
UNIVERSITY OF CAPE TOWN


DEPARTMENT OF COMPUTER SCIENCE

# CS Honours Project Final Paper 2024

Title: *Benchmarking Classical Methods against the Quantum Approximate Optimisation Algorithm to Solve the Vehicle Routing Problem: Branch and Bound*

Author: Tayla Rogers

Project Abbreviation: VRP

Supervisor(s): Krupa Prag

| **Category** | *Min* | *Max* | Chosen |
|---|---|---|---|
| Requirement Analysis and Design | *0* | *20* | 0 |
| Theoretical Analysis | *0* | *25* | 0 |
| Experiment Design and Execution | *0* | *20* | 20 |
| System Development and Implementation | *0* | *20* | 10 |
| Results, Findings and Conclusions | *10* | *20* | 20 |
| Aim Formulation and Background Work | *10* | *15* | 10 |
| Quality of Paper Writing and Presentation | *10* | | 10 |
| Quality of Deliverables | *10* | | 10 |
| Overall General Project Evaluation (*this section allowed only with motivation letter from supervisor*) | *0* | *10* | 0 |
| **Total marks** | | **80** | |

# Benchmarking Classical Methods against the Quantum Approximate Optimisation Algorithm to Solve the Vehicle Routing Problem: Branch and Bound

Tayla Rogers
University of Cape Town
Cape Town, South Africa

## ABSTRACT

A comparative analysis of classical and quantum techniques for solving the Vehicle Routing Problem (VRP) is presented in this paper. The VRP, a well-known *NP*-hard problem, is described as having significant real-world implications. The Quantum Approximate Optimisation Algorithm (QAOA) is benchmarked against the classical Branch and Bound (B&B) algorithm. The comparison is focused on solution quality, computational time, scalability, feasibility, and success rate. Insights into the practical feasibility of QAOA are revealed using a custom-created dataset, tailored for small-scale quantum computing capabilities. QAOA's current limitations and potential for future applications in combinatorial optimisation are included in this analysis.

## 1 INTRODUCTION

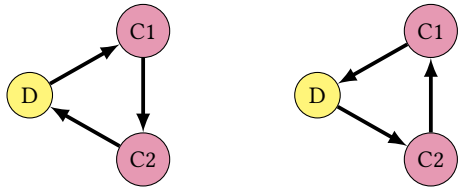
The Vehicle Routing Problem (VRP) is recognised as a significant challenge in combinatorial optimisation. The most cost-effective routes for a fleet of vehicles delivering goods or services to a set of customers are established as the primary objective. To illustrate this concept, various routes for a single vehicle delivering to two customers from a depot are shown in Figure 1. The problem has been mathematically formulated, as depicted in Figure 10 [3], with an emphasis placed on minimising the objective function detailed in Equation 1 [3].

$$VRP(n,k) = min_{\{x_{ij}\}_{i->j} \in \{0,1\}} \sum_{i->j} w_{ij} x_{ij}. \quad (1)$$

Rich VRPs (RVRPs) incorporate real-world constraints such as congestion and delays into the traditional VRP [4]. Through these advancements, a broad range of real-world challenges beyond delivery services, including in-home health care and waste collection, have been addressed by the problem [16]. As a result, the efficient solving of VRPs is considered vital for businesses and individuals dependent on prompt transport services, especially for small and medium-sized enterprises constrained by budget and resources [4].

The VRP has been recognised as highly significant, but it has also been notably difficult to solve because it has been classified as an $\mathcal{NP}$-hard problem [15]. The difficulty has been caused by the exponential increase in problem size with the input size, rendering the VRP impractical to solve for large-scale instances.

Recent advances in quantum computing have sparked interest in applying quantum principles, such as entanglement and superposition, to solve combinatorial optimization problems such as the VRP. Quantum algorithms on IBM's Noisy Intermediate-Scale Quantum (NISQ) devices have been enabled by these principles to explore multiple potential routes concurrently across a large solution space. Recent work [10] has shown that exact brute-force methods can be outperformed by quantum algorithms on NISQ devices for certain problems. The potential for quantum computing to become more applicable to solving combinatorial optimisation problems, where brute-force methods are currently impractical, is suggested. However, challenges have been faced by current NISQ devices due to noise, errors, and limited qubit resources. As a result, hybrid quantum-classical approaches have been considered the most practical application of quantum computing at this stage.




**(a) Route A** **(b) Route B**

**Figure 1: Alternative routing options A and B for a single vehicle travelling from the depot to visit two customers.**

IBM's most recent release of Qiskit `1.0` has been recognised as offering a more stable and performant way to explore current quantum capabilities [7]. Significant memory and performance improvements have been made in Qiskit `1.0`. The groundwork for future workloads involving 1000+ qubits has been laid by these improvements, and more effective solutions to complex optimisation challenges are being paved as a result [7].

The VRP's representation as an energy function is illustrated in Figure 11 [3], enabling the creation of a Quadratic Unconstrained Binary Optimisation (QUBO) formulation, as depicted in Figure 12 [3]. The QUBO formulation is then mapped to an Ising Hamiltonian, which is shown in Figure 13 [3], enabling the VRP to be solved using quantum methods. The energy associated with specific qubit configurations within a quantum circuit is represented by the Ising formulation, detailed in Equation 2 [3].

$$H_{\text{Ising}} = -\sum_i \sum_{j<i} I_{ij} s_i s_j + \sum_i h_i s_i + d. \quad (2)$$

Insights into the current capabilities of quantum algorithms have been provided through their benchmarking, which has been limited by the number of available qubits [1, 14]. The effectiveness of these quantum algorithms has been assessed by comparing their performance against standard classical benchmarks.

Prior work on quantum VRP algorithms [2, 3] has been extended in this study. A comprehensive comparison between the hybrid

quantum-classical Quantum Approximate Optimisation Algorithm (QAOA) and the Branch and Bound (B&B) algorithm across various metrics is offered in this paper.

### 1.1 Problem Statement

The performance of QAOA and B&B when applied to the VRP across various instance sizes is evaluated and compared in this study. The solution quality, computational efficiency, scalability, success rate, and feasibility of QAOA are aimed to be determined in comparison to B&B as the complexity of VRP instances increases. The strengths and limitations of quantum versus classical approaches in solving combinatorial optimisation problems will be identified through this research.

### 1.2 Research Questions

Specifically, this research addresses the following questions:

(1) How does QAOA perform compared to B&B regarding solution quality, computational efficiency, scalability, feasibility and success rate across various VRP problem sizes?
(2) Can enhancements in QAOA's optimiser (e.g., COBYLA vs. SPSA) lead to performance improvements?
(3) What are the limitations of QAOA in solving VRP instances compared to B&B?

## 2 RELATED WORK

Growing interest in quantum computing has resulted in several significant studies evaluating its potential in solving combinatorial optimisation problems. An in-depth analysis of the feasibility of Variational Quantum Eigensolver (VQE) and QAOA in addressing the Travelling Salesman Problem (TSP) and the Quadratic Assignment Problem (QAP) was provided by Khumalo et al. [9]. B&B and Simulated Annealing (SA) were used to benchmark the quantum algorithms based on solution quality, feasibility, success rate, and computational time. Thirty trials of each experiment were conducted, ensuring the validity of the results. The classical methods were found to significantly outperform the quantum algorithms on all metrics. This was largely due to the limited number of qubits, which restricted the size and complexity of problems that could be effectively handled by quantum methods. The need for further research into the scalability of quantum algorithms for more complex combinatorial optimisation problems such as the VRP was emphasised by this study.

A study comparing the performance of VQE and QAOA against IBM's CPLEX classical optimisation solver for solving the VRP was conducted by Alsaiyari and Felemban [2]. CPLEX, known for its exact algorithms, was used as a benchmark for high-quality solutions. It was indicated by their results that QAOA could achieve solution quality comparable to CPLEX. However, only solution quality was considered in their analysis, excluding other important metrics such as computational time and success rate. The experiments were conducted using a single data input for three different problem sizes: (3 customers, 2 vehicles), (4 customers, 2 vehicles), and (5 customers, 2 vehicles). While the exact distance matrices were not provided, they could be inferred from graphs showing customer locations on a grid map. Additionally, only problem instances involving two vehicles were focused on, excluding other potential problem sizes that might be effectively handled by the algorithms.

A benchmark study evaluating the performance of QAOA in solving the VRP was conducted by Azad et al. [3]. The performance of QAOA using different classical optimisers and varying numbers of repetitions ($p$) was compared against IBM's CPLEX optimisation solver. It was indicated by the results that the gradient-free optimisers, COBYLA and SPSA, performed the best, with improvements in solution quality being shown with increased $p$-values. It was also found that the probability of obtaining a feasible solution increased with higher $p$-values. An increase in p was noted to be related to an increase in time. However, the average optimal cost produced by QAOA variants was unable to match that of the CPLEX solver. It was observed that QAOA sometimes favours a lower-energy solution that does not satisfy all problem constraints over a feasible solution, which can lead to infeasible results. Similar to previous research, an in-depth analysis of QAOA's solution quality in comparison to the CPLEX solver was provided, but a broader evaluation across multiple metrics was lacking. Although the analysis was limited to three specific problem sizes — (3 customers, 2 vehicles), (4 customers, 2 vehicles), and (5 customers, 2 vehicles) — multiple distance matrices for each were included, providing a thorough evaluation of each problem instance. The reproducibility of the results is limited by the use of randomly generated distance matrices and the absence of a publicly available dataset.

## 3 METHODOLOGY

The methodologies used to solve the VRP with two approaches, the classical B&B algorithm and QAOA, have been described in this section. B&B, as an exact algorithm, has been used to systematically explore solutions through branching and pruning to ensure optimal outcomes. QAOA, tailored for quantum hardware, uses quantum states and Hamiltonians to optimise solutions. The methodologies have been explained with figures and pseudo-code, showcasing the distinct strategies each algorithm has employed to address the VRP.

### 3.1 Branch and Bound

B&B has been recognised as a highly effective exact algorithm for solving combinatorial optimisation problems [15]. The discovery of the optimal solution is guaranteed by B&B by exploring all potential solutions in a structured manner. A tree-like structure, represented in Figure 2, is used to explore these potential solutions.

In B&B, a branching strategy is used to iteratively divide the problem into sub-problems. In Figure 2, the branching strategy has been illustrated by constructing routes from parent nodes to their children. Each branch is represented as a sub-problem of deciding which node should be visited next, based on the customers yet to be visited in the current path.

Bounding techniques are used to prune sub-optimal branches. Performance is enhanced by the elimination of non-promising routes that would not lead to an optimal solution. In Figure 2, bounding strategies have been applied to prevent the further exploration of certain sub-routes, such as those beginning with a visit to customer 2. In symmetrical distance matrices, non-memory-based dominance rules [8] are used as bounding techniques to eliminate duplicate sub-problems, such as when the distance from A to B

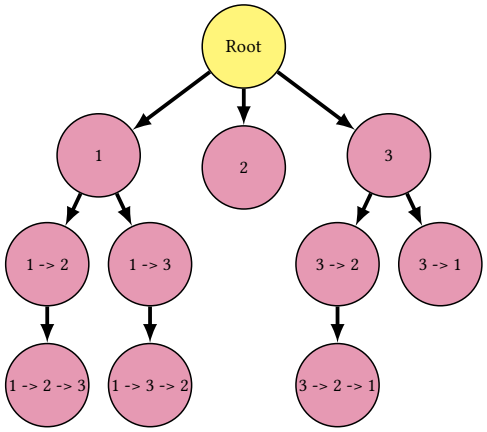



**Figure 2: Tree-like structure of the B&B algorithm constructing sub-problems for a three customer one vehicle VRP.**

is identical to the distance from `B` to `A`, thereby preventing the redundant exploration of equivalent sub-routes.

Selection rules in B&B are used to determine the next sub-problem to explore. As illustrated in Figure 2, these rules have guided the decision on whether to expand a sub-route, such as '`1 -> 2`' or '`3 -> 1`'. The decision is based on chosen criteria, such as the lowest cost or the most promising branch.

The algorithm is run until all sub-problems have been explored or a time limit has been reached, indicating infeasibility within a reasonable time frame. Algorithm 1, adapted from Morrison et al. [13], has outlined solving the VRP with B&B. An explanation of the variables used has been provided in Table 9.

## 3.2 Quantum Approximate Optimisation Algorithm

QAOA, specifically developed for combinatorial optimisation problems, has been extensively researched as a method for approximating solutions [2]. The Mixer Hamiltonian, $H_M$, is considered central to QAOA. The initial quantum state is prepared, and the exploration of the solution space is facilitated, ensuring that the algorithm does not become stuck in local minima. The Cost Hamiltonian, $H_C$, encodes the objective function and guides the algorithm in refining the solution by exploiting problem-specific information. To enforce the constraints of the classical representation of the problem, penalty terms are added to $H_C$. This ensures that all necessary conditions are adhered to in the final solution. The Hamiltonians are used together to balance exploration and exploitation, driving the algorithm towards optimal solutions.

The ansatz, modelled as a parameterised quantum circuit, is used to prepare the initial quantum state [3]. The quantum state is evolved by the algorithm through the application of the ansatz and the utilisation of $H_M$ and $H_C$ to explore and refine potential solutions. The number of layers of the quantum circuit, $p$, is considered a key hyper-parameter to balance the trade-off between the quality of the approximation and the complexity of the quantum circuit. A higher $p$ is typically associated with a more accurate approximation but requires more resources. Once the final quantum state has been determined, it is transformed into a classical representation for the evaluation of the objective function. The parameters of the ansatz are then iteratively optimised by a classical optimiser. This optimisation is used to fine-tune the balance between $H_M$ and $H_C$ to enhance the algorithm's performance.

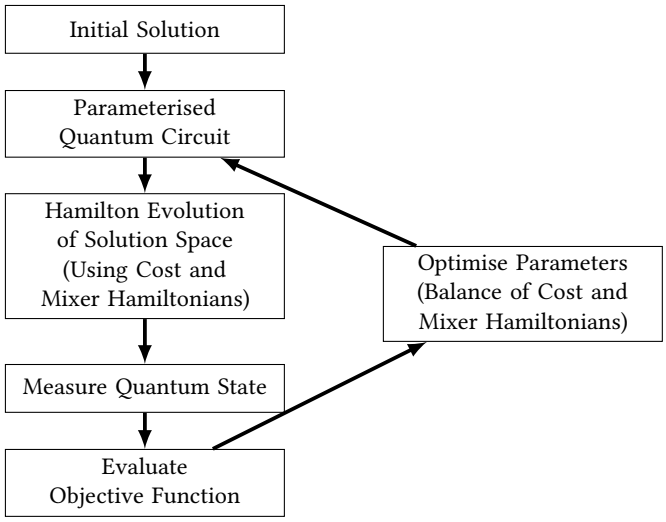



**Figure 3: Flowchart of the QAOA process, illustrating steps from initialisation to iterative parameter optimization.**

Figure 3 summarises the QAOA process, illustrating the iterative steps from initialisation to parameter optimisation. Pseudo-code for QAOA, as seen in Algorithm 2, adapted from Khumalo et al. [9], has been provided. It offers a practical approach for the utilisation of quantum computing in combinatorial optimisation problems, including the VRP. An explanation of the variables used has been provided in Table 10.

## 3.3 Design and Implementation

This section outlines the methodologies and experimental setups that have been used to evaluate B&B and QAOA for solving the VRP. Key aspects include experimental settings, data preprocessing, and algorithm implementations. Information on the metrics that have been used to compare the performance of both algorithms has also been provided.

*3.3.1 Experimental Settings.* To assess the performance of both algorithms, all experiments have been conducted on a classical computing device. As part of an Honours research project, direct access to quantum computing devices has not been available. Instead, a quantum simulator provided by Qiskit version 1.1.0 has been used to provide a classical approximation of the quantum algorithms. The behaviour of quantum algorithms has been effectively mimicked by the quantum simulator running on classical hardware. Valuable insights into their scalability and performance under real-world conditions have been obtained by this. For reference, IBM's current quantum limits support up to 127 qubits, offering insight into the state of available quantum computing at the time of writing.

Each algorithm has been run six times per data input to ensure consistency. A six-hour timeout has been set to manage computation time and resources due to the complexity of the problems.

The full specifications of the computing device used have been detailed in Table 1, outlining the hardware required to replicate these experiments.

**Table 1: Computer specifications**

| Specification | Details |
|---|---|
| Operating System | Linux Server |
| Processor | AMD Ryzen 7 3800X 8-Core Processor |
| Threads | 16 |
| Memory (RAM) | 32 GB |
| GPU | NVIDIA GeForce RTX 2060 |

*3.3.2 Dataset and Data Preprocessing.* A custom dataset has been created from a classical Capacitated VRP (CVRP) dataset [12] due to the scarcity of publicly accessible datasets suitable for quantum computing's small-scale capabilities. The original dataset [11] has been modified by removing CVRP-specific constraints, such as capacity and demand, to generate generalised VRP instances. For this research, it has also been specified that each vehicle must visit at least one customer, adding a constraint to ensure that every vehicle is used in the solution. Both symmetric and asymmetric data inputs are included in the dataset. This allows for the evaluation of algorithm versatility across different routing challenges to be conducted. The number of vehicles and destinations has been reduced to stay within the practical limits of the QAOA and B&B algorithms. Problem sizes ranging from 2 customers and 1 vehicle to 15 customers and 8 vehicles have resulted from this. This dataset has been made publicly available to support further research and facilitate comparison with this study's results. In the dataset and the analysis throughout this paper, a shorthand notation of customers and vehicles has been used to indicate problem size. Using this shorthand a problem size of 15 customers and 2 vehicles is represented as 15-2.

*3.3.3 Branch and Bound.* A branching strategy has been used in the B&B implementation to generate all possible customer assignments to vehicles and explore every route option, ensuring that no potential solution has been missed. The reliable answers provided by B&B are crucial for the evaluation of quantum solutions through this exhaustive approach. Each unvisited customer is systematically considered for each vehicle, with new potential solutions being branched into by the algorithm.

A cost-based pruning method, or upper-bound pruning, has been used to manage the mix of asymmetrical and symmetrical data inputs. This strategy has been used to prune suboptimal routes while minimising the risk of discarding promising ones, ensuring the algorithm reliably finds the correct solution. The cost of each partial solution is compared to the best-known solution, and any that exceed it are discarded, avoiding unnecessary exploration.

A Best-First Search (BFS) strategy has been used as the selection rule [6] since the lowest-cost nodes are prioritised. The algorithm's quicker convergence on an optimal solution is helped by this. This approach has been found to improve the likelihood of finding the best solution faster, making it suitable for complex optimisation problems such as the VRP.

To verify accuracy, solutions for various distance matrices, ranging from 2-1 (2 customers and 1 vehicle) to 3-2 (3 customers and 2 vehicles), have been manually checked. This verification has confirmed that the correct optimal routes have been consistently produced by the algorithm, providing a reliable foundation for comparison in further analyses.

*3.3.4 Quantum Approximate Optimisation Algorithm.* Previous research by Azad et al. [3] has demonstrated that increasing the $p$ hyper-parameter from 1 to 10 improves solution quality but also increases the time involved. To evaluate both the accuracy of the algorithm in solving the problem and its feasibility compared to more established methods, a $p$-value of 5 has been selected. By this choice, a balance between the algorithm's speed and the quality of the results has been ensured, allowing for a fair comparison across multiple metrics.

The performance of various classical optimisers has also been examined in this research, revealing that the best results have been achieved by COBYLA and SPSA, particularly with an increased $p$-value. Consequently, both optimisers have been selected for this study to comprehensively evaluate the algorithm's performance across multiple metrics, using those identified as best suited for the algorithm.

*3.3.5 Metrics.* The performance of QAOA and B&B have been compared using several key metrics.

- **Solution quality** is analysed with an assessment of whether QAOA can match the optimal solutions identified by B&B. Consistency and accuracy are evaluated using best-case performance and standard deviation across multiple runs.
- **CPU time** is used to evaluate computational efficiency by measuring the duration each algorithm requires to solve problems of varying sizes. Metrics such as average, best, worst-case times, and standard deviation have been used to assess variability in computational demands. The practical feasibility of QAOA and B&B is determined by this assessment.
- **Scalability** is assessed by examining each algorithm's capacity to handle larger problem sizes and increasing complexity, highlighting their potential to manage more extensive datasets and complex instances.
- **Feasibility** percentage indicates the proportion of instances where the algorithm has successfully met all problem constraints. In contrast to previous research [9], scalability and feasibility have been treated as separate metrics in this study to provide a clearer analysis of both algorithms' performances. A higher feasibility percentage, as defined here, has been suggested to indicate greater reliability in producing valid solutions, thereby enhancing the algorithm's practical applicability.
- **Success rate** is measured to determine how often QAOA has produced solutions within 95% and 99% of the optimal results from B&B. This provides insights into QAOA's reliability in approximating near-optimal solutions. This

measurement is based on the number of instances where a solution has been generated by each algorithm, excluding cases where no solution has been found.

*3.3.6 System Development and Implementation.* All algorithms in this study have been implemented in Python, with significant reliance placed on IBM's Algorithms and Optimisation packages for the implementation of QAOA. The implementations have incorporated the latest updates introduced with the release of Qiskit `1.0`. Transparency and reproducibility are ensured by making all code, experimental results, and data input files publicly available in a GitHub repository linked here.

## 4 EXPERIMENTAL RESULTS AND DISCUSSION

In this section, the experimental results comparing QAOA and B&B in solving the VRP are presented. Performance is evaluated across several metrics, including solution quality, computational efficiency, scalability, feasibility, and success rate. A brief comparison with SA is also included to provide further context on their effectiveness. The strengths and limitations of both quantum and classical methods are highlighted by the findings, indicating where further development of QAOA is required to compete with B&B.

Throughout this section, a shorthand is used to represent problem sizes, with 15 customers and 2 vehicles represented as `15-2`.

### 4.1 Solution Quality

Robustness and reliability in finding globally optimal solutions have been consistently demonstrated by the B&B algorithm, as shown in Table 2. Optimal results across various data inputs are guaranteed due to its deterministic nature, providing a definitive benchmark, especially with the newly created dataset. A standard deviation of `0` for every instance confirms the effectiveness of B&B in identifying optimal solutions.

Both QAOA variants have been shown to perform well on smaller instances, with the optimal costs achieved by B&B closely matched for problem sizes up to `3-2`, as indicated in Table 2. As complexity has increased, larger instances have posed challenges for both optimisers, with higher-than-optimal costs produced in two cases and an invalid solution generated in one case — `4-2` for COBYLA and `4-1` for SPSA. Different sensitivities to problem complexity between the optimisers are suggested by this outcome. Solutions closer to those of B&B have generally been yielded by SPSA, indicating advantages over COBYLA in specific scenarios. However, consistent matching of B&B's performance for complex problems has not been achieved by either optimiser. A standard deviation of `0` has been maintained by both optimisers, showing consistent performance across runs. Overall, while effectiveness has been demonstrated by QAOA for simpler problems, a decrease in solution quality and feasibility has been observed with increased complexity. This is consistent with prior research [2, 3, 3, 9].

In Figure 4, the optimal costs achieved by the algorithms for problem instances that have been solved by QAOA are compared. Consistent outperformance of both QAOA variants by B&B in solution quality has been observed as problem size has increased. QAOA's limitations in finding globally optimal solutions for more complex VRP instances are highlighted by this. A noticeable decline in QAOA solution quality has been observed as the problem size has exceeded `3-2`. Infeasible solutions have been observed from both COBYLA and SPSA where the optimal costs have dipped below those produced by B&B. Despite relatively small differences in optimal costs, which could be acceptable if advantages in other metrics were provided by QAOA, the early performance divergence has suggested limited applicability for QAOA in real-world scenarios.

**Table 2: Final Route costs of each algorithm**[1]

| Size | B&B | QAOA COBLYA | QAOA SPSA |
|---|---|---|---|
| 2-1 | 637 | 637 | 637 |
| 3-1 | 1096 | 1096 | 1096 |
| 3-2 | 1442 | 1442 | 1442 |
| 4-1 | 1003 | 1121 | 693 |
| 4-2 | 1766 | 1580 | 1766 |
| 4-3 | 2241 | 2337 | 2317 |
| 5-1 | 1260 | - | - |
| 5-2 | 1409 | - | - |
| 5-3 | 1943 | - | - |
| 5-4 | 2871 | - | - |
| 6-1 | 1405 | - | - |
| 6-2 | 1320 | - | - |
| 6-3 | 890 | - | - |
| 6-4 | 1747 | - | - |
| 6-5 | 1462 | - | - |
| 7-4 | 1079 | - | - |
| 7-5 | 1797 | - | - |
| 7-6 | 2925 | - | - |
| 8-1 | 762 | - | - |
| 8-2 | 482 | - | - |
| 8-3 | 1636 | - | - |
| 8-4 | 876 | - | - |
| 8-5 | 1777 | - | - |
| 8-6 | 2005 | - | - |
| 8-7 | 3241 | - | - |
| 10-1 | 1523 | - | - |
| 10-3 | 1179 | - | - |
| 10-4 | 1853 | - | - |
| 10-5 | 1609 | - | - |
| 10-6 | 2159 | - | - |
| 10-7 | 2964 | - | - |
| 10-8 | 4691 | - | - |
| 15-1 | 564 | - | - |

Figure 5 presents the optimal route visualisations calculated by each algorithm for problem size `4-3`. The routes assigned to each vehicle and the distances between stops have been shown. The most efficient route, minimising total travel distance for each vehicle, has been generated by B&B. Higher costing routes have been produced by both QAOA variants. The impact of slight routing differences on overall solution quality has been highlighted in this diagram.

[1] Only the costs for problem sizes solved by each algorithm are displayed. For unsolved problem sizes and their experimental outcomes, refer to Tables 11, 12, and 13 available in the Appendix.

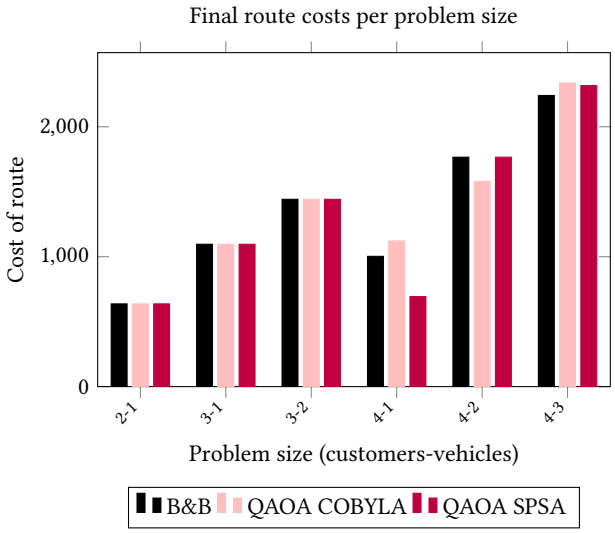
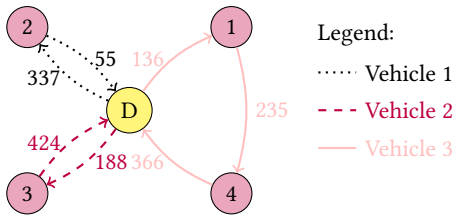
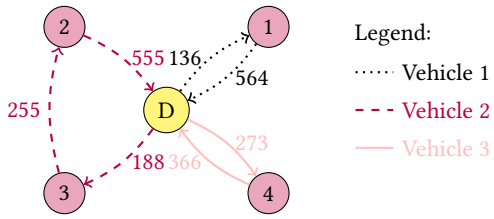


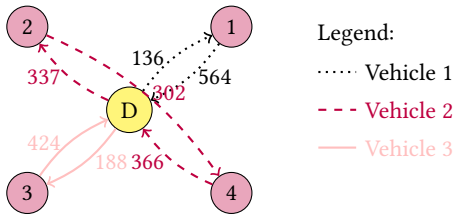

**Figure 4: The differences in final route costs across different problem sizes.**




**(a) Final route generated by B&B**




**(b) Final route generated by QAOA with COBLYA**




**(c) Final route generated by QAOA with SPSA**

**Figure 5: The final routes generated by each algorithm for an input of four customers and three vehicles.**

## 4.2 CPU Time

Substantial differences in computational efficiency and scalability have been highlighted by the analysis of CPU time across different algorithms, as shown in Tables 3, 4, and 5. Relatively strong computational efficiency has been demonstrated by B&B, with low CPU times maintained across most problem sizes. For smaller problems, negligible CPU time has been observed for B&B, with minimal variation. As problem sizes have increased, an upward trend in B&B's CPU time has been recorded, particularly at sizes `8-3` and `10-1`, though this growth has remained relatively gradual.

**Table 3: CPU time statistics for B&B[2]**

| Size | Best (s) | Average (s) | Worst (s) | STD |
|---|---|---|---|---|
| 2-1 | 0.0000 | 0.0000 | 0.0000 | 0.0000 |
| 3-1 | 0.0001 | 0.0001 | 0.0001 | 0.0000 |
| 3-2 | 0.0001 | 0.0001 | 0.0001 | 0.0000 |
| 4-1 | 0.0001 | 0.0002 | 0.0003 | 0.0000 |
| 4-2 | 0.0004 | 0.0005 | 0.0006 | 0.0001 |
| 4-3 | 0.0006 | 0.0007 | 0.0007 | 0.0000 |
| 5-1 | 0.0009 | 0.0009 | 0.0009 | 0.0000 |
| 5-2 | 0.0025 | 0.0025 | 0.0026 | 0.0000 |
| 5-3 | 0.0059 | 0.0060 | 0.0060 | 0.0000 |
| 5-4 | 0.0051 | 0.0059 | 0.0071 | 0.0008 |
| 6-1 | 0.0057 | 0.0059 | 0.0061 | 0.0001 |
| 6-2 | 0.0123 | 0.0139 | 0.0142 | 0.0007 |
| 6-3 | 0.0309 | 0.0312 | 0.0316 | 0.0002 |
| 6-4 | 0.0505 | 0.0512 | 0.0521 | 0.0006 |
| 6-5 | 0.0432 | 0.0575 | 0.0799 | 0.0114 |
| 7-4 | 0.3590 | 0.3647 | 0.3691 | 0.0035 |
| 7-5 | 0.5593 | 0.5660 | 0.5741 | 0.0056 |
| 7-6 | 0.5041 | 0.5249 | 0.5628 | 0.0195 |
| 8-1 | 0.0227 | 0.0236 | 0.0246 | 0.0006 |
| 8-2 | 0.1177 | 0.1200 | 0.1209 | 0.0011 |
| 8-3 | 2.0053 | 2.0247 | 2.0332 | 0.0098 |
| 8-4 | 2.5533 | 2.5924 | 2.6548 | 0.0353 |
| 8-5 | 6.3270 | 6.3940 | 6.4789 | 0.0463 |
| 8-6 | 7.1929 | 7.3594 | 7.4987 | 0.1034 |
| 8-7 | 4.7770 | 5.3018 | 5.7773 | 0.4057 |
| 10-1 | 1229.2973 | 1241.6108 | 1248.3768 | 6.5205 |
| 10-3 | 44.3235 | 45.2894 | 46.0346 | 0.5855 |
| 10-4 | 653.8471 | 662.6753 | 671.4617 | 5.5375 |
| 10-5 | 476.5849 | 484.8464 | 493.7667 | 7.2183 |
| 10-6 | 1247.9466 | 1267.3614 | 1291.4012 | 15.237 |
| 10-7 | 1606.9999 | 1653.3820 | 1678.1007 | 22.8200 |
| 10-8 | 1340.2807 | 1374.6141 | 1394.3114 | 17.7948 |
| 15-1 | 3392.7250 | 3426.6246 | 3480.4246 | 29.9229 |

Significantly higher CPU times have been shown across all tested problem sizes by both QAOA variants with COBYLA and SPSA optimisers, especially for larger instances. With COBYLA, CPU times have sharply increased from `9.04` seconds for size `2-1` to over `6000`

[2]Only the CPU time statistics for problem sizes solved by B&B are displayed. For unsolved problem sizes and their experimental outcomes, refer to Table 11 available in the Appendix.

seconds for sizes 4-1 and larger. An even more pronounced increase has been observed with SPSA, where CPU times have escalated from 20 seconds for size 2-1 to over 14000 seconds for sizes above 4-1, largely surpassing both COBYLA and B&B. Even for the smallest problem size, SPSA has required approximately twice the CPU time of COBYLA. For the largest solvable instance, SPSA has been more computationally demanding, taking 14550 seconds compared to COBYLA's 6608 seconds. Overall, while COBYLA has been more efficient than SPSA in terms of CPU time, both have fallen short of the computational performance achieved by B&B.

**Table 4: CPU time statistics for QAOA with COBLYA[3]**

| Size | Best (s) | Average (s) | Worst (s) | STD |
|---|---|---|---|---|
| 2-1 | 9.0368 | 9.4568 | 9.9533 | 0.3434 |
| 3-1 | 51.9174 | 53.7953 | 55.9595 | 1.6798 |
| 3-2 | 50.2398 | 52.3134 | 54.1024 | 1.5605 |
| 4-1 | 6957.8015 | 7023.4335 | 7078.6253 | 38.8989 |
| 4-2 | 6277.3285 | 6378.0725 | 6443.0287 | 51.2725 |
| 4-3 | 6608.7774 | 6645.0226 | 6672.2721 | 24.8904 |

**Table 5: CPU time statistics for QAOA with SPSA[4]**

| Size | Best (s) | Average (s) | Worst (s) | STD |
|---|---|---|---|---|
| 2-1 | 20.2237 | 21.0713 | 22.2761 | 0.7653 |
| 3-1 | 113.9934 | 117.8591 | 122.6308 | 3.4134 |
| 3-2 | 113.4740 | 118.8100 | 126.0478 | 4.8201 |
| 4-1 | 14646.5182 | 14698.8419 | 14765.4618 | 40.0098 |
| 4-2 | 14543.7797 | 14689.0642 | 14832.6404 | 90.3534 |
| 4-3 | 14550.1156 | 14669.9708 | 14819.5857 | 100.7905 |

The increase in CPU time for each algorithm as the problem size has grown is shown in Figure 6. A sharp rise has been observed for both QAOA variants at instances with 4 customers. B&B has been proven to be more efficient, with a problem size of 15-1 being solved in half the time required by QAOA with COBYLA for a 4-1 size. B&B's computational time has also been observed to increase significantly for larger instances, reflecting its exponential growth. This trend has been further demonstrated in Figure 7, where a new experiment presenting a fully executed problem size of 15-2 without a time limit reveals a dramatic rise in computation time between 15-1 and 15-2. The rapid escalation in B&B's time requirements with increasing problem complexity contrasts with the QAOA variants. Although initially competitive, significant slowdowns with even small increases in problem size have been experienced by QAOA. The scalability limitations of both quantum and classical algorithms have been highlighted by these findings.

The standard deviation in computational times for each algorithm for all problem sizes is illustrated in Figure 8. Consistent performance with low variability has been shown by B&B for smaller problem sizes. A noticeable increase in standard deviation for B&B has only been observed as the problem size has approached 10 customers. In contrast, significantly higher variability has been demonstrated by both QAOA variants, particularly with the SPSA optimiser, where the standard deviation has reached nearly 100 seconds for larger instances. This variability has suggested that, unlike B&B, consistent performance is not maintained by QAOA as problem size and complexity increase. Despite the increase in standard deviation, it has been noted that it remains relatively small for all algorithms compared to the total computation time. This indicates that overall fluctuations have not noticeably impacted the computation times in relation to the total runtime.

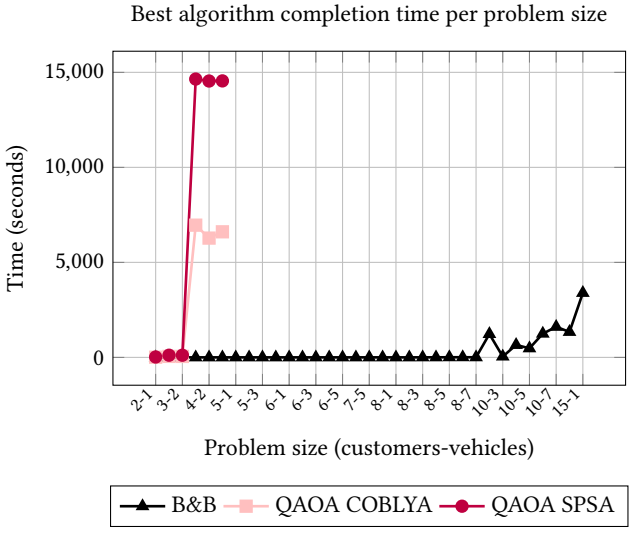



**Figure 6: The shortest times taken by each algorithm to solve each problem size.**

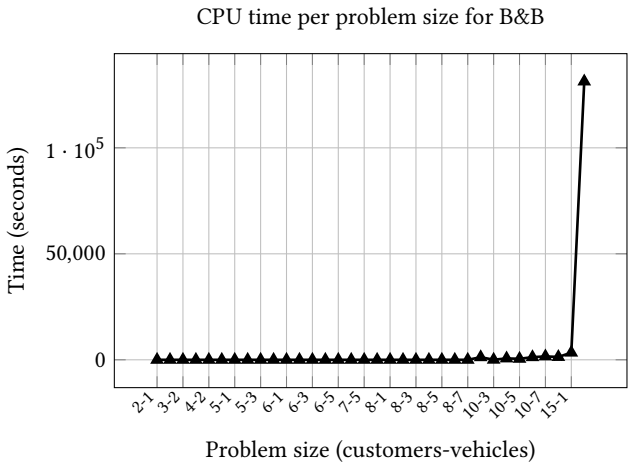



**Figure 7: Showing the exponential growth of B&B including an experiment run without a timeout.**

[3]Only the CPU time statistics for problem sizes solved by QAOA with COBLYA are displayed. For unsolved problem sizes and their experimental outcomes, refer to Table 12 available in the Appendix.

[4]Only the CPU time statistics for problem sizes solved by QAOA with SPSA are displayed. For unsolved problem sizes and their experimental outcomes, refer to Table 13 available in the Appendix.

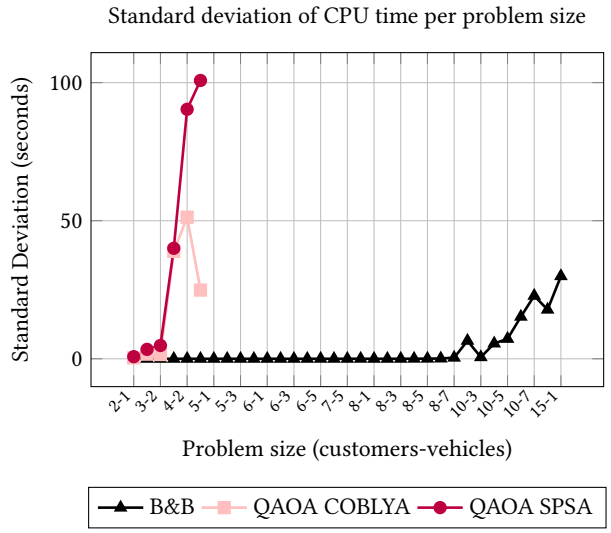



**Figure 8: The standard deviation of CPU time for each algorithm on different problem sizes.**

Overall, the results have suggested that QAOA is feasible for smaller problem sizes; however, its computational cost and variability increase significantly as problem complexity grows. B&B has consistently outperformed QAOA in both computational efficiency and consistency, making it the preferred choice for solving larger, more complex VRP instances.

## 4.3 Scalability

The scalability limitations of the algorithms evaluated in this study are presented in Table 6. This shows the largest problem sizes that have been solved by each algorithm. Instances in which the algorithms have faced timeouts, memory shortages, or other resource-related failures have been excluded.

**Table 6: Scalability of each algorithm**

| Algorithm | Largest problem size solved |
|---|---|
| B&B | 15-1 |
| QAOA COBLYA | 4-3 |
| QAOA SPSA | 4-3 |

The highest scalability has been demonstrated by B&B, with problem instances up to size 15-1 being solved within the allocated timeout. It is shown in Table 11 that instances of sizes 15-2 and 15-3 have hit the six-hour time limit and have been terminated, while instances larger than 15-3 have exhausted the 32GB memory limit. This outcome highlights the primary limitation of B&B — substantial computational resources are required for larger problem sizes, which are still relatively small compared to real-world scenarios.

Instances up to size 4-3 have been solved by QAOA variants. Instances with 5 customers are shown in Tables 12 and 13 to have exhausted available memory. It is also shown that those with 6 or more customers have exceeded current quantum algorithm limits, restricting the problem sizes that can be represented in a quantum state. The 32GB of memory has been exhausted by QAOA at much smaller problem sizes than by B&B, indicating substantial computational resource requirements. QAOA's solvable limits have been reached at relatively small problem sizes. The constraints involved in complex problems such as the VRP are likely the cause of this. The rapid decline in problem sizes solvable by both QAOA variants is illustrated in Figure 6, contrasting with B&B's capabilities and highlighting the current limitations of QAOA in handling larger, more complex instances.

These findings show that QAOA's scalability is restricted by current quantum limitations and algorithmic inefficiencies. B&B proves to be a more reliable choice for trying to solve larger VRP instances, reinforcing its role as a benchmark for assessing emerging quantum-based approaches.

## 4.4 Feasibility

Table 7 presents the feasibility percentages for B&B and QAOA. A feasibility percentage of 100% has been achieved by B&B, consistently finding valid solutions across all solvable instances. This aligns with its deterministic nature that guarantees a valid solution given sufficient resources.

**Table 7: Feasibility percentages of each algorithm**

| Algorithm | Feasibility percentage |
|---|---|
| B&B | 100% |
| QAOA COBLYA | 83.33% |
| QAOA SPSA | 83.33% |

A feasibility percentage of 83.33% has been achieved by both QAOA variants. Valid solutions have been generated for five out of six problem sizes, indicating occasional failures. Previous research [3] has suggested that this lower feasibility is linked to the $p$-value; increasing $p$ could improve valid solutions but would raise computational time. The need for further refinement in quantum algorithm design to enhance reliability and applicability for more complex problems is highlighted by these findings.

## 4.5 Success Rate

The success rates of the QAOA variants are detailed in Table 8. This metric is considered critical for evaluating the reliability of these algorithms in consistently finding solutions within a certain percentage of the optimal cost.

**Table 8: QAOA variant success rates at various thresholds**

| Algorithm | Success rate (95%) | Success rate (99%) |
|---|---|---|
| QAOA COBLYA | 83.33% | 66.67% |
| QAOA SPSA | 100% | 83.33% |

At a 95% threshold, a 100% success rate has been achieved by SPSA, consistently finding solutions within 95% of the optimal cost. A lower success rate of 83.33% has been achieved by COBYLA, indicating reduced reliability. At a stricter threshold of 99%, success rates for both algorithms have decreased, with SPSA at 83.33%

and COBYLA dropping to 66.67%. The challenges faced by these QAOA variants in consistently reaching highly accurate solutions are emphasised by this decline in success rates as the threshold increases. This is particularly the case as problem size increases or the cost function landscape becomes more complex.

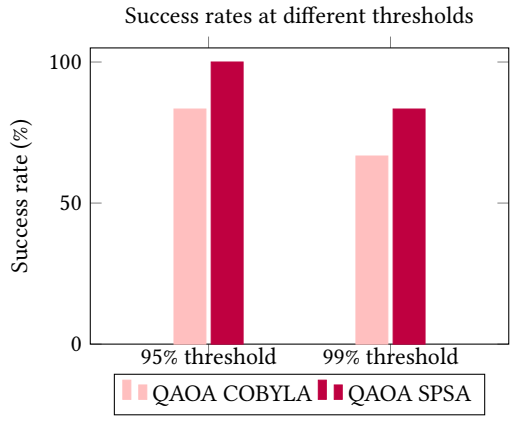



**Figure 9: The success rates for both QAOA Variants at 95% and 99% thresholds.**

Figure 9 illustrates the performance of both algorithms. The results have shown that while SPSA has been more reliable in achieving solutions closer to the optimal cost compared to COBYLA, neither optimiser has enabled QAOA to consistently match the accuracy of B&B, particularly at higher accuracy thresholds. The lower success rates at higher thresholds suggest that further optimisation and refinement of these quantum algorithms may be necessary to enhance their effectiveness and reliability for solving larger, more complex VRP instances.

## 4.6 Comparison with Simulated Annealing

While QAOA and B&B are the primary focus of this study, a comparison with SA, a popular heuristic for combinatorial optimisation problems such as the VRP, has been provided for valuable insights. It has been shown in a recent study [5] on the same dataset that all problem sizes, including the largest instance (15-8), have been consistently solved by SA. Negligible CPU time has been obtained in all instances, demonstrating greater computational efficiency than both QAOA and B&B as problem sizes have increased.

A 100% feasibility rate has been maintained by SA, consistently finding valid solutions comparable to B&B and exceeding the 83.33% feasibility rate of QAOA. However, more variability in the solution quality of SA has been observed with larger problem sizes, as indicated by an increasing standard deviation. These findings suggest that while efficiency and scalability have been demonstrated by SA, the stability of B&B and QAOA in consistently producing the same solution has not been matched.

Overall, SA has been shown to excel in computational efficiency and scalability but has not matched B&B in precision. QAOA has been outperformed by SA in feasibility and computational efficiency. This comparison highlights the trade-offs between classical methods and their performance relative to QAOA.

## 4.7 Summary of Main Points

Below are some of the main points to note from the above discussion of the experimental results:

- **Solution Quality**: B&B has consistently found optimal solutions, while QAOA's performance has declined with increasing complexity, with SPSA generally outperforming COBYLA.
- **CPU Time**: B&B has demonstrated greater time efficiency, especially for larger problems, while QAOA, particularly with SPSA, has been slower.
- **Scalability**: B&B has handled larger problem sizes better, whereas QAOA has been constrained by qubit limits, circuit depth, and algorithm efficiency.
- **Feasibility**: B&B has achieved 100% feasibility, consistently finding feasible solutions, while QAOA variants have shown 83.33% feasibility, indicating occasional failures and a need for further refinement.
- **Success Rate**: SPSA has achieved a higher success rate than COBYLA, but both have struggled with accuracy on larger, more complex problems.
- **Comparison with Simulated Annealing**: SA has exhibited greater efficiency and feasibility, but its solution quality has varied with problem size, lacking B&B's precision while outperforming QAOA in practical metrics.

# 5 CONCLUSIONS

Based on the findings of this study, several conclusions have been drawn about the performance of QAOA compared to the classical B&B method for solving VRP instances.

It has been consistently demonstrated that B&B outperforms QAOA in solution quality, computational efficiency, scalability, feasibility, and success rate, especially as problem size has increased. Competitiveness has been shown by QAOA on smaller problems, but a significant decline in performance has been observed with increased complexity. This indicates that further development is needed for QAOA to match classical methods in solving the VRP.

The impact of optimiser choice on QAOA's performance has also been explored in this study. Adequate performance has been observed for both optimisers on smaller problems. Significant differences have emerged only for instances larger than 3 customers and 2 vehicles. Solutions closer to the optimal cost have generally been yielded by SPSA, while lower computational times have been demonstrated by COBYLA. However, consistent matching of B&B's performance has not been achieved by either optimiser as problem size has grown. This suggests that optimiser choice alone has not been enough to bridge the gap with classical methods.

QAOA's main limitations, compared to B&B, have been highlighted by its lower feasibility percentage and reduced scalability. Current quantum limitations, such as qubit count, circuit depth, and noise levels, have forced constraints on QAOA, restricting its ability to solve larger problems effectively. A significant increase in computational time has also been observed with problem size, making QAOA less practical for larger, more complex VRP instances. In contrast, manageable computational times have been maintained by B&B, demonstrating superior scalability and efficiency.

Overall, promise has been shown by QAOA for smaller VRP instances, but it has remained limited by current quantum constraints and algorithmic inefficiencies. Evidence has indicated that advantages can be offered by quantum computing in certain scenarios [10], but it has yet to consistently outperform classical methods in solving combinatorial optimisation problems. The more robust choice for larger problems has been B&B, underscoring its status as a benchmark for evaluating new quantum approaches. Future advancements in quantum hardware and algorithms are needed to improve the feasibility and applicability of quantum algorithms such as QAOA for complex optimisation problems.

## 5.1 Limitations

Several limitations of this study have been identified and should be considered when interpreting the results. Firstly, the experiments have been conducted using quantum simulators rather than physical quantum devices due to the constraints of an Honours research project. While valuable insights into quantum algorithm performance are provided by simulators, they do not fully replicate the potential advantages of real quantum hardware, such as quantum speedup. This could have affected the results and conclusions about the effectiveness of algorithms like QAOA.

It has also been noted that each algorithm has been tested on only one instance per problem size. Testing multiple data inputs for each size would have provided a more comprehensive understanding of the algorithms' performance, ensuring that results have not been biased by specific problem instances and making the conclusions more reliable.

Lastly, the evaluation of the algorithms, particularly for larger problem sizes, has been limited by the available classical computing power. Insufficient computational resources have caused memory limitations, restricting the amount of data and insights that could have been obtained. Greater access to computational resources would have allowed for more extensive experiments and deeper insights into the scalability and performance of both B&B and QAOA.

## 5.2 Future Work

Future research should focus on experiments being conducted on physical quantum devices to accurately assess quantum advantages that simulators cannot fully replicate. A more realistic evaluation of quantum algorithms, such as QAOA, for solving complex optimisation problems such as the VRP would be provided, allowing for direct comparisons between simulators and actual quantum hardware.

Enhancements to classical computing resources should also be considered for further research. Greater computational power would enable larger problem sizes to be tested without memory constraints. A more comprehensive exploration of both quantum and classical algorithms would therefore be possible, providing deeper insights into their scalability and performance.

These areas should be addressed to help future research build on the findings of this study, offering a fuller understanding of the comparative performance of quantum and classical algorithms in solving the VRP.

# REFERENCES


[1] Akshay Ajagekar and Fengqi You. 2019. Quantum computing for energy systems optimization: Challenges and opportunities. *Energy* 179 (07 2019), 76–89. https://doi.org/10.1016/j.energy.2019.04.186

[2] Muhammad Alsaiyari and Muhamad Felemban. 2023. Variational Quantum Algorithms for Solving Vehicle Routing Problem. , 4 pages. https://doi.org/10.1109/ICSCA57840.2023.10087522

[3] Utkarsh Azad, Bikash K. Behera, Emad A. Ahmed, Prasanta K. Panigrahi, and Ahmed Farouk. 2022. Solving Vehicle Routing Problem Using Quantum Approximate Optimization Algorithm. *IEEE Transactions on Intelligent Transportation Systems* 24 (2022), 1–10. https://doi.org/10.1109/TITS.2022.3172241

[4] Jose Caceres-Cruz, Pol Arias, Daniel Guimarans, Daniel Riera, and Angel A. Juan. 2014. Rich Vehicle Routing Problem. *Comput. Surveys* 47 (12 2014), 1–28. https://doi.org/10.1145/2666003

[5] Ben Cleveland. 2024. *Benchmarking Classical Methods against Variational Quantum Eigensolver to Solve the Vehicle Routing Problem: Simulated Annealing.* Ph. D. Dissertation.

[6] Rina Dechter and Judea Pearl. 1985. Generalized best-first search strategies and the optimality of A*. *J. ACM* 32 (07 1985), 505–536. https://doi.org/10.1145/3828.3830

[7] Kaelyn Ferris. 2024. Qiskit 1.0 release summary | IBM Quantum Computing Blog. https://www.ibm.com/quantum/blog/qiskit-1-0-release-summary

[8] Matteo Fischetti and Domenico Salvagnin. 2010. Pruning Moves. *INFORMS journal on computing* 22 (02 2010), 108–119. https://doi.org/10.1287/ijoc.1090.0329

[9] Maxine T. Khumalo, Hazel A. Chieza, Krupa Prag, and Matthew Woolway. 2022. An investigation of IBM quantum computing device performance on combinatorial optimisation problems. *Neural Computing and Applications* (06 2022). https://doi.org/10.1007/s00521-022-07438-4

[10] Youngseok Kim, Andrew Eddins, Sajant Anand, Ken Xuan Wei, Ewout van den Berg, Sami Rosenblatt, Hasan Nayfeh, Yantao Wu, Michael Zaletel, Kristan Temme, and Abhinav Kandala. 2023. Evidence for the utility of quantum computing before fault tolerance. *Nature* 618 (06 2023), 500–505. https://doi.org/10.1038/s41586-023-06096-3

[11] Adam Letchford and Juan-Jose Salazar-Gonzalez. 2018. CVRP instances. *research-data.ull.es* 1 (06 2018). https://doi.org/10.17632/kcc52cw4zs.1

[12] Adam N. Letchford and Juan-José Salazar-González. 2019. The Capacitated Vehicle Routing Problem: Stronger bounds in pseudo-polynomial time. *European Journal of Operational Research* 272 (01 2019), 24–31. https://doi.org/10.1016/j.ejor.2018.06.002

[13] David R. Morrison, Sheldon H. Jacobson, Jason J. Sauppe, and Edward C. Sewell. 2016. Branch-and-bound algorithms: A survey of recent advances in searching, branching, and pruning. *Discrete Optimization* 19 (02 2016), 79–102. https://doi.org/10.1016/j.disopt.2016.01.005

[14] Karthik K Srinivasan, Saipriya Satyajit, Bikash K Behera, and Prasanta K Panigrahi. 2018. Efficient quantum algorithm for solving travelling salesman problem: An IBM quantum experience. *arXiv (Cornell University)* (05 2018). https://doi.org/10.48550/arxiv.1805.10928

[15] Paolo Toth and Daniele Vigo (Eds.). 2002. *The Vehicle Routing Problem.* Society for Industrial and Applied Mathematics. https://doi.org/10.1137/1.9780898718515

[16] Paolo Toth and Daniele Vigo. 2015. *Vehicle routing : problems, methods, and applications.* Society for Industrial and Applied Mathematics (SIAM, 3600 Market Street, Floor 6, Philadelphia, PA 19104).

## APPENDIX

$$\min \quad \sum_{i \to j} w_{ij} x_{ij} \tag{3}$$

$$\text{s.t.} \quad \sum_{i \in source[i]} x_{ij} = 1 \qquad \forall i \in \{1, ..., n-1\} \tag{4}$$

$$\sum_{j \in target[i]} x_{ji} = 1 \qquad \forall i \in \{1, ..., n-1\} \tag{5}$$

$$\sum_{j \in source[0]} x_{0j} = k \tag{6}$$

$$\sum_{j \in target[0]} x_{j0} = k \tag{7}$$

**Figure 10: Mathematical model of the Vehicle Routing Problem**

$$H_{VRP} = H_A + H_B + H_C + H_D + H_E \tag{8}$$

$$H_A = \sum_{i \to j} w_{ij} x_{ij} \tag{9}$$

$$H_B = A \sum_{i \in 1,...,n-1} \left( 1 - \sum_{j \in source[i]} x_{ij} \right)^2 \tag{10}$$

$$H_C = A \sum_{i \in 1,...,n-1} \left( 1 - \sum_{j \in target[i]} x_{ji} \right)^2 \tag{11}$$

$$H_D = A \left( k - \sum_{j \in source[0]} x_{0j} \right)^2 \tag{12}$$

$$H_E = A \left( k - \sum_{j \in target[0]} x_{j0} \right)^2 \tag{13}$$

**Figure 11: Energy function of the Vehicle Routing Problem**

$$\min \quad \vec{x}^T Q \vec{x} + \vec{g}^T \vec{x} + c \tag{14}$$

$$\vec{x} = [x_{(0,1)}, x_{(0,2)}, \dots x_{(1,0)}, x_{(1,2)}, \dots x_{(n-1,n-2)}]^T \tag{15}$$

$$\sum_{j \in source[i]} x_{ij} = \vec{z}_{S[i]}^T \vec{x} \tag{16}$$

$$\sum_{j \in target[i]} x_{ji} = \vec{z}_{T[i]}^T \vec{x} \tag{17}$$

$$Q = A \Bigg( [\vec{z}_{T[1]}, \vec{z}_{T[2]}, \dots, \vec{z}_{T[n-1]}, \vec{z}_{T[0]}, \vec{z}_{T[2]}, \dots, \vec{z}_{T[n-2]}]^T, + [[\vec{z}_{S[0]}]^{\times(n-1)} [\vec{z}_{S[1]}]^{\times(n-1)} \dots [\vec{z}_{S[n-1]}]^{\times(n-1)}] \Bigg) \tag{18}$$

$$\vec{g} = \vec{w} - 2A(\vec{J} + \vec{K}) - 2Ak(\vec{z}_{S[0]} + \vec{z}_{T[0]}) \tag{19}$$

$$c = 2A(n-1) + 2Ak^2 \tag{20}$$

$$\vec{J} = n \times n \text{ matrix with first } n-1 \text{ elements } = 0 \text{ and next } (n-1)^2 \text{ elements } = 1 \tag{21}$$

$$\vec{K} = \vec{x} \text{ with } x_{ij} = 1 \text{ if } j \neq 0, \forall i \in \{0, \dots, n-1\}, \text{ else } 0 \tag{22}$$

$$\vec{w} = \text{ weight vector} \tag{23}$$

**Figure 12: QUBO formulation of the Vehicle Routing Problem**

$$H_{\text{Ising}} = -\sum_i \sum_{j<i} I_{ij} s_i s_j + \sum_i h_i s_i + d \tag{24}$$

$$x_{ij} = \frac{s_{ij} + 1}{2} \qquad s_{ij} \in \{-1, 1\} \tag{25}$$

$$I_{ij} = -\frac{Q_{ij}}{4} \qquad \forall i < j, \quad I_{ii} = 0 \quad \forall i \tag{26}$$

$$h_i = \frac{g_i}{2} + \sum_j \frac{Q_{ij}}{4} + \sum_j \frac{Q_{ji}}{4} \tag{27}$$

$$d = c + \sum_i \frac{g_i}{2} + \sum_i \frac{Q_{ii}}{4} + \sum_i \sum_j \frac{Q_{ij}}{4} \tag{28}$$

**Figure 13: Ising formulation of the Vehicle Routing Problem**

**Algorithm 1** Branch and Bound

```
activeSet ← Initial Node
bestVal ← Initial Value
currentBest ← Initial Solution
T0 ← Tmax
while activeSet not empty do
    Choose branching node, node k ∈ activeSet
    Remove node k from activeSet
    Generate children of node k, child i, where i ← 1, ..., nk
    and their respective lower bounds, lb_i
    for i = 1 to nk do
        if child i dominated by another then
            reject child i as possible solution
        else if child i is a complete solution then
            bestVal ← lb_i
            currentBest ← child i
        else
            add child i to activeSet
        end if
    end for
end while
return currentBest
```

| Symbol | Description |
|---|---|
| $activeSet$ | Set of nodes currently being considered |
| $bestVal$ | Current best objective function value |
| $currentBest$ | Current best allocation of routes to vehicles |
| $T0$ | Maximum time allowed to run |
| $i$ | Current sub-problem being analysed |
| $nk$ | Number of sub-problems of current node |
| $lb$ | Lower-bound of current sub-problem |

**Table 9: Table of symbols and descriptions for Branch and Bound.**

**Algorithm 2** Quantum Approximate Optimisation Algorithm

**Input:**
$inputmatrix(ces), initialpoint, maxiter$
**Output:**
$\pi^*, cost^*$
**initialisation**
$qubitOpdocplex \leftarrow$ BuildModel($inputmatrix(ces)$)
$num \leftarrow$ number of qubits of $qubitOpdocplex$
$spsa \leftarrow$ SPSA($maxiter$)
$qaoa \leftarrow$ QAOA($qubitOpdocplex, spsa, initialpoint$)
$quantuminstance \leftarrow$ Backend(1024 shots)
$result \leftarrow$ Run($qaoa, quantuminstance$)
$\pi^*, cost^* \leftarrow$ FeasibleOutput($result[eigenstate]$))

| Symbol | Description |
|---|---|
| $inputmatrix(ces)$ | Cost matrix of the VRP |
| $initialpoint$ | Initial parameters for the quantum circuit that will be optimised |
| $maxiter$ | Maximum number of iterations for optimisation process |
| $\pi^*$ | Solution's allocation of routes to vehicles |
| $cost^*$ | Cost associated with solution |
| $qubitOpdocplex$ | Quantum operator |
| $num$ | Number of qubits required for quantum operator to represent VRP |
| $spsa$ | Variable assigned to instance of optimiser |
| $qaoa$ | Instance of QAOA algorithm |
| $quantuminstance$ | Quantum environment where QAOA algorithm will be executed |
| $result$ | Output from executing the QAOA instance on quantum instance |

**Table 10: Table of symbols and descriptions for Quantum Approximate Optimisation Algorithm.**

**Table 11: Experimental outcomes for unsolved B&B experiments**

| Size | Experiment outcome |
|---|---|
| 15-2 | Timeout |
| 15-3 | Timeout |
| 15-4 | Out of Memory |
| 15-5 | Out of Memory |
| 15-6 | Out of Memory |
| 15-7 | Out of Memory |
| 15-8 | Out of Memory |

**Table 12: Experimental outcomes for unsolved QAOA COBLYA experiments**

| Size | Experiment outcome |
|---|---|
| 5-1 | Out of Memory |
| 5-2 | Out of Memory |
| 5-3 | Out of Memory |
| 5-4 | Out of Memory |
| 6-1 | Qubit Limit |
| 6-2 | Qubit Limit |
| 6-3 | Qubit Limit |
| 6-4 | Qubit Limit |
| 6-5 | Qubit Limit |
| 7-4 | Qubit Limit |
| 7-5 | Qubit Limit |
| 7-6 | Qubit Limit |
| 8-1 | Qubit Limit |
| 8-2 | Qubit Limit |
| 8-3 | Qubit Limit |
| 8-4 | Qubit Limit |
| 8-5 | Qubit Limit |
| 8-6 | Qubit Limit |
| 8-7 | Qubit Limit |
| 10-1 | Qubit Limit |
| 10-3 | Qubit Limit |
| 10-4 | Qubit Limit |
| 10-5 | Qubit Limit |
| 10-6 | Qubit Limit |
| 10-7 | Qubit Limit |
| 10-8 | Qubit Limit |
| 15-1 | Qubit Limit |
| 15-2 | Qubit Limit |
| 15-3 | Qubit Limit |
| 15-4 | Qubit Limit |
| 15-5 | Qubit Limit |
| 15-6 | Qubit Limit |
| 15-7 | Qubit Limit |
| 15-8 | Qubit Limit |

**Table 13: Experimental outcomes for unsolved QAOA SPSA experiments**

| Size | Experiment outcome |
|---|---|
| 5-1 | Out of Memory |
| 5-2 | Out of Memory |
| 5-3 | Out of Memory |
| 5-4 | Out of Memory |
| 6-1 | Qubit Limit |
| 6-2 | Qubit Limit |
| 6-3 | Qubit Limit |
| 6-4 | Qubit Limit |
| 6-5 | Qubit Limit |
| 7-4 | Qubit Limit |
| 7-5 | Qubit Limit |
| 7-6 | Qubit Limit |
| 8-1 | Qubit Limit |
| 8-2 | Qubit Limit |
| 8-3 | Qubit Limit |
| 8-4 | Qubit Limit |
| 8-5 | Qubit Limit |
| 8-6 | Qubit Limit |
| 8-7 | Qubit Limit |
| 10-1 | Qubit Limit |
| 10-3 | Qubit Limit |
| 10-4 | Qubit Limit |
| 10-5 | Qubit Limit |
| 10-6 | Qubit Limit |
| 10-7 | Qubit Limit |
| 10-8 | Qubit Limit |
| 15-1 | Qubit Limit |
| 15-2 | Qubit Limit |
| 15-3 | Qubit Limit |
| 15-4 | Qubit Limit |
| 15-5 | Qubit Limit |
| 15-6 | Qubit Limit |
| 15-7 | Qubit Limit |
| 15-8 | Qubit Limit |