Implementation of Deutsch-Jozsa Algorithm using Qiskit

Anup Khanal, Gaurab Subedi, Laxman Paudel, Pratyush Pradhan, Ronaj Pradhan, Siddhi Bajracharya, Unish Rajkarnikar
Department of Computer Science, University of South Dakota, Vermillion, SD 57069

***Abstract*—Deutsch-Jozsa (DJ) algorithm is a groundbreaking quantum computing algorithm that showcases the power of quantum parallelism. This report contains a comprehensive overview of the DJ algorithm encompassing its introduction, methodology, implementation, and results. The implementation of this algorithm was achieved through Qiskit, a popular quantum computing framework. In this report, we delve into the fundamental principles underlying the DJ algorithm, highlighting its innovative approach to determining whether a given binary function is balanced or constant. Furthermore, an in-depth explanation of the implementation of the DJ algorithm in Qiskit for a provided 5-bit qubit string, consisting of the steps involved and the circuitry required.**

***Index Terms*—Deutsch-Jozsa, quantum computing, quantum parallelism**

I. Introduction

Quantum computing has been in research and development for many years. From 1905 when Albert Einstein explained the photoelectric effect, till 2019 when Google claimed quantum supremacy, it has come a very long way [1]. The development of quantum computing systems focuses on solving complex problems which are too difficult to handle for classical computers. Well-known scientists and physicists such as Richard Feynman, Paul Benioff, and Yuri Manin have researched and worked on the development of systems for quantum computations. With all three researchers investigating the nature of algorithms that could be run on a quantum computer, a physicist at Oxford, David Deutsch, prepared a comprehensive framework for quantum computing in 1985. After working on an algorithm that would run faster on a quantum computer than a classical computer, Deutsch collaborated with Richard Jozsa, which allowed the algorithm to solve the problem in one step on a quantum computer while Deutsch's problem would take the worst-case O(n) on a classical computer [2].

The Deutsch-Jozsa (DJ) algorithm was introduced by Deutsch and Jozsa in 1992 [3]. The DJ algorithm is a special kind of math problem solver. It uses quantum computers, which are super-fast and powerful. The algorithm can tell if a function is balanced or not in just one try, while classical computers need many tries (2n-1+1 for n bit bitstring). It does this by using special quantum operations on quantum bits (qubits). However, it only works for certain types of problems and does not make all computations faster. Overall, the DJ algorithm is a crucial step in quantum computing and is one of the seminal quantum algorithms that exemplifies the potential advantage of quantum computing over classical computing for specific problem classes. It provides a clear demonstration of quantum parallelism and its ability to solve the DJ problem efficiently.

II. Methodology

The main aim of the DJ algorithm is to determine whether the given function yields 0 or 1 for the given n-bitstring. Consider the function f, which takes the n-bit string x. Suppose that we are promised f(x) is either a constant function that returns the same values (either 0 or 1) for all input x or a balanced function that returns an equal number of 1s and 0s. The goal then becomes how to decide whether the function is constant or balanced. If we were to solve this problem classically, for a n-bit string, we would have to perform $2^{n-1}+1$ evaluation in the worst case. Let us take an example of a function that takes in a 4-bit string. Classically, it would take us $2^{4-1}+1 = 9$ evaluations to confirm with 100% accuracy. We would start evaluating the function with all inputs starting from 0000, 0001, ... and so on.

$$f(0000) = 1, f(0001) = 1, ...and\ so\ on.$$

In the worst case, we will have to evaluate more than half of the possible inputs to accurately determine whether the function is constant or balanced. But in the best case, as few as 2 evaluations can solve this problem. We can reduce the number of evaluations but with the risk of accuracy. However, we can use quantum computers to solve this problem with just one evaluation.

We see that the DJ algorithm is designed to operate on a black-box function. This function can either be balanced or constant. [4]. The goal of the algorithm is to figure out which one it is without opening the box. As classical computers require multiple evaluations of the function to determine its nature, the quantum algorithm can accomplish this task in a single execution by using quantum parallel processing. By employing quantum gates, such as the Hadamard gate and an oracle function evaluation, the algorithm creates superposition and entanglement [4]. The detail of the implementation of the algorithm is explained in the Implementation section below.

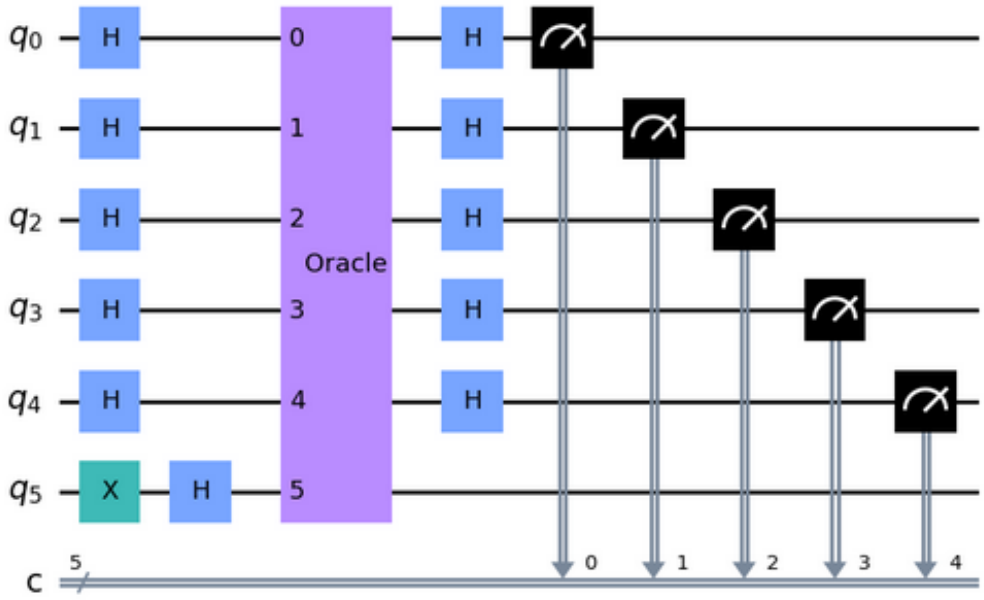

Fig. 1: 5-bit Deutsch-Josza algorithm implementation

III. Implementation

Implementing the DJ algorithm can be accomplished through various quantum computing platforms, including gate-based quantum computers or quantum simulators. The choice of implementation depends on the available hardware or software resources. Programming language frameworks such as Qiskit [6], an open-source software development kit (SDK), are commonly used to write code for executing algorithms on specific quantum platforms. We have implemented the DJ algorithm in Qiskit, as it allows developers to work with circuits, pulses, and algorithms. It also provides tools for creating, manipulating, and running quantum programs on IBM Quantum Experience [7] or local simulators. The following steps are carried out to implement the algorithm in Qiskit for a 5-bit string.

The steps of the algorithm are as follows:

1) Prepare two quantum registers. The first is a 4-qubit register initialized to $|0000\rangle$, and the last is a one-qubit register initialized to $|1\rangle$.
2) Apply a Hadamard gate to each qubit.
3) Apply the quantum oracle. The oracle returns either a constant or balanced function.
4) Apply a Hadamard gate to each qubit in the first register.
5) Measure the first register. The probability measures to 1 if the function is constant and 0 if the function is balanced.

In the implementation using Qiskit, two functions have been created:

```
def dj_oracle(case, n):
```

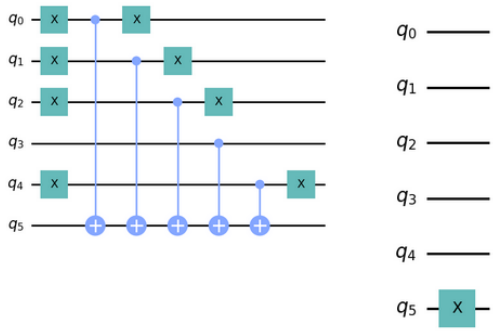

Fig. 2: From left to right. Balanced oracle and constant oracle.

This is an oracle function that generates the random circuit (in the figure). It takes two parameters: case and n. Case determines which circuit to return, whether constant or balanced, whereas n determines the number of bits in the bit string.

```
def dj_algorithm(oracle, n)
```

This is the implementation of the actual DJ algorithm. The algorithm is inspired by Qiskit's official documentation.

IV. Results and Discussion

The experiment for each type of oracle (either constant or balanced) is run 1024 times which is the default number of shots in the measurement method for the quantum circuits in Qiskit. We measure the first n registers of the circuit. For a constant oracle, the measurement

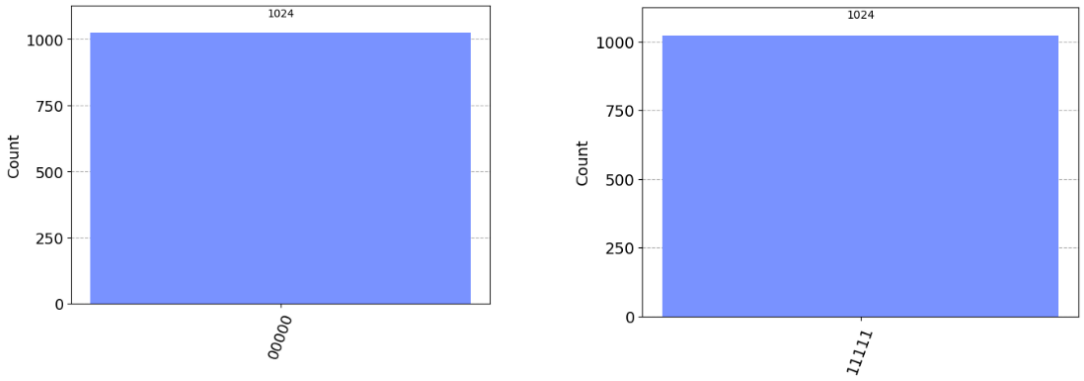

Fig. 3: From left to right. Measurement for constant oracle and balanced oracle.

probability is 100% for 00000, and for a balanced oracle, the measurement probability is 100% for 11111.

The implementation of the DJ algorithm showed the superiority of the DJ algorithm over classical algorithms. By employing a single function evaluation, regardless of the function's complexity or input size, the algorithm efficiently determines whether the black-box function is constant or balanced. The DJ algorithm's ability to provide a solution with a single-function evaluation is a remarkable achievement. It illustrates the inherent advantage of quantum parallelism, where multiple computations are simultaneously performed. However, it is important to note that the algorithm's scope is limited to solving specific types of problems and does not offer a general speedup for all computational tasks.

V. Conclusion

This work demonstrates that the DJ algorithm has reached a significant milestone in the advancement of quantum computing. It showcases the potential of quantum parallelism. It also shows the potential of achieving exponential speedup in targeted problem domains. While further research is necessary to fully unlock the potential of quantum computing, the DJ algorithm serves as a foundation for future developments in the field of quantum computing.

References

[1] Press, G. (2021, May 18). 27 milestones in the history of quantum computing. Forbes. Retrieved from https://www.forbes.com/sites/gilpress/2021/05/18/27-milestones-in-the-history-of-quantum-computing/?sh=610e61197b23

[2] Hidary, J.D. (2021). A Brief History of Quantum Computing. In: Quantum Computing: An Applied Approach. Springer, Cham. https://doi.org/10.1007/978-3-030-83274-2_2

[3] Deutsch, D., Jozsa, R. (1992). Rapid solution of problems by quantum computation. Proceedings of the Royal Society of London. Series A: Mathematical and Physical Sciences, 439(1907), 553-558.

[4] Nielsen, M. A., Chuang, I. L. (2010). Quantum Computation and Quantum Information. Cambridge University Press.

[5] Cleve, R., Ekert, A., Macchiavello, C., Mosca, M. (1998). Quantum algorithms revisited. Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences, 454(1969), 339-354.

[6] Qiskit: An Open-source Framework for Quantum Computing, 2023, 10.5281/zenodo.2573505

[7] "IBM Quantum." (n.d.). Retrieved from https://www.ibm.com/quantum/