مجلة ساحل المعرفة للعلوم الإنسانية والتطبيقية
Sahel Almarifah Journal of Humanities and Applied Sciences

تصدر عن الاكاديمية الليبية فرع الساحل الغربي
المجلد الثاني–عدد خاص-S1
الصفحات (E-54 - E-49)

Quantum computing

M. M Deban[1]*, M. M Abdo-Alslam[2]
[2,1]Department of Physics, Faculty of Education, Qaser Bin Ghashir, University of Tripoli, Libya
mabroukadeban@gmail.com, masarra.ef@gmail.com

الحوسبة الكمومية

امبروكة مسعود ذبان* مسرة محمد عبد السلام[2]
[1,2]قسم الفيزياء، كلية التربية قصر بن غشير، جامعة طرابلس، طرابلس، ليبيا

Abstract
This study investigates the efficiency of Grover's algorithm in identifying a specific element within a given search space. The experiment compares the performance of classical search versus quantum search using Grover's algorithm, focusing on the time required to locate the target element in an unsorted database of 100,000 entries. The results demonstrate that the quantum search exhibits a significantly slower rise in computational time compared to classical search, highlighting its superior efficiency in processing large-scale datasets. This performance disparity arises because classical search algorithms operate by sequentially checking each element until the target is found, resulting in a linear increase in the number of operations as the dataset grows. In contrast, Grover's quantum search algorithm achieves a quadratic speedup over classical approaches, making it particularly advantageous for large, unstructured databases.
Keywords: - Quantum Computing, Quantum entanglement, Quantum State, Qubit.

الملخص
يختبر هذا البحث فعالية خوارزمية جروفر في إيجاد عنصر مميز ضمن فضاء بحث معين حيث تهدف التجربة إلى مقارنة أداء البحث الكلاسيكي مقابل البحث الكمومي باستخدام خوارزمية جروفر، وذلك من حيث الزمن المستغرق للوصول إلى العنصر المطلوب داخل قاعدة بيانات غير مرتبة تتكوّن من 100,000 عنصـــر. حيث ظهر بوضـــوح أن خط البحث الكمومي يرتفع ببطء أكبر مقارنة بالبحث التقليدي، مما يعكس تفوقه في معالجة أحجام كبيرة من البيانات، هذا الاختلاف في الأداء يرجع إلى إن البحث التقليدي في الخوارزميات الكلاسيكية، يتم عن طريق المرور على كل عنصر على حدي حتى يتم العثور على العنصر المطلوب. وبالتالي كلما زاد عدد العناصر زادت عدد العمليات اللازمة بمعدل خطي, في المقابل تحقق خوارزمية البحث الكمومي لجروفر تســـارعًا تربيعيًا مقارنة بالأســـاليب الكلاسيكية، مما يجعلها مفيدة بشكل خاص لقواعد البيانات الكبيرة غير المنظمة.

الكلمات الدالة: - الحوسبة الكمومية، التشابك الكمومي، الحالة الكمومية، الكيوبت.

1. Introduction

Traditional computing as we know it today is the result of decades of development, as it relies on processing information in binary (0 and 1) through electrical circuits that carry out sequential logical operations. However, with the acceleration of progress in the field of information technology, the computer industry has faced difficulties related to the miniaturization of electronic components, especially transistors, which are the main component of microprocessors, because the continuous reduction of transistors to nanometer scales faces practical restrictions imposed by the laws of matter that govern the universe at these precise scales within the framework of the laws of classical physics,including the need for a new computational model that transcends the constraints of classical physics. Here comes the role of quantum mechanics which introduces a different concept of information processing.

The principles of quantum mechanics redefine the concept of computing through phenomena that do not exist in the classical world. While the traditional unit of information (bit) is limited to one state at a time, qubits - the basic units in quantum computing - allow for multiple states to exist simultaneously via the principle of quantum superposition. The qubits can also link together via quantum entanglement, creating unprecedented computational possibilities. (Wong, 2022)

This transformation in the computing infrastructure has led to the emergence of a new class of algorithms that take advantage of quantum properties. Computations are no longer constrained by the traditional linear path, but are able to explore several parallel computational paths thanks to the phenomenon of quantum superposition. These algorithms are characterized by their ability to address problems in an indefinite probabilistic way, reflecting the intrinsic nature of the quantum world.

2. Research Methodology

This research relied on the use of a comparative approach to compare quantum computing with classical computing, in terms of speed, efficiency, and practical applications, with the aim of highlighting the differences and unique potential of quantum computing. The effectiveness of the Grover algorithm in finding a distinctive element within a particular search space was tested. The experiment aims to compare the performance of classical search against quantum search using the Grover algorithm, in terms of the time it takes to reach the desired element within an unordered database consisting of 100,000 elements. This experiment comes to support the theoretical framework related to the performance of quantum algorithms, and to show the practical difference that Grover's algorithm can make compared to traditional methods based on serial examination. Grover's algorithm searches for one item in an unordered N-size database to represent this search quantitatively and to search for a particular item in an unordered N-size database. We prepare a quantum system consisting of n qubits to represent the search space as a first step where:

$$N = 2^n$$

The previous relationship enables us to know the number of n qubits required to find the element as:

$n = \log_2 N$

In addition, we need an auxiliary qubit to $|1\rangle$ implement Oracle, so the initial state of the system is as follows:

$$|\psi_0\rangle = |0\rangle^{\otimes n} \otimes |1\rangle$$

The next step in the algorithm is to apply the Hadamard Gateway to each search qubit to create an equal superposition of all possible states (Rieffel& Polak, 2011 p.27).

$$|\psi_1\rangle = (H^{\otimes n}\,|0\rangle^{\otimes n}) \otimes (H\,|1\rangle)$$

When deciphering this tensor multiplication, we get the sum of all possible states

$$H^{\otimes n}|0\rangle^{\otimes n} = \left(\frac{1}{\sqrt{2^n}}\right)\Sigma_{x=0}^{2^n-1}|x\rangle = \left(\frac{1}{\sqrt{N}}\right)\Sigma_{x=0}^{N-1}|x\rangle$$

The overall state of the system becomes:

$$|\psi_1\rangle = \left(\frac{1}{\sqrt{N}}\,\Sigma_{x=0}^{N-1}|x\rangle\right) \otimes \left(\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right)$$

$\Sigma_{x=0}^{N-1}$ Represents an equal superposition of all possible states in the research space Each case $|x\rangle$ has an amplitude of $\frac{1}{\sqrt{N}}$

We learned earlier that in quantum physics, states are not as discrete and obvious as in classical computing. Rather, they resemble waves that can interfere with each other. If there is a slight change in the phase of a particular wave such as a change in the direction of its oscillation, then that wave will later behave differently when interfering with other waves. (Hidary, , 2019 p.2O)

Here comes the function of the orcal, which is to distinguish the desired state from all quantum states in a superposition state by making a slight modification in the properties of the correct state by reversing the property of the phase in this case without other cases. This can be expressed mathematically in the following relationship:

$$U f\,|x\rangle = (-1)^{f(x)}|x\rangle$$

$$f(x)\begin{cases}1 & if\, x = w\\ 0 & otherwise\end{cases}$$

Where $x = w$ the orcal effect inverts the phase $|w\rangle$ U f $|w\rangle$=-$|w\rangle$

If it $x \neq w$ is, then the phase remains the same and does not change $|x\rangle$ U f $|x\rangle$ When the orcal are applied to,$|\psi_1\rangle$ we get :

$$|\psi_2\rangle = Uf\,|\psi 1\rangle = \left(\frac{\mathbf{1}}{\sqrt{\boldsymbol{N}}}\,\boldsymbol{\Sigma}_{\boldsymbol{x=0}}^{\boldsymbol{N-1}}|\boldsymbol{x}\rangle(-\mathbf{1})^{f(x)}\,|\,\boldsymbol{x}\rangle\right) \otimes \left(\frac{|\mathbf{0}\rangle - |\mathbf{1}\rangle}{\sqrt{\mathbf{2}}}\right)$$

All cases $|x\rangle$ retain their original capacity$\left(+\frac{1}{\sqrt{N}}\right)$except for $|w\rangle$ which is reflected in $\left(-\frac{1}{\sqrt{N}}\right)$

It is the reflection of the w phase that will later allow its amplitude to be amplified.
Since f(x) = 1only when$x = w$ this case can be excluded and therefore can be written

$$|\,\psi_2\rangle = \left(\frac{1}{\sqrt{N}}\,\Sigma_{x=0}^{N-1}|x\rangle + \frac{1}{\sqrt{N}}(-1)\,|\,w\rangle\right)\otimes\left(\frac{|0\rangle-\ |1\rangle}{\sqrt{2}}\right)$$

$$= \left(\frac{1}{\sqrt{N}}\,\Sigma_{x=0}^{N-1}|x\rangle - \frac{1}{\sqrt{N}}\,|\,w\rangle\right)\otimes\left(\frac{|0\rangle-\ |1\rangle}{\sqrt{2}}\right)$$

All quantum states are stored in the form of probability amplifications, and the orcal reverses the phase of the correct state, but when the system is measured, all states will be equal in amplitude, which causes difficulty in distinguishing the required state, including the need to amplify the amplitude (Amplitude Amplification), which is a process aimed at increasing the amplitude, and thus the probability of the state that represents the correct solution to the problem, while reducing the amplitudes of other cases. This amplification is done through a coefficient called the propagation coefficient, also known as the Grover Diffusion Operator, which reflects about the average amplitude. This happens by measuring the extent of the deviation of each case from the average, and then reflects it for this average, and because the correct state was inverted due to the application of the oracle to it. Therefore, when it is reversed around the average, the case wave has constructive interference with the average wave and its amplitude increases. As for the other cases, when its condition is reversed around the average, destructive interference occurs and its amplitude decreases. (Ganguly, 2022)
As a final step, we perform a process known as Grover iteration, which is a coordinated repetition of the orcal and diffusion coefficient steps together as a complete cycle. Each cycle causes a change in quantum capacities so that the amplitude of the correct state gradually increases and the amplitudes of other states decrease, and the amplitude of the correct state continues to increase, until it reaches the maximum possible value close to 1, making the probability of the correct measurement almost certain.
This rise in amplitude does not continue indefinitely; it is subject to an oscillation similar to the movement of the sine wave, that is, the amplitude begins to decline again if we exceed the ideal number of repetitions. This phenomenon occurs because quantum superposition follows periodic cycles caused by interference. If the amplification continues unabated, the benefit turns into harm, so it is necessary to accurately determine the number of iterations through a mathematical relationship that depends on the size of the database. If this number is exceeded, the algorithm loses its effectiveness and reduces the probability of success. Therefore, the number of repetitions of the cycle is not random, but proportional to the size of the database, and is calculated approximately according to the relationship:

$$\mathrm{r}\ \approx\ \left\lfloor\left(\frac{\pi}{4}\right)\times\ \sqrt{\mathrm{N}}\right\rfloor$$

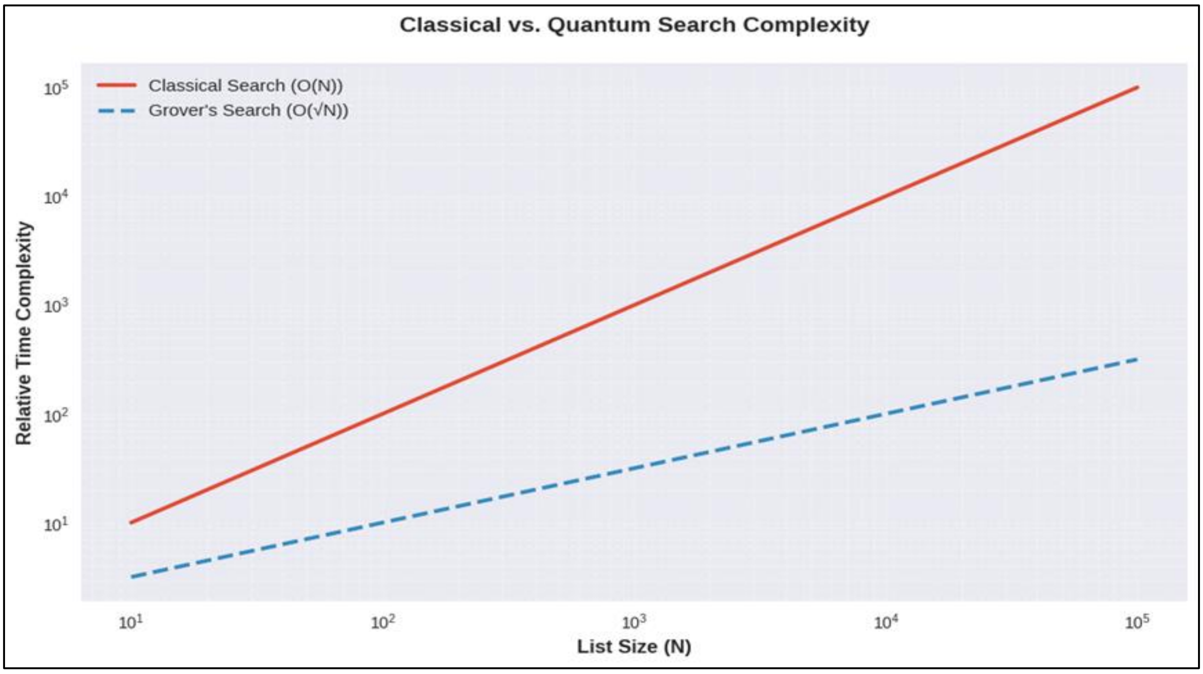
The experiment was carried out on the Google Colab website and is a free cloud interface in two stages:
1-Applying a traditional search algorithm using linear repetition by writing code in Python to find a specific item within an unordered list of 100,000 items
2- SimulatingGrover's algorithm quantumically for the same task, estimating the quantum time required to solve the problem approximately, and then comparing the two times graphically.

```
        # Formatting
        ax.set_xscale('log')
        ax.set_yscale('log')
        ax.set_xlabel('List Size (N)', fontsize=12, fontweight='bold')
        ax.set_ylabel('Relative Time Complexity', fontsize=12, fontweight='bold')
        ax.set_title('Classical vs. Quantum Search Complexity',
                    fontsize=14, pad=20, fontweight='bold')

        ax.grid(True, which="both", ls="--", alpha=0.3)
        ax.legend(fontsize=11, framealpha=1)

        # Professional touches
        ax.spines['top'].set_visible(False)
        ax.spines['right'].set_visible(False)
        plt.tight_layout()

        return fig

if __name__ == "__main__":
    visualizer = SearchPerformanceVisualizer()
    fig = visualizer.plot_comparison()
    plt.savefig('search_complexity_comparison.png', dpi=300, bbox_inches='tight')
    plt.show()
```

This code aims to create a graph that compares classic traditional search performance with quantum search performance, using the graph matplotlib library and the numpy library to perform numerical calculations. In addition, typing. Tuple is used to define data types returned from some functions.

```
import matplotlib.pyplot as plt
import numpy as np
from typing import List, Tuple

class SearchPerformanceVisualizer:
    """
    A class to visualize and compare classical vs. quantum search performance.
    """

    def __init__(self, max_list_size: int = 100000):
        """
        Initialize with default maximum list size.

        Args:
            max_list_size: Maximum N to visualize (default: 100,000)
        """
        self.max_list_size = max_list_size
        self.sizes = self._generate_size_array()

    def _generate_size_array(self) -> np.ndarray:
        """Generate logarithmic spaced array of list sizes."""
        return np.logspace(1, np.log10(self.max_list_size), num=50, dtype=int)

    def calculate_performance(self) -> Tuple[np.ndarray, np.ndarray]:
        """
        Calculate classical and quantum search complexities.

        Returns:
            Tuple of (classical_times, quantum_times) arrays
        """
        classical_times = self.sizes  # O(N)
        quantum_times = np.sqrt(self.sizes)  # O(√N)
        return classical_times, quantum_times

    def plot_comparison(self) -> plt.Figure:
        """
        Generate comparison plot with professional styling.

        Returns:
            matplotlib Figure object
        """
        classical_times, quantum_times = self.calculate_performance()

        plt.style.use('seaborn-v0_8')
        fig, ax = plt.subplots(figsize=(10, 6))

        # Plot data
        ax.plot(self.sizes, classical_times,
                color='#E24A33', linestyle='-',
                linewidth=2.5, label='Classical Search (O(N))')

        ax.plot(self.sizes, quantum_times,
                color='#348ABD', linestyle='--',
                linewidth=2.5, label="Grover's Search (O(√N))")
```

Results:



When executing the code, we were shown a graph in which two lines representing the time complexity of each of the two algorithms appeared:

The red straight line represents the performance of classical research as time increases linearly with the size of the list, expressing a time complexity of the order O(N)

The blue dotted line represents the quantum performance of Grover's algorithm, as time increases more slowly with the size of the list, following a time complexity of rank r.

The graph clearly shows that the quantum search line is rising more slowly than traditional research, reflecting its superiority in processing large volumes of data.

This difference in performance is due to the fact that the traditional search in classical algorithms is done by going through each item separately until the desired item is found. Thus, the greater the number of elements, the greater the number of operations needed at a linear rate. This model translates into mathematical temporal complexity given in the formula:

$$T_{classical}(N) \propto N \ldots\ldots\ldots\ldots\ldots (1)$$

Whereas

$T_{classical}(N)$ means the time taken by a traditional search algorithm when the number of elements is N means the time taken by a traditional search algorithm when the number of elements is N.

This means that the time doubles as the number of items doubles. In the graph, this is shown as a straight line using a logarithmic scale, reflecting the linear relationship

In quantum research, quantum superposition is used to speed up the search process. Grover's algorithm can find the desired element after a number of steps proportional to the square root of the number of elements:

$$T_{quantum}(N) \propto \sqrt{N} \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (2)$$

$T_{quantum}(N)$ means the time taken by a quantum search algorithm when the number of elements is N

This leads to a significant reduction in the number of processes needed compared to traditional research, especially with large lists. In the chart, this is shown as a slower rising line compared to the traditional line,.As the list grows in size, the difference accelerates Quadratic, so that quantum research becomes almost exponentially superior compared to traditional research.

At huge sizes such as millions or trillions of items, the difference between quantum and traditional research is in days versus seconds and this is where the achieved time efficiency stands out

The simulation showed that the practical application of Grover's algorithm on a real quantum computer is possible and gives accurate results on the simulator, indicating the possibility of transitioning to execution on actual quantum devices.

3. Conclusion

This research represents an applied study to compare the efficiency of Grover's quantum algorithm with classical research methods in processing huge unordered databases. A simulation experiment was carried out on a 100,000-item database, with the results showing that Grover's algorithm achieves time acceleration of a quadratic nature compared to the linear complexity of conventional research, making it more effective at handling large-scale data. This superiority is due to the exploitation of the principles of superposition and quantum interference to increase the likelihood of reaching the desired element after a smaller number of steps. The study demonstrates the great potential of quantum computing in multiple fields, including complex data analysis, drug design, and advanced material development, emphasizing that future developments in quantum hardware will enhance the applicability of these algorithms on a wide practical scale.

References

1. Ganguly, S. (2022). Quantum machine learning: An applied approach. Apress.
2. Hidary, Jack. 2019. Quantum Computing: An Applied Approach. Cham, Switzerland: Springer Nature Gewerbestrasse 11, 6330 Cham, Switzerland
3. Lai, C-Y and Cheng, H-C (2022). " Learning quantum circuits of some T gates". Journal of IEEE Transactions on Information Theory, 68(6):3951–3964.
4. Rieffel, Eleanor., & Polak, Wolfgang. (2011). Quantum computing: A gentle introduction. The MIT Press Cambridge, Massachusetts London, England
5. Wong Thomas G. 2022. Introduction to Classical and Quantum Computing. Published by Rooted Grove, Omaha, Nebraska.