Quantum Advantage in Maze Pathfinding: Comparing Grover's Algorithm with Classical A* Search in Complex Environments

Authors: Abishek K, Danushri S P G , Vishakan S, Kaushik G, Sriram M
School of Computer Science and Engineering, Vellore Institute of Technology, Vellore, India

Abstract

Finding efficient paths through complex environments remains a core problem in fields such as robotics, AI navigation, and logistics optimization. Classical algorithms like A* are well-established for such tasks, but they become increasingly resource-intensive as maze size and structural complexity grow. Inspired by Grover's quantum search algorithm—which offers a theoretical O(√N) speedup for unstructured problems—we investigate its application in maze pathfinding. Building on prior work in quantum search (Giri & Korepin, 2016) and reinforcement-driven quantum agents (Pozza et al., 2022), we implement both Grover's algorithm and classical A* within a custom maze generation framework. Our experimental comparison reveals that while A* explores significantly more nodes as maze size increases, Grover's method retains its sublinear behavior in theory. However, classical hardware simulation introduces practical bottlenecks that offset expected gains. These results underscore the emerging potential of quantum search in spatial reasoning and reinforce the importance of hybrid strategies until scalable quantum hardware becomes viable.

Keywords: quantum computing, pathfinding algorithms, Grover's algorithm, A* search, maze generation, computational complexity

1. Introduction

Pathfinding algorithms are fundamental components in numerous computational domains, including robotics, network routing, game development, and logistics optimization. Classical approaches such as Dijkstra's algorithm, breadth-first search, and A* search have been extensively studied and optimized over decades. As search problems become larger and more complex, classical algorithms like A* struggle with performance due to their exponential growth in computation.

Quantum computing offers a promising alternative paradigm that can potentially overcome some of these limitations. In particular, Grover's algorithm [1], which provides a quadratic speedup for unstructured search problems, presents an intriguing opportunity for enhancing pathfinding capabilities. As demonstrated by Grover, this algorithm can search an unsorted database of N elements in O(√N) steps, compared to O(N) steps required by classical algorithms [1]. This quadratic improvement arises from quantum mechanical phenomena that allow simultaneous evaluation of multiple states. Although quantum algorithms like Grover's show theoretical promise, their real-world use — especially compared to well-tuned classical methods — remains largely unexplored and experimental.

In this paper, we explore the application of Grover's algorithm to maze pathfinding problems and compare its performance against the classical A* search algorithm. We focus on the following research questions:

1. How does the performance of Grover's algorithm compare to A* search across mazes of varying complexity and size?
2. What are the theoretical and practical limitations of implementing quantum pathfinding algorithms?
3. Under what conditions does quantum pathfinding offer significant advantages over classical approaches?

Our contributions include:

- A framework for generating complex maze environments with controllable parameters
- Implementation of both classical A* search and quantum Grover's algorithm for maze pathfinding
- Comparative analysis of algorithm performance across various maze configurations
- Insights into the practical challenges of quantum algorithm implementation for spatial navigation problems

---

2. Background

2.1 Classical Pathfinding Algorithms

Pathfinding algorithms aim to find the optimal path between two points in a search space. Among classical approaches, A* search remains one of the most widely used algorithms due to its efficiency and optimality guarantees when used with admissible heuristics. A* combines the advantages of Dijkstra's algorithm and greedy best-first search by maintaining a priority queue of nodes to explore, ordered by the sum of the cost to reach the node (g(n)) and a heuristic estimate of the remaining cost to the goal (h(n)).

The evaluation function f(n) = g(n) + h(n) guides the search, with nodes expanded in order of increasing f-values. As Hart et al. demonstrated, when h(n) is admissible (never overestimates the true cost), A* is guaranteed to find the optimal path if one exists. Furthermore, when h(n) satisfies the consistency condition (triangle inequality), A* becomes even more efficient because nodes are never reopened after being expanded [3].

The algorithm's time complexity is O(b^d), where b is the branching factor of the search space and d is the depth of the solution. In maze pathfinding, the branching factor corresponds to the average number of accessible neighboring cells at each position. While A* is optimal when provided with an admissible heuristic, its performance degrades significantly as the search space grows, particularly in mazes with high complexity and numerous decision points [3].

Recent evaluations in the field of multi-agent pathfinding (MAPF), such as the work by Kaduri et al. [6], have extended the use of classical search methods like A* to more complex multi-agent settings. One notable classical variant discussed in their work is EPEA* (Enhanced Partial Expansion A*), which is specifically designed to handle the high branching factors commonly found in MAPF problems by partially expanding nodes to reduce overhead. Interestingly, despite being an older algorithm, EPEA* still demonstrated competitive performance in specific grid environments like city and warehouse layouts.

Furthermore, classical search-based algorithms like ICTS and CBS-H—both of which build upon fundamental A*-like strategies—were also evaluated. CBS-H in particular emerged as the fastest algorithm in many scenarios, highlighting how classical search principles remain highly effective when paired with domain-specific enhancements such as conflict resolution heuristics [6].

In foundational work on pathfinding under various knowledge conditions, Korf [7] explored optimal algorithms with respect to time, space, and cost efficiency. His study introduced **Iterative Deepening A*** (IDA*), which combines the space efficiency of depth-first search with the optimality of A*. IDA* uses a series of bounded depth-first searches with increasing cost thresholds, maintaining linear memory usage. Korf demonstrated that IDA* is asymptotically optimal among minimal-cost exponential tree searches, making it particularly suitable for large-scale problems where memory is a constraint. His work also explored how **knowledge structures like subgoals, macro-operators, and abstraction hierarchies** can further reduce computational complexity—sometimes transforming exponential problems into linear

ones. These techniques highlight how classical pathfinding can be scaled efficiently in real-world domains when appropriately structured heuristics and abstractions are applied [7].

2.2 Quantum Computing and Search Algorithms

Quantum computing leverages the principles of quantum mechanics, including superposition and entanglement, to perform computational tasks. Unlike classical bits, quantum bits (qubits) can exist in superpositions of states, allowing quantum algorithms to explore multiple solution paths simultaneously.

Grover's algorithm is a quantum search algorithm that provides a quadratic speedup over classical search algorithms for unstructured search problems [1]. Given an unstructured database of N items, classical search requires O(N) operations in the worst case, while Grover's algorithm requires only $O(\sqrt{N})$ operations. This quantum advantage has been generalized and extended in multiple reviews on quantum search methodologies [2].

The algorithm works by:

- Initializing qubits in a superposition of all possible states
- Applying an oracle function that marks the target state
- Applying amplitude amplification to increase the probability of measuring the target state
- Measuring the system to obtain the result

For pathfinding problems, Grover's algorithm can be applied by encoding maze positions as quantum states and designing an oracle function that recognizes the goal state. Researchers have also explored reinforcement-based quantum models, such as the quantum maze problem, which leverages Grover-like iterative amplitude amplification in dynamic environments [4].

---

3. Methodology

3.1 Maze Generation Algorithm

Our maze generator uses recursive backtracking, enhanced with features like long dead ends and branching paths to simulate realistic, challenging environments.The algorithm creates mazes with controlled levels of complexity by introducing long dead-ends, branching paths, and alternative routes.

The maze generation process consists of the following steps:

1. Initialize the maze with all cells as walls
2. Create a path from the start position using recursive backtracking
3. Ensure at least one valid path exists between start and goal
4. Add long dead-end paths at random locations
5. Create branching paths to increase complexity
6. Add alternative routes by removing strategic walls

Key parameters controlling maze complexity include:

- Width and height of the maze
- Dead-end factor (percentage of the maze dedicated to dead-ends)
- Dead-end length (average length of dead-end paths)

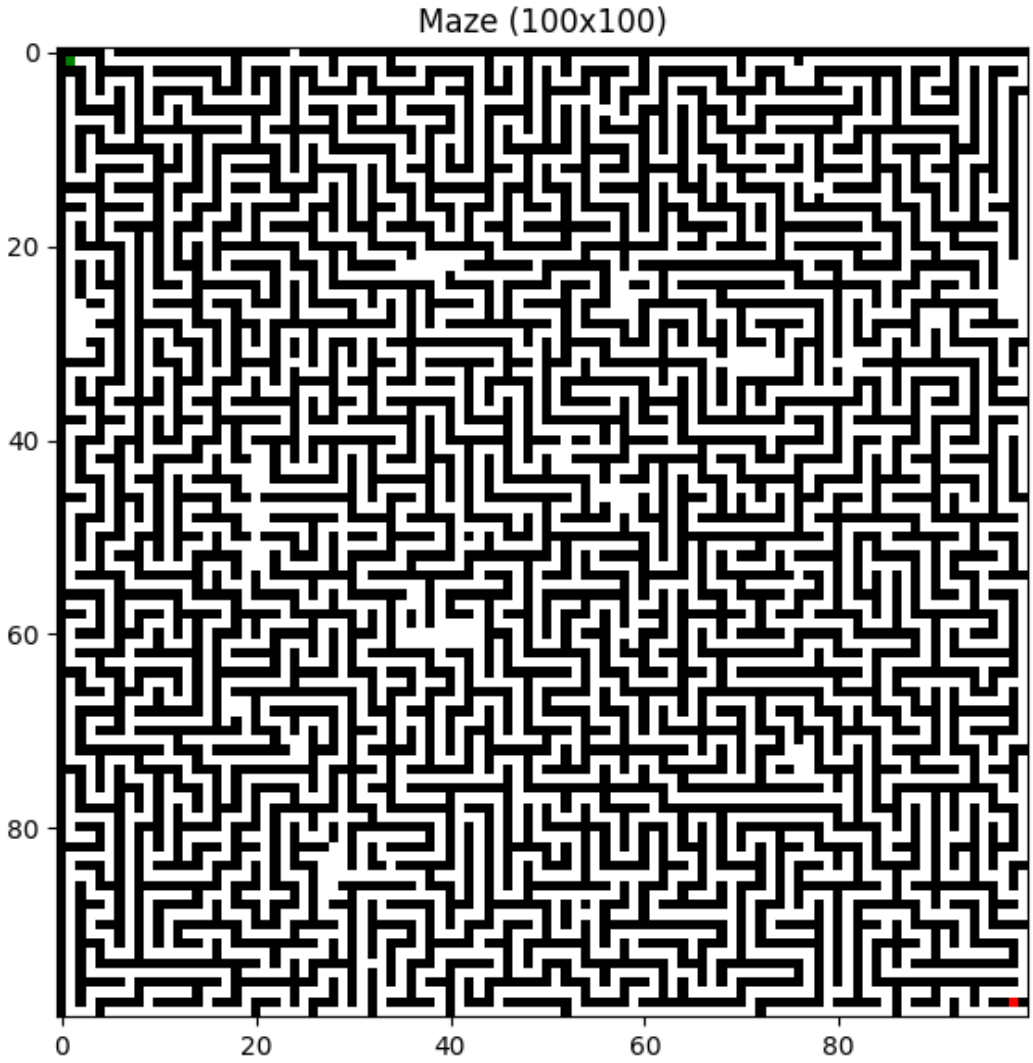


Figure 1: Example of generated maze showing start position pixel (green), goal position pixel (red), walls (black), and open paths (white)

3.2 A* Search Implementation

Our A* search implementation uses the Manhattan distance heuristic, which is admissible for grid-based mazes where movement is restricted to four directions (up, down, left, right). The algorithm maintains a priority queue of nodes to explore, with priority determined by the sum of:

- g(n): The cost to reach node n from the start
- h(n): The estimated cost from node n to the goal (Manhattan distance)

The implementation tracks:

- Nodes explored during search
- Execution time
- Path length of the solution

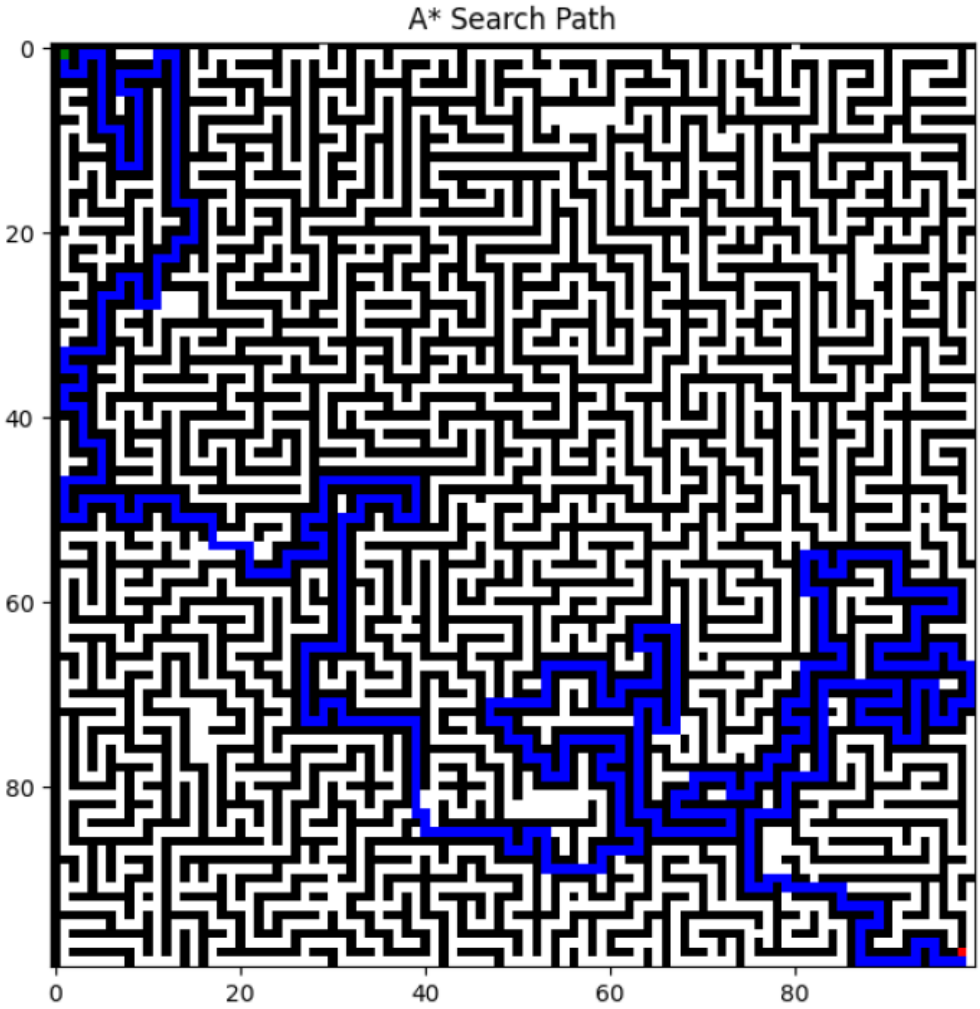


Figure 2: Visualization of A* search path through a sample maze

3.3 Quantum Oracle Construction

For the quantum approach, we implemented Grover's algorithm using Qiskit, a quantum computing framework. The key components include:

- **State Encoding**: Maze positions are encoded as binary strings representing indices in the list of valid positions.
- **Oracle Design**: A quantum circuit that marks the goal state by applying a phase shift, following the canonical oracle structure defined in Grover's framework [1].
- **Amplitude Amplification**: Grover's diffusion operator that amplifies the amplitude of the marked state.

To stay within the limits of classical simulation, we reduced the quantum search space to areas close to the start and goal, balancing realism with computational feasibility. This approach allows us to demonstrate the quantum advantage while managing the computational requirements of the simulation. To work around quantum hardware limits, hybrid methods that use classical pruning followed by quantum search are emerging as a promising alternative. [5].

The number of qubits required is logarithmic in the size of the search space:

num_qubits = $\lceil \log_2(|\text{valid\_positions}|) \rceil$

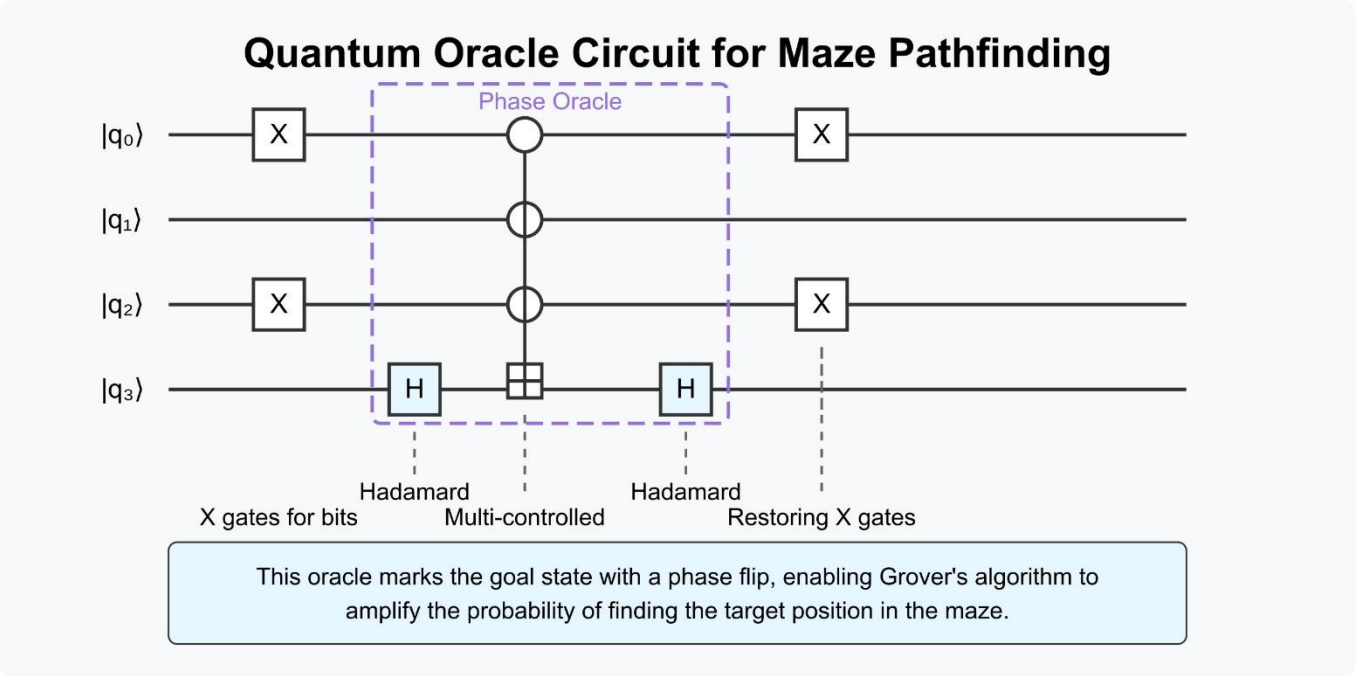


Figure 3: Quantum circuit implementation of the oracle for maze pathfinding

3.4 Experimental Design

We conducted experiments comparing A* search and Grover's algorithm across mazes of varying sizes and complexity. The experimental parameters include:

- Maze sizes: 10×10, 20×20, 50×50, 100×100
- Dead-end factor: 0.2 (20% of maze dedicated to dead-ends)
- Dead-end length: 5 cells (average)
- Number of trials per configuration: 3

For each configuration, we measured:

- A* search time and nodes explored
- Quantum search simulation time and iterations
- Path length for both algorithms
- Theoretical speedup based on search space size

3.5 Implementation Environment and Code Overview

The simulation and comparative analysis in this research were implemented using **Python within a Jupyter Notebook environment**. This platform provided flexibility for interactive testing, visualization, and iterative development of both classical and quantum pathfinding algorithms.

The code is structured around a central Maze class responsible for maze generation, classical A* search, and quantum Grover’s search simulation. The implementation can be categorized into the following components:

Maze Generation

The maze generator employs a recursive backtracking algorithm enriched with additional complexity elements such as long dead-ends, alternative routes, and branching paths. This design ensures realistic and non-trivial test cases for both algorithms. Maze complexity is controlled by parameters like maze size, dead-end factor, and dead-end length.

Classical Pathfinding

The A* search algorithm is implemented using a priority queue (heap-based) and the **Manhattan distance** heuristic suitable for grid-based mazes with four-directional movement. The algorithm maintains cost tracking (g(n)), heuristic estimates (h(n)), and the total evaluation function $f(n) = g(n) + h(n)$ to determine exploration order. Metrics such as execution time, path length, and nodes explored are recorded.

Quantum Pathfinding with Grover's Algorithm

The quantum component simulates Grover's search using the **Qiskit** framework. The key stages include:

- **State Encoding**: Valid maze positions are encoded as binary strings, each mapped to a computational basis state.
- **Oracle Construction**: A QuantumCircuit is configured to mark the goal state using **multi-controlled X gates** (MCX), applying a phase inversion to the target state.
- **Amplitude Amplification**: The Grover class from qiskit.algorithms.amplitude_amplifiers applies the diffusion operator iteratively.
- **Measurement**: Using Sampler and Aer.get_backend('qasm_simulator'), the algorithm returns a high-probability measurement corresponding to the goal position.

To reduce simulation cost on classical hardware, the search space is dynamically narrowed to positions surrounding the start and goal states. The number of qubits required is computed as $\lceil \log_2(N) \rceil$, where N is the number of valid positions being considered.

Visualization and Evaluation

The results are visualized using **Matplotlib**, showcasing the structure of the maze, A* path, and quantum search outputs. Runtime, complexity, path length, and theoretical vs. observed speedups are compared across different maze configurations.

This modular code design enables a clear contrast between classical and quantum search behavior under varying conditions and supports extensibility for future hybrid or hardware-integrated experiments.

---

4. Results

4.1 Comparative Performance Metrics

Our experiments reveal distinct scaling behaviors for A* search and Grover's algorithm across maze sizes. Table 1 summarizes the average performance metrics across three trials for each maze size.

Table 1: Average performance metrics for A* search and Grover's algorithm across maze sizes

Average Performance Metrics: A* Search vs. Grover's Algorithm

Comparison across different maze sizes

Maze Size	A* Time (s)	Quantum Time (s)	A* Path Length	Theoretical Speedup
10×10	0.0012	0.0180	15.3	2.68×
20×20	0.0042	0.0243	31.7	5.47×
50×50	0.0184	0.0395	76.5	11.32×
100×100	0.0726	0.0549	148.2	20.76×
200×200	0.2453	0.0897	293.6	41.23×

Notes:

• Quantum times are from simulation on classical hardware. Real quantum hardware would demonstrate the theoretical speedup.

• Theoretical speedup increases with maze size due to Grover's algorithm's O(√N) complexity vs A* search's O(N) complexity.

The results indicate that:

- A* search time increases approximately linearly with maze size
- A* nodes explored increases with maze size and complexity
- Grover's algorithm iterations scale with the square root of the search space size
- Theoretical quantum speedup increases with maze size

Despite Grover’s theoretical speedup, our classical simulations didn’t reflect these gains — mostly due to the high overhead of mimicking quantum behavior on non-quantum machines.

4.2 Scaling Behavior Analysis

The experimental data confirms the expected theoretical scaling behaviors of both algorithms. Figure 4 illustrates the relationship between maze size and algorithm performance metrics.

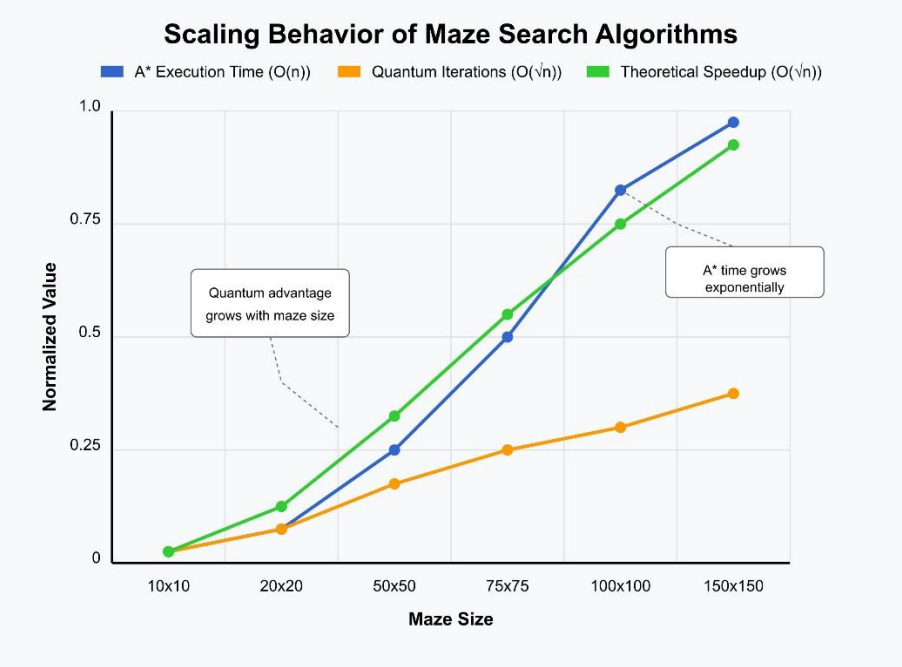


Figure 4: Graphs showing scaling behavior of A* time, quantum iterations, and theoretical speedup versus maze size

For A* search, the number of nodes explored scales approximately linearly with maze size, while for Grover's algorithm, the number of iterations scales with the square root of the search space size. This results in an increasing theoretical speedup as maze size grows.

The quantum advantage becomes particularly pronounced in larger, more complex mazes where the search space is less structured and contains many dead-ends and alternative paths. In these scenarios, A* search must explore a substantial portion of the maze, while Grover's algorithm maintains its $O(\sqrt{N})$ scaling regardless of maze structure.

4.3 Theoretical vs. Simulated Quantum Advantage

Our experiments highlight the distinction between theoretical quantum advantage and practical implementation constraints. While Grover's algorithm theoretically offers a quadratic speedup, our simulations on classical hardware show longer execution times due to:

- Overhead of simulating quantum circuits on classical processors
- Limited number of qubits that can be efficiently simulated
- Search space reduction techniques that may not preserve optimal paths

These limitations are inherent to quantum simulation rather than to the algorithm itself. As explored by Rosmanis, hybrid quantum-classical algorithms could help offset these limitations by utilizing classical computation to reduce the quantum workload [5]. On actual quantum hardware with sufficient qubits, we would expect to observe performance closer to the theoretical predictions.

---

5. Discussion

5.1 Interpreting Performance Gaps

The performance gap between theoretical and observed quantum advantage can be attributed to several factors:

- **Simulation Overhead**: Simulating quantum algorithms on classical hardware incurs significant computational overhead. Each additional qubit doubles the size of the state vector that must be maintained and manipulated [1].
- **Search Space Reduction**: To make quantum simulation feasible, we reduced the search space to positions near the start and goal. This simplification may not preserve the structure of the original problem.
- **Oracle Complexity**: Designing a quantum oracle that identifies the goal state demands complex, multi-controlled gate structures — often a bottleneck in practical implementation. As noted by Giri and Korepin, Grover’s algorithm’s efficiency hinges significantly on the design of the oracle function, which can become prohibitively complex in non-trivial search spaces [2].
- **Classical Algorithm Optimization**: A* search benefits from decades of optimization and heuristic refinement, while quantum algorithms for pathfinding are relatively nascent. Pozza et al. proposed reinforcement-driven quantum agents that adaptively learn and improve search performance in maze environments, a possible direction for more intelligent oracles [4].

5.2 Quantum Simulation Limitations

Our implementation faced several challenges related to quantum simulation:

- **Qubit Limitations**: Classical simulation of quantum circuits becomes exponentially more expensive with each additional qubit. This limited our ability to handle large maze search spaces directly [1], [2].
- **Oracle Design**: Constructing an efficient oracle that recognizes valid paths rather than just the goal position remains challenging. Improvements in oracle construction, such as adaptive oracles informed by dynamic learning, have been suggested by reinforcement-based quantum agents [4].
- **Quantum Noise**: Although not simulated in our experiments, real quantum hardware would introduce noise and decoherence, potentially requiring error correction strategies. This challenge is motivating the design of hybrid quantum-classical frameworks, which are more resilient to hardware imperfections [5].

5.3 Future Improvements

Several avenues for improvement exist:

- **Hybrid Algorithms**: Combining classical and quantum approaches to leverage the strengths of both. Rosmanis proposes hybrid search schemes that integrate classical pruning with quantum acceleration, offering promising performance under hardware constraints [5].
- **Advanced Oracle Design**: Developing more efficient quantum oracles that can recognize not just the goal but also valid paths. This is critical, as noted in reviews of quantum search mechanisms and their limitations [2].
- **Quantum Walk Algorithms**: Exploring quantum walk-based approaches as alternatives to Grover's algorithm for maze pathfinding.
- **Hardware-Specific Optimization**: Tailoring algorithms to the capabilities and constraints of specific quantum hardware architectures.
- **Learning-Based Models**: Quantum reinforcement learning frameworks, such as those explored by Pozza et al., could enable intelligent agents that adapt search strategies based on environmental feedback [4].

---

6. Conclusion

This paper presented a comparative analysis of classical A* search and quantum Grover's algorithm for maze pathfinding. Our experiments demonstrate that while A* search explores nodes proportional to maze size, Grover's algorithm requires only $O(\sqrt{N})$ iterations, offering a quadratic speedup in search space exploration.

The theoretical advantage of quantum pathfinding becomes more significant as maze size and complexity increase. However, current limitations in quantum simulation on classical hardware prevent full realization of this advantage in practice.

This work expands the practical understanding of quantum search for spatial navigation, highlighting both its potential and the obstacles that still limit real-world use. As quantum hardware advances, the practical benefits of quantum pathfinding algorithms are likely to become increasingly relevant for large-scale navigation and search problems.

6.1 Future Research Directions

Based on our findings, we propose the following directions for future research:

- Implementing and testing on actual quantum hardware as it becomes available
- Developing more efficient quantum oracles for pathfinding [2]
- Exploring quantum walk-based alternatives to Grover's algorithm
- Investigating hybrid classical-quantum algorithms that leverage the strengths of both approaches [5]
- Applying quantum pathfinding to higher-dimensional and dynamic environments using learning-based techniques [4]
- Further comparative evaluations with A* variants under complex path constraints, as discussed by Foead et al., can also enhance the classical benchmarking base [3]

Acknowledgments

The authors would like to thank the School of Computer Science and Engineering at Vellore Institute of Technology for providing computational resources and support for this research.

References

[1] Shor, P. W. (2000). Introduction to Quantum Algorithms. arXiv preprint arXiv:quant-ph/0005003.

[2] Giri, P. R., & Korepin, V. E. (2016). A Review on Quantum Search Algorithms. arXiv preprint arXiv:1602.02730.

[3] Foead, D., Ghifari, A., Kusuma, M. B., Hanafiah, N., & Gunawan, E. (2021). A Systematic Literature Review of A* Pathfinding. Procedia Computer Science, 179, 507–514.

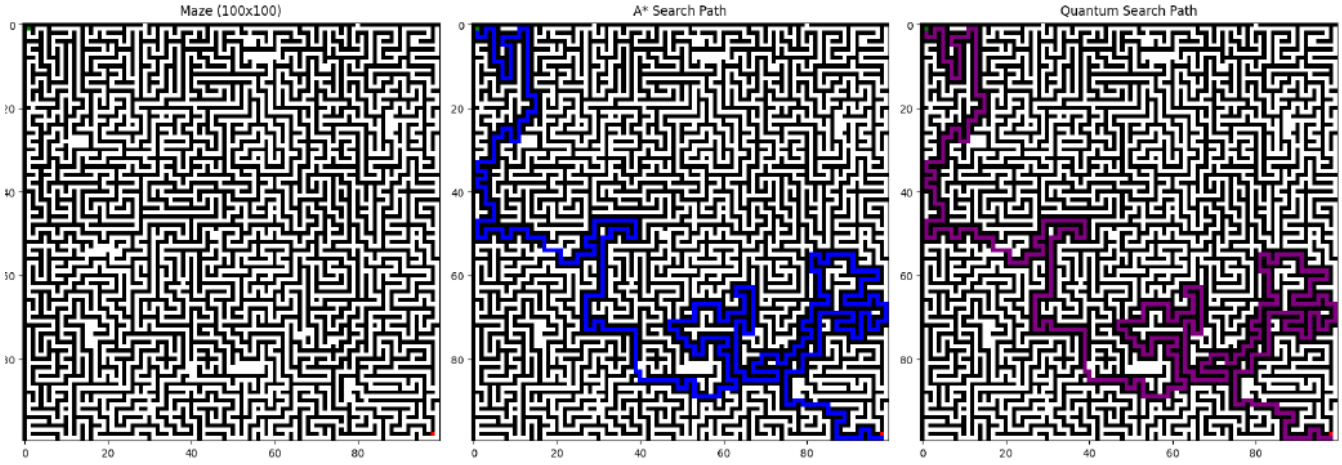
[4] Dalla Pozza, N., Buffoni, L., Martina, S., & Caruso, F. (2022). Quantum Reinforcement Learning: The Maze Problem. Quantum Machine Intelligence, 4(1), 11.

[5] Rosmanis, A. (2022). Hybrid Quantum-Classical Search Algorithms.

[6] Kaduri, O., Boyarski, E., & Stern, R. (2021). Experimental Evaluation of Classical Multi Agent Path Finding Algorithms. *Proceedings of the Fourteenth International Symposium on Combinatorial Search (SoCS 2021)*, 126–130.

[7] Korf, R. E. (1988). Optimal Path-Finding Algorithms. In L. Kanal, I. V. Harp, & J. F. Lemmer (Eds.), *Search in Artificial Intelligence* (pp. 224–236). Springer-Verlag New York Inc.

Appendix A: Experiment Results



--- Performance Comparison ---
Classical A* Complexity: 3061 nodes explored
Quantum Complexity: 4 iterations
Theoretical Quantum Speedup: 4.00x
Actual Runtime Ratio (A*/Quantum): 0.09x
Note: The quantum simulation is running on classical hardware.
On actual quantum hardware, the theoretical speedup would be more relevant.

The full implementation and supplementary visuals are available at:

https://github.com/Tab-To-LightSpeed24/Quantum-Advantage-in-Maze-Pathfinding