Degree Project in Computer Science and Engineering

First cycle, 15 credits

Impact of Noise on Grover's Algorithm When Searching for Multiple Queried Items

ALFRED MALMSTRÖM, ANAHITA SHAFIEI

Impact of Noise on Grover’s Algorithm When Searching for Multiple Queried Items

ALFRED MALMSTRÖM
ANAHITA SHAFIEI

Degree Programme in Computer Science and Engineering
Date: July 2, 2025
Supervisor: Ivy Bo Peng
Examiner: Pawel Andrzej Herman
School of Electrical Engineering and Computer Science
Swedish title: Brusets påverkan på Grovers algoritm vid sökning av flera markerade element

Abstract

Quantum computing promises more efficient solutions to certain problems, exemplified by Grover's algorithm, which achieves a quadratic speed-up in unstructured database search. However, its practical implementation is challenged by noise in today's Noisy Intermediate-Scale Quantum (NISQ) era devices. This study investigates how various types of quantum noise affect Grover's performance when searching for multiple queried elements in an unsorted database. Using the Qiskit framework, simulations were run under both ideal and noisy conditions, including real hardware noise from IBM and custom error models (amplitude damping, depolarization, and bit-flip). The study explores two versions of Grover's algorithm, optimistic and pessimistic, that differ in how the number of iterations is calculated for multiple marked elements. Performance was measured by the average number of oracle calls needed. Results show that noise significantly degrades performance as database size increases. The pessimistic variant outperformed the optimistic one in all tests, however, noise had a detrimental impact on both variants. These findings underscore the importance of noise-aware algorithm design in the NISQ era.

Sammanfattning

Kvantalgoritmer har potential att lösa vissa problem mer effektivt än klassiska metoder, där Grovers algoritm är ett exempel genom sin kvadratiska prestandaförbättring vid sökning i ostrukturerade databaser. I praktiken begränsas dock dess tillämpning av brus i dagens så kallade NISQ-era datorer (Noisy Intermediate-Scale Quantum). Denna studie undersöker hur olika typer av brus påverkar Grovers algoritms prestanda vid sökning efter flera markerade element i en ostrukturerad databas. Med hjälp av Qiskit-ramverket simulerades algoritmen under både brusfria och brusiga förhållanden, inklusive realistiska brusmodeller från IBMs kvantdatorer samt egenkonstruerade modeller som simulerar amplituddämpning, depolarisering och bitflip-fel. Studien undersöker två versioner av Grovers algoritm, en optimistisk och en pessimistisk, som skiljer sig åt i hur antalet iterationer beräknas för att hitta flera markerade element. Prestandan mättes som det genomsnittliga antalet orakelanrop som krävdes för att hitta alla markerade element. Resultaten visar att bruset kraftigt försämrar prestandan i takt med att databasens storlek ökar. Den pessimistiska varianten presterade bättre än den optimistiska i samtliga tester, men bruset hade en stark negativ inverkan på båda varianterna. Dessa resultat understryker vikten av att utforma algoritmer med brusanpassning i åtanke för den nuvarande NISQ-eran.

Contents

Chapter 1

Introduction

Quantum computing is an emerging technology that offers the ability to tackle computational problems that are too complex or resource-intensive for classical computers. It could enable major advances in fields such as cryptography [1], drug discovery [2], and optimization [3], making it a topic of significant interest for further research. Quantum mechanics, the theory describing atomic and subatomic behaviour, form the foundation of quantum computing. Rather than storing and manipulating data using bits set to 0 or 1, quantum computing performs operations on quantum bits, also known as qubits, that exist in a state between 0 and 1, collapsing into either value upon measurement. This makes it so that quantum computers can evaluate multiple options at once, leading to a fundamentally different way of processing information.

This, in theory, gives certain quantum algorithms the potential to outperform classical ones. A well-known example is Grover's search algorithm, which offers a quadratic speed-up for solving unstructured search problems. It can be used to search through a list of unsorted items, like trying to find "a needle in a haystack" [4]. A classical implementation of linear search would need to search every item one by one, which would take $O(N)$ for a list of size N. Grover's algorithm can perform the same task in roughly $O(\sqrt{N})$, significantly reducing the amount of queries.

Despite its promise, quantum computing is still in its early stages, with the current stage it's in being called the NISQ (Noisy Intermediate-Scale Quantum) era. Current hardware is limited in scale and precision, largely because of the instability of qubits. Quantum systems are highly sensitive to their environment and external disturbances known as noise can cause computational errors. Understanding and mitigating the effects of noise is therefore essential for advancing practical quantum algorithms and realizing their full potential.

1.1 Aim and Research Question

The aim of this study is to investigate how different types and levels of quantum noise affect the performance of Grover's algorithm when searching for multiple queried elements. Through simulations under various noise models, database sizes, and numbers of queried items, the work seeks to quantify how noise influences the average number of oracle calls needed to successfully identify all marked elements in an unsorted database. This leads to the research question:

> *How does quantum noise affect the average number of oracle calls required to find all marked elements in an unsorted database using Grover's algorithm? In particular, how do different noise models and levels influence performance across varying database sizes and algorithmic variants?*

The study further explores which types of noise most significantly impact performance and which version of Grover's algorithm is most robust. By varying the number of queried elements, qubits, and iterations, the goal is to clarify the algorithm's practical limitations and sensitivity to noise in the context of today's NISQ-era quantum hardware.

1.2 Scope

This study focuses on simulating Grover's algorithm under both ideal noiseless and noisy conditions to assess its theoretical performance without actual implementation on quantum hardware. Elements that have been found will be excluded from subsequent searches.

1.3 Approach

The quantum computing framework *Qiskit*, along with *Qiskit Aer*, is used to simulate Grover's algorithm under both noiseless and noisy conditions. Noise is modelled using pre-defined noise models from IBM's fake back-ends and custom-built noise models simulating different types of noise. Each run attempts to find one of k marked elements in an unsorted database of size N, giving the average number of oracle calls needed to find it. The total amount of oracle calls to identify all k elements is then calculated. The various results will be compared to one another, as well as the performance of linear search.

Chapter 2

Background

2.1 Quantum Computing Basics

This section outlines the basics of quantum computing, covering qubits, superposition, quantum registers, and how quantum gates manipulate qubit states for computation.

2.1.1 Qubits and Superposition

Qubits are the quantum equivalent of classical bits and are commonly represented using Dirac (Bra-ket) notation as $|\psi\rangle$. Upon measurement they can collapse into one of two basis states, $|0\rangle$ or $|1\rangle$. When unmeasured, however, the probability of a qubit collapsing into either of these states can be described by the formula:

$$|\psi\rangle = \alpha \cdot |0\rangle + \beta \cdot |1\rangle$$

where α^2 and β^2 are the probabilities that $|0\rangle$ and $|1\rangle$ respectively will be the measured state.
They also satisfy the condition:

$$\begin{cases} |\alpha|^2 + |\beta|^2 = 1, & \text{if } \alpha, \beta \text{ are complex} \\ \alpha^2 + \beta^2 = 1, & \text{if } \alpha, \beta \text{ are real} \end{cases}$$

While the coefficients, also referred to as amplitudes, can be complex in the general case, this study focuses on Grover's algorithm, where only the real case is relevant [5].

2.1.2 Quantum Registers

Multiple qubits can be grouped together to form a quantum register, denoted as $|\psi_n\rangle$, where n represents the number of qubits. A quantum registers exists in a superposition of all possible states of the qubits, and once measured, can collapse into $N = 2^n$ possible states. For example:
1-qubit register: $|0\rangle, |1\rangle$
2-qubit register: $|00\rangle, |01\rangle, |10\rangle, |11\rangle$
3-qubit register: $|000\rangle, |001\rangle, |010\rangle, |011\rangle, |100\rangle, |101\rangle, |110\rangle, |111\rangle$
All states in Dirac notation, $|x\rangle$, are representations of the combinations of states the two qubits can collapse into upon measurement. The example of a 2-qubit quantum register can be used to describe a register's superpositions:

$$|\psi\rangle = \alpha_0 \cdot |00\rangle + \alpha_1 \cdot |01\rangle + \alpha_2 \cdot |10\rangle + \alpha_3 \cdot |11\rangle$$

A state i's amplitude squared, α_i^2, represents the possibility that the register will end up in that state when measured. As with a singular qubit, the sum of all probabilities must equal one [5], [6]:

$$\sum_{i=0}^{2^n-1} \alpha_i^2 = 1, \text{case } \alpha \text{ real}$$

In this study, an grouping of qubits/register, regardless of size, will be referred to as $|Q\rangle$ and the set of all its possible basis states as U_n.

2.1.3 Matrix and Geometric Representations

To aid in visualising qubit behaviour, various representations can be used. The state of a qubit $|\psi\rangle$, is formally written as a unit vector in C^2, the two-dimensional complex vector space. It can be written in vector notation as:

$$|\psi\rangle = \begin{bmatrix} \alpha \\ \beta \end{bmatrix}$$

The states $|0\rangle$ or $|1\rangle$ serve as the orthonormal basis for the space, referred to as the computational basis, represented as:

$$|0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \text{ and } |1\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

in matrix form. In this representation, $|\psi\rangle = \alpha \cdot |0\rangle + \beta \cdot |1\rangle$ still holds true, but can now be interpreted as a linear combination of the basis states [1,2]. Given the amplitudes' complex nature, the computational basis is typically represented as a Bloch

Sphere [6], where each pure qubit state corresponds to a surface on the surface of a unit sphere (See Fig. 2.1). The poles represent the basis states $|0\rangle$ and $|1\rangle$ and the vector $|\psi\rangle = cos\theta \cdot |0\rangle + e^{i\phi} sin\theta \cdot |1\rangle$ represents the state of the qubit.

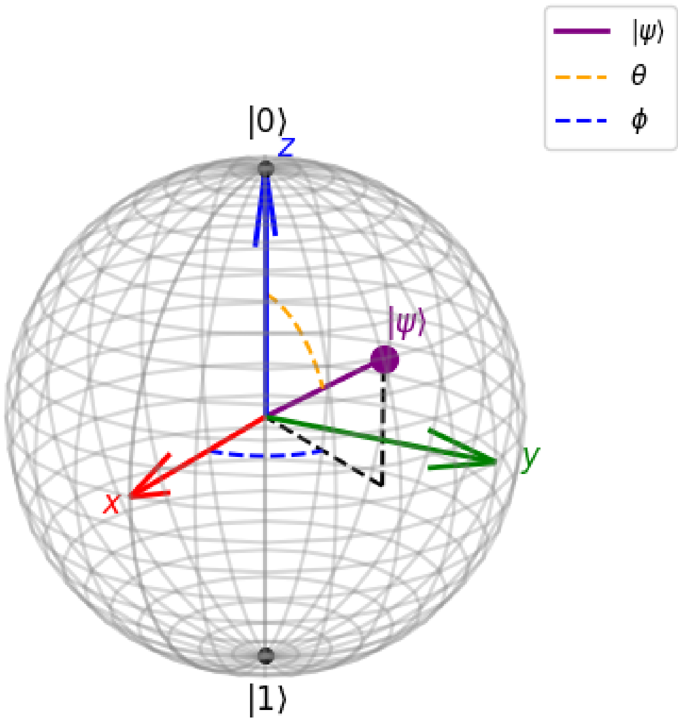

Figure 2.1: A representation of a Bloch Sphere.

When describing Grover's algorithm, however, multiple studies opt to represent qubits' space as a vector on a two-dimensional plane, akin to a unit circle [5], [7]. This representation (Fig. 2.2) will be used throughout the background of this study.

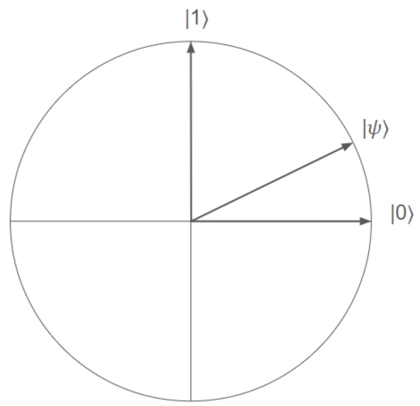

Figure 2.2: A 2-dimensional representation

2.1.4 Quantum Gates and Circuits

In order to perform the operations which are necessary for quantum computing, qubits' state probability amplitudes need to be manipulated. This is done through quantum gates, analogous to logical gates in classical computing.

Because of the continuous state probability amplitudes of qubits, which is more complex than a discrete binary bit, a lot of quantum gates are simply 1-qubit gates. They are also required to be unitary, meaning that their operations are reversible [5], [6], to be able to preserve the total quantum information in the system.

As a result of this, quantum gates can be represented as $n * n$ unitary matrices, where n is the amount of qubits in a register. These matrices can then be multiplied with the qubit state vector to change its amplitudes [6].

2.1.5 Hadamard Gate and Transform

The Hadamard gate is a 1-qubit gate often used in quantum computing transform the computational basis states into an equal superposition. The unitary matrix used to represent a Hadamard gate takes on the following form [8]:

$$H = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

providing these resulting states when multiplied with the basis states:

$$H|0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$$

$$H|1\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$$

When applied to a register (a system of n qubits), it is called the Hadamard Transform. The following notation is used when it is performed on a register only consisting of qubits in the $|0\rangle$ state with an NxN Hadamard Matrix [6]:

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

where $|x\rangle$ represents all permutations of states of the register, U_n [6], [8]. This results in the "Hadamard State", $|u\rangle$, where all values are equally likely to be measured.

2.2 Grover's Algorithm

Grover's algorithm, introduced by Lov Grover in 1996 [4], provides a quantum solution to the problem of unstructured search in an unsorted database. Classical linear search requires $O(N)$ queries to find a marked item, but Grover's algorithm reduces this to $O\left(\sqrt{\frac{N}{k}}\right)$ [4], [9], where k is the number of marked elements. It uses quantum superposition and interference to amplify the probability of measuring a correct solution, relying on two key operations: phase inversion, which flips the sign of marked states, and diffusion, which boosts their amplitudes [4].

The circuit for Grover's algorithm (Fig. 2.3) begins with a Hadamard transform applied to each qubit in the quantum register $Q = \{q_1, q_2, \ldots, q_n\}$ to create a uniform superposition. This is followed by a series of Grover iterations, each consisting of a phase inversion (Z-gate) and a diffusion transform (R-gate). After the final iteration, the qubits are measured (M) into a classical register $C = \{c_1, c_2, \ldots, c_n\}$, producing outcomes that have a high probability of matching the marked elements.

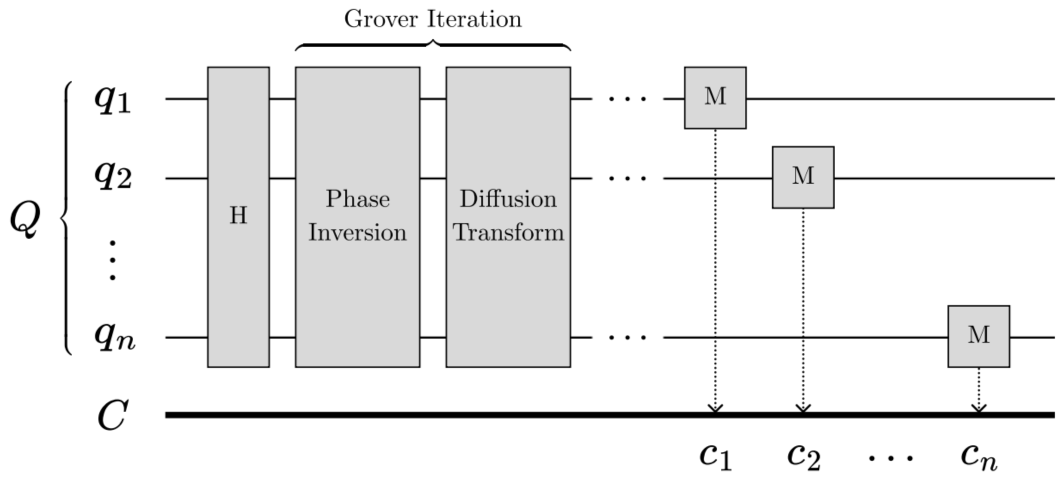

Figure 2.3: An abstracted visual representation of Grover's algorithm. Adapted from [7].

The following sections outline the central mechanisms of Grover's algorithm: starting with the oracle function, then the Grover iteration, comprising of phase inversion and diffusion, and concluding with strategies for choosing the number of iterations .

2.2.1 Oracle Function and State Decomposition

The oracle function is a key component of Grover's algorithm, used within the phase inversion step to identify and distinguish marked elements. For the purposes of this study, the oracle is defined as a function $f(|x\rangle)$, where:

$$f(|x\rangle) = \begin{cases} true, & \text{if } |x\rangle \text{ is queried} \\ false, & \text{if } |x\rangle \text{ is not queried.} \end{cases}$$

The oracle function $f(|x\rangle)$ leads to a partition of all basis states of a register, U_n, into two completely disjoint sets. The set $\mathcal{A}$ consists of all queried states $|a\rangle \in U_n$ for which $f(|a\rangle) = true$. $\mathcal{B}$, the set of non-queried states, can be defined as all states $|b\rangle \in U_n$ where $f(|b\rangle) = false$. The states these two sets contain can then be combined defined as their own quantum states, $|\mathcal{A}\rangle$ and $|\mathcal{B}\rangle$ [7].A general quantum register $|Q\rangle$ can then be expressed as a linear combination of these two orthogonal states:

$$|Q\rangle = \alpha \cdot |\mathcal{A}\rangle + \beta \cdot |\mathcal{B}\rangle, \quad \text{where } \alpha^2 + \beta^2 = 1.$$

This decomposition allows Grover's algorithm to be visualised similarly to the single-qubit unit circle example in section 2.1.3 (Fig. 2.2), however with its orthogonal basis states $|1\rangle$ and $|0\rangle$ being replaced by $|\mathcal{A}\rangle$ and $|\mathcal{B}\rangle$ (Fig 2.4).

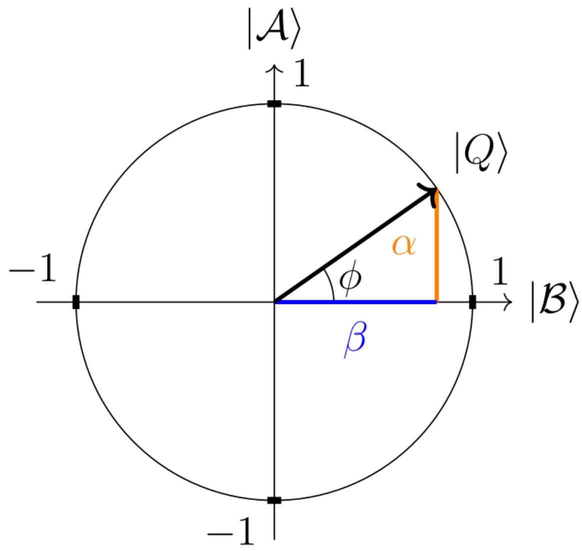

Figure 2.4: A visualisation of a register $|Q\rangle$ as a vector in the two-dimensional subspace spanned by orthogonal states $|\mathcal{A}\rangle$ (queried states) and $|\mathcal{B}\rangle$ (non-queried states). Adapted from [7].

To ensure that both states are in an equal superposition, the register $|Q\rangle$ is initialised using the Hadamard transform, resulting in the following equal superpositions:

$$|\mathcal{A}\rangle = \frac{1}{\sqrt{|\mathcal{A}|}} \sum_{|a\rangle \in \mathcal{A}} |a\rangle \textit{ and } \quad |\mathcal{B}\rangle = \frac{1}{\sqrt{|\mathcal{B}|}} \sum_{|b\rangle \in \mathcal{B}} |b\rangle$$

Seeing as the goal of Grover's algorithm is to maximise the probability of the marked elements being measured, this can be translated as increasing α^2 as much as possible, ideally having the value of ϕ reach $\frac{\pi}{2}$ so that $|Q\rangle$ approaches $|\mathcal{A}\rangle$[4], [5], [7]. This amplification is performed in the following steps of the algorithm, during the phase inversion and diffusion transform operations of the Grover Iterate.

2.2.2 First reflection - Phase Inversion

The first step of the Grover Iterate is phase inversion (Z-gate/operator), which uses the oracle function f to invert the phases of queried states $|a\rangle$ in $|\mathcal{A}\rangle$ [4], [5], [7]. This is referred to as the phase shift or phase inversion and can be written as such:

$$Z(|x\rangle) = \begin{cases} -|x\rangle, & \text{if } |x\rangle \text{ is queried} \\ |x\rangle, & \text{otherwise.} \end{cases}$$

Due to all states $|a\rangle$ in $|\mathcal{A}\rangle$ being queried, the following generalisation can be derived,

$$Z(|Q\rangle) = \alpha \cdot Z(|\mathcal{A}\rangle) + \beta \cdot Z(|\mathcal{B}\rangle) = -\alpha \cdot |\mathcal{A}\rangle + \beta \cdot |\mathcal{B}\rangle$$

which can be visualised in the unit circle representation as the vector for $|Q\rangle$ being "flipped" along the x-axis:

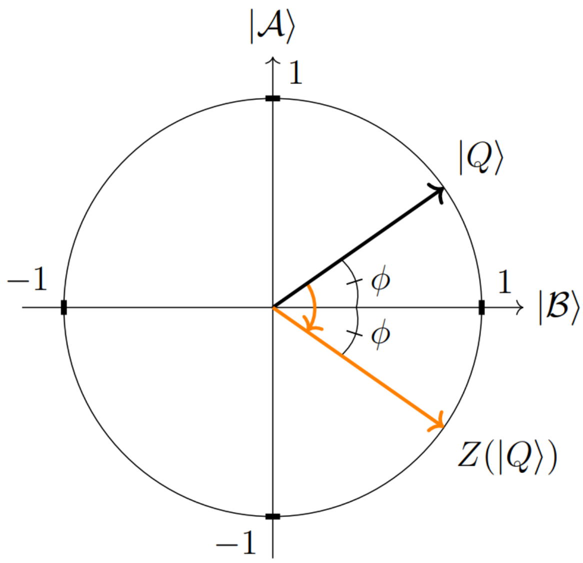

Figure 2.5: Visualisation of the result of a phase-inversion operation (Z-gate) on a quantum register $|Q\rangle$. Adapted from [7].

2.2.3 Second Reflection - Diffusion Transform

The second step of the Grover Iterate is known as the diffusion transform. This operation is used to increase the amplitudes of all phases that were inverted in the prior step. Geometrically, this operation can be understood as a reflection of $|Q\rangle$ around the initial Hadamard state $|u\rangle$, rotating it closer to $|\mathcal{A}\rangle$.

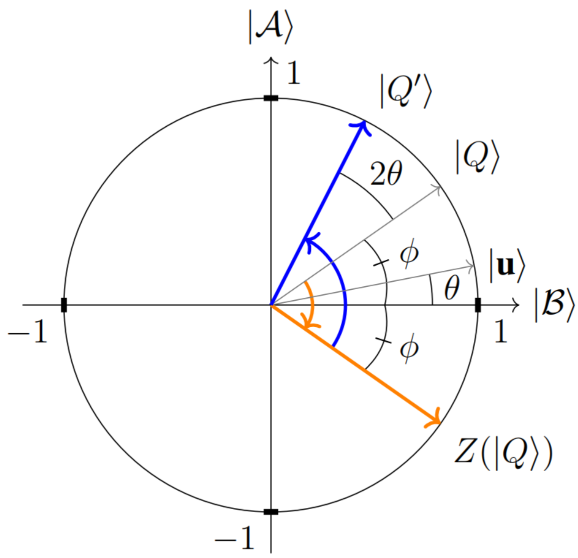

Figure 2.6: Visualisation of the Phase Inversion and Diffusion transform being applied to register $|Q\rangle$, resulting in $|Q'\rangle$. Adapted from [7].

As visualised in Figure 2.6, if the angle between $|Q\rangle$ and $|\mathcal{B}\rangle$ is ϕ, and the angle between the Hadamard state $|u\rangle$ and $|\mathcal{B}\rangle$ is θ, then the diffusion operator reflects the phase-inverted state $Z(|Q\rangle)$ over $|u\rangle$, $R(Z(|Q\rangle))$, producing a new state $|Q'\rangle$ at angle of $\phi + 2\theta$ from $|\mathcal{B}\rangle$ [4], [5], [7].

2.2.4 Optimistic and Pessimistic Grover

Optimistic and Pessimistic Grover are terms introduced and used by Aldenbro and Skalski [7], based on the work of Boyer et al. [9], to describe two different approaches when calculating the number of iterations *m* used to find multiple queried elements in an unsorted database using Grover. They were developed to minimize the number of oracle calls, as in uses of the oracle function, needed while still giving a accurate result.

Optimistic Grover assumes that the number of targeted elements $\mathcal{A}$ is known or can be estimated accurately. Grover's algorithm is highly sensitive to the number of solutions, as its performance improves when the number of solutions is larger. Optimistic Grover leverages this by tailoring the number of iterations of the algorithm to the expected number of solutions, maximizing efficiency. This approach is "optimistic" because it assumes the best-case scenario where the algorithm can be fine-tuned for optimal performance. They calculate that the number of iterations is:

$$m \approx \frac{\pi}{4 \times arcsin(\sqrt{\frac{|\mathcal{A}|}{N}})}$$

In contrast, the Pessimistic Grover approach assumes that the number of target elements is unknown or could be smaller than expected. This leads to a more conservative application of Grover's algorithm, where the number of iterations is chosen to account for the possibility of fewer solutions. This approach is "pessimistic" because it prepares for the worst-case scenario, ensuring that the algorithm still works even if the number of solutions is lower than anticipated. The number of iterations according to the pessimistic method is:

$$m = 0.58278\sqrt{\frac{N}{|\mathcal{A}|}}$$

2.3 Noise

Noise is a significant challenge that arises in quantum computers from various sources, each contributing to the degradation of quantum information and the reliability of quantum computations. Noise refers to any kind of disturbance that can cause errors in quantum operations or computation errors. These disturbances can manifest as unwanted interactions with the environment, imperfections in hardware, or fluctuations in control parameters. This affects the result of algorithms run on quantum computers which is a fundamental obstacle in the development of practical quantum computers.

Addressing noise through improved hardware design, error correction techniques, and environmental control is crucial for realizing the full potential of quantum computing. Researchers are actively exploring various strategies, such as topological qubits, better materials, and advanced error correction methods, to mitigate the effects of noise and build more robust quantum systems [10].

This report considers three key common and widely studied types of quantum noise: amplitude damping, depolarizing noise, and bit-flip errors.

2.3.1 Bit-flip Error

The *bit-flip channel* represents a simple type of noise where a qubit flips from $|0\rangle$ to $|1\rangle$ or vice versa with some probability. This can be thought of as a quantum version of a classical bit flip. If the flip occurs randomly with probability $1-p$, the effect on the Bloch sphere is a contraction toward the x-axis: the sphere is compressed in the yz-plane, while points along the x-axis are left unaffected (Fig. 2.7). This reflects the loss of coherence caused by unpredictable bit flips [6].

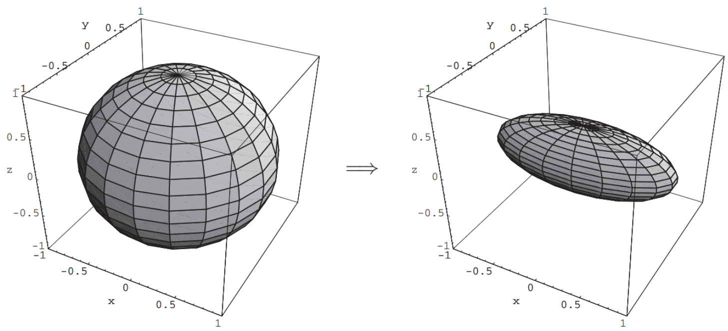

Figure 2.7: Effect of bit-flip noise with $p = 0.3$ on the Bloch sphere. The original sphere (left) shows all pure qubit states, while the distorted sphere (right) shows the result after noise is applied. Adapted from [6].

2.3.2 Depolarization

The *depolarizing channel* is a common model for uniform noise in a quantum system. With probability p, a qubit is completely randomized, replaced by the maximally mixed state, and with probability $1 - p$, it remains unchanged. On the Bloch sphere, this causes all states to shrink uniformly toward the center, reflecting a general loss of information about the qubit's state (Fig. 2.8). The result is reduced purity, as the qubit becomes less distinguishable from a completely random state [6].

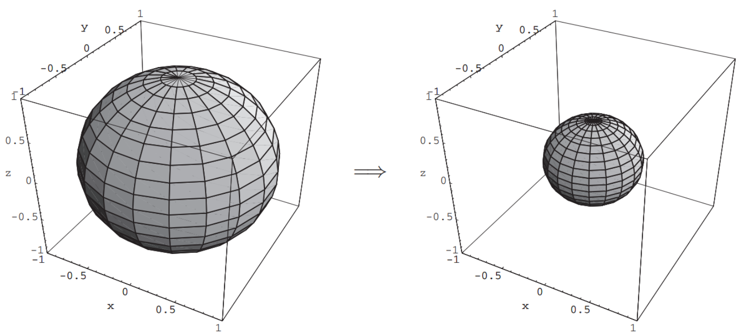

Figure 2.8: Effect of depolarizing noise with $p = 0.5$ on the Bloch sphere. The entire sphere shrinks uniformly as a function of p. Adapted from [6].

2.3.3 Amplitude damping

Amplitude damping describes energy loss in a quantum system, such as a qubit spontaneously emitting a photon and falling from an excited state $|1\rangle$ to the ground state $|0\rangle$. As this process occurs, the qubit becomes more likely to be found in $|0\rangle$, and the Bloch sphere representation reflects this by pulling all states upward toward the $|0\rangle$ state (Fig. 2.9). Over time, the sphere effectively shrinks in this direction, with $|0\rangle$ as the stable final point. This models realistic physical scenarios like photon loss or atomic decay [6].

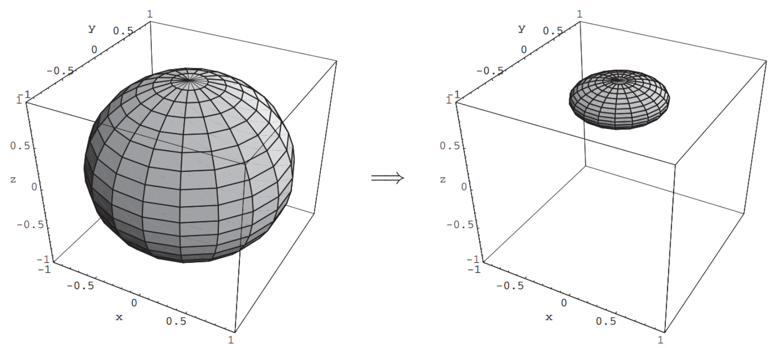

Figure 2.9: Effect of amplitude damping noise with $p = 0.8$ on the Bloch sphere. The sphere contracts toward the north pole, representing the $|0\rangle$ state, showing a loss of energy in the system. Adapted from [6].

2.4 Literature Study

2.4.1 Grover's algorithm

Grover's 1996 study Grover [4] introduced the fundamental concepts of quantum search algorithms using amplitude amplification. The paper establishes key results, such as an upper bound on the required number of queries and a method for designing the phase-inversion and diffusion operators. However, this study mainly focuses on the case of a single marked element, only briefly discussing how this algorithm could be modified for multi-query searches.

Boyer, Brassard, Høyer, *et al.* [9] formally analysed Grover's search algorithm with multiple marked elements, also deriving theoretical bounds on the number of queries needed. Their work demonstrated that for k marked elements in an unsorted database of N items, the number of required queries scales as $O(\sqrt{N/k})$, providing a generalization of Grover's algorithm. In this era of quantum computing, these algorithms were seen as promising for future applications rather than methods that could be immediately implemented on a large scale, given the technical limitations at the time.

2.4.2 Grover's algorithm in practice

The first experimental implementation of Grover's algorithm was conducted by Chuang, Gershenfeld, and Kubinec [11], who used nuclear magnetic resonance (NMR) techniques to implement the search algorithm, albeit only on a very small scale. Chloroform molecules in acetone were utilized to create a 2-qubit system. This study acted as the first bridge between theory and practice for Grover's algorithm, demonstrating that it could feasibly be applied.

In 1999, Nakamura, Pashkin, and Tsai [12] established a new standard when they demonstrated that superconducting circuits could be used as qubits. This represented a paradigm shift, given that most modern quantum computers rely on this technology. Since then, quantum computing has become an ever-growing field, with commercial quantum computers becoming increasingly available, even accessible via cloud services such as IBM's quantum systems [13].

2.4.3 Future Applications of Grover's Algorithm

Looking forward, Grover's algorithm holds significant potential across a range of applications where unstructured search problems are prevalent. One major area is cryptography, while Grover's algorithm does not completely break modern cryptographic systems like RSA (which is threatened by Shor's algorithm), it does pose a substantial risk to symmetric cryptographic schemes. Specifically, it can reduce the effective key length of symmetric encryption algorithms from n^2 to approximately $n/2$ operations, necessitating longer key sizes to maintain security levels [14].

In addition to cryptography, Grover's algorithm can be applied to optimization problems encountered in areas such as logistics, machine learning, and financial modelling [15]. Though classical heuristics may still outperform quantum approaches in certain structured tasks, Grover's algorithm remains promising for domains where little is known about the problem's structure. Furthermore, it can be integrated into hybrid quantum-classical algorithms to accelerate tasks like pattern matching, unsorted database search, and clustering, which are central in big data and AI applications [16].

2.4.4 Impact of Noise

The impact of random Gaussian noise on Grover's algorithm was studied by Pablo-Norman and Ruiz-Altaba [17]. The researchers found that the presence of Gaussian noise leads to a reduction in the algorithm's efficiency. Specifically, they determined that the maximum tolerable noise level scales with the size of the search space N as $N^{-2/3}$, showing that noise sensitivity increases with database size.

Shapira, Mozes, and Biham [18] investigated the impact of unitary noise, which arises from fluctuations and drift in quantum gate operations. Their study found that the algorithm maintains significant efficiency as long as the noise level remains below a critical threshold, $\epsilon < O(n^{-1/2}N^{-1/4})$, based on the number of qubits n and the search space size N. Above this threshold, the algorithm's performance deteriorates significantly.

Salas [19] conducted a numerical analysis of Grover's algorithm under depolarizing noise conditions. The study revealed that the success probability of the algorithm decays exponentially with increasing noise levels. An error threshold was identified, indicating that the permissible noise level scales inversely with the dataset size, approximately as $N^{-1.1}$.

Chapter 3

Methods

This study aims to investigate the effect of noise on Grover's Quantum Search Algorithm by comparing its performance under noisy conditions to both a noiseless implementation of Grover's algorithm and linear search. The noiseless implementation of Grover's algorithm will be borrowed from the work of Aldenbro and Skalski [7] to create a baseline, serving as the foundation for the noiseless implementation, which will then be adapted to incorporate noise models and simulate noisy quantum environments.

Two types of noise models will be used in this study: real noise models and custom noise models. Real noise models will be sourced from Qiskit Runtime, reflecting the actual noise experienced on IBM quantum hardware. Custom noise models will be created to simulate specific types of noise, specifically depolarizing noise, amplitude damping, and phase damping, enabling more precise measurements of how different noise types affect Grover's algorithm. Each test case will be defined by three parameters: N, k, m and and v, where N represents the size of the database, k represents the number of target items to search for, m represents the noise model applied (either real or custom), and v indicates the version of Grover's algorithm used (either optimistic or pessimistic). Test cases will be designed by varying these parameters within specific ranges to ensure a comprehensive analysis. Each test case will be run multiple times, and the average results will be used for comparison to ensure statistical reliability.

3.1 Quantum Computing Programming Framework

- `Python(v= 3.11.1)`: Programming language used for writing the code.
- `Qiskit(v=1.4.2)`: An open-source framework for working with quantum computers at the level of circuits, pulses, and algorithms.
- `Qiskit Aer(v=0.16.4)`: An extension of Qiskit made for high-performance simulator for quantum circuits, allowing efficient execution of simulated noise on classical hardware.
- `Qiskit Runtime(v=0.36.1):` A cloud-based execution environment optimised for running quantum programs with improved performance and lower latency.

Qiskit [20] was chosen due to it being a more popular quantum computing simulator, and because the code we received from Aldenbro and Skalski was written using it. Qiskit Aer [21] and Runtime are extensions of the Qiskit framework that were used in order to use IBM's pre-existing noise models made from the noise of real quantum computers. We used an older version of Python because some of the functions of Qiskit Aer were not compatible with the newer releases of Python.

3.2 Algorithm

As previously mentioned, the original algorithm was borrowed from Aldenbro and Skalski [7]. Their method for a multi-queried Grover's Algorithm that excludes previously found elements served as the foundation for this study. The algorithm was updated to the newest versions of Qiskit and modified in order to support noise simulation. Here is the main algorithm, directly adapted from [7]:

Algorithm 1 Description of the Grover Simulation

```
function GroverIteration(n, A)
    G ← QuantumCircuit(n)                    ▷ Instantiate a quantum circuit of n qubits
    G.Append(Z(n, A))                        ▷ Apply phase inversion
    G.Append(R(n))                           ▷ Apply diffusion operator
    return G
function GroverExperiment(n, k, m)
    N ← 2^n
    A ← CreateQuery(N, k)                    ▷ Create query of k marked states
    qc ← QuantumCircuit(n)
    G ← GroverIteration(n, A)
    qc.Append(Hadamard(n))                   ▷ Initial Hadamard transform
    for i = 1 to m do
        qc.Append(G)
    qc.Append(Measure(n))                    ▷ Final measurement gates
    for i = 1 to 100 do
        s ← SimulateCircuit(qc, 1000)
        successes.Append(s)                  ▷ Collect results
    return successes
```

`GroverExperiment`(n, k, m) is the main algorithm used in this study, taking the number of qubits n, marked items k and number of Grover iterations m. Its purpose is to simulate multiple executions of Grover's algorithm and report how many times a marked item is successfully identified. The implementation also relies on a helper subroutine, `GroverIteration`(n, A), which constructs a complete Grover iteration circuit tailored to a specific set of target elements A.

The helper functions used throughout the implementation are:

- `CreateQuery`(N, k) generates a random subset A containing k unique integers chosen uniformly from the range $[0, N-1]$.
- `Z`(n, A) constructs an oracle gate that flips the phase of the computational basis states corresponding to the elements in A, while `R`(n) creates the diffusion operator used in Grover's amplitude amplification.

- `Hadamard`(n) produces a quantum gate applying the Hadamard transform to all n qubits, and `Measure`(n) returns a set of measurement operations for each qubit in the register.

- `QuantumCircuit`(n) initializes an empty quantum circuit consisting of n qubits and allows for sequential gate construction. Circuit execution is handled by `SimulateCircuit`(qc, shots), which runs the circuit qc a specified number of times and records the outcomes. The initial quantum state is assumed to be the all-zero state $|0^n\rangle$.

3.3 Experiments

The tests are structured around four key parameters:

- **N**: The size of the database.
- **k**: The number of target items in the search.
- **m**: The noise model applied.
- **v**: The version of Grover's algorithm used.

The experiments are organized by noise model type. For each model, we evaluate databases of size $N = 2^n$, where $2 \leq n \leq 6$. Within each database, we vary the number of target elements k from 1 up to $\frac{N}{2}$. To ensure statistical reliability, each configuration is run 100 times, with 10,000 measurements per run.

In each run i, we record the number of successful measurements s_i, where $1 \leq i \leq 100$. From this, we compute the number of repetitions required to obtain a successful result:

$$r_i = \frac{10000}{s_i}.$$

Each repetition includes m oracle calls during the Grover iterations, plus one final oracle call to verify the result. The total number of oracle calls per successful outcome is then:

$$x_i = r_i \cdot (m + 1).$$

To assess performance, we calculate the average number of oracle calls across all runs:

$$\tilde{x} = \frac{1}{100} \sum_{i=1}^{100} x_i.$$

Both the average hit probability and the average oracle call count $\tilde{x}$ are recorded for each configuration, allowing for comparison across different noise models. This method of calculating the number of oracle calls required was based on [7]. The number of runs and measurements per configuration was also chosen to match the setup used by Aldenbro and Skalski [7], to ensure comparable precision without excessive simulation time.

3.4 Noise Models

We examine two distinct noise categories: real quantum hardware noise and custom-simulated noise. The first is sourced from Qiskit Runtime, specifically the *FakeAthensV2* backend, based on IBM's retired Athens system. This provides a realistic benchmark by replicating gate errors, decoherence, and readout noise from actual hardware. The second category comprises controlled noise models simulated via Qiskit Aer, each isolating a single noise type to assess its individual impact. These include depolarizing noise, amplitude damping, and bit-flip noise, with possibility of occurring that is systematically increased from 0.05% (minimum) and 0.5% (maximum) in increments of 0.05%. This range was chosen based on the typical single-gate error rates reported in [22], which generally fall between 0.1% and 0.3%. Extending the range slightly below and above this interval allows for a more comprehensive evaluation that includes both better/near-ideal and worse noise scenarios.

3.5 Comparison with Classical Linear Search

Following the methodology of Aldenbro and Skalski [7], who focused on comparing the performance of classical and quantum search, the quantum simulation results are also compared to the average-case performance of classical linear search, as a baseline. For each test case (N, k), the goal is to determine the smallest number of oracle calls m such that the probability of identifying all k marked elements is at least 0.5. This corresponds to the smallest positive integer satisfying:

$$\frac{\binom{m}{k}}{\binom{N}{k}} \geq 0.5.$$

Chapter 4

Results

4.1 Comparing Noise Models

In this section, we compare the performance of Grover's algorithm by analyzing the average number of oracle calls under different noise models. Figure 4.1 presents the aggregated average number of oracle calls required to find k queried items using both the pessimistic and optimistic versions of Grover's algorithm. The specific noise models are labeled alongside the graphs. For comparison, the performance of classical linear search ("Classical") and the database size N are also included.

Multiple tests were conducted for each noise model, but only the results for the lowest and highest tested noise levels are shown. For example, "Min Depol" refers to the depolarization model set to a minimum error rate of 0.05%, while "Max Amp" refers to the amplitude damping model with a maximum error rate of 0.5%. "Athens" corresponds to the simulated backend *FakeAthensV2*.

The results in Figure 4.1 show that the impact of noise becomes more significant as the number of qubits increases. This is evident from the increasing gap between the highest and lowest noise level results, as the upper bounds rise significantly with more qubits. Furthermore, the graphs show that at higher noise levels, the algorithm's performance is severely degraded beyond N=16, leading to results substantially deviating from both the classical baseline and the noise-free algorithm. Additionally, at $N = 64$, even the lowest tested noise level causes the algorithms to perform significantly slower than classical linear search.

Based on the graphs, we observe the same general effects of noise in both Optimistic and Pessimistic Grover. However we also observe that in the later graphs the pessimistic version has a better result than the optimistic version.

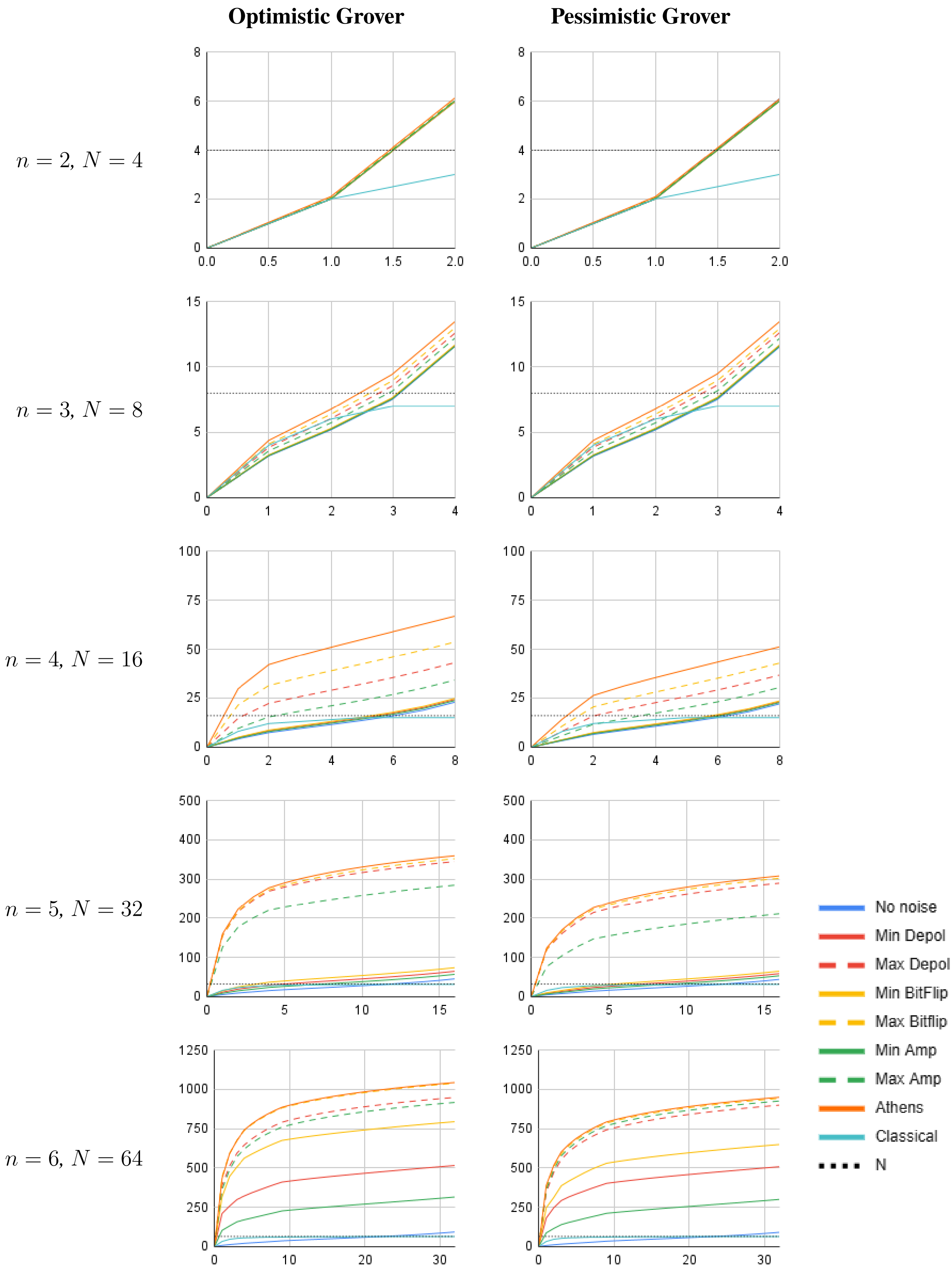

Figure 4.1: Noisy and noiseless simulations of the Optimistic and Pessimistic variants of Grover's algorithm. Each subplot corresponds to a different number of qubits n, determining the database size $N = 2^n$. The x-axis represents the number of queried elements k, and the y-axis shows the aggregated average number of oracle calls required to find all marked elements.

4.2 Comparing Optimistic and Pessimistic Variants with a Single Custom Noise Model

When comparing the noise sensitivity of the two algorithms, the pessimistic version exhibited smaller differences across noise levels. For example, in the amplitude damping tests with four qubits, increasing the noise from noiseless to the lowest level resulted in a 3.19% increase in oracle calls for the pessimistic version, compared to a 3.85% increase for the optimistic version.

These differences in performance persist as the noise increases, leading to a total difference of 38.19% for pessimistic and 49.54% difference for optimistic, when comparing the noiseless algorithms against the highest noise tested. Figure 4.2 illustrates this as the graphs clearly show that the optimistic version exhibits a larger spread in its results, indicating that increases in noise levels have a greater impact on the number of oracle calls compared to the pessimistic version. Similar trends can be observed across all custom noise models.

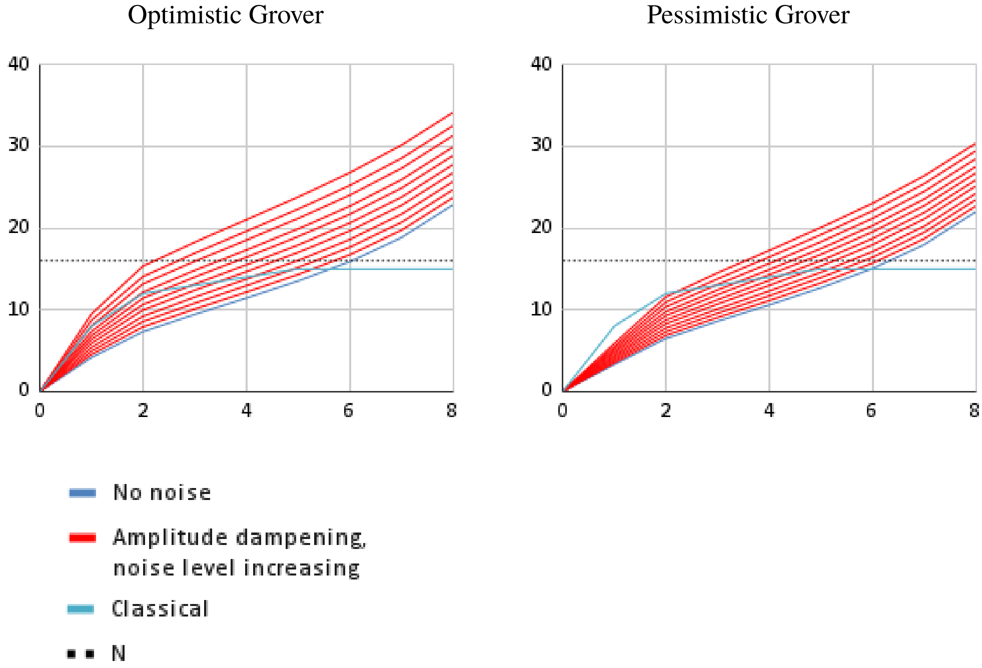

Figure 4.2: Average number of oracle calls for the Optimistic and Pessimistic variants of Grover's algorithm under varying amplitude damping noise levels, using 4 qubits. Noise levels range from 0.05% to 0.5% in increments of 0.05%. As the noise increases, each corresponding line in the plot shifts upward, indicating a higher number of oracle calls required. The x-axis represents the number of queried elements k, and the y-axis shows the aggregated average number of oracle calls required to find all marked elements.

4.3 Comparing Noise Levels

Table 4.1 presents the numerical data from the tests using the Bitflip model with 5 qubits, employing the Optimistic version of Grover's algorithm. The values in the table represent the unaggregated average number of oracle calls required to find the final queried item, given that there are k marked items.

Upon reviewing the data in Table 4.1, it is found that for some higher noise levels (≥ 0.003) and lower values of k (≤ 5), the results do not follow the expected trend, where noise being increased immediately leads to worse performance. This trend breaking is visible in the data with n=5 and n=6 for the Bitflip model and n=6 for the other models, being even more evident in the data for n=6. We also observe that the number of oracle calls remains relatively stable when k is closer to its maximum. For example, when k = 5, the difference between no noise and the maximum noise level is 9.61 oracle calls, or a 423.35% increase. In contrast, when k = 1, the difference is 150.52 oracle calls, or a 3010.4% increase, representing a substantial deviation.

K	**0.0**	**0.0005**	**0.001**	**0.0015**	**0.002**	**0.0025**	**0.003**	**0.0035**	**0.004**	**0.0045**	**0.005**
1	5.00	14.54	35.04	69.63	100.68	125.75	148.54	148.54	153.75	156.99	155.52
2	4.16	10.32	21.09	33.65	43.58	51.59	56.76	59.79	63.10	62.01	62.23
3	3.00	5.74	9.81	14.37	20.20	23.85	26.87	28.62	30.38	30.90	31.47
4	3.17	5.99	9.85	14.29	17.50	20.49	21.76	22.82	22.93	23.62	23.24
5	2.27	3.19	4.29	5.51	6.85	8.03	9.18	10.19	10.71	11.45	11.88
6	2.11	2.93	3.95	5.02	6.10	7.17	8.11	8.73	9.29	9.66	9.95
7	2.02	2.80	3.73	4.71	5.66	6.50	7.22	7.75	8.15	8.49	8.60
8	2.00	2.73	3.59	4.39	5.29	5.96	6.56	6.97	7.30	7.56	7.64
9	2.02	2.72	3.49	4.30	5.00	5.58	6.02	6.34	6.64	6.79	6.83
10	2.09	2.77	3.47	4.16	4.75	5.22	5.59	5.86	6.06	6.18	6.21
11	2.20	2.84	3.48	4.08	4.54	4.95	5.23	5.40	5.58	5.64	5.68
12	2.37	2.97	3.54	4.03	4.40	4.73	4.90	5.04	5.12	5.20	5.20
13	2.60	3.16	3.60	3.99	4.29	4.49	4.64	4.76	4.79	4.80	4.89
14	2.93	3.36	3.75	4.00	4.18	4.31	4.43	4.45	4.46	4.52	4.57
15	3.36	3.64	3.84	3.98	4.10	4.14	4.20	4.20	4.23	4.26	4.25
16	4.02	3.99	3.98	4.00	3.99	4.02	4.01	3.99	3.99	4.03	4.02

Table 4.1: Average number of oracle calls for Optimistic grover with 5 qubits and bit-flip noise for all noise levels tested. The first column shows the value of k during the test and the top row shows the level of noise used during the test.

Chapter 5

Discussion

As the database size N (where $N = n^2$) increases, the effect of quantum noise on Grover's algorithm becomes increasingly significant. As shown in Figure 4.1, performance begins to degrade starting at $n = 4$ in noisy conditions. For higher noise levels (such as `FakeAthensV2` or the maximum custom noise settings) the algorithm's performance deteriorates remarkably, requiring a very high amount of average oracle calls, indicating that Grover's algorithm has ceased to function as intended. This gap in performance worsens as the database size increases. These findings are consistent with previous research showing that noise effects scale with the size of the database [17]–[19]. No consistent trend was found showing that increasing the number of queried elements k changes the impact of noise on total performance when searching for all elements.

5.1 Noise Models and Levels

The choice and intensity of the noise model have a substantial impact on performance. The `FakeAthensV2` model and the custom models at max noise level (0.5%) already renders the algorithm entirely ineffective from $n = 4$ onwards, except from the amplitude damping model for pessimistic grover. It still significantly worsens but it is usable while $k < 3$. In contrast, under minimum custom noise settings (0.05%), the algorithm remains similar to the noiseless variant up to $n = 5$. However, in $n = 6$, performance declines sharply even under mild noise conditions.

Among custom noise types, the bit-flip model consistently resulted in the worst performance of the custom models, often comparable to `FakeAthensV2` at its highest levels. This is likely because bit-flip errors can directly "unmark" correct states by flipping their bits, which would dramatically disrupt the algorithm. In contrast, amplitude damping showed the best performance under noise, possibly because it primarily affects amplitudes without entirely flipping computational states.

As shown in Table 4.1, once the noise level exceeds approximately 0.003, the impact on algorithmic performance becomes less predictable. In cases with 5 and 6 qubits, performance does not consistently degrade with increasing noise, this is possibly due to the limited sample size or because the noise is strong enough to completely disrupt the algorithm so that differences in performance become harder to observe. This suggests that there is a limit to how much noise can effect the algorithm, which could simply be a product of the error threshold being surpassed.

5.2 Optimistic and Pessimistic Grover

The pessimistic version of Grover's algorithm performed better across all test, as shown in Figures 4.1 and 4.2. This suggests that Aldenbro and Skalski's [7] theory that, the pessimistic version of Grover's algorithm experiences less negative effects of noise is true. This is most likely due to the pessimistic circuit being shorter when it reduces the number of Grover iterations used, giving the noise less iterations to effect the algorithm, as opposed to the optimistic version where the noise can multiply more and have a larger effect. A shorter circuit limits the opportunity for noise to corrupt the quantum state and gives the better result.

While this may suggest that the pessimistic variant is preferable, both versions performed poorly under realistic or moderately high noise levels as the database size increased, which complicates the practical choice between them in real-world applications, where larger databases are likely to be the norm.

5.3 Limitations and Future Work

This study was restricted to simulations involving up to six qubits. While this range was sufficient to observe clear trends in how quantum noise affects Grover's algorithm, it does not capture the algorithm's behaviour on larger database sizes. Investigating higher qubit counts could offer a more comprehensive understanding of performance at scales closer to those expected in future quantum systems.

Additionally, the total number of simulation runs was constrained by time and computational resources. For instance, simulating Grover's algorithm with noise at six qubits required approximately 450 minutes, or around seven and a half hours, for each noise level. A greater number of repeated tests would strengthen the reliability of the results.

Another limitation lies in the choice of noise models. The realistic noise used in this study was based on a single simulated backend, `FakeAthensV2`, which mirrors the behaviour of a specific quantum device. While this provides a useful approximation of real-world noise, it does not account for the diversity of noise characteristics present in different quantum devices. Similarly, the custom noise models were simplified and isolated, representing specific types of errors individually.

Future work could expand this study in several directions. First, testing Grover's algorithm on a wider range of hardware backends and noise profiles would help assess the generality of the findings. Second, increasing the number of qubits and extending the simulations to more complex problem instances would provide a deeper understanding of performance limits. Lastly, perhaps incorporating more sophisticated error mitigation strategies or noise-adaptive variants of the algorithm could lead to practical improvements in robustness, contributing to more effective use of Grover's algorithm on near-term quantum devices.

5.4 Ethics and Sustainability

While the ethical and sustainability challenges of quantum computing are broad and still evolving, this study involved only limited simulations on classical hardware, without the use of real-world data, users, or applied systems. As such, it does not raise any immediate ethical concerns. Nonetheless, as quantum technologies move closer to real-world deployment, they pose significant implications for cybersecurity, environmental sustainability, and global equality that warrant continued attention.

One of the most direct ethical concerns is the impact of quantum technologies on cybersecurity. Algorithms such as Grover's and Shor's offer powerful capabilities that could undermine classical encryption schemes. Grover's algorithm, for example, can quadratically accelerate brute-force attacks on symmetric encryption [23], potentially compromising existing cryptographic infrastructure once noise-resilient, scalable quantum hardware becomes available. This, in turn, could have serious implications for the privacy of personal data and the confidentiality of sensitive government information [24], and would likely necessitate the development of more quantum-resistant cryptographic methods.

The high cost and complexity of quantum technologies currently restrict access to a small number of powerful companies and countries. This may give rise to a "quantum divide," analogous to the digital divide, in which those lacking access fall behind in technological and economic development [25]. Promoting equitable access through international collaboration, education, and open innovation will be essential to ensuring that quantum advancements contribute to socially sustainable progress.

From an environmental perspective, the most relevant sustainability issue for quantum computing is energy consumption. Quantum computers have the potential to use significantly less energy than classical supercomputers for certain problems, due to the lower time complexity of some quantum algorithms [26]. However, there is currently no standardized method for measuring or comparing the energy use of quantum systems, which makes concrete assessments difficult [26]. Still, early indications suggest that quantum systems could help reduce the growing energy footprint of AI and data centers [27]. If developed with energy efficiency in mind, quantum computing could play an important role in making high-performance computing more sustainable.

Chapter 6

Conclusions

This study set out to investigate how quantum noise influences the number of oracle calls required by Grover's algorithm to successfully identify all marked elements in an unsorted database. Using simulations in Qiskit, we calculated the total number of oracle calls required under three conditions: ideal (noise-free), custom noise models, and realistic noise derived from quantum hardware. These results were compared to one another and classical linear search to assess the algorithm's performance in noisy conditions.

Our findings demonstrate that quantum noise significantly degrades the performance of Grover's algorithm, particularly as the database size increases. The pessimistic variant of the algorithm consistently outperformed the optimistic version under noisy conditions, likely due to its shorter circuit depth, which reduces the cumulative effect of noise. Among the noise models tested, bit-flip errors had the most detrimental impact, while amplitude damping was the least disruptive. Notably, even low noise levels (0.05%) led to substantial performance degradation for larger databases (e.g., N=64), rendering the algorithm impractical in such cases.

These results highlight the challenges of implementing Grover's algorithm on current NISQ-era quantum devices, where noise remains a critical limitation. The study underscores the importance of developing noise-resilient quantum algorithms and error mitigation strategies to harness the potential of quantum computing in practical applications. Future research should explore larger-scale simulations, diverse noise environments, and adaptive algorithmic approaches to improve robustness in noisy settings.

Bibliography

[1] Microsoft Azure Quantum Team, *Cryptography in the era of quantum computers*, 2020. [Online]. Available: https://azure.microsoft.com/en-us/blog/quantum/2020/02/26/cryptography-quantum-computers/ (Accessed: 2025-07-01).

[2] M. Guenot, "Can quantum computing crack the biggest challenges in health?", *Nat Med*, vol. 31, pp. 4–7, 2025. DOI: 10.1038/s41591-024-03369-w.

[3] A. Abbas, et al., "Challenges and opportunities in quantum optimization", *Nat Rev Phys*, vol. 6, pp. 718–735, 2024. DOI: 10.1038/s42254-024-00770-9.

[4] L. K. Grover, *A fast quantum mechanical algorithm for database search*, 1996. DOI: 10.1145/237814.237866.

[5] C. Lavor, L. R. U. Manssur, and R. Portugal, *Grover's algorithm: Quantum database search*, 2003. arXiv: quant-ph/0301079 [quant-ph].

[6] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information: 10th Anniversary Edition*. Cambridge University Press, 2010.

[7] D. Aldenbro and A. Skalski, "Comparing grover's algorithm to linear search when finding multiple queried elements", Dissertation, 2024. [Online]. Available: https://kth.diva-portal.org/smash/record.jsf?pid=diva2:1886690 (Accessed: 2025-07-01).

[8] N. S. Yanofsky and M. A. Mannucci, *Quantum Computing for Computer Scientists*. Cambridge Univ. Press, 2008.

[9] M. Boyer, G. Brassard, P. Høyer, and A. Tapp, "Tight bounds on quantum searching", *Fortschritte der Physik*, vol. 46, no. 4-5, pp. 493–505, 1998. DOI: 10.1002/(SICI)1521-3978(199806)46:4/5<493::AID-PROP493>3.0.CO;2-P.

[10] Z. Cai, et. al., "Quantum error mitigation", *Rev. Mod. Phys.*, vol. 95, p. 045 005, 2023. DOI: 10.1103/RevModPhys.95.045005.

[11] I. L. Chuang, N. Gershenfeld, and M. Kubinec, “Experimental implementation of fast quantum searching”, *Phys. Rev. Lett.*, vol. 80, pp. 3408–3411, 1998. DOI: 10.1103/PhysRevLett.80.3408.

[12] Y. Nakamura, Y. A. Pashkin, and J. S. Tsai, “Coherent control of macroscopic quantum states in a single-cooper-pair box”, *Nature*, vol. 398, pp. 786–788, 1999. DOI: 10.1038/19718.

[13] IBM, *IBM quantum platform*, 2025. [Online]. Available: https://www.ibm.com/quantum (Accessed: 2025-07-01).

[14] D. J. Bernstein, *Post-Quantum Cryptography*. Springer Berlin Heidelberg, 2009. DOI: 10.1007/978-3-540-88702-7_1.

[15] A. Daskin and S. Kais, “Decomposition of unitary matrices for finding quantum circuits: Application to molecular hamiltonians”, *The Journal of Chemical Physics*, vol. 134, no. 14, 2011. DOI: 10.1063/1.3575402.

[16] M. Schuld, I. Sinayskiy, and F. Petruccione, “An introduction to quantum machine learning”, *Contemporary Physics*, vol. 56, no. 2, pp. 172–185, 2014. DOI: 10.1080/00107514.2014.964942.

[17] B. Pablo-Norman and M. Ruiz-Altaba, “Noise in grover’s quantum search algorithm”, *Phys. Rev. A*, vol. 61, p. 012 301, 1999. DOI: 10.1103/PhysRevA.61.012301.

[18] D. Shapira, S. Mozes, and O. Biham, “Effect of unitary noise on grover’s quantum search algorithm”, *Phys. Rev. A*, vol. 67, p. 042 301, 2003. DOI: 10.1103/PhysRevA.67.042301.

[19] P. Salas, “Noise effect on grover algorithm”, *Eur. Phys. J. D*, vol. 46, pp. 365–373, 2008. DOI: 10.1140/epjd/e2007-00295-1.

[20] IBM Quantum, *IBM quantum documentation and guides*, 2024. [Online]. Available: https://quantum.ibm.com/docs (Accessed: 2025-07-01).

[21] Qiskit Aer Project, *Qiskit aer documentation*, 2024. [Online]. Available: https://qiskit.github.io/qiskit-aer/ (Accessed: 2025-07-01).

[22] G. Turati, A. Marruzzo, M. F. Dacrema, and P. Cremonesi, *An empirical analysis on the effectiveness of the variational quantum linear solver*, 2024. arXiv: 2409.06339 [quant-ph].

[23] M. Kumar, “Post-quantum cryptography algorithm’s standardization and performance analysis”, *Array*, vol. 15, p. 100 242, 2022. DOI: 10.1016/j.array.2022.100242.

[24] K. P. M. Taddeo A. Blanchard, "Consider the ethical impacts of quantum technologies in defence — before it's too late", *Nature*, vol. 634, pp. 779–781, 2024. DOI: 10.1038/d41586-024-03376-4.

[25] K. Holley, "Bridging the quantum divide", *Forbes*, 2023. [Online]. Available: https://www.forbes.com/councils/forbestechcouncil/2023/06/20/bridging-the-quantum-divide/ (Accessed: 2025-07-01).

[26] S. Chen, "Are quantum computers really energy efficient?", *Nature Computational Science*, vol. 3, pp. 457–460, 2023. DOI: 10.1038/s43588-023-00459-6.

[27] T. Constantino, "Is quantum computing an unlikely answer to ai's looming energy crisis?", *Forbes*, 2024. [Online]. Available: https://www.forbes.com/sites/torconstantino/2024/10/02/is-quantum-computing-an-unlikely-answer-to-ais-looming-energy-crisis/ (Accessed: 2025-07-01).

TRITA-EECS-EX-2025:309
Stockholm, Sverige 2025

www.kth.se