SudoQ

Ángel Gómez González, Alberto Martínez Gallardo, Javier López Roda

Indice

Algoritmo de Grover

El algoritmo de Grover es un algoritmo de búsqueda.
Lo que consigue este algoritmo es encontrar un objeto concreto dentro de un conjunto desordenado.

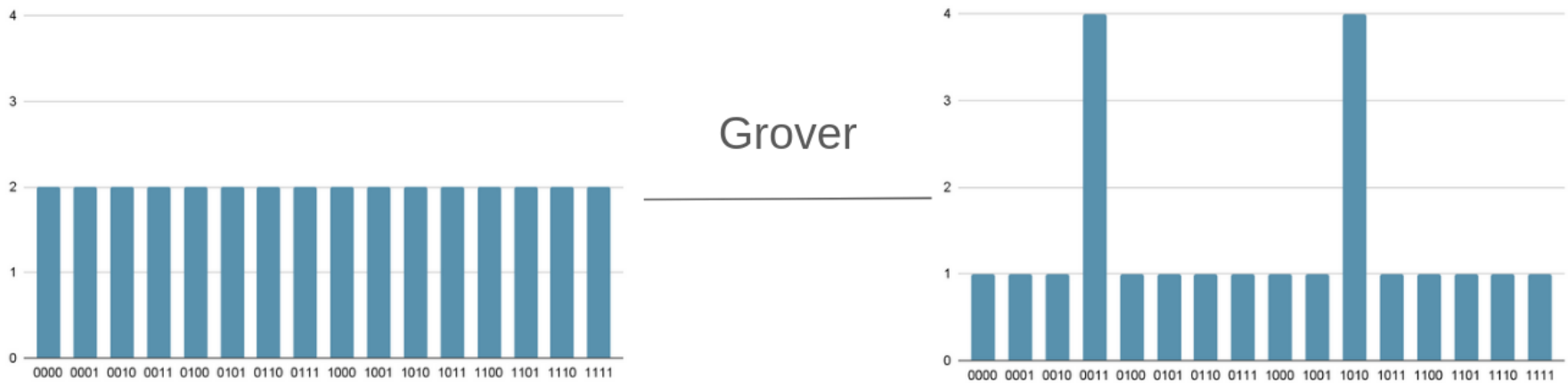


Figure: Funcionamiento del Algoritmo de Grover

Grover paso a paso

1. En primer lugar se aplica a todos los estados una puerta Hadamar, que incializa los estados de manera equiprobable.
2. Luego, aplica un Oracle para invertir el estado que se quiere buscar.
3. Finalmente, se emplea un circuito Reflector para amplificar la amplitud de ese estado.
4. Los pasos 2 y 3 se pueden repetir varias veces para obtener una mayor probabilidad en el estado correcto.

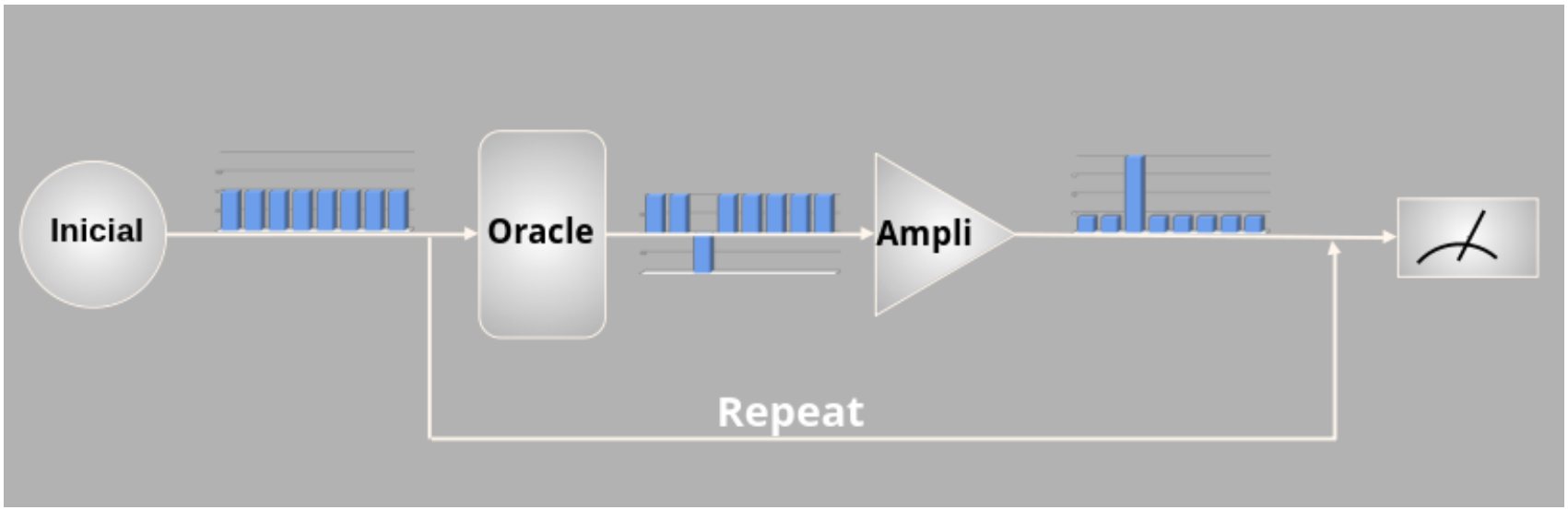


Figure: Esquema del Funcionamiento del Algoritmo

Resolución de un Sudoku

Un sudoku es un juego matemático que consiste en una malla de números y huecos. La manera de resolverlo es conociendo cuales son las condiciones que tienen que cumplir. Estas son:

- No se puede repetir el mismo número en una fila.
- No se puede repetir el mismo número en una columna
- No se puede repetir el mismo número en un bloque.

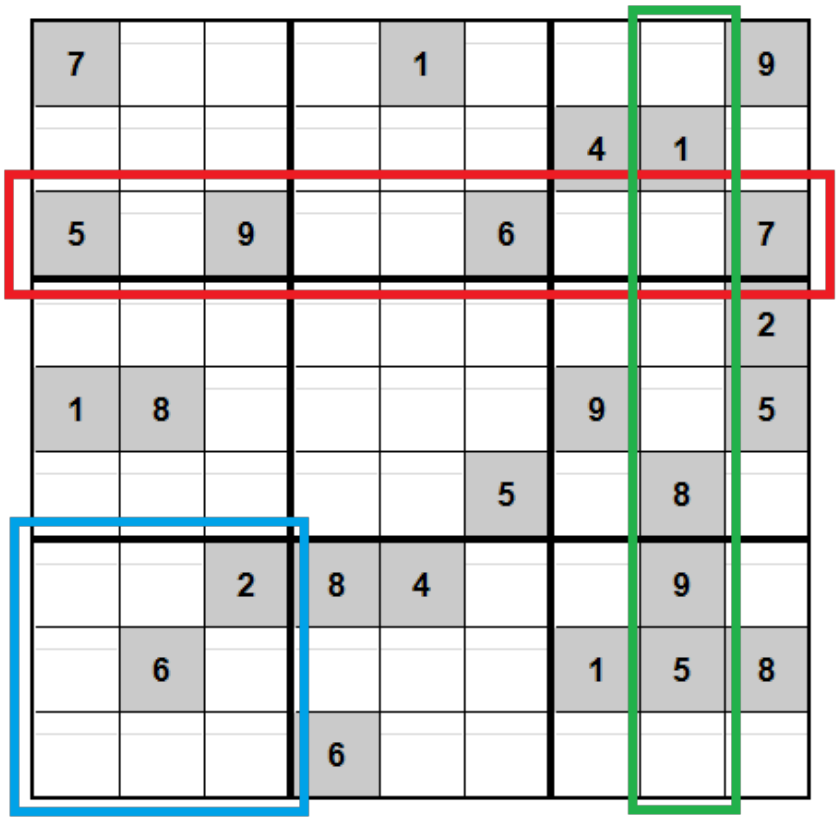


Figure: Condiciones de un Sudoku

Teoria del algoritmo

- Primer paso:

El primer paso que vamos a hacer es generar nuestro sudoku. Después de esto, tendremos que restarle 1 a todo el sudoku para que coincidir con el orden de conteo de python.

	1	3	2
3			4
	4		3
2		4	

Figure: Nuestro SudoQ

$\Rightarrow$

	0	2	1
2			3
	3		2
1		3	

Figure: SudoQ con índices 0,1,2,3.

Teoria del Algoritmo

- Segundo paso:

El segundo paso es codificar en binario los valores del sudoku. Donde tendremos:

- 0 - 00
- 1 - 01
- 2 - 10
- 3 - 11

Estos valores en binario, representan los valores que tendremos que obtener en nuestro circuito y con los que trabajaremos en el programa.

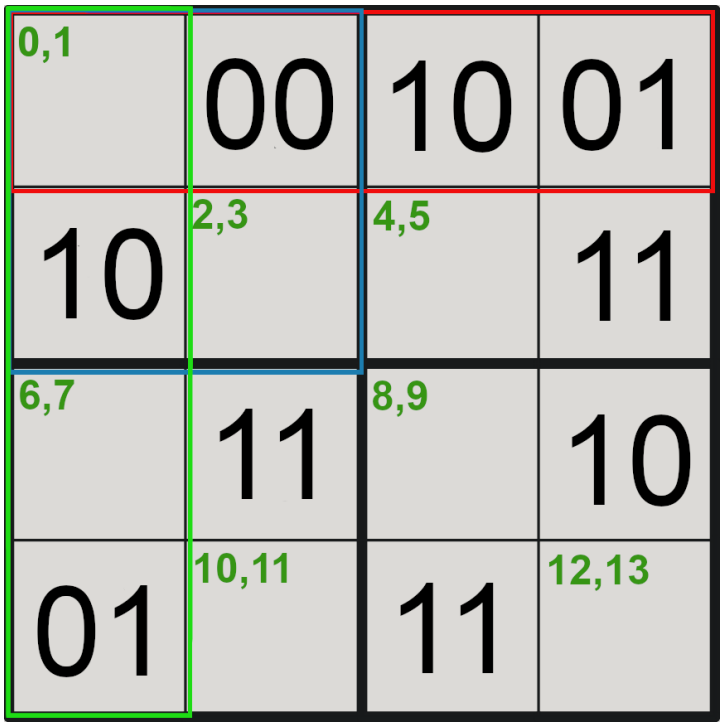
	00	10	01
10			11
	11		10
01		11	

Figure: SudoQ codificado en binario.

Teoría del algoritmo

- Tercer paso:

Codificamos también cada uno de los bits de los huecos y definimos las condiciones para cada uno de ellos. Entonces, aplicamos el algoritmo para cada uno de los huecos y sus condiciones.



- Si tenemos en cuenta la primera fila, vemos que hay un hueco que no puede ser 00, 10 o 01, por tanto ha de ser 11.
- Si tenemos en cuenta la columna, vemos que hay dos dos huecos y que estos no pueden ser 10 o 01. Además, estos huecos no pueden tener el mismo número entre ellos, por lo tanto los estados posibles son 1100 y 0011.

Teoría del algoritmo

- Cuarto paso:

Para interpretar el resultado de qiskit, primero tenemos que tener en cuenta que nos va a devolver un estado con todos los valores en él, dicho estado será

$$|0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13\rangle \tag{1}$$

No obstante, qiskit nos lo da al revés, es decir, que vamos a recibir el estado

$$|13, 12, 11, 10, 9, 8, 7, 6, 5, 4, 3, 2, 1, 0\rangle \tag{2}$$

Tras ejecutar el circuito obtenemos el siguiente estado más probable:

$$|00011000001011\rangle \xrightarrow{invertimos} |11010000011000\rangle \tag{3}$$

Teoría del algoritmo

$$|11\ 01\ 00\ 00\ 01\ 10\ 00\rangle$$

0,1 11	00	10	01
10	2,3 01	4,5 00	11
6,7 00	11	8,9 01	10
01	10,11 10	11	12,13 00

$\Rightarrow$

4	1	3	2
3	2	1	4
1	4	2	3
2	3	4	1

Figure: SudoQ resuelto en binario.

Figure: SudoQ resuelto.

Implementación en Qiskit

Veamos la implementación en qiskit.