C}\left(\frac{5}{2}, \frac{3}{2}, 0, \pi\right) \otimes \hat{I} = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \otimes \hat{I},$$

$$\hat{O}_2 = \hat{C}\left(\frac{13}{2}, \frac{3}{2}, 0, 0\right) \otimes \hat{I} = \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix} \otimes \hat{I},$$

$$\hat{O}_3 = e^{i\pi}\left(\hat{C}(0, 0, 0, 0) \otimes \hat{I}\right) = e^{i\pi} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \otimes \hat{I}.$$

$$\mathbb{I} = \hat{C}(0, 0, 0, 0) \otimes \hat{I} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \otimes \hat{I}.$$

La siguiente figura muestra la implementación del oráculo basadas en caminatas cuánticas sin qubit auxiliar para todas las funciones booleanas de dos bits que son constantes o balanceadas, dadas por la Figura 6, utilizando los operadores anteriores.

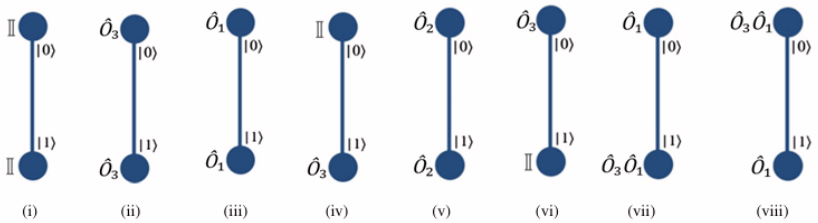


Figura 10: Ilustración esquemática de las implementaciones de oráculo basadas en caminatas cuánticas sin qubit auxiliar para todas las funciones booleanas de dos bits que son constantes o balanceadas, dadas por la Figura 6, utilizando la operación de identidad y tres operadores dependientes de la posición.

2.1.5. Implementación física del algoritmo Deutsch-Jozsa usando un qubit auxiliar

La implementación física del algoritmo bajo el marco de Caminata Cuántica de Tiempo Discreto se fundamenta en la codificación de la información cuántica en los grados de libertad de una única partícula, específicamente un fotón. En este esquema, los elementos abstractos del algoritmo se mapean a propiedades ópticas observables y manipulables mediante óptica lineal.

Para el esquema estándar que incluye un qubit auxiliar, la correspondencia física es la siguiente:

1. **Espacio de la Moneda ($\mathcal{H}_c$):** Representa el estado del qubit auxiliar. Físicamente, este espacio se implementa utilizando los estados de *polarización* del fotón.

$$|0\rangle_c \to |H\rangle \text{ (Horizontal)}, \quad |1\rangle_c \to |V\rangle \text{ (Vertical)} \tag{20}$$

2. **Espacio de Posición ($\mathcal{H}_p$):** Representa el registro de entrada (qubits de trabajo). Físicamente, se corresponde con las *trayectorias espaciales* (modos longitudinales) que el fotón puede ocupar dentro del interferómetro.

La evolución del sistema se rige por operadores unitarios implementados mediante componentes ópticos estándar:

- **Operador de Moneda ($\hat{C}$):** Las rotaciones en el espacio de Hilbert de la moneda (como la compuerta Hadamard o Pauli-X) se realizan mediante **Placas de Onda (Waveplates)**.
  - Una placa de media onda (HWP, $\lambda/2$) permite rotar la polarización lineal, implementando operaciones como σ_x.
  - Una combinación de placas HWP y de cuarto de onda (QWP, $\lambda/4$) permite realizar rotaciones arbitrarias en la esfera de Bloch del espacio de polarización.
- **Operador de Desplazamiento ($\hat{S}$):** El entrelazamiento condicional entre el estado de la moneda y la posición se logra mediante **Divisores de Haz Polarizantes (PBS)**. Un PBS transmite el fotón si su polarización es horizontal ($|H\rangle$) y lo refleja si es vertical ($|V\rangle$), acoplando efectivamente el grado de libertad interno con la trayectoria espacial.

En este montaje, el oráculo actúa modificando la polarización del fotón (qubit auxiliar) dependiendo de la trayectoria por la que viaja, lo cual requiere un control preciso de la interferencia y componentes adicionales para mantener la coherencia entre los caminos.

2.1.6. Implementación física del algoritmo Deutsch-Jozsa sin qubit auxiliar

La innovación principal analizada en este trabajo es la eliminación del qubit auxiliar, lo que permite una implementación más eficiente en términos de recursos físicos[cite: 406, 422]. En este enfoque, la información de la función $f(x)$ se imprime directamente en la fase global de la partícula, eliminando la necesidad de manipular el grado de libertad de polarización para codificar la salida del oráculo.

Esquema Experimental

Tal como se ilustra en la Figura 8, la implementación física se basa en una estructura interferométrica (análoga a un interferómetro Mach-Zehnder) que utiliza óptica lineal. El proceso físico se describe en tres etapas:

1. **Preparación y Superposición:** Un fotón individual incide en el sistema y atraviesa un primer *Divisor de Haz* (Beam Splitter, BS). Este componente realiza la operación de Hadamard sobre el espacio de posición, generando una superposición coherente de trayectorias espaciales:

$$|\psi_{in}\rangle \xrightarrow{BS} \frac{1}{\sqrt{2}}(|\text{Camino}_0\rangle + |\text{Camino}_1\rangle) \tag{21}$$

2. **Implementación del Oráculo de Fase (U_f):** A diferencia del enfoque estándar que requiere compuertas CNOT, este esquema implementa el oráculo $U_f |x\rangle = (-1)^{f(x)} |x\rangle$ mediante **Desfasadores (Phase Shifters)** y configuraciones de placas de onda (Q1, Q2, O) ubicadas en los brazos del interferómetro.

   - Si $f(x) = 0$, el fotón viaja por la trayectoria sin sufrir cambios de fase relativos.
   - Si $f(x) = 1$, se introduce un desfase de π (cambio de signo) en la trayectoria correspondiente.

   Físicamente, esto se logra ajustando el índice de refracción o la longitud del camino óptico en uno de los brazos, o mediante la orientación específica de las láminas retardadoras que actúan sobre la fase geométrica del fotón.

3. **Interferencia y Medición:** Las trayectorias se recombinan en un segundo Beam Splitter. Debido a la naturaleza ondulatoria del fotón, se produce interferencia:

   - Si la función es *constante* ($f(0) = f(1)$), la interferencia es constructiva hacia un puerto de salida y destructiva hacia el otro.
   - Si la función es *balanceada* ($f(0) \neq f(1)$), el desfase relativo de π invierte las condiciones de interferencia, dirigiendo el fotón hacia el puerto opuesto.

Finalmente, la medición se realiza mediante fotodetectores (DET) colocados a la salida. La detección del fotón en un puerto específico permite determinar con certeza (probabilidad 1) si la función es constante o balanceada con una sola consulta al oráculo, utilizando un número reducido de componentes ópticos (menor cantidad de Waveplates y PBS en comparación con la versión con qubit auxiliar).

2.2. Algoritmo Bernstein-Vazirani

En esta sección analizaremos un problema conocido como el problema de Bernstein-Vazirani. Antes de abordar el problema, vamos a introducir la siguiente notación. Sean $x = x_{n-1} \cdots x_0$ y $y = y_{n-1} \cdots y_0$ dos cadenas binarias de longitud n, definimos:

$$x \cdot y = x_{n-1}y_{n-1} \oplus \cdots \oplus x_0y_0 = x \cdot y \quad (\text{mód } 2).$$

Nos referiremos a esta operación como el *producto escalar binario*. Una forma alternativa de definirlo es la siguiente:

$$x \cdot y = \begin{cases} 1 & x_{n-1}y_{n-1} + \cdots + x_0y_0 \text{ es impar} \\ 0 & x_{n-1}y_{n-1} + \cdots + x_0y_0 \text{ es par} \end{cases}$$

En el planteamiento del problema Bernstein-Vazirani, la entrada es una función $f : \{0,1\}^n \to \{0,1\}$. De entrada, hay una promesa de que existe una cadena binaria secreta $s = s_{n-1} \cdots s_0$ para la cual $f(x) = s \cdot x$ para todo $x \in \Sigma^n$. El resultado debe de ser el valor de la cadena s.

El algoritmo Deutsch-Jozsa es suficiente para resolver este problema; no se requiere un nuevo algoritmo cuántico. Para facilitar la referencia, llamaremos al circuito cuántico mostrado anteriormente el circuito Deutsch-Jozsa.

Comenzaremos nuestro análisis del comportamiento del circuito Deutsch-Jozsa, concretamente, cuando la función de entrada cumple los requisitos del problema de Bernstein-Vazirani señalando un aspecto rápido. Este aspecto es que el efecto de aplicar n compuertas Hadamard a los estados de base estándar de n qubits puede representarse de forma equivalente utilizando el producto escalar binario. Esto se logra de la siguiente manera:

$$H^{\otimes n}|x\rangle = \frac{1}{\sqrt{2^n}} \sum_{y \in \Sigma^n} (-1)^{x \cdot y}|y\rangle$$

Al igual que lo que vimos al analizar el algoritmo de Deutsch, esto se debe a que el valor $(-1)^k$ para cualquier número entero k depende únicamente de si k es par o impar.

Pasando al circuito Deutsch-Jozsa, después de realizar la primera capa de compuertas Hadamard, el estado de los $n + 1$ qubits es

$$|-\rangle \otimes \frac{1}{\sqrt{2^n}} \sum_{x \in \Sigma^n} |x\rangle.$$

A continuación, se implementa la compuerta de consulta, que (a través del fenómeno de retroceso de fase) transforma el estado en

$$|-\rangle \otimes \frac{1}{\sqrt{2^n}} \sum_{x \in \Sigma^n} (-1)^{f(x)}|x\rangle.$$

Utilizando nuestra fórmula para la acción de una capa de compuertas Hadamard, vemos que la segunda capa de compuertas Hadamard transforma entonces este estado en

$$|-\rangle \otimes \frac{1}{2^n} \sum_{x \in \Sigma^n} \sum_{y \in \Sigma^n} (-1)^{f(x)+x \cdot y} |y\rangle$$

Ahora podemos hacer algunas simplificaciones en el exponente de -1 dentro de la suma. Se nos garantiza que $f(x) = s \cdot x$ para alguna cadena $s = s_{n-1} \cdots s_0$, por lo que podemos expresar el estado como

$$|-\rangle \otimes \frac{1}{2^n} \sum_{x \in \Sigma^n} \sum_{y \in \Sigma^n} (-1)^{s \cdot x + x \cdot y} |y\rangle.$$

Dado que $s \cdot x$ y $x \cdot y$ son valores binarios, podemos sustituir la suma por el OR exclusivo, ya que lo único que importa para un entero en el exponente de -1 es si es par o impar. Aprovechando la simetría del producto escalar binario, obtenemos esta expresión para el estado:

$$|-\rangle \otimes \frac{1}{2^n} \sum_{x \in \Sigma^n} \sum_{y \in \Sigma^n} (-1)^{(s \cdot x) \oplus (y \cdot x)} |y\rangle.$$

Se han añadido paréntesis para mayor claridad, aunque en realidad no son necesarios, ya que es habitual considerar que el producto binario tiene mayor prioridad que el OR exclusivo.

En este punto utilizaremos la siguiente fórmula.

$$(s \cdot x) \oplus (y \cdot x) = (s \oplus y) \cdot x$$

Podemos obtener la fórmula a través de una formula similar para bits,

$$(ac) \oplus (bc) = (a \oplus b)c,$$

junto con una expansión del producto binario y el OR exclusivo bit a bit.

$$\begin{aligned}(s \cdot x) \oplus (y \cdot x) &= (s_{n-1}x_{n-1}) \oplus \cdots \oplus (s_0 x_0) \oplus (y_{n-1}x_{n-1}) \oplus \cdots \oplus (y_0 x_0) \\ &= (s_{n-1} \oplus y_{n-1})x_{n-1} \oplus \cdots \oplus (s_0 \oplus y_0)x_0 \\ &= (s \oplus y) \cdot x\end{aligned}$$

Esto nos permite expresar el estado del circuito inmediatamente antes de las mediciones de la siguiente manera:

$$|-\rangle \otimes \frac{1}{2^n} \sum_{x \in \Sigma^n} \sum_{y \in \Sigma^n} (-1)^{(s \oplus y) \cdot x} |y\rangle.$$

El último paso es utilizar otra fórmula más, que funciona para cada cadena binaria $z = z_{n-1} \cdots z_0$.

$$\frac{1}{2^n} \sum_{x \in \Sigma^n} (-1)^{z \cdot x} = \begin{cases} 1 & \text{if } z = 0^n \\ 0 & \text{if } z \neq 0^n \end{cases}$$

Aquí utilizamos una notación sencilla para las cadenas que usaremos durante el resto de la discusión sobre este algoritmo: 0^n es la cadena compuesta únicamente por ceros de longitud n.

Una forma sencilla de demostrar que esta fórmula funciona es considerar los dos casos por separado. Si $z = 0^n$, entonces $z \cdot x = 0$ para cada cadena $x \in \Sigma^n$, por lo que el valor de cada término de la suma es 1, y obtenemos 1 sumando y dividiendo por 2^n. Por otro lado, si cualquiera de los bits de z es igual a 1, entonces el producto escalar binario $z \cdot x$ es igual a 0 para exactamente la mitad de las posibles opciones de $x \in \Sigma^n$ y 1 para la otra mitad, porque el valor del producto escalar binario $z \cdot x$ cambia (de 0 a 1 o de 1 a 0) si cambiamos cualquier bit de x en una posición en la que z tiene un 1.

Si ahora aplicamos esta fórmula para simplificar el estado del circuito antes de las mediciones, obtenemos

$$|-\rangle \otimes \frac{1}{2^n} \sum_{x \in \Sigma^n} \sum_{y \in \Sigma^n} (-1)^{(s \oplus y) \cdot x} |y\rangle = |-\rangle \otimes |s\rangle,$$

debido al hecho de que $s \oplus y = 0^n$ si y solo si $y = s$. Por lo tanto, las mediciones revelan precisamente la cadena s que estamos buscando.

2.2.1. Transición hacia la Implementación Física

Para implementar este algoritmo en el marco de la Caminata Cuántica de Tiempo Discreto sin qubit auxiliar, reinterpretamos la acción del oráculo en términos de componentes ópticos.

En el esquema teórico, la cadena oculta s determina qué estados $|x\rangle$ reciben un cambio de fase de π (multiplicación por -1). En la implementación física con un solo fotón, esto se traduce en la configuración de la *fase óptica* en los distintos caminos del interferómetro.

Específicamente, si el bit j-ésimo de la cadena secreta es 1 ($s_j = 1$), esto implica físicamente la inserción de un elemento que modifique la fase (como un *Phase Shifter* o una lámina retardadora) en el camino óptico correspondiente al modo j.

2.2.2. Implementación Física del Algoritmo

La realización experimental propuesta utiliza la misma arquitectura interferométrica base descrita para el algoritmo Deutsch-Jozsa (ver Figura 11), pero con una configuración específica del oráculo para codificar s.

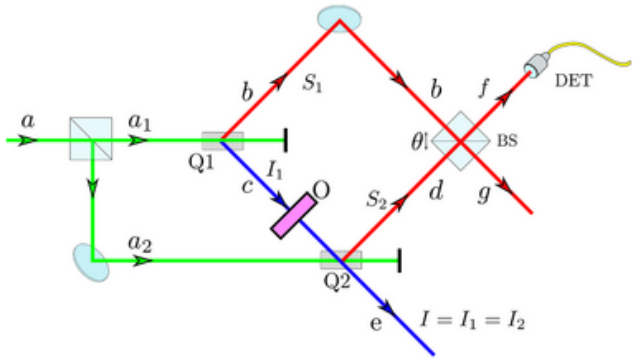
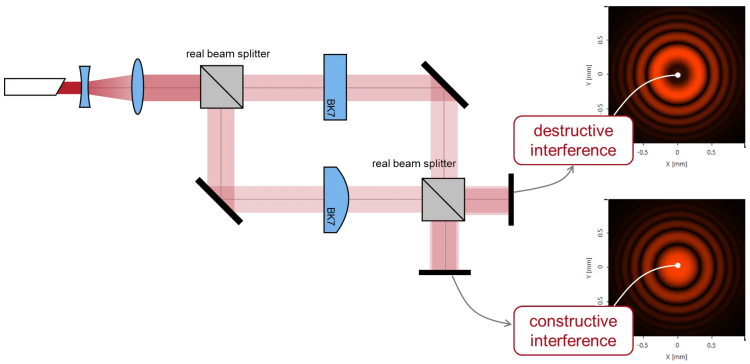


Figura 11: Implementacion sin Qubit auxiliar.

- **Codificación del Oráculo (U_f):** El oráculo ya no distingue simplemente entre constante o balanceada, sino que codifica la cadena s. Esto se logra colocando desfasadores en los brazos del interferómetro.
  - Para descubrir un bit específico de s, el oráculo introduce un desfase de π en el camino óptico si y solo si el bit correspondiente de la cadena oculta es 1.
  - En términos de la óptica lineal, esto corresponde a la operación $e^{i\pi} = -1$ aplicada selectivamente a los modos espaciales del fotón.
- **Proceso de Medición:** Tras pasar por el segundo Beam Splitter (que realiza la operación de mezcla/interferencia final), la posición en la que se detecta el fotón revela la información de s.



  - En un sistema de 1 qubit (simplificado), la detección en el puerto "superior"podría indicar $s = 0$, mientras que la detección en el puerto ïnferiorïndicaría $s = 1$.
  - La "magia"del algoritmo radica en que la interferencia óptica redirige el fotón al puerto de salida que corresponde binariamente al valor de s, permitiendo identificar la configuración del oráculo (la cadena secreta) en un solo paso de vuelo del fotón.

Esta implementación demuestra cómo la computación cuántica fotónica puede resolver el problema de Bernstein-Vazirani aprovechando la interferencia de un solo fotón, reduciendo

la complejidad de recursos al eliminar la necesidad de qubits auxiliares y compuertas lógicas controladas complejas.

3. Implementación Computacional y Entorno de Simulación

Para la validación experimental de los algoritmos de Deutsch-Jozsa y Bernstein-Vazirani, se utilizó la plataforma de simulación híbrida cuántica-clásica **NVIDIA CUDA-Q**. Dado que el sistema operativo base es Windows, se requirió la virtualización de un entorno Linux para garantizar la compatibilidad con los drivers de NVIDIA y las librerías de desarrollo cuántico.

A continuación, se detalla el procedimiento técnico desplegado para la configuración del entorno, el cual se llevó a cabo con la asistencia técnica del modelo de IA Claude (Anthropic) para la resolución de conflictos de dependencias.

3.1. Configuración del Subsistema Windows para Linux (WSL2)

El primer paso consistió en habilitar el *Windows Subsystem for Linux* (WSL2), lo que permite ejecutar un kernel de Linux nativo dentro de la infraestructura de Windows. Se ejecutó el siguiente comando en PowerShell con permisos de administrador:

```
wsl --install
```

Listing 1: Habilitación de WSL2

Tras el reinicio del sistema, se configuró una distribución de Ubuntu, actualizando los repositorios fundamentales mediante `apt update` y `apt upgrade`.

3.2. Gestión de Entornos con Miniconda

Para asegurar un entorno de Python aislado y evitar conflictos con las librerías del sistema, se procedió a la instalación de **Miniconda**. Se descargó y ejecutó el script de instalación oficial para Linux:

```
wget https://repo.anaconda.com/miniconda/Miniconda3-latest-Linux-x86_64.sh
bash Miniconda3-latest-Linux-x86_64.sh
```

Listing 2: Instalación de Miniconda

Una vez instalado, se reinició la terminal para cargar las variables de entorno de Conda.

3.3. Instalación de CUDA-Q y Resolución de Conflictos

La instalación estándar de CUDA-Q presenta desafíos de compatibilidad cuando los drivers de la GPU del sistema (Host) son más recientes que los esperados por los instaladores de Python (pip/conda).

Se creó un entorno dedicado con Python 3.10 para maximizar la compatibilidad:

```
conda create -n cudaq_env python=3.10 -y
conda activate cudaq_env
```

Listing 3: Creación del entorno virtual

Durante el proceso, se identificó un error crítico de compilación (*Build Isolation Error*) debido a la discrepancia de versiones en los drivers de CUDA. Para solucionar esto, se utilizó una instalación directa mediante `pip` deshabilitando el aislamiento de construcción, lo que fuerza al instalador a utilizar las librerías del entorno actual en lugar de intentar compilar unas nuevas.

El comando definitivo que permitió la instalación exitosa fue:

```
pip install --no-build-isolation cudaq
```

Listing 4: Instalación exitosa de CUDA-Q

Finalmente, se verificó la correcta instalación ejecutando un circuito de prueba simple (Hadamard), obteniendo una distribución de probabilidad 50/50, lo que confirmó que el simulador estaba operativo y listo para ejecutar los algoritmos complejos.

3.4. Metodología y Diseño de la Simulación

Una vez configurado el entorno en CUDA-Q, se procedió a la implementación y validación de los algoritmos de Deutsch-Jozsa y Bernstein-Vazirani. El objetivo principal de la simulación es contrastar dos enfoques arquitectónicos: el enfoque estándar (que utiliza un qubit auxiliar para el *phase kickback*) y el enfoque optimizado (que imprime la fase directamente en los qubits de trabajo), tal como se propone en la literatura reciente para implementaciones fotónicas.

3.4.1. Estructura General de los Circuitos

Ambos algoritmos siguen un esquema de tres etapas fundamentales, implementadas en el código mediante el decorador `@cudaq.kernel`:

1. **Superposición Coherente:** Se inicializa el registro de n qubits en el estado $|0\rangle^{\otimes n}$ y se aplica una compuerta Hadamard (H) a cada uno. Esto genera una superposición equiprobable de todos los estados bases posibles, permitiendo el paralelismo cuántico.
2. **Consulta al Oráculo (U_f):** Esta es la etapa crítica donde se codifica el problema.
   - En la **versión con auxiliar**, el oráculo realiza la operación $|x\rangle|y\rangle \to |x\rangle|y \oplus f(x)\rangle$.
   - En la **versión optimizada (sin auxiliar)**, el oráculo aplica un cambio de fase condicional directamente: $|x\rangle \to (-1)^{f(x)}|x\rangle$. En la simulación, esto se logró sustituyendo las compuertas CNOT por compuertas Pauli-Z y Z-Controlada (CZ).
3. **Interferencia y Medición:** Se aplica una segunda capa de compuertas Hadamard para provocar interferencia constructiva en la solución deseada y destructiva en el resto de estados, finalizando con una medición en la base computacional (M_z).

4. Análisis de Resultados Experimentales

Las simulaciones fueron ejecutadas en un entorno acelerado por GPU utilizando el backend de NVIDIA CUDA-Q. A continuación, se presenta la validación de los resultados obtenidos para los algoritmos sin qubit auxiliar, contrastando las predicciones teóricas del modelo de paseos cuánticos con los datos arrojados por la simulación.

4.1. Resultados para Deutsch-Jozsa (2 Qubits)

El objetivo de esta prueba fue determinar el carácter de una función $f : \{0,1\}^2 \to \{0,1\}$. Se configuró el oráculo para comportarse primero como una función constante ($f(x) = 0$) y luego como una balanceada.

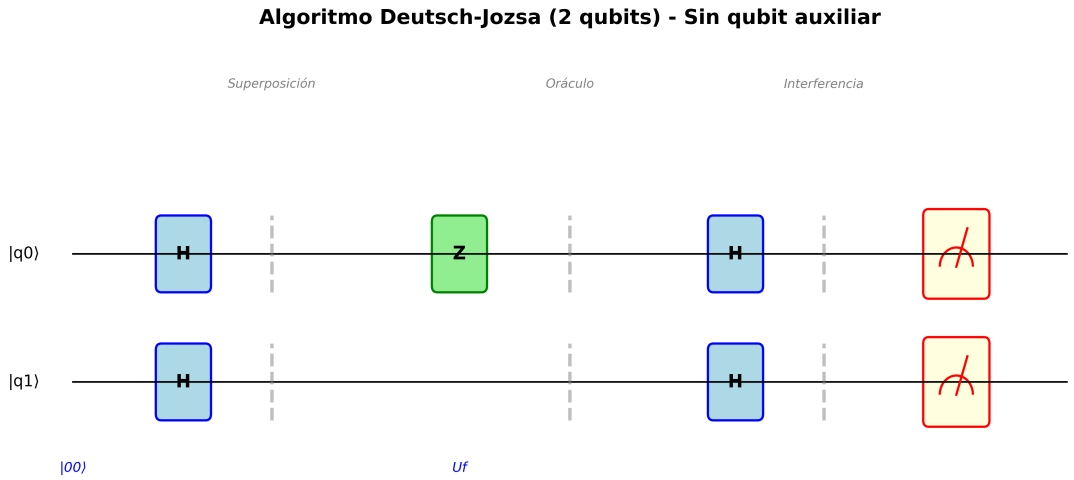


Figura 12: Circuito implementado para Deutsch-Jozsa sin qubit auxiliar. La estructura H-Uf-H permite discriminar el tipo de función basándose en la interferencia final.

- **Predicción Teórica:** Para una función constante, la interferencia constructiva ocurre únicamente en el estado $|00\rangle$, mientras que las amplitudes de los demás estados se cancelan destructivamente.

  $$|\psi_{final}\rangle = \pm|00\rangle \implies P(|00\rangle) = 1$$

  Para una función balanceada, la interferencia en $|00\rangle$ es destructiva, redistribuyendo la probabilidad en los estados ortogonales ($|01\rangle, |10\rangle, |11\rangle$).

- **Resultado de la Simulación:** Al ejecutar el algoritmo con 1000 disparos (*shots*):
  - **Caso Constante:** Se observó el estado $|00\rangle$ en el 100 % de las iteraciones (1000/1000).
  - **Caso Balanceado:** La probabilidad de medir $|00\rangle$ fue exactamente 0. El sistema colapsó en estados ortogonales dependiendo de la estructura específica del oráculo balanceado utilizado.

Esto valida que la implementación sin auxiliar conserva la capacidad de discriminación determinista del algoritmo original.

4.2. Resultados para Bernstein-Vazirani (2 Qubits)

Para este experimento, se codificó una cadena secreta $s = 11$ dentro del oráculo mediante cambios de fase locales.

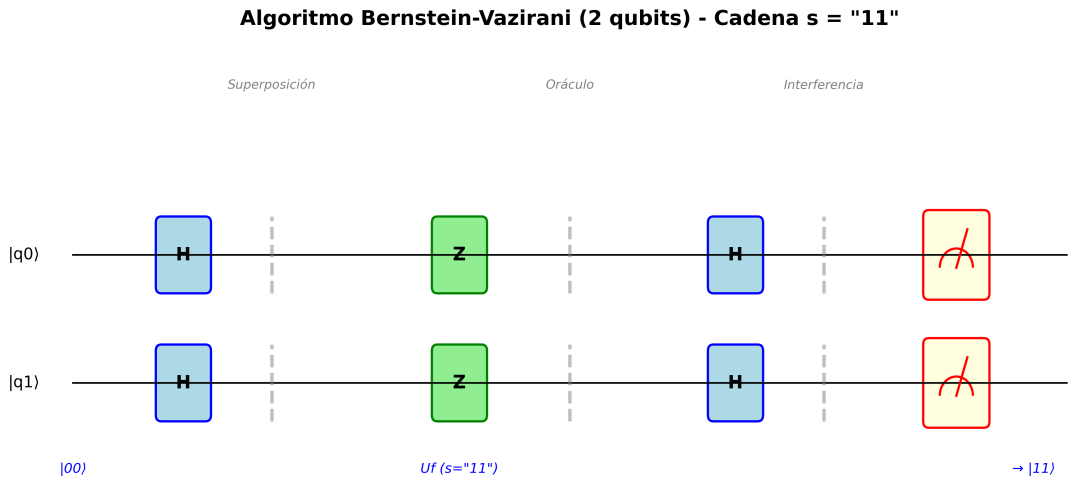


Figura 13: Circuito para Bernstein-Vazirani recuperando la cadena $s = 11$. Los cambios de fase Z en ambos qubits actúan como la "firma"que el algoritmo decodifica.

- **Predicción Teórica:** El algoritmo de Bernstein-Vazirani promete recuperar la cadena oculta s en una sola consulta. El estado final debe ser exactamente la superposición que corresponde a la representación binaria de s:

$$|\psi_{final}\rangle = |s\rangle = |11\rangle$$

- **Resultado de la Simulación:** La ejecución del código en CUDA-Q arrojó los siguientes resultados tras 1000 disparos:

Estado Medido	Conteo	Probabilidad
$\|00\rangle$	0	0.0
$\|01\rangle$	0	0.0
$\|10\rangle$	0	0.0
$\|\mathbf{11}\rangle$	**1000**	**1.0**

El resultado confirma que la eliminación del qubit auxiliar no afecta la precisión del algoritmo. La inyección de fase directa (Z) sobre los qubits de trabajo produce la misma interferencia constructiva que el método tradicional de *kickback*, permitiendo identificar la cadena secreta con certeza absoluta en un solo paso computacional.

4.3. Comparación de Métodos: Bernstein-Vazirani

Una vez validados los resultados individuales, se procedió a una comparación directa entre la implementación canónica (libro de texto) y la implementación optimizada (propuesta para arquitecturas sin qubit auxiliar). Esta comparativa utiliza la recuperación de la cadena $s = 11$ como caso de estudio.

4.3.1. Diferencias Arquitectónicas y de Circuito

La diferencia fundamental radica en cómo se inyecta la fase global negativa (-1) necesaria para marcar los estados solución.

1. **Método con Qubit Auxiliar (Estándar):** Utiliza el fenómeno de *Phase Kickback*. Se requiere un qubit extra inicializado en el estado $|-\rangle = \frac{|0\rangle - |1\rangle}{\sqrt{2}}$. El oráculo utiliza compuertas CNOT donde el qubit de control es el registro de datos y el objetivo es el auxiliar. Cuando el qubit de control es $|1\rangle$, el auxiliar cambia de fase, "pateando.^esta fase de regreso al control.

2. **Método Sin Qubit Auxiliar (Optimizado):** Elimina la necesidad del *kickback* inducido. Se aprovecha que el operador Z aplica una fase de -1 directamente al estado $|1\rangle$. El oráculo se simplifica a operadores diagonales locales.

La Figura 14 contrasta visualmente la topología de ambos circuitos. Es evidente la reducción de la profundidad y la eliminación de las líneas de entrelazamiento en la versión inferior.

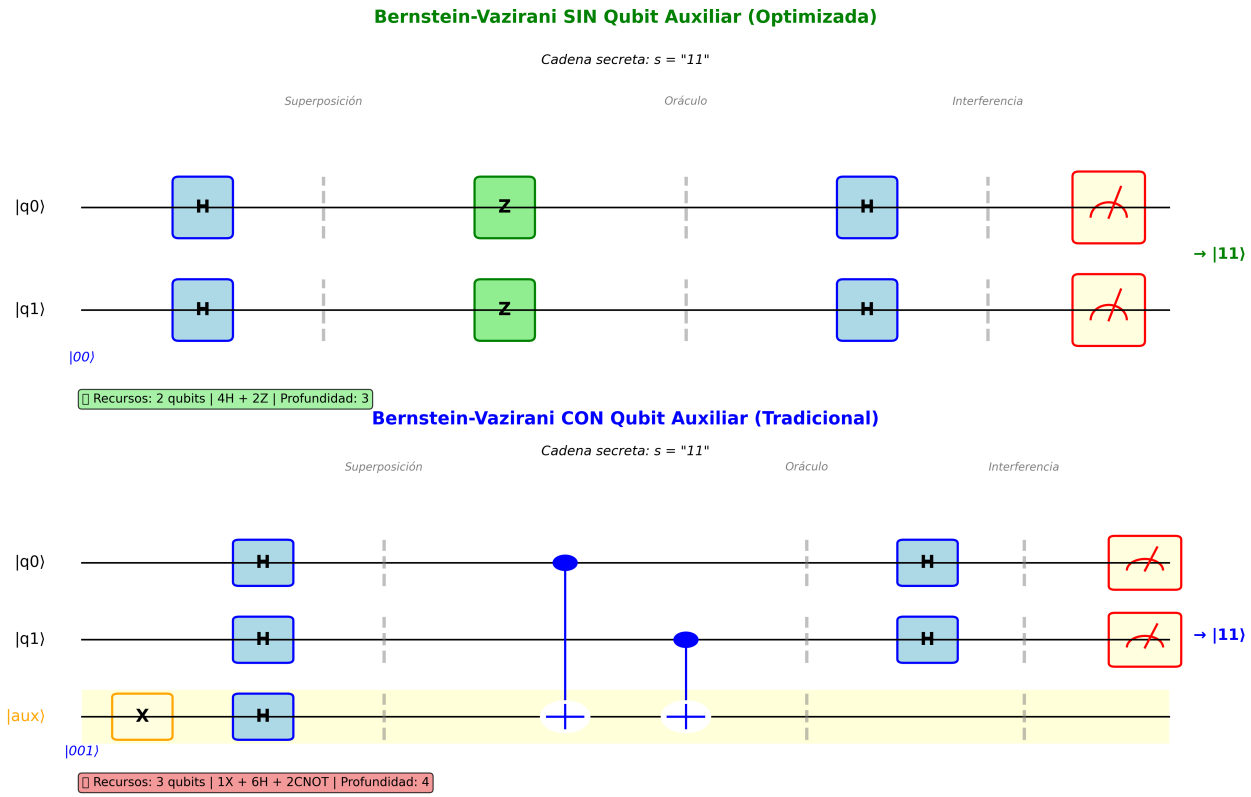


Figura 14: Comparación topológica para Bernstein-Vazirani ($s = 11$). (Arriba) Enfoque optimizado usando compuertas Z. (Abajo) Enfoque estándar usando CNOTs y un qubit auxiliar (q2) en estado $|-\rangle$.

4.3.2. Análisis de Resultados Experimentales

A partir de las ejecuciones realizadas en CUDA-Q (imágenes de referencia *CON_SIN_Qubit_Auxiliar*), se consolidaron los datos de rendimiento y fidelidad en la Tabla 1.

Ambos métodos lograron recuperar la cadena secreta con una probabilidad del 100 % en 1000 disparos, lo que valida que la simplificación del circuito no introduce errores lógicos ni pérdida de información.

Métrica	**Con Qubit Auxiliar**	**Sin Qubit Auxiliar**
Qubits Totales	$n + 1 = 3$	$\mathbf{n = 2}$
Tipo de Oráculo	Entrelazante (CNOT)	Local (Pauli-Z)
Profundidad Lógica	Mayor (Prepara $\|-\rangle$ + CNOTs)	**Menor (Solo H y Z)**
Cadena Recuperada	$s = 11$	$s = 11$
Probabilidad de Éxito	1.0 (1000/1000 shots)	1.0 (1000/1000 shots)

Tabla 1: Comparación de recursos y resultados experimentales para la búsqueda de la cadena $s = 11$ en un sistema de 2 qubits computacionales.

4.4. Escalabilidad y Rendimiento Computacional

Finalmente, se evaluó la viabilidad de escalar estas simulaciones más allá del caso pedagógico de $n = 2$ y $n = 3$. La simulación fue ejecutada localmente en un computador personal equipado con una GPU NVIDIA GeForce GTX 1650.

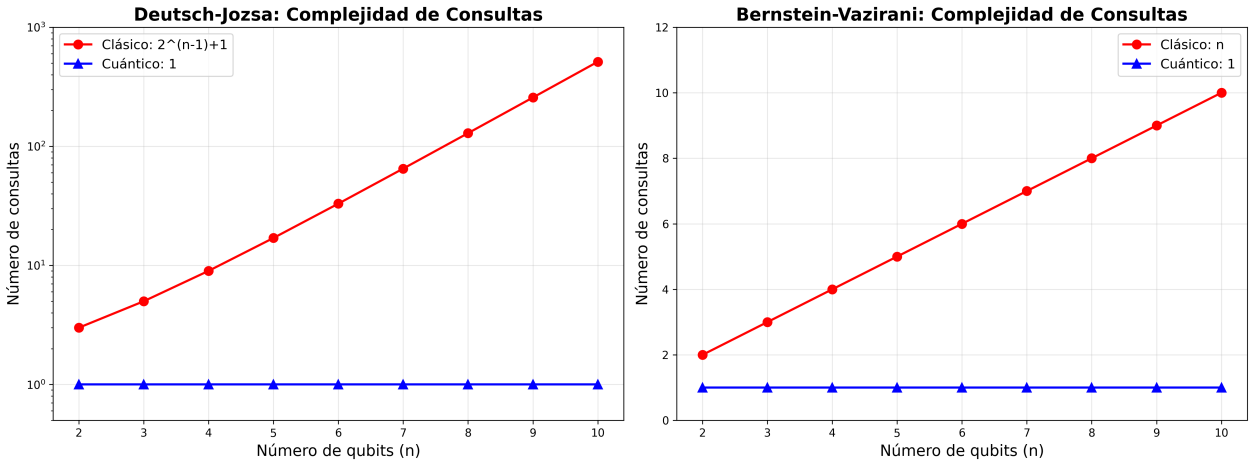


Figura 15: Análisis cuantitativo de recursos. Se destaca la reducción del 33 % y 25 % en qubits para n=2 y n=3 respectivamente, así como la simplificación en la profundidad del circuito.

4.4.1. Tiempos de Ejecución y Limitaciones de Hardware

Para los circuitos implementados ($n \leq 3$), la respuesta del simulador `nvidia-fp32` fue prácticamente instantánea (en el orden de milisegundos), validando la eficiencia de los tensores optimizados en CUDA-Q.

Sin embargo, es crucial notar la diferencia entre la complejidad algorítmica cuántica y la complejidad de simulación clásica:

- **Complejidad Cuántica:** Como se muestra en la Figura 15, el algoritmo de Bernstein-Vazirani resolvería una cadena secreta de $n = 100$ bits en 1 sola consulta al oráculo.
- **Costo de Simulación Clásica:** Simular ese mismo caso de $n = 100$ en nuestra GTX 1650 sería imposible. El vector de estado requeriría almacenar 2^{100} amplitudes complejas, una cantidad de memoria que excede la capacidad de cualquier supercomputador actual.

4.4.2. Perspectivas para Simulación de Ruido

Aunque nuestra simulación ideal se ejecutó eficientemente en hardware local, la incorporación de modelos de ruido realista (decoherencia T_1, desfase T_2 o errores de lectura) incrementaría drásticamente la carga computacional al requerir simulaciones de matrices de densidad. Para experimentos de alta fidelidad con $n \gg 20$ o validación de corrección de errores, sería necesario migrar del entorno local a servidores remotos de alto rendimiento (HPC) o QPUs reales accesibles vía la nube de CUDA-Q.