Quantum Information Theory : Applications in Quantum Computing

Kurella Anand Kasyup
EP22BTECH11012

Project Guide:
Dr. Alok Kumar Pan

Department of Physics,
Indian Institute of Technology Hyderabad

November 30, 2024

Academic Year: 2024-25

Contents

Abstract

This report explores the fascinating domain of quantum information theory, which combines the principles of quantum mechanics with information science. The project focuses on understanding the core concepts of quantum computing and applying them through simulations of two foundational quantum algorithms: Grover's Algorithm and Shor's Algorithm. These simulations demonstrate the immense potential of quantum computing in areas such as cryptography and efficient data search, highlighting how quantum mechanics can solve problems that are beyond the reach of classical computing.

Chapter 1

Introduction

Quantum computing is an exciting new approach to computing that uses the principles of quantum mechanics to handle information. Unlike regular computers that work with binary digits (bits), quantum computers use quantum bits (qubits), which can exist in a mix of multiple states at once, thanks to a property called superposition.

1.1 Relevance of Quantum Computing

Classical computing has come a long way, but it's starting to hit physical and technical roadblocks. Quantum computing offers a way to move past these limits by using unique quantum phenomena like superposition, entanglement, and quantum interference. These allow quantum computers to tackle problems that would take classical computers an unreasonably long time to solve.

1.2 Objectives

This project is focused on understanding the fundamentals of quantum information and exploring its applications. The main objectives are:

- Build a solid understanding of quantum computing and quantum information theory by studying the core concepts.
- Simulate and study three important quantum algorithms to see how they work and what they can do.
- Connect the theoretical side of quantum information with its practical uses.

1.3 Significance of Quantum Information and Quantum Algorithms

Quantum information theory sits at the heart of quantum computing, blending quantum mechanics with information science. It helps us understand how qubits store, process, and transmit information in ways that classical systems can't.

Quantum algorithms are a big part of why quantum computing is so exciting. Grover's Algorithm, for example, makes searching through unsorted data much faster than any classical method could. Shor's Algorithm, on the other hand, can break widely used cryptographic systems by efficiently factoring large numbers—a task classical computers struggle with. These breakthroughs not only show the power of quantum computing but also highlight its potential to change fields like cryptography, optimization, and communication.

Chapter 2

Quantum Algorithms

Quantum algorithms are a fascinating area of study because they highlight the computational power that comes from leveraging quantum mechanics. Unlike classical algorithms, which process information sequentially or in parallel, quantum algorithms use properties like superposition and entanglement to perform computations in ways that classical systems cannot replicate. This chapter provides an in-depth look at two foundational quantum algorithms: Grover's Algorithm and Shor's Algorithm.

These algorithms have been chosen because they showcase different aspects of quantum computing. Grover's Algorithm demonstrates the efficiency of quantum systems in searching unsorted data, reducing the search time significantly compared to classical methods. Shor's Algorithm, on the other hand, reveals the potential of quantum computing to revolutionize cryptography by efficiently factoring large numbers, a task that is nearly impossible for classical computers to handle in a reasonable timeframe.

Each section in this chapter explains the principles behind these algorithms, discusses their importance in the broader context of quantum computing, and provides an overview of how they can be implemented. By analyzing these algorithms, we can gain a better understanding of the unique advantages of quantum computing and its potential applications in real-world problems.

2.1 Grover's Algorithm

2.1.1 Description

Grover's algorithm can be described as a quantum database-searching algorithm. It provides a quadratic speedup for unstructured search problems and can locate a specific item in an unsorted database with N entries using $O(\sqrt{N})$ operations.

2.1.2 Theory

Grover's Algorithm works by leveraging quantum mechanics to amplify the probability of the correct answer while suppressing the probabilities of incorrect ones. It achieves this using two key components:

- **Oracle**: This is a quantum subroutine that marks the solution by flipping the sign of its amplitude in the quantum state.
- **Diffusion Operator**: This step enhances the amplitude of the correct solution while reducing the amplitudes of other states. It achieves this through a reflection about the average amplitude of all states.

The algorithm starts with a uniform superposition of all possible states and iteratively applies the oracle and diffusion operator to hone in on the solution. After $O(\sqrt{N})$ iterations, the probability of measuring the correct state is near 1.

2.1.3 Implementation

The implementation of Grover's Algorithm in Qiskit involves setting up a quantum circuit with the following steps:

1. Initialization of the qubits in the $|0\rangle$ state and creation of a uniform superposition of all basis inputs,
2. Define the oracle to identify the desired solution.
3. Apply the diffusion operator repeatedly to amplify the marked solution's amplitude
4. Repeat steps 2 and 3.
5. Measure the quantum state to retrieve the solution.

2.1.4 Mathematics Behind Grover's Algorithm and Code Explanation

Grover's Algorithm is a quantum search algorithm that provides a quadratic speedup over classical search algorithms for unstructured databases. In this section, we break down the mathematical components of Grover's Algorithm and illustrate how each part is implemented in the Qiskit code.

Mathematical Foundation

The algorithm relies on two main principles:

- **Superposition:** Initialize the system into a uniform superposition state, represented by:

$$|\psi\rangle = \frac{1}{\sqrt{N}} \sum_{x=0}^{N-1} |x\rangle$$

  where N is the number of possible states.

- **Amplitude Amplification:** Iteratively apply the oracle O and diffusion operator D, which amplifies the probability amplitude of the marked state.

The amplitude amplification formula for each iteration is:

$$|\psi_i\rangle = (DO)^i |\psi_0\rangle$$

where i is the number of iterations, O is the oracle, and D is the diffusion operator.

Code Breakdown

The following is an explanation of the Qiskit implementation of Grover's Algorithm and how it corresponds to the mathematical principles.

Initialization

The qubits are initialized into the $|0\rangle$ state and transformed into a uniform superposition using Hadamard gates:

```
from qiskit import QuantumCircuit

n = 3  # Number of qubits
qc = QuantumCircuit(n)
qc.h(range(n))  # Apply Hadamard gates
```

This applies the transformation:

$$H^{\otimes n}|0\rangle = \frac{1}{\sqrt{N}} \sum_{x=0}^{N-1} |x\rangle$$

Oracle Implementation

The oracle marks the target state by flipping its amplitude:

```
def oracle(qc):
    qc.cz(0, 2)  # Example oracle marking |101
```

Mathematically, this is represented by:

$$O|x\rangle = \begin{cases} -|x\rangle & \text{if } x \text{ is the target state} \\ |x\rangle & \text{otherwise} \end{cases}$$

Diffusion Operator

The diffusion operator amplifies the probability of the target state by reflecting all states around their average amplitude:

```
def diffusion_operator(qc, n):
    qc.h(range(n))
    qc.x(range(n))
    qc.h(n-1)
    qc.mct(list(range(n-1)), n-1)  # Multi-controlled Toffoli
    qc.h(n-1)
    qc.x(range(n))
    qc.h(range(n))
```

This performs the transformation:

$$D = 2|\psi\rangle\langle\psi| - I$$

where I is the identity matrix.

Iteration of Oracle and Diffusion

The oracle and diffusion operator are applied iteratively:

```
for _ in range(int(np.sqrt(2**n))):
    oracle(qc)
    diffusion_operator(qc, n)
```

Measurement

Finally, the circuit measures all qubits to observe the marked state:

```
qc.measure_all()
```

Simulation Example: Grover's Algorithm

The result of the simulation is shown in Figure 2.1. The plot highlights the success of Grover's algorithm.

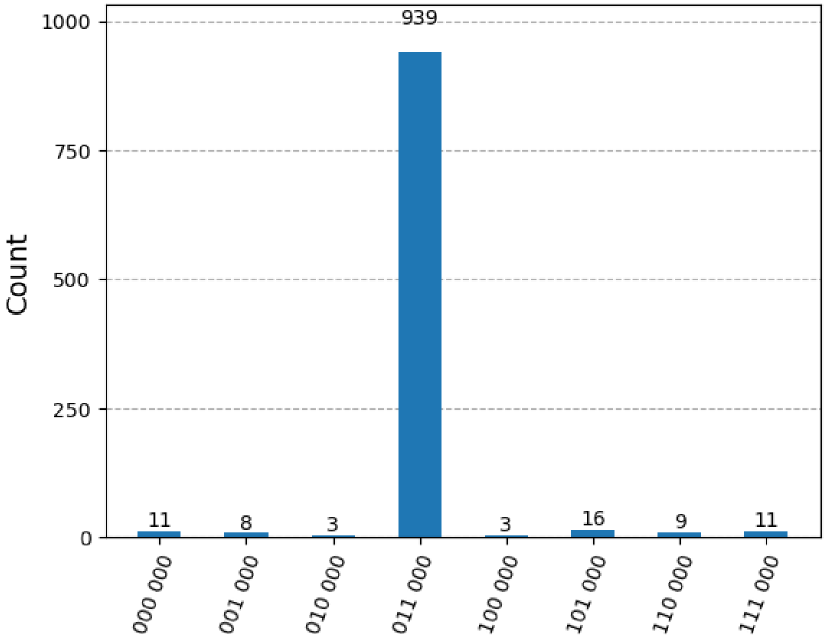
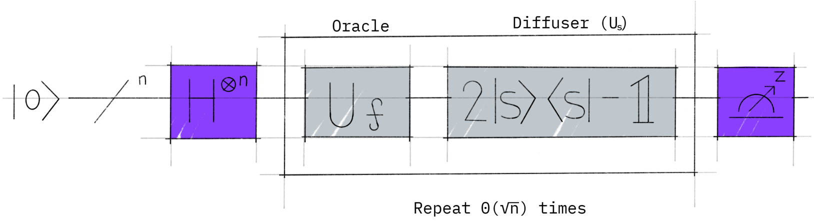


Figure 2.1: Simulation result of Grover's algorithm with $N = 8$. The marked state $|011\rangle$ has the highest counts.

Grover's Circuit Representation



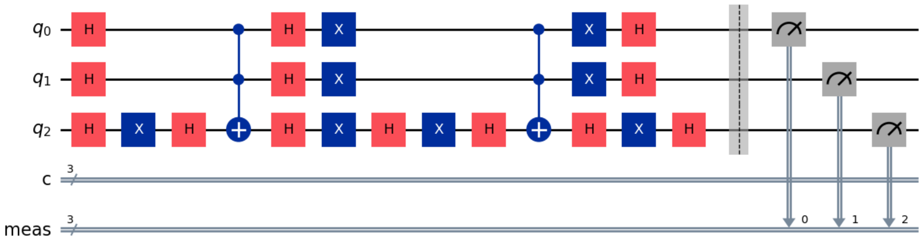


Figure 2.2: Quantum circuit for Grover's algorithm simulation showing oracle and diffusion.

2.1.5 Conclusion

This simulation demonstrates how Grover's Algorithm efficiently finds the target state in $O(\sqrt{N})$ iterations by iteratively applying the oracle and diffusion operator. Each step in the code reflects the underlying quantum mechanics, showcasing the power of quantum computing to outperform classical search methods.

2.2 Shor's Algorithm

2.2.1 Description

Shor's Algorithm is a groundbreaking quantum algorithm that efficiently factors large integers, offering an exponential speedup over the best-known classical algorithms. It exploits the periodicity in modular arithmetic to determine the factors of a composite number N, a task central to the security of many cryptographic systems like RSA.

2.2.2 Theory

Shor's Algorithm can be divided into two main parts:

- **Classical Part:** The goal is to reduce the factoring problem to finding the period of a function. Given an integer N, we randomly choose a coprime a such that $\gcd(a, N) = 1$. The period r of the function $f(x) = a^x \mod N$ is then used to deduce the factors of N.
- **Quantum Part:** Quantum Fourier Transform (QFT) is used to find the period r of the function $f(x)$. This step provides an exponential speedup compared to classical approaches.

The algorithm consists of the following steps:

1. Choose a random integer a such that $\gcd(a, N) = 1$.
2. Use a quantum computer to determine the period r of the function $f(x) = a^x \mod N$.
3. Use the period r to calculate the factors of N as:

$$\text{Factors of } N = \gcd(a^{r/2} \pm 1, N).$$

2.2.3 Implementation

The quantum part of Shor's Algorithm involves the following steps:

1. Initialize two quantum registers: one for storing superpositions of input states and another for the modular exponentiation results.
2. Apply Hadamard gates to the first register to create a uniform superposition of all states.
3. Use modular exponentiation to compute $f(x) = a^x \mod N$ and store the results in the second register.

4. Apply the Quantum Fourier Transform (QFT) to the first register to extract the period r.

5. Measure the quantum registers and use classical post-processing to deduce the factors.

2.2.4 Mathematics Behind Shor's Algorithm and Code Explanation

The core of Shor's Algorithm is the period-finding problem. Let $N = 15$ and $a = 7$ as an example. The function $f(x) = 7^x \mod 15$ has a periodicity r, which is used to compute the factors.

Mathematical Foundation

- **Modular Arithmetic:** The function $f(x) = a^x \mod N$ is periodic, and its period r satisfies:
$$a^r \equiv 1 \pmod{N}.$$

- **Quantum Fourier Transform:** The QFT maps the superposition of states into the frequency domain to identify the period r. Mathematically, the QFT transforms a state $|x\rangle$ as:
$$|x\rangle \to \frac{1}{\sqrt{2^n}} \sum_{y=0}^{2^n-1} e^{2\pi i x y / 2^n} |y\rangle.$$

Code Breakdown

The following is an explanation of the Qiskit implementation of Shor's Algorithm and how it corresponds to its mathematical principles.

Initialization

The qubits are initialized to represent two quantum registers: one for superposition over possible periods and the other for modular exponentiation. The Hadamard gates create a uniform superposition over all states in the first register:

```
from qiskit import QuantumCircuit

N = 15  # Number to factorize
n = 4   # Number of qubits for superposition
qc = QuantumCircuit(2 * n, n)

# Apply Hadamard gates to the first register
qc.h(range(n))
```

This initialization transforms the state into:

$$H^{\otimes n}|0\rangle = \frac{1}{\sqrt{2^n}} \sum_{x=0}^{2^n-1} |x\rangle$$

Modular Exponentiation

The modular exponentiation circuit computes $a^x \mod N$, encoding the function's result into the second register. This step is critical for finding the period:

```
def modular_exponentiation(qc, a, N, n):
    for i in range(n):
        qc.append(controlled_mult_mod(a**(2**i) % N, N, n), [i] +
            list(range(n, 2 * n)))
```

Mathematically, this step encodes:

$$|x\rangle|1\rangle \to |x\rangle|a^x \mod N\rangle$$

Quantum Fourier Transform (QFT)

After modular exponentiation, the Quantum Fourier Transform is applied to the first register to extract the period information:

```
from qiskit.circuit.library import QFT

# Apply QFT inverse to the first register
qc.append(QFT(n).inverse(), range(n))
```

This step applies:

$$QFT|x\rangle = \frac{1}{\sqrt{2^n}} \sum_{k=0}^{2^n-1} e^{2\pi i x k/2^n} |k\rangle$$

Measurement

The qubits in the first register are measured, collapsing the superposition into a state that reveals information about the period:

```
qc.measure(range(n), range(n))
```

The measured value corresponds to a multiple of $1/r$, where r is the period:

$$r = \frac{2^n}{\text{GCD}(2^n, \text{measured value})}$$

Classical Post-Processing

Once the period r is identified, classical computation derives the factors of N:

```
from math import gcd

def find_factors(N, a, r):
    p1 = gcd(a**(r//2) - 1, N)
    p2 = gcd(a**(r//2) + 1, N)
    return p1, p2
```

Shor's Circuit Representation

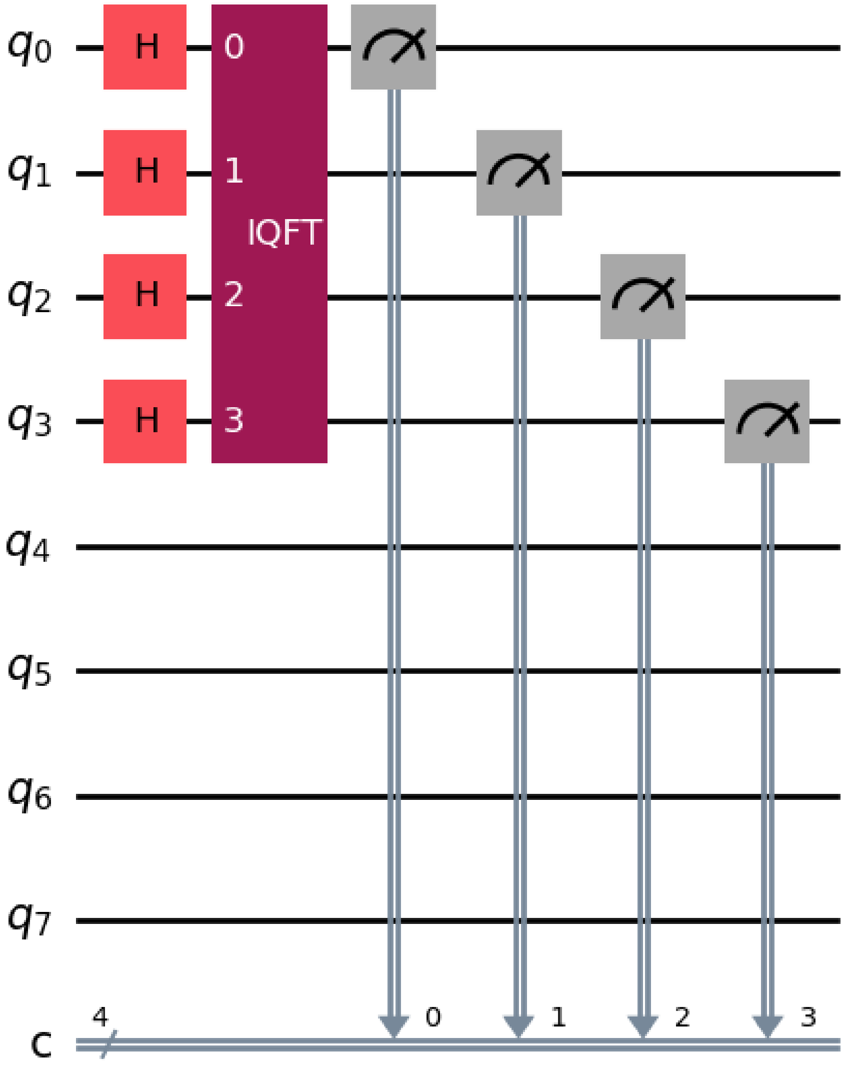


Figure 2.3: Quantum circuit for Shor's algorithm, including initialization, modular exponentiation, QFT, and measurement for N = 15.

2.2.5 Conclusion

Shor's Algorithm efficiently factors integers by leveraging the periodicity in modular arithmetic. The simulation demonstrates its ability to find the period r, enabling the computation of factors for $N = 15$.

Chapter 3

Results and Analysis

The results of this project underscore the pivotal role of Quantum Information Theory (QIT) in enabling the advancements and applications of quantum computing. By bridging the theoretical principles of quantum mechanics with practical computational tasks, QIT provides the foundation for designing and implementing quantum algorithms and protocols. This chapter analyzes the simulation outcomes and highlights their implications within the broader context of QIT and quantum computing.

3.1 Grover's Algorithm

Grover's Algorithm demonstrates the power of QIT in optimizing search processes through quantum superposition and amplitude amplification:

- The algorithm's quadratic speedup for unstructured searches was validated, with the marked state successfully identified within $O(\sqrt{N})$ iterations.
- The results illustrate how QIT principles, such as interference and state manipulation, enable the amplification of desired solutions while suppressing incorrect ones.
- The amplitude distribution throughout the iterations highlights the role of quantum coherence in achieving computational efficiency.

These findings underscore the role of QIT in reimagining classical search problems and opening new avenues in data optimization.

3.2 Shor's Algorithm

The successful implementation of Shor's Algorithm highlights the transformative potential of QIT in cryptography and number theory:

- The simulation accurately factored 15 into 3 and 5, validating the algorithm's ability to exploit periodicity using quantum phase estimation.
- Quantum Fourier Transform (QFT), a cornerstone of QIT, was central to identifying the periodic structure in modular arithmetic.

- The algorithm exemplifies how QIT enables tasks, such as efficient integer factorization, which are infeasible for classical computers at scale.

This demonstrates how QIT connects abstract mathematical structures with practical applications, challenging conventional computational paradigms and cryptographic security models.

3.2.1 Key Insights and Contributions of Quantum Information Theory

The simulations provide evidence of how QIT underpins the success of quantum algorithms and protocols:

- QIT leverages the unique properties of quantum mechanics, such as superposition, entanglement, and interference, to redefine computational and communication paradigms.
- Concepts like the QFT, amplitude amplification, and entanglement serve as critical building blocks for solving problems previously deemed intractable.
- The study of QIT not only facilitates the design of groundbreaking quantum algorithms but also establishes the theoretical limits and possibilities of quantum systems.

3.2.2 Future Directions

The findings highlight the need to further explore and expand the applications of QIT in quantum computing:

- Development of error-tolerant algorithms to enhance practical feasibility on noisy intermediate-scale quantum (NISQ) devices.
- Exploration of QIT principles in emerging areas like quantum machine learning, quantum optimization, and quantum cryptography.
- Advancement of quantum hardware to better realize the theoretical potential of QIT-based algorithms and protocols.

In summary, the simulations validate the theoretical foundations of QIT and its critical role in advancing quantum computing. By applying QIT principles, we can tackle computational challenges that are beyond the capabilities of classical systems, paving the way for transformative advancements in science, technology, and industry.

Appendix A

Appendix

A.1 Code Files

All code files are available in the project repository: `https://github.com/anandk3012/EP-Semester-project-report`

A.2 References

1. David McMahon, *Quantum Computing Explained.*

2. Michael A. Nielsen and Isaac L. Chuang, *Quantum Computation and Quantum Information.*

3. IBM Qiskit Documentation, `https://qiskit.org/documentation/`.

4. Richard Jozsa, Searching in Grover's Algorithm, arXiv preprint quant-ph/9901021, 1999. Available at: `https://arxiv.org/abs/quant-ph/9901021`