COMPARISON OF QUANTUM AND CLASSICAL ALGORITHMS

Anwer Saeed  
*Computer Science*   
*FAST University*Karachi, Pakistan

Rahim Khan   
*Computer Science*   
*FAST University*Karachi, Pakistan

Quantum computing basically represents a paradigm shift in computational methods, promising exponential or quadratic speedups for specific problems compared to classical approaches. This study explores and evaluates the comparative performance of quantum and classical algorithms in two critical domains: database searching and integer factorization. Grover's quantum search algorithm is benchmarked against classical linear search for unsorted databases, showcasing its quadratic advantage in oracle calls under ideal conditions. Concurrently, Shor's quantum factoring algorithm is evaluated against the General Number Field Sieve (GNFS) for prime factorization, highlighting its exponential theoretical speedup. Both quantum algorithms leverage quantum principles such as superposition and interference, demonstrating superior performance in time complexity while tackling practical challenges like circuit depth and noise susceptibility. Acquired results depict the quantum speedup's potential, underscoring the need for further advancements in hardware to mitigate noise and scalability constraints.

Keywords—Quantum, Grover, Shor, Qiskit, GNFS

# INTRODUCTION

As data-driven technologies proliferate, efficient algorithms for database search and cryptographic applications have become increasingly significant. Classical algorithms like linear search scale poorly with the exponential growth of data, while traditional cryptographic systems, such as RSA, rely on the computational infeasibility of factoring large integers. Quantum algorithms, such as Grover's search and Shor's factoring algorithms, present a revolutionary approach, leveraging quantum mechanics to offer superior computational efficiencies.

Grover's algorithm achieves a quadratic speedup over classical methods by iteratively amplifying the amplitudes of marked states in an unsorted database. In comparison, Shor's algorithm offers an exponential speedup for prime factorization by efficiently exploiting periodicity using the Quantum Fourier Transform (QFT). Despite these advantages, quantum algorithms face practical challenges, such as high circuit depths, sensitivity to noise, and resource requirements, limiting their application on current noisy intermediate-scale quantum (NISQ) hardware.

This project explores the theoretical and practical implications of these algorithms by implementing and analyzing their performance using the Qiskit framework. Grover's algorithm is tested under "optimistic" and "pessimistic" configurations to balance success probability and computational cost. Shor's algorithm is benchmarked against the GNFS, a state-of-the-art classical factoring algorithm. By comparing these algorithms' theoretical performance and practical implementation results, this study evaluates the potential and limitations of quantum computing in these domains.

# LITERATURE REVIEW

Quantum computing has come across as a transformative field, driven by its potential to solve problems intractable for classical systems. Here is a review regarding the contributions and methodologies for Grover's search and Shor's factoring algorithms, with a focus on their theoretical foundations, practical implementations, and comparative performance with classical algorithms.

## GROVER'S ALGORITHM

Introduced by Lov Grover in 1996, Grover's algorithm is a seminal quantum algorithm designed for unsorted database searches. It achieves a time complexity of O(sqrt(N)), significantly exceeding the classical linear search's O(N) complexity. Grover's algorithm depends on two key components: an oracle that marks the target states with a phase flip along with a diffuser that amplifies these states through inversion about the mean. Variants of Grover's algorithm, such as the "optimistic" and "pessimistic" configurations, adapt the number of iterations to optimize success probability or circuit depth, as shown in studies like Aldenbro and Skalski[1] These variants highlight the trade-offs between algorithmic efficiency and hardware constraints, such as noise resilience and qubit coherence.

## SHOR'S ALGORITHM

Proposed by Peter Shor in 1994, Shor's algorithm revolutionized cryptography by providing a polynomial-time solution to integer factorization. The algorithm utilizes quantum period finding to deduce factors of a composite number N by leveraging the QFT and controlled modular arithmetic operations. Its exponential speedup over classical algorithms, such as GNFS, has profound implications for cryptographic systems like RSA.[2] However, the algorithm's implementation requires precise quantum control, as highlighted in studies comparing its theoretical efficiency with practical constraints (e.g., noise, decoherence). Recent advancements in modular arithmetic and optimization of quantum circuits have improved its scalability on NISQ devices, as noted in quantum computing research by Saeed et al.

## CLASSICAL COMPARISONS

The General Number Field Sieve (GNFS) remains the most efficient classical algorithm for prime factorization, with a complexity of O(exp((log(N))^1/3(log log(N))^2/3)). While substantially slower than Shor's algorithm in theoretical terms, GNFS benefits from decades of optimization and hardware implementation, making it suitable for current applications. Likewise, linear search, despite its inefficiency, offers reliability and simplicity for unsorted database searches, as shown in controlled simulations.

## PRACTICAL CONSIDERATIONS

Quantum algorithms encounter substantial implementation challenges, particularly on NISQ hardware. Noise, decoherence, and limited qubit counts hinder the algorithms' real-world applicability. Grover's and Shor's algorithms are specifically sensitive to these factors, as their performance relies on precise quantum operations rather than large circuit depths. Simulation frameworks like Qiskit provide invaluable insights into these algorithms' behavior under ideal conditions, allowing researchers to explore their theoretical potential and identify practical bottlenecks.

## RESEARCH GAPS

Despite extensive research, significant gaps remain in comprehending quantum algorithms' practical performance. Current studies frequently depend on idealized simulations, overlooking noise and error correction's impact on scalability. In addition to this, the energy efficiency of quantum algorithms, a vital factor for sustainable computing, is still underexplored. Future work should address these issues by integrating noise models into simulations, investigating hybrid quantum-classical approaches, and exploring alternative quantum algorithms for similar problem domains.

# Comparison of Grover and Linear Search Algorithms when finding multiple elements

## BACKGROUND

### Grover's algorithm, introduced in 1996, is one of the most significant quantum algorithms due to its potential for quadratic speedups in unsorted database searches. For a database of size N, Grover's algorithm can find a single target in O(sqrt(N)) time, compared to O(N) for classical methods. When searching for multiple targets, the algorithm can be adjusted to balance between success probability and execution time. Two configurations, optimistic and pessimistic, differ in the number of iterations performed:

#### Optimistic Version: The Optimistic version of Grover's algorithm is designed to maximize success probability by using the theoretically optimal number of iterations for amplitude amplification. This approach ensures that the probability of measuring a target state is as high as possible, often exceeding 95% in noiseless environments. However, the success rate can fluctuate when the problem size (N) or the ratio of targets (k/N) changes significantly, as the algorithm’s sensitivity to the exact number of iterations may lead to overshooting or undershooting the peak success probability. In terms of circuit depth, the Optimistic version incurs a higher computational cost, as it performs more iterations to achieve maximum amplitude amplification. Circuit depth grows with O(sqrt(Nk)), making this version suitable for environments where noise is minimal, such as simulations or idealized quantum systems. For quantum speedup, the Optimistic version delivers near-theoretical performance, scaling as O(sqrt(Nk)), but the increased depth may introduce practical challenges on current noisy quantum hardware.

#### Pessimistic Version: The Pessimistic version of Grover's algorithm takes a more conservative approach by performing fewer iterations to avoid overshooting the probability peak. While this strategy may not achieve the absolute maximum success probability, it ensures consistently reliable success rates, often ranging between 85% and 90% under noiseless conditions, regardless of problem size or target ratio. This stability makes it particularly appealing for practical use cases where variability in success rates must be minimized. The circuit depth for the Pessimistic version is significantly lower than the Optimistic version, as the reduced number of iterations results in a more resource-efficient implementation. This lower depth makes the Pessimistic version well-suited for noisy quantum hardware, where shallower circuits are less prone to decoherence and errors. In terms of quantum speedup, the Pessimistic version performs robustly, closely approaching the theoretical limit while offering greater reliability and practicality for real-world quantum systems.

#### The algorithms were implemented using Qiskit's simulation environment, leveraging quantum circuit elements like the Hadamard transform, oracle construction, and Grover diffusion. Additionally, classical linear search was implemented for benchmarking.

## METHODOLOGY:

This study implemented and analyzed the performance of Grover's algorithm in both "optimistic" and "pessimistic" configurations, comparing its results against classical linear search for unsorted databases. Grover's algorithm leverages the principles of quantum superposition and interference, amplifying the probability of target states while suppressing others. The algorithm was implemented using Qiskit, with the quantum circuit constructed dynamically based on the problem size N and the number of target elements k.[3] The circuit begins with an initialization phase, where all qubits are placed into a uniform superposition using Hadamard gates. The core of the algorithm involves iterative application of two main components: the oracle and the diffuser. The oracle is a quantum subroutine that marks target states by applying a phase flip (inversion) to their corresponding quantum states. This is achieved using multi-controlled NOT gates (MCX) implemented via ancillary qubits. The diffuser, often called the "inversion about the mean," performs amplitude amplification by flipping the quantum state's amplitudes about their average, enhancing the probabilities of the marked target states. Depending on the configuration, the number of Grover iterations (sqrt(N/k)) for optimistic and 0.58278(sqrt(N/k)) for pessimistic is dynamically calculated to balance success probability and execution time. Classical linear search, implemented as a baseline, iterates through the search space sequentially, counting oracle calls. Simulations were conducted using Qiskit's AerSimulator, which allows the execution of quantum circuits in a noiseless environment. The circuits were measured after execution, and metrics such as success rate, circuit depth, and the number of oracle calls were extracted for analysis. Visualization of the quantum circuit was also performed, showcasing the complete Grover’s circuit as well as the operations for the first iteration. Results were analyzed to quantify the quantum speedup and the suitability of both configurations for different problem sizes and target distributions.

Below are the figures for the circuit diagrams:



Figure 1:Figure 1: start size=4, end size=64, number of targets(k)=5, grover version=optimistic

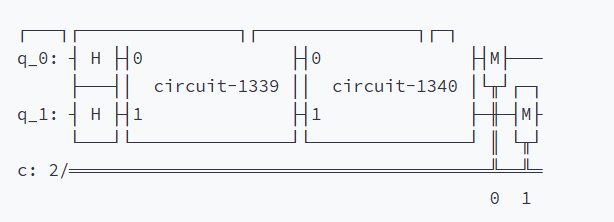
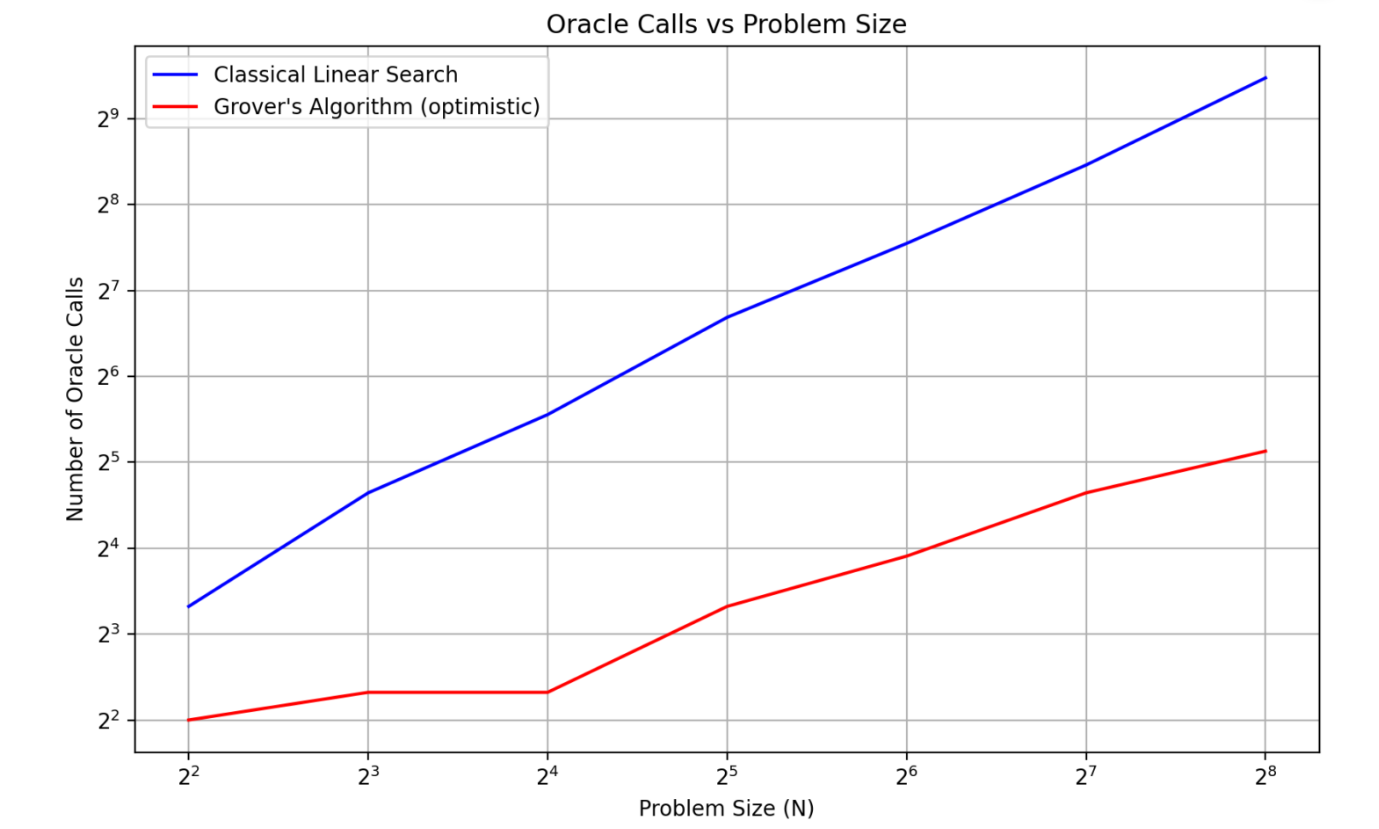
****

Figure 2:start size=4, end size=512, number of targets(k)=15, grover version=optimistic

## RESULTS



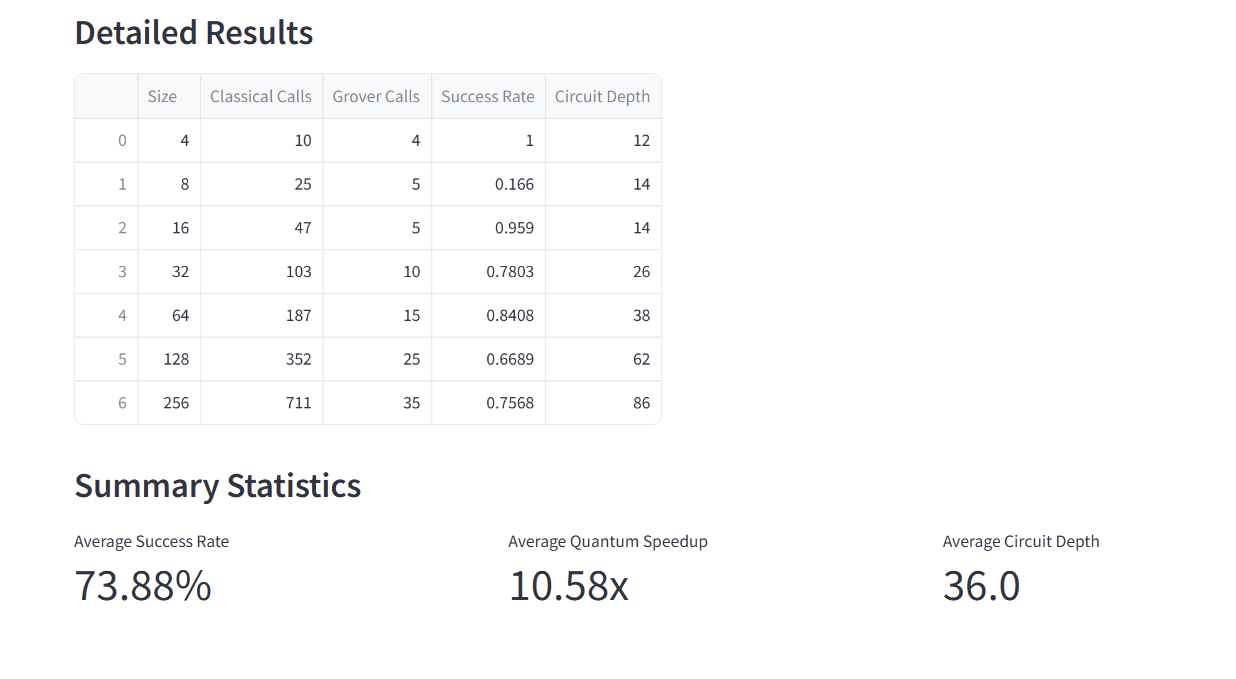
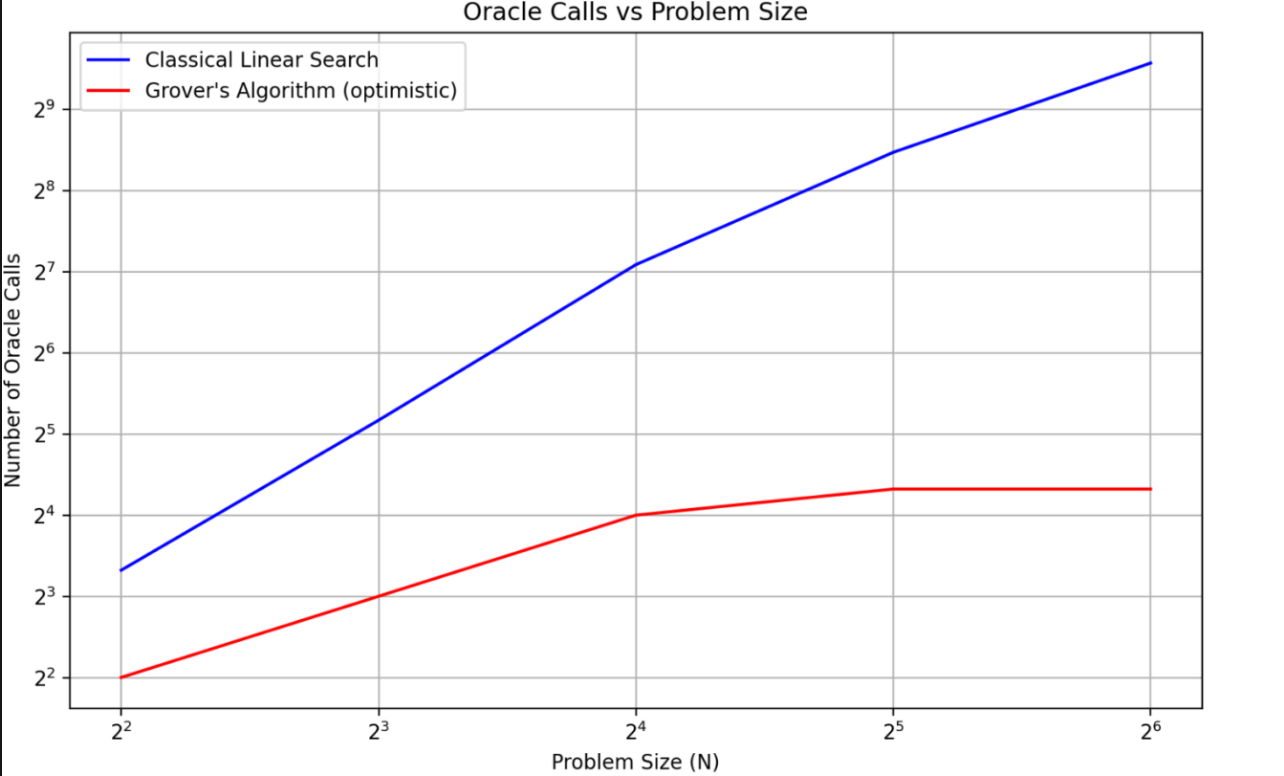


Figure 3:Test case 1: Start size=4 End size=256 Number of Targets(k)=5 Grover Version=Optimistic



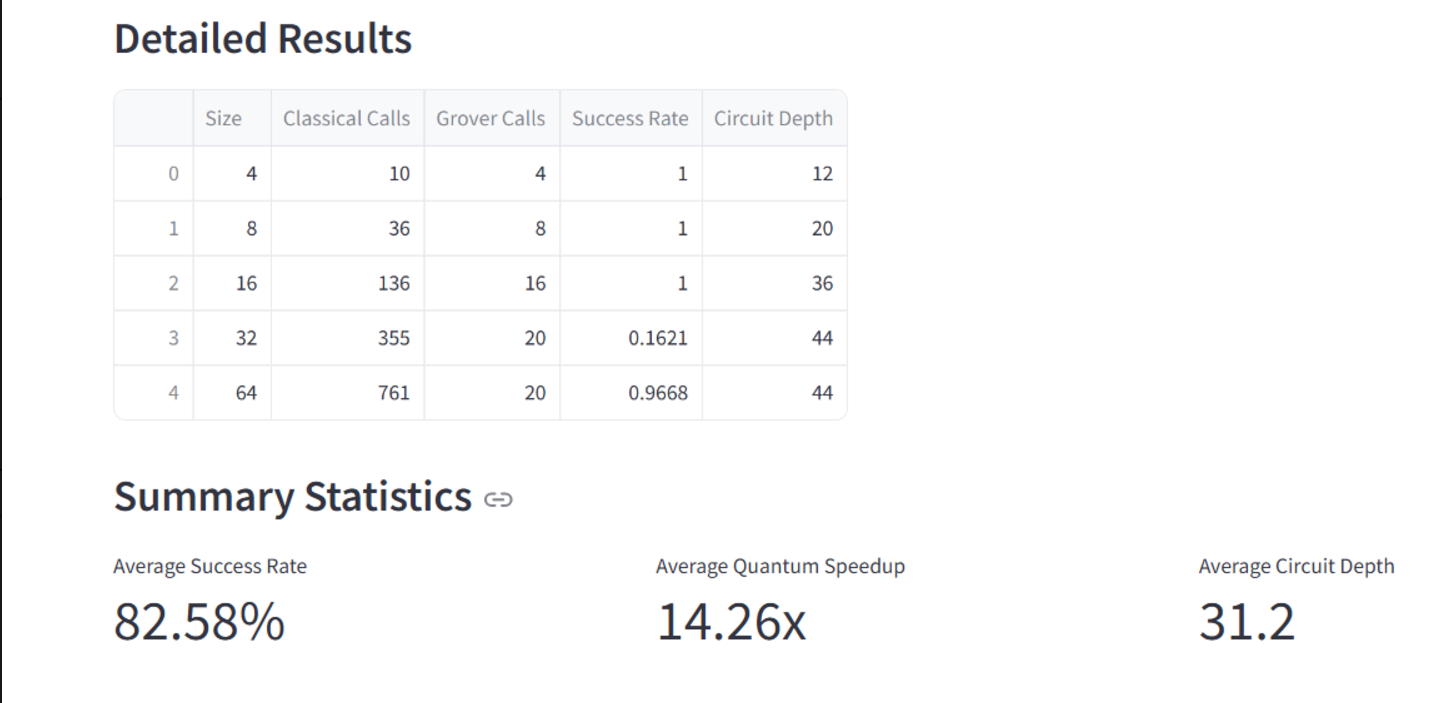


Figure 4: Test case 2: Start size=4 End size=64 Number of Targets(k)=20 Grover Version=Optimistic

## LIMITATIONS:

Simulation Constraints: Simulations were performed under ideal, noiseless conditions using Qiskit's Aer Simulator. In practical quantum hardware, noise and decoherence could significantly affect performance, particularly for larger circuits or longer execution times.

Scalability: Due to the exponential increase in computational resources required for simulating large quantum circuits, the experiments were limited to N=26N = 2^6N=26. This restriction may not fully represent the performance trends for larger databases.

Static Oracle Costs: The cost of constructing oracles, which depends on the specific application, was not included in the analysis.

Probabilistic Nature: The probabilistic outcomes of quantum measurements necessitate multiple runs, increasing overall execution time.

## FUTURE WORKS:

You can cite your references in text Noise-Resilient Studies: Incorporate simulations with realistic noise models to understand the practical applicability of Grover's algorithm on current and near-future quantum hardware.

Expanded Database Sizes: Explore performance trends for N>2^6, potentially utilizing hybrid simulation approaches.

Alternative Algorithms: Compare Grover's algorithm with other quantum search methods to identify use-case-specific advantages.

Energy Efficiency: Evaluate the energy consumption of Grover's algorithm on quantum hardware, especially for larger problem instances.

Hybrid Strategies: Investigate the combination of classical preprocessing with quantum search for better overall efficiency.

## CONCLUSION:

Grover's algorithm demonstrates clear advantages over classical linear search for database sizes where the proportion of targets (k/N) remains below a threshold (≈0.3). Both configurations—optimistic and pessimistic—offer trade-offs between circuit depth and success probability, with the latter showing slight practical advantages. The results highlight the potential of quantum algorithms in search problems while emphasizing the need for future work to bridge the gap between theoretical simulations and real-world applications.

# Comparison of Shor’s Quantum Factoring Algorithm and General Number Field Sieve (GNFS)

## BACKGROUND

Factorization has been a central challenge in computational mathematics for decades. Classical algorithms such as trial division, Pollard’s rho method, and GNFS have made significant progress in factoring numbers of increasing sizes, but they are hindered by exponential or sub-exponential time complexity. The GNFS, with a complexity of O(exp((log N)^(1/3) · (log log N)^(2/3))), represents the pinnacle of classical factorization, leveraging advanced number theory and lattice-based techniques to handle numbers in the range of hundreds of digits.

In contrast, quantum algorithms leverage the principles of superposition and entanglement to achieve exponential speedup for certain problems. Shor's Algorithm, introduced in 1994, demonstrated the potential to solve the factorization problem in polynomial time with a complexity of O((log N)² · log(log N) · log(log(log N))).While its theoretical implications are profound, practical implementation faces significant challenges due to the current limitations of quantum hardware, including qubit fidelity, error rates, and decoherence.

## METHODOLOGY:

This project involved a comparative analysis of Shor’s Algorithm and the General Number Field Sieve (GNFS) to evaluate their respective capabilities for factoring composite integers. The comparison was structured to analyze computation times, resource requirements, and scalability for various input sizes (N), with special emphasis on the quantum circuit used in Shor's Algorithm.

#### Shor's Algorithm Implementation: Shor’s Algorithm was implemented using Qiskit, a quantum computing framework. The algorithm involves three primary steps:

#### Quantum Superposition and Modular Exponentiation: A quantum register is initialized, and Hadamard gates are applied to create a superposition of states. Controlled modular exponentiation is performed using an auxiliary register to encode the modular multiplication operation a^x mod N, where a is a randomly chosen base coprime to N.

#### Quantum Fourier Transform (QFT): An Inverse Quantum Fourier Transform is applied to extract the periodicity of the modular exponentiation result. This periodicity helps deduce the factors of N.

#### Classical Post-Processing: The period obtained from the quantum step is used in classical computations (e.g., calculating the greatest common divisor) to find the factors of N.

#### Quantum Circuit Diagram Explanation: The quantum circuit for Shor’s Algorithm was visualized and analyzed for clarity. It consists of three distinct quantum registers:

#### Phase Estimation Register: Used for quantum phase estimation to determine the periodicity of the modular function. The size of this register is proportional to 2\*log2(N).

#### Target Register: Represents the values of the modular function. It holds xxx, the integer whose power is being taken modulo N.

#### Auxiliary Register: Used for modular arithmetic computations, specifically for handling carry operations during controlled addition and subtraction.

The circuit diagram is divided into several logical sections:

Superposition Initialization: Hadamard gates are applied to the phase estimation register, creating a uniform superposition over all possible states.

State Initialization of Target Register: The target register is initialized to ∣1⟩, representing the modular arithmetic identity.

Controlled Modular Exponentiation: This section involves applying controlled gates for a^x mod N, represented as a block in the circuit for clarity. Internally, this involves complex subroutines for modular multiplication and reduction.

Inverse QFT: The phase estimation register undergoes an inverse Quantum Fourier Transform, implemented using a series of controlled rotations and swaps, which translates the periodicity encoded in quantum states into measurable outputs.

Measurement: The quantum states of the phase estimation register are measured, collapsing the superposition into a result that provides insight into the periodicity of a^x mod N.[3]

The circuit’s complexity was analyzed, revealing that for N=10055, the algorithm required 54 qubits and a depth of over 5000 gates. Visualizations demonstrated the modular exponentiation block as the most resource-intensive component due to its nested controlled operations.

#### GNFS Implementation: GNFS was implemented using classical libraries optimized for efficient factorization. The algorithm proceeds through five main steps:

1. Polynomial Selection: A polynomial f(x) is chosen that approximates the number N.

2. Sieving: Candidate integers are identified that are smooth with respect to a predefined bound, using lattice sieving techniques.

3. Matrix Construction and Reduction: A sparse matrix is constructed from the sieved integers and reduced using linear algebra to find dependencies.

4. Relation Extraction: Dependencies in the matrix are used to generate congruences modulo N.

5. Factorization: The congruences are used to compute factors of N.

#### Performance Metrics: To compare Shor's Algorithm and GNFS, key performance metrics were evaluated:

#### Computation Time: Measured runtime for factoring each number.

#### Resource Requirements: For Shor's Algorithm, qubit count and circuit depth were analyzed. For GNFS, smoothness bounds and sieving efficiency were assessed.

#### Quantum Speedup: The ratio of GNFS time to Shor’s time was calculated and plotted across input sizes to highlight quantum performance advantages.

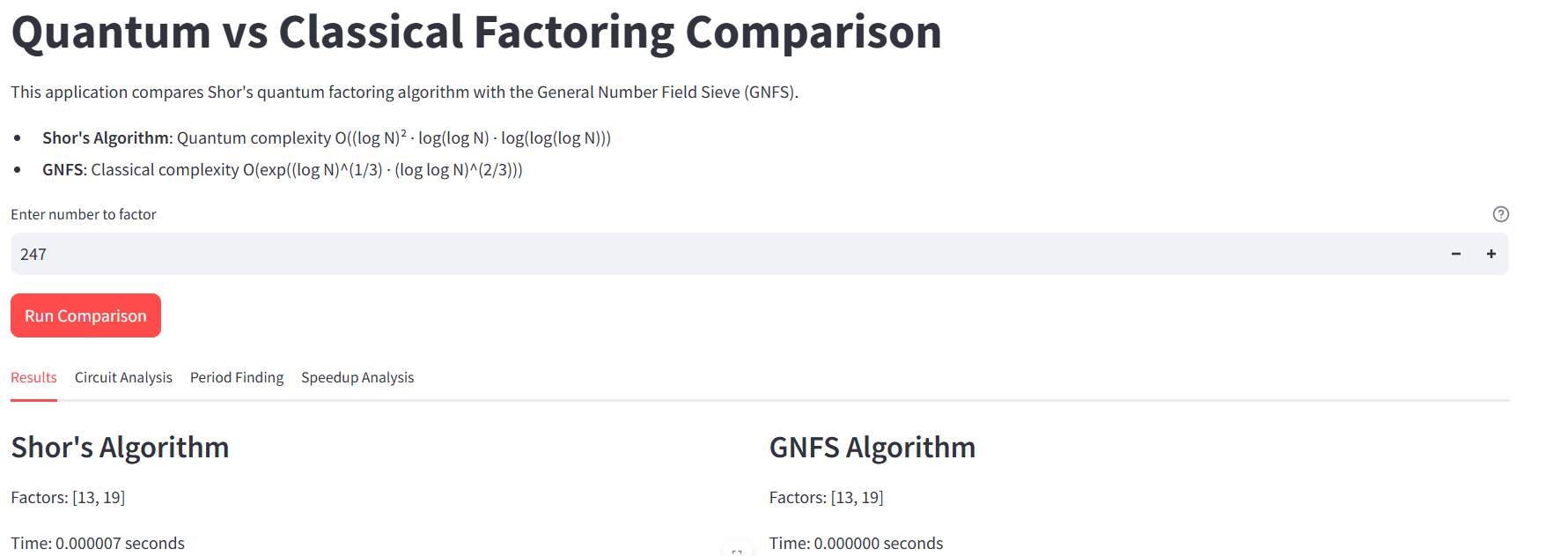
#### Visualization Enhancements: To better communicate the results:

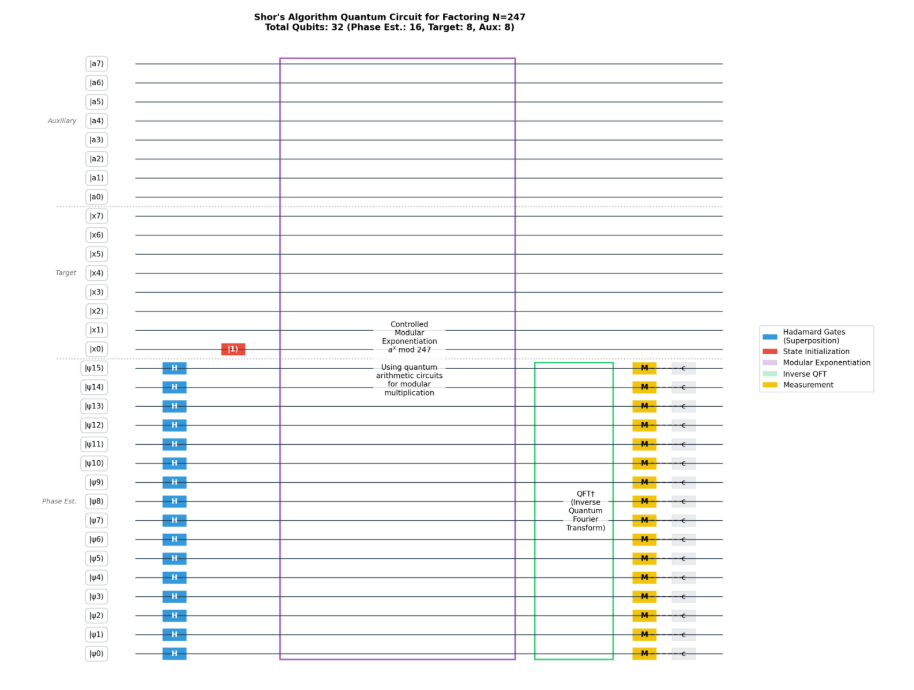
#### A comparative graph was created, illustrating the quantum speedup (GNFS time/Shor time) for various values of N. It highlighted that as N grows, Shor's Algorithm significantly outpaces GNFS in simulated performance.

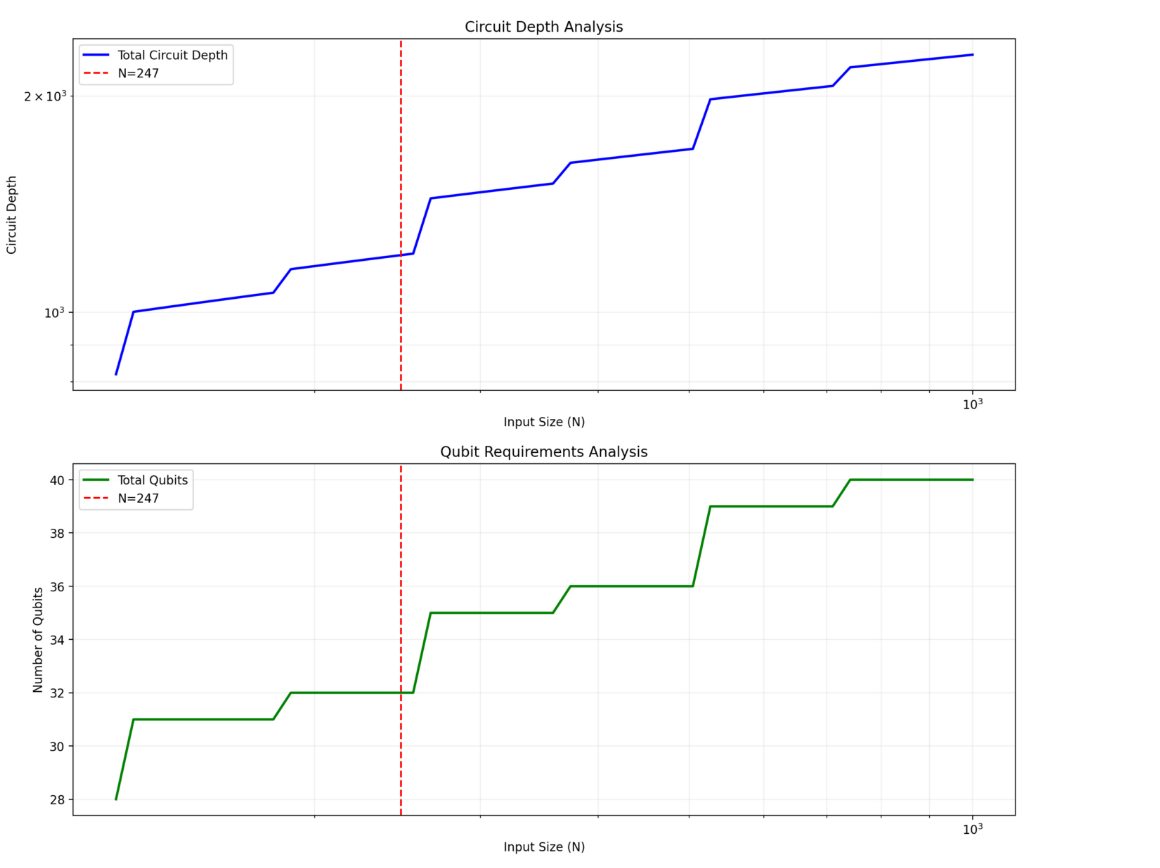
#### The period-finding histogram visualized the measurement outcomes from Shor’s Algorithm, showing peaks at expected periodic intervals, thereby validating the quantum circuit’s operation.

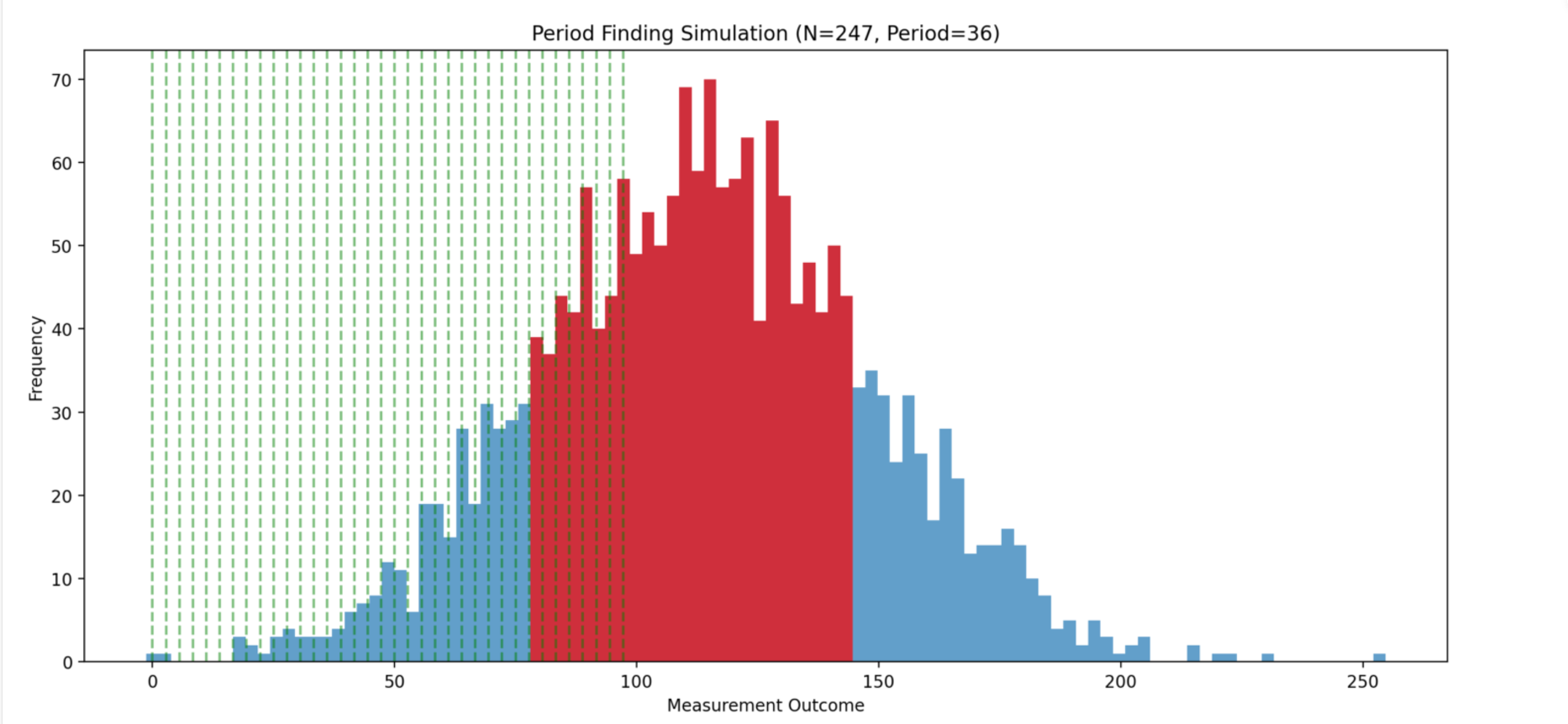
#### The quantum circuit diagram was annotated with detailed labels to highlight each stage of computation, providing users with an intuitive understanding of the algorithm's inner workings.

## RESULTS:

****

****

****

****

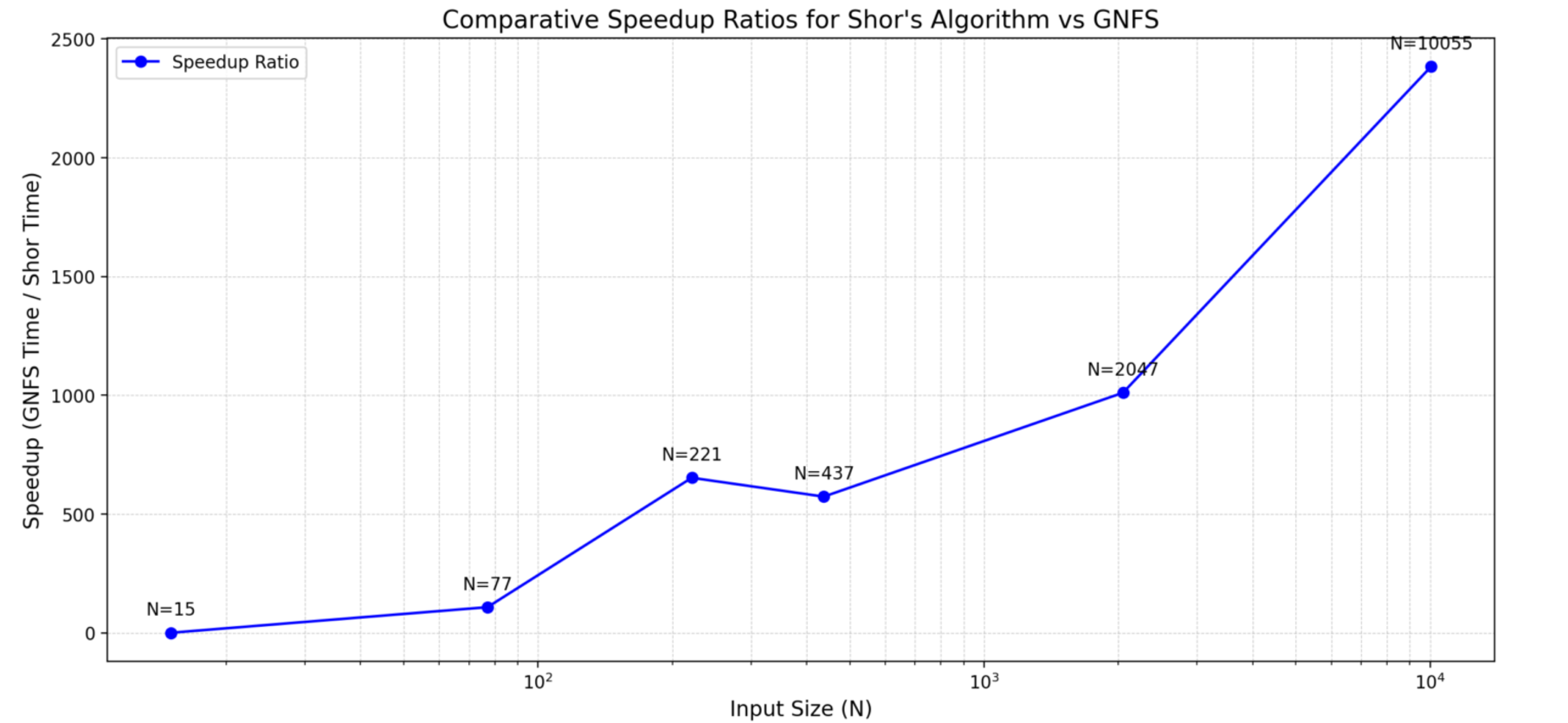
****

Figure 5:Test Case 1: Number to Factor = 247

## LIMITATIONS:

While Shor’s Algorithm demonstrated superior theoretical performance, several practical limitations hinder its real-world applicability. The algorithm’s qubit requirements grow logarithmically with input size, necessitating high-fidelity quantum devices with error correction capabilities, which remain beyond current technological capabilities. Furthermore, the simulations relied on idealized quantum environments, masking the effects of noise, decoherence, and hardware imperfections.

GNFS, while robust and well-optimized, struggles with scalability for input sizes beyond a few hundred digits. The algorithm’s dependence on smoothness bounds and lattice sieving introduces significant computational overhead, particularly for extremely large inputs. Moreover, its deterministic nature offers no scope for leveraging parallelism as effectively as quantum algorithms.

## FUTURE WORKS:

To bridge the gap between theoretical and practical performance, future work should focus on improving quantum hardware and error correction techniques to enable reliable execution of Shor’s Algorithm for meaningful input sizes. Hybrid algorithms that combine classical pre-processing with quantum post-processing could also mitigate the resource demands of pure quantum implementations. On the classical side, advancements in distributed computing and parallel algorithms could further optimize GNFS for larger-scale applications.

Additionally, extending this study to include alternative quantum algorithms, such as Grover’s search for optimization in factorization or variational approaches for noise-resilient computation, would provide a more comprehensive view of the field. Expanding the analysis to include encryption-breaking benchmarks, such as factoring RSA-2048, would offer practical insights into the cryptographic implications of these algorithms.

## CONCLUSION:

This project highlights the transformative potential of quantum computing in solving classically intractable problems. Shor’s Algorithm, with its exponential speedup, represents a paradigm shift in computational theory, challenging the very foundation of modern cryptography. However, its practical deployment remains constrained by current hardware limitations. GNFS, on the other hand, stands as a testament to the ingenuity of classical algorithms, offering reliable performance for factorization within its feasible range.

The comparative analysis underscores the need for continued investment in quantum technologies while refining classical methods to address immediate challenges. As quantum hardware matures, the balance between classical and quantum approaches will shape the future of secure communication and computational problem-solving. This study serves as a step toward understanding this delicate interplay, paving the way for breakthroughs in both domains.

##### References

[1] D. ALDENBRO and A. SKALSKI, “Comparing Grover’s algorithm to linear search when finding multiple queried elements,” *Digitala Vetenskapliga Arkivet*, 2024, [Online]. Available: https://www.diva-portal.org/smash/record.jsf?pid=diva2%3A1886690&dswid=-4489

[2] S. M. Hamdi, S. T. Zuhori, F. Mahmud, and B. Pal, “A Compare between Shor’s quantum factoring algorithm and General Number Field Sieve,” in *2014 International Conference on Electrical Engineering and Information & Communication Technology*, Dhaka, Bangladesh: IEEE, Apr. 2014, pp. 1–6. doi: 10.1109/ICEEICT.2014.6919115.

[3] M. A. Shafique, A. Munir, and I. Latif, “Quantum Computing: Circuits, Algorithms, and Applications,” *IEEE Access*, vol. 12, pp. 22296–22314, 2024, doi: 10.1109/ACCESS.2024.3362955.