emen ta zabal zazu
Universidad del País Vasco Euskal Herriko Unibertsitatea

UNIVERSIDAD DEL PAÍS VASCO
FACULTAD DE INFORMÁTICA

Practical work

Grover’s algorithm

Justinas Bliujus
jbliujus001@ikasle.ehu.eus

2025

Contents

1. FIRST SECTION

1.1. The task

The task is to understand what the Grover's algorithm achieves, how it works and implementing it in Qiskit.

1.2. The purpose of algorithm

Grover‘s algorithm was found in 1996. It is a searching algorithm with a speed of $O = \sqrt{N}$, where N is the number of elements, which is faster than classical algorithms.

So, assuming we would have a database of elements, as seen in Table 1, we can define the problem as: find x such that f(x) = 1.

Table 1

Element	0	0	....	1	...	0
Index	1	2	...	W	...	N-1

The searched element would be at index W.

Classical algorithms would find it $\frac{N}{2}$, so in $O = \sqrt{N}$ speed. Grover‘s algorithm seeks to improve that and finds the searched element faster.

1.3. Oracles

Oracle is a function, a quantum operation with a black-box principle, meaning we give it inputs and receive and output. Grover‘s algorithm uses two oracles – one for phase inversion and one for diffusion.

1.4. Grover‘s algorithm

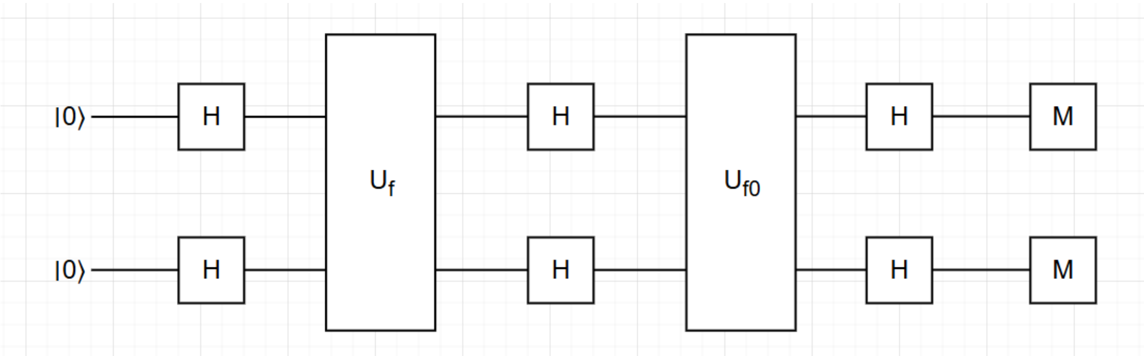

Figure 1

In Figure 1 we can see the full Grover's algorithm's quantum circuit.

- First step is to define starting states to |0⟩.
- Second step is to apply Hadamard gates to get a superposition on all the states. Thus, we get state $|s\rangle$. As in Equation 1.

Equation 1

$$H^{\otimes n}|0\rangle = \frac{1}{\sqrt{2^n}} \sum_{k\in\{0,1\}^n} |x\rangle = |s\rangle$$

Which means that all the states have equal probability as seen in Figure 2.

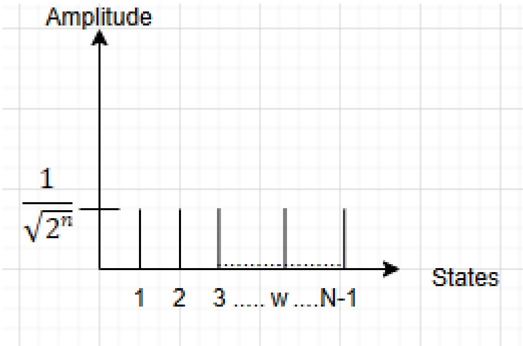

Figure 2

Let's define a space spanned by $|s\rangle$ and $|w\rangle$.

We define a perpendicular that is orthogonal to $|w\rangle$ as $\langle w^{\perp}|w\rangle = 0$

Inner product of the two is zero.

Equation 2

$$|w^{\perp}\rangle = \frac{\sqrt{2^n - 1}}{\sqrt{2^n}} \sum_{x\neq w} |x\rangle$$

Thus, it is a vector of all other values except one which is W as shown in Equation 2.

We can write $|s\rangle$ as a projection: $|s\rangle = \frac{\sqrt{2^n-1}}{\sqrt{2^n}}|w^{\perp}\rangle + \frac{1}{\sqrt{2^n}}|w\rangle = \cos\frac{\theta}{2}|w^{\perp}\rangle + \sin\frac{\theta}{2}|w\rangle =$ $\theta = 2\,\mathrm{asin}\frac{1}{\sqrt{2^n}}$

Then the angle between $|w^{\perp}\rangle$ and $|s\rangle$ is $\frac{\theta}{2}$ as shown in Figure 3.

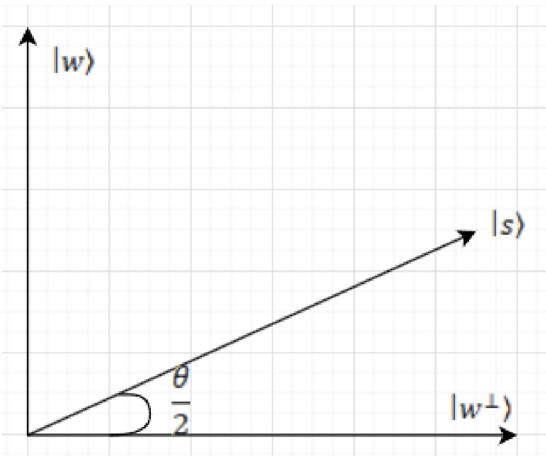

Figure 3

- Third step is the phase inversion, which means applying oracle U_f.

It is a black box principle and applying it gives:

$$U_f|w\rangle = -|w\rangle$$

$$U_f|x\rangle = |x\rangle$$

Which shifts the amplitude of the searched state as shown in Figure 4.

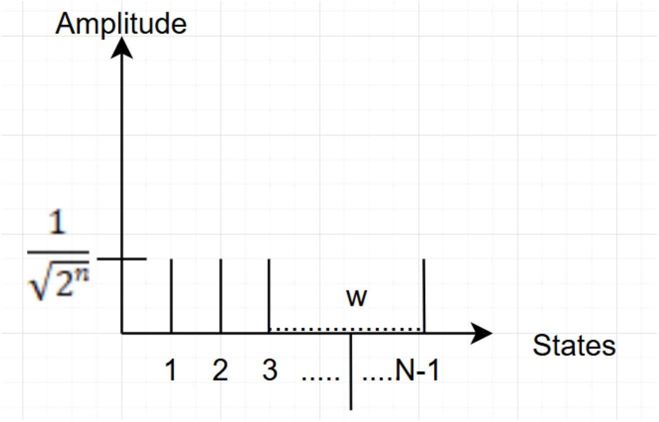

Figure 4

The oracle is defined as: $U_f = (I - 2\langle w|w\rangle)|s\rangle$. We take $|s\rangle$ projection on $|w\rangle$ twice, which results, that values of $|s\rangle$ are not changed, but $|w\rangle$ states are now negative as shown in Figure 5.

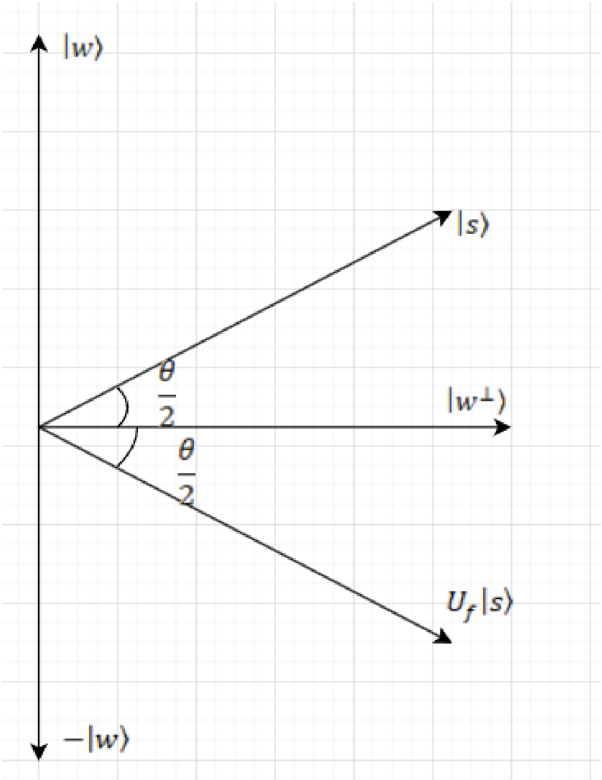

Figure 5

- Fourth step is inversion around the mean.

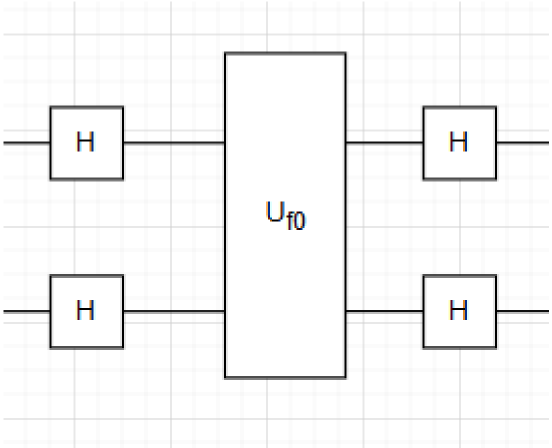

Figure 6

Let's denote the inversion part (Figure 6) as V. After applying it we get state $V.U_f|s\rangle$. V is a projection of $U_f|s\rangle$ onto $|s\rangle$. It is expressed as $V = 2\langle s|s\rangle - I$.

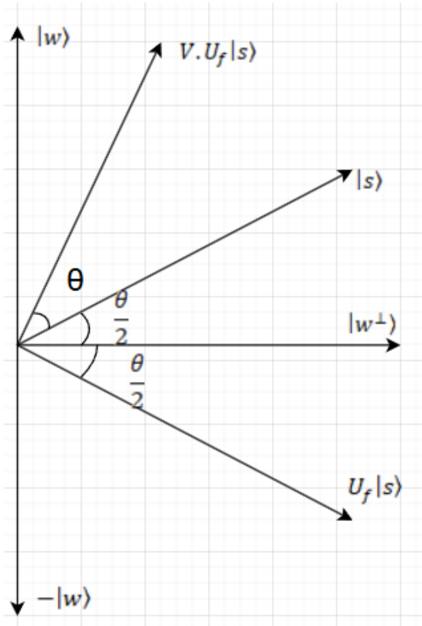

Figure 7

Then we get that current state is shifted by $\theta + \frac{\theta}{2}$ angle after one operation as shown in Figure 7. We repeat third and fourth steps r times. The state is shifted by θ r times, until it is aligned with $|w\rangle$. Since $\theta = 2asin\frac{1}{\sqrt{2^n}}$ and $r\theta + \frac{\theta}{2} = \frac{\pi}{2}$, so $r = \frac{\pi}{2\theta} - \frac{1}{2}$. Then N = 2^n value is very big, θ is very small and then $sin\theta \approx \theta$, then $r = \frac{\pi}{4}\sqrt{2^n}$ which is $O(\sqrt{N})$ iterations.

So, in the end we find the desired element in $O(\sqrt{N})$ iterations.

1.5. Implementing Grover's algorithm in Qiskit

To implement the algorithm, we need to design oracle and a diffuser. The chosen state we will search for is $|11\rangle$.

1.5.1. Making the oracle

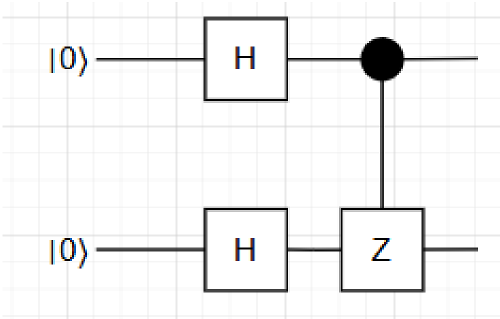

Figure 8

As depicted in Figure 8, the two qubits start in the state $|00\rangle$. Applying Hadamard gate transforms the qubits from $|0\rangle$ to the superposition state $|+\rangle$, which is $\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$. After applying Hadamard gate to both qubits the state will be $\frac{1}{2}(|00\rangle + |01\rangle + |10\rangle + |11\rangle)$. Then we apply CZ gate to both qubits. It applies a phase of -1 to the second qubit, if the first qubit is 1. So now our state is $\frac{1}{2}(|00\rangle + |01\rangle + |10\rangle - |11\rangle)$, marking the desired state by a phase shift.

1.5.2. Making the diffuser

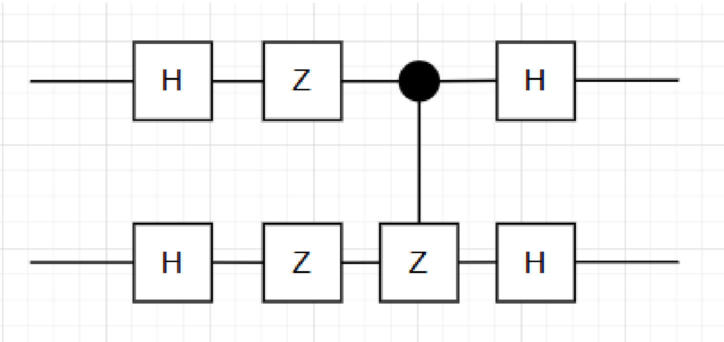

Figure 9

As shown in Figure 9, we apply Hadamard gates to transform qubits into superposition once again. Then, Z gates transform the state by performing phase shift and CZ gate making additional phase flip on $|11\rangle$ state, which is the target. The final Hadamard gate completes the inversion about the mean by amplifying the amplitude of the target state, while reducing the amplitude of other states.

1.6. Results

After running the implemented algorithm a hundred times, the results are (Figure 10) as follows: state $|01\rangle$ was measured 3% of the time. State $|10\rangle$ – 5% of the time and $|11\rangle$ – 95 % of the time, meaning that the desired results were achieved, since the target state is found with a high probability.

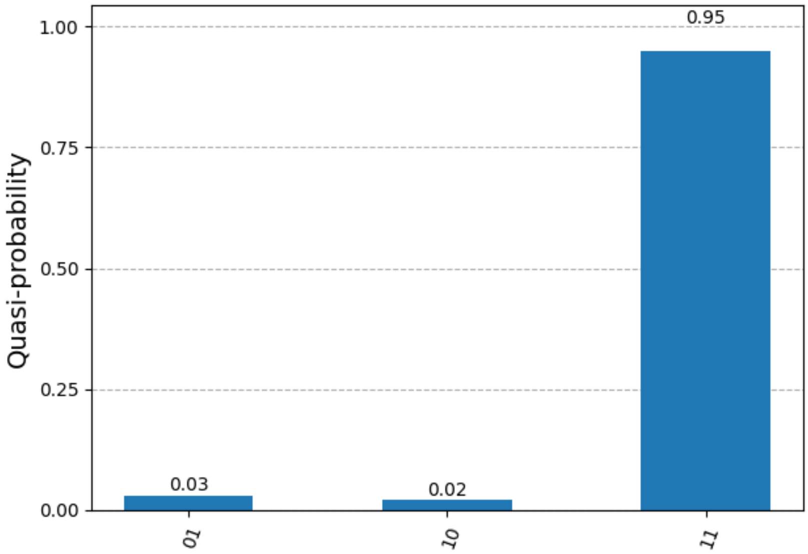

Figure 10

1.7. Conclusions

The Grover's algorithm successfully demonstrates a quantum speedup for search problems, achieving an $O = \sqrt{N}$ complexity compared to the classical an $O = N$. The implementation in Qiskit verifies theoretical predictions of finding the target state, since the target state was found with a high probability of 95%. The phase inversion oracle and inversion about the mean were implemented to work together to iteratively shift the state vector towards the desired solution.

Attachment 1. Code implementation.

https://github.com/JustinasBliujus/Grover-s-algorithm.git