IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 28 October 2024, accepted 17 January 2025, date of publication 27 January 2025, date of current version 5 February 2025.

Digital Object Identifier 10.1109/ACCESS.2025.3534677

RESEARCH ARTICLE

Graph Coarsening Approach to the Vehicle Routing Problem: An Approximation Strategy

KATARZYNA NAŁĘCZ-CHARKIEWICZ[1], ARNAV DAS[2,3], TURBASU CHATTERJEE[4], JOSHUA KEENE[5], PAWEŁ GORA[6], AND CARLOS C. N. KUHN[5,7]

[1]Faculty of Electronics and Information Technology, Warsaw University of Technology, 00-661 Warsaw, Poland
[2]Department of Computer Science, St. Xavier's University, Kolkata 700160, India
[3]India Internet Foundation, Kolkata, West Bengal 700091, India
[4]Department of Computer Science, Virginia Tech, Blacksburg, VA 24061, USA
[5]12thLevel Pty Ltd., Barton, ACT 2600, Australia
[6]Fundacja Quantum AI, 02-110 Warsaw, Poland
[7]Faculty of Science and Technology, University of Canberra, Canberra, ACT 2617, Australia

Corresponding author: Katarzyna Nałęcz-Charkiewicz (katarzyna.nalecz-charkiewicz.dokt@pw.edu.pl)

This work was supported by the Australian Capital Territory (ACT) Government, Future Jobs Fund–Open Source Institute (OpenSI) under Grant R01553.

ABSTRACT In the Noisy Intermediate-Scale Quantum (NISQ) era of quantum computing, solving complex optimization problems such as the Vehicle Routing Problem (VRP) remains a formidable challenge. To overcome this obstacle, we introduce a novel method in this paper that focuses on reducing the number of edges in a graph based on Euclidean distances between vertices while preserving the crucial characteristics needed for solving the VRP. Our graph coarsening approach combined with the inflation method enables the reduced graph to be processed efficiently by classical, quantum or hybrid (quantum-classical) solvers, resulting in a faster computation time. When using the proposed method with hybrid algorithms, we have demonstrated an improvement of approximately 50% in the solution found by the hybrid solver. The practicality of our method on quantum hardware highlights its potential to contribute to the advancement of quantum and hybrid algorithms and applications during the NISQ era.

INDEX TERMS Combinatorial optimization, graph reduction, vehicle routing, quantum computing.

I. INTRODUCTION

The Vehicle Routing Problem (VRP), introduced by Dantzig and Ramser over 60 years ago, is a fundamental combinatorial optimization challenge with lasting implications in areas such as logistics [1]. It involves finding the optimal way to deliver goods to a set of locations using a fleet of vehicles, minimizing costs and satisfying constraints. Despite its importance and numerous solution approaches over the decades, the *NP*-hard nature of VRP and its variants means that solving its large instances on classical computers remains a challenge [2]. Therefore, there is still a need to search for new, efficient solvers for this type of combinatorial problem in logistics.

It is noteworthy to mention that there exist many variants of VRP, among the most popular are: Capacitated Vehicle Routing Problem (CVRP), which aims to minimize the total route distance while adhering to vehicle capacity constraints, and the Vehicle Routing Problem with Time Windows (VRPTW), which extends the VRP by stipulating specific time windows for customer service.

Although not all combinatorial optimization problems are inherently difficult for classical computers, those classified as NP-hard, such as the various forms of VRP, pose a substantial challenge for traditional computing methodologies. This provides an appeal for solving the vehicle routing problems using quantum computing. In this context, quantum computing has been shown to provide quadratic speedup over its classical counterparts by applying Grover's search algorithm [3] to solve the Traveling Salesman Problem (TSP), which is a special case of VRP. Moylett et al. demonstrated that quadratic quantum speedup can be achieved by applying a quantum backtracking algorithm to a classical approach [4]. Raj and Shivakumar employed Grover's quantum algorithm

The associate editor coordinating the review of this manuscript and approving it for publication was Binit Lukose.

to accelerate the solution of the Hamiltonian cycle problem, which is closely related to TSP [5]. Furthermore, attempts have been made to apply the quantum computing paradigm to solve the CVRP, an example of which is the work focusing on the application of the Quantum Approximate Optimization Algorithm (QAOA) [6].

However, present-day quantum computers fall under the aptly coined Noisy Intermediate Scale Quantum (NISQ) computing era [7], characterized by noise, limited coherence times, and a small number of operating qubits. This significantly limits the size and complexity of specific problem instances that can be solved using quantum computers. To address this issue, various techniques have been proposed to reduce the size of the problem instance while preserving its essential features, thereby ensuring the feasibility of the Integer Linear Programming (ILP) model used to solve the VRP. Classical methods such as Benders' decomposition [8] and Dantzig-Wolfe decomposition [9], which have a long history, have been employed in studies such as [10], [11], and [12]. In the field of quantum optimization, recent decomposition-based approaches, such as the one proposed by Gambella et al. [13] aim to extend the applicability of mixed binary optimization problems. Ponce et al. [14] presented a graph decomposition technique for combinatorial optimization using QAOA.

Another method is to coarsen the graph by reducing the number of edges based on the Euclidean distances between the vertices. This approach can then feed the coarsened graph into a classical solver to reduce the computation time or even provide the reduced graph to a quantum computer, which is the original inspiration for this study. Gilbert et al. extensively explored efficient graph coarsening techniques (outside the quantum realm), specifically highlighting the effectiveness of the Heavy Edge Coarsening method in large-scale graph analysis tasks [15].

The VRP exhibits a well-defined geometric and topological structure, making it amenable to graph representation. Graph coarsening techniques have the inherent capability to preserve this essential structure [16], ensuring that the fundamental characteristics of the problem are retained during problem size reduction. Graph coarsening methods have shown promising results in reducing the size and complexity of VRP instances for quantum computing [17], [18].

In line with these works, this study underscores the impact of our algorithm, particularly highlighting its dual strengths: not only does it employ a streamlined coarsening technique grounded in Euclidean distances to effectively diminish the scale of CVRP instances, but it also introduces a methodical inflation method (a way to return to the original graph, maintaining its original structure and characteristics). This systematic approach is crucial in the NISQ era of quantum computing.

The rest of the paper is organized as follows: Section II outlines the Integer Linear Programming (ILP) model for CVRP, our graph coarsening technique, and the inflation method. Section III describes our computational experiments. Section IV discusses the solution trends, results for different instances, and time variations. Section V concludes the paper with prospects for future research.

II. PRELIMINARY CONCEPTS

A. ILP DEFINITION OF THE CVRP

The ILP model implemented in this study is known as the Miller-Tucker-Zemlin (MTZ) formulation [19], and we followed a few steps from [20]. Mathematically, the CVRP is a combinatorial optimization problem, and the goal is to minimise the total distance the vehicles travel under a capacity constraint while visiting all vertices and starting and ending in the depot.

The goal is then to minimise the value of the objective function F:

$$F(x) = \sum_{i,j\in\mathcal{V}, i\neq j} w_{ij}x_{ij}, \tag{1}$$

where x is a set of binary decision variables x_{ij} that corresponds to the edge connecting vertex i to j ($x_{ij} = 1$ if the solution includes a direct link between i and j, and 0 otherwise), w_{ij} is the cost of traveling between i and j, $\mathcal{V}$ is the set of vertices $\{0, 1, 2, \ldots, n\}$. The depot is identified as vertex 0.

This is subject to subtour elimination constraints, which also take into consideration the truck capacities, Q, and demands of the vertices, q_i.

$$u_j - u_i \geq q_j - Q(1 - x_{ij}) \qquad \forall i, j \in \mathcal{V} \setminus \{0\} \qquad i \neq j \tag{2}$$

$$q_i \leq u_i \leq Q \qquad \forall i \in \mathcal{V} \setminus \{0\} \tag{3}$$

Here, u_i is a continuous variable used to track the capacities of the vehicle when it visits the vertices, while the goal of these constraints is to ensure that all routes forming cycles have to pass through the depot (the details are in [19]).

The model also incorporates the vertices visiting constraint:

$$\sum_{i\in\mathcal{V}} x_{ij} = 1 \qquad \forall j \in \mathcal{V} \setminus 0 \qquad i \neq j \tag{4}$$

$$\sum_{j\in\mathcal{V}} x_{ij} = 1 \qquad \forall i \in \mathcal{V} \setminus 0 \qquad i \neq j \tag{5}$$

$$\tag{6}$$

and the depot visiting constraints:

$$\sum_{i\in\mathcal{V}} x_{0i} = K \tag{7}$$

$$\sum_{j\in\mathcal{V}} x_{j0} = K \tag{8}$$

where K is the number of vehicles.

B. COARSENING METHOD FOR CVRP

Assuming that we have a fully-connected graph $G = (\mathcal{V}, \mathcal{E})$ with $|\mathcal{V}|$ vertices and $|\mathcal{E}|$ edges, the goal is to find a smaller

network that retains sufficient information to find a good solution for a CVRP instance. This process can be performed using the method described in Algorithm 1.

Each vertex v_i has Cartesian coordinates defined by (x_i, y_i). Thus, the length of the edge between vertices v_i and v_j is defined as

$$\omega_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \tag{9}$$

First, all edges are sorted by increasing length, using the *SortEdges* procedure. Then, the function *IdentifyPairs* determines which node pairs fall within the calculated *coarsenRadius* (i.e., the lengths of their edges are lower than the length of the edge determined using parameters *bounds* and *radiusCoefficient*) and should therefore be considered for merging. Finally, the *Coarsen* function applies the coarsening procedure to the identified pairs of nodes, effectively merging them and updating the graph to its coarser form. The coarsening process is illustrated in Figure 1.

Algorithm 1 Graph Coarsening Algorithm

Input: Original graph $G(V, E)$, coarsening rate P, radiusCoefficient
Output: Coarsened graph
Initialization:
Initialize coarsened graph $G'(V', E') \leftarrow G(V, E)$ to start the coarsening process;
Coarsening Loop:
while $|V'| > P \cdot |V|$ **do**
  sortedEdges $\leftarrow$ SortEdges(G');
  pairsToCoarsen $\leftarrow \emptyset$, bounds $\leftarrow 1$, loopCounter $\leftarrow 0$;
  while pairsToCoarsen $= \emptyset$ and loopCounter < 10 **do**
    part $\leftarrow \lfloor |V'| \cdot$ bounds $\cdot$ radiusCoefficient$\rfloor$;
    if part < Size(sortedEdges) **then**
      coarsenRadius $\leftarrow$ Length(sortedEdges[part]);
      pairsToCoarsen $\leftarrow$ IdentifyPairs(sortedEdges, coarsenRadius);
      bounds $\leftarrow$ bounds $\cdot$ 1.4;
      loopCounter $\leftarrow$ loopCounter + 1;
    else
      return G';
    end if
  end while
  $G'' \leftarrow$ Coarsen(pairsToCoarsen);
  Update node capacities for G'' by sequentially summing capacities of fine nodes that are merged into coarsened nodes during the coarsening process
  Record mapping from G to G''
  if $|V''| = |V'|$ **then**
    break ▷ Break the loop to prevent infinite iteration;
  end if
  $G' \leftarrow G''$;
end while
return G';

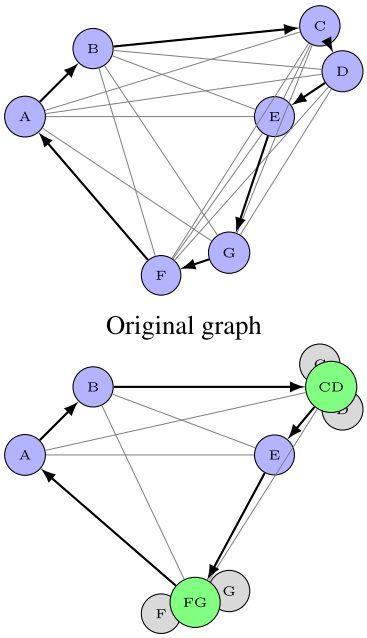


FIGURE 1. This figure serves as an illustration of an instance where vertex A represents the depot. On the upper image, we have the original network, while on the lower image, an example demonstrates the algorithm's identification of pairs of vertices within a certain threshold distance of each other. In this case, vertices C and D, as well as vertices F and G (vertices in gray), are identified and coarsened. This process involves creating temporary vertices positioned at the midpoint between the original vertices (vertices in green). By coarsening these vertices, the size of the network is reduced while preserving the essential characteristics necessary for further analysis and optimization.

When ω_{ij} is smaller than the threshold, it indicates that the vertices v_i and v_j should collapse to create a single vertex, which generates a new vertex v'_{ij} located at

$$\left(x'_{ij}, y'_{ij}\right) = \left(\frac{x_i + x_j}{2}, \frac{y_i + y_j}{2}\right) \tag{10}$$

Moreover, if any vertex is a neighbor of either v_i or v_j, it becomes a neighbor of the newly created vertex v'_{ij}. Recalculation of the edge lengths is carried out using the newly created vertices and Equation (9). It is important to note that the coarsening process occurs in pairs, with the identification of vertices to be coarsened taking place in layers. This implies that one pair is coarsened in each iteration (in ascending order by edge length) until the desired coarsening ratio is achieved. In the example depicted in Figure 1, the coarsening process begins by coarsening vertices C and D, subsequently, vertices F and G are coarsened, which is followed by the recalculation of the new network. Throughout this process, the coarsening layers are tracked to ensure proper inflation back to the original network.

C. INFLATION METHOD

Since the above method is performed in layers of coarsening, it is reversible and easy to inflate back to the original network. As the goal is to minimize the distance traveled by the

vehicles, the inflation method on the route after the coarsened network has been optimized is based on the total distance a vehicle needs to cover to go from the origin vertex to the destination vertex. As shown in Figure 1, the vehicle route is:

$$v_B \rightarrow v'_{C,D} \rightarrow v_E \tag{11}$$

Then, the inflation algorithm is:

$$\begin{aligned} &\textit{if:} \quad \omega_{BC} + \omega_{DE} < \omega_{CE} + \omega_{BD} \\ &\qquad v'_{C,D}: \quad v_C \rightarrow v_D \\ &\textit{else:} \\ &\qquad v'_{C,D}: \quad v_D \rightarrow v_C \end{aligned} \tag{12}$$

Here, the notation $v_C \rightarrow v_D$ indicates that if one solves the coarsened instance and wishes to recreate a route for the original graph, the route should proceed from v_C to v_D.

III. EXPERIMENTAL EVALUATION

To empirically validate and analyze our coarsening method, we conducted a set of experiments using two distinct problem formulations: one based on the ILP formulation of the CVRP (hereafter referred to as ILP-based), and another grounded in the Binary Quadratic Model (BQM) approach (subsequently termed BQM-based). The last approach assumes that we transform the CVRP by encoding the objective function and constraints in the form of BQM (violations of constraints are penalized by adding penalty terms to the objective function) and use it to solve it with a quantum annealer (see Section III-D1).

A. THE APPROACH TO THE FLEET SIZE ISSUE

Various CVRP formulations exist, our approach outlined in [21], aims to minimise the total vehicle distance while using the smallest fleet, without setting a maximum limit on the number of vehicles used.

To define the CVRP for the ILP-based solver, the number of vehicles was fixed according to the fleet size known for the current best solution. This reduces the solution space, but helps to solve the problem more efficiently. Such an approach was also used in other works, e.g. in [22].

On the other hand, for the BQM-based solver, the fleet size for optimal solutions has been treated as a predetermined number of vehicles, which can be increased if necessary (see Section III-D2).

B. DATASET

We used CVRP benchmarks from the Christofides dataset [23], which is widely recognized and commonly used as a benchmark for various optimization problems. Our study focused on the instances CMT01-CMT05 (with all vertices randomly and uniformly distributed), CMT11, and CMT12 (which represent more ''practical'' cases [23]), where all vehicles have the same capacity. We omitted cases with time constraints to avoid the transition to the Capacitated Vehicle Routing Problem with Time Windows (CVRPTW).

Appendix A provides comprehensive details, including the input data characteristics and post-coarsening graph vertices.

C. ILP-BASED FORMULATION

In this subsection, we utilize the formulation presented in Section II-A, along with our algorithm presented in Section II-B, to assess its performance.

1) SOLVER

In our comparative analysis, we applied the ILP framework in conjunction with the Gurobi solver [24]. The Gurobi solver, established by the trio of **Gu**, **Ro**thberg, and **Bi**xby, stands out in the realm of optimization problems, particularly for Linear Programming (LP) and Mixed Integer Linear Programming (MILP). Gurobi takes the mathematical representation of a given problem to be solved and streamlines and enhances its representation in the so-called ''presolve'' stage. Gurobi then employs heuristic methods to pinpoint viable ILP solutions early on. The optimization process is propelled forward using the branch-and-bound technique, which systematically narrows down to the optimal solution. This solver is designed to harness the power of multi-core processors, significantly boosting its parallel computational capabilities. The superiority of Gurobi in solving MILP problems is attributed to its advanced branch-and-bound algorithm, which is particularly effective for problems involving a large number of variables and constraints [24]. This algorithm not only finds feasible solutions quickly but also ensures that the solutions are optimal by systematically eliminating suboptimal branches. Moreover, Gurobi's performance is further enhanced by its ability to perform parallel computations, which reduces the time required to reach an optimal solution [24]. The solver's presolve phase is crucial as it simplifies the problem, which can significantly reduce the solution space and, consequently, the computational effort [24]. These features collectively make Gurobi a robust and efficient tool for addressing complex MILP challenges.

2) RESULTS

This phase of the experiments aimed to evaluate the efficiency of the graph coarsening approach for CVRP on the Christofides dataset using the Gurobi solver. For this purpose, an attempt was made to compute the optimal solution for each instance among CMT01-05, 11, and 12, using the coarsening rate values of {0.3, 0.5, 0.7, 0.9, 1.0}, with a timeout of 6000 s. Three runs were performed for each parameter set. It is worth noting that although Gurobi is deterministic by design, the introduction of a time limit can lead to different behavior between runs, particularly if the machine load varies. Therefore, multiple runs were essential to assess the range of possible solutions.

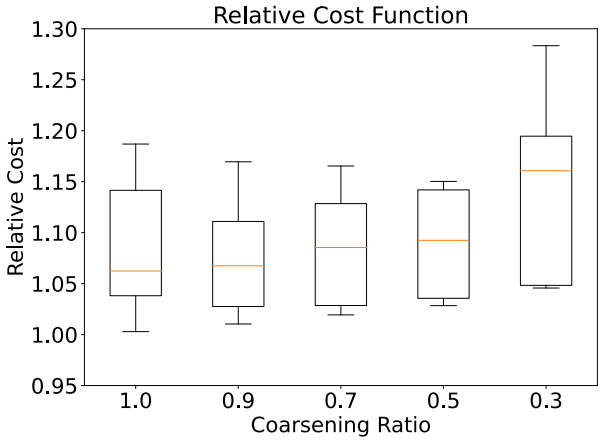


FIGURE 2. Summary of experiment results for the Gurobi solver on the Christofides dataset. Relative cost represents the normalized difference between the solution found using the graph coarsening technique and the best-known solution for a given run (averaged over all runs). It is crucial to observe that as we engage in coarsening, the outcomes diverge further from the known optimal results as expected. On the other hand, when employing quantum computing, we observe a different trend: the results actually enhance as the network is subjected to coarsening, as elaborated in Section IV of this study.

The results are shown in Figure 2. It can be observed that decreasing the coarsening rate leads to an increase in the relative cost, indicating a greater deviation from the known optimal solution. In almost all cases, the timeout limit was reached.

D. BQM-BASED FORMULATION

The second phase of our research involved conducting experiments to assess the impact of reducing the original graph dimensions through the graph coarsening algorithm outlined in Section II-B on the results obtained when solving CVRP using BQM-based solvers, based on the quantum and simulated annealing phenomena. By systematically reducing the graph dimensions, we aimed to evaluate the trade-off between graph size reduction and the quality of the solutions achieved.

Quantum annealing is a method designed to solve optimization problems by leveraging the principles of quantum mechanics. The essence of its operation resembles that of the known classical simulated annealing algorithm [25]. However, instead of using thermal fluctuations to transition from the initial state to a random end-state, which account for a properly defined objective function, quantum fluctuations are employed [26].

1) SOLVER

We used CVRP solvers specifically designed for use with quantum annealers which were implemented in the VRP-exploration project [27]. The project was established during the QOSF Quantum Computing Mentorship Program 2021 Cohort 4 and is currently being developed by a research group at QWorld [28]. Among the various solvers in this project designed to tackle different VRP variants, we focused on those based on the BQM that address the CVRP: Capacitated Full Qubo Solver (CFQS), Capacitated Average Partitioning Solver (CAPS), Capacitated Route Activation Solver (CRAS), Capacitated Clustered TSP Solver (CCTS), Capacitated Solution Partitioning Solver (CSPS).

These solvers use different formulations of the CVRP as Quadratic Unconstrained Binary Optimization (QUBO) problems. The QUBO approach belongs to the BQM class and is essential for formulating problems suitable for quantum annealers. By transforming the CVRP into a QUBO formulation, we can exploit the capabilities of the quantum annealer to search for optimal or near-optimal solutions to CVRP instances. An effective QUBO formulation is characterized by the minimal use of slack variables to avoid overwhelming solvers with complexity. In addition, the limited parameter resolution of quantum computing devices, such as quantum annealers, must be considered to prevent integrated control errors and ensure optimal solutions. Opting for constraint enforcement methods that generate fewer sub-constraints can lead to a reduction in slack variables, thereby streamlining the solution process and enhancing the efficiency.

The solvers are compatible with a number of backends, both quantum and classical. For our experiments, due to the size of the examined instances of the CVRP problem in Christofides' dataset, we were particularly interested in the backend based on the Leap Hybrid Sampler [29] (we refer to it as `leap`) and the Simulated Annealing Sampler (hereinafter referred to as `neal`).

Based on the information contained in [27] and the results of preliminary experiments (detailed in Appendix B), CSPS was selected for the main experiments.

The standard Solution Partitioning Solver (SPS) operates as follows. Initially, a single vehicle is assumed and the TSP problem is addressed (e.g., using a quantum solver, such as `leap`, or a classical solver, like `neal`). Subsequently, the solution is partitioned based on the number of vehicles in the original problem formulation using a dynamic programming approach (this part is performed classically). Therefore, the VRP can be solved entirely using classical or hybrid (quantum-classical) solvers. The Capacitated Solution Partitioning Solver (CSPS), tailored to the CVRP, appends capacity constraints to the existing ones (ensuring single delivery per client and that each vehicle is at a distinct location at a given time). This methodology was introduced in [30].

2) RESULTS

We conducted the main part of the experiments using the CSPS for both the `neal` and `leap` backends. For both backends, given their non-deterministic operations, calculations were performed three times for each problem instance and every examined coarsening rate. We used the default

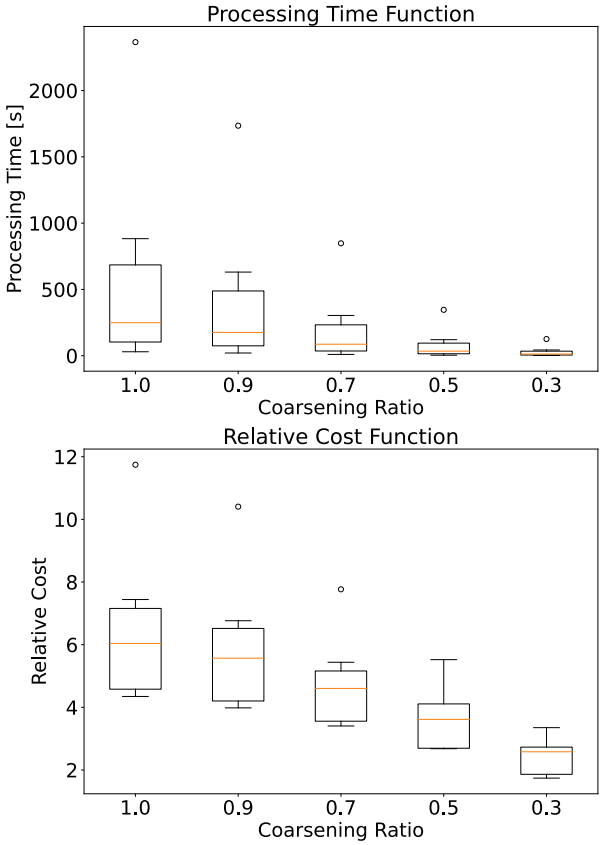


FIGURE 3. Summary of experiment results for CSPS solver and `neal` backend for the Christofides dataset. It's worth noting that the relative costs tend to approach the known optimum as the original graph is further coarsened, which is actually counterintuitive.

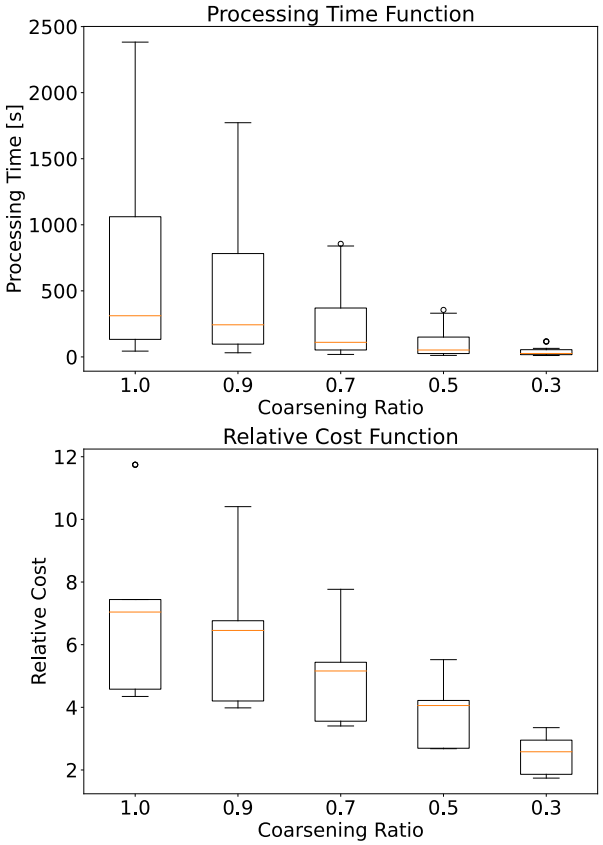


FIGURE 4. Summary of experiment results for CSPS solver and `leap` backend for the Christofides dataset.

settings for the `LeapHybridSampler` solver, particularly the minimal time limit, which was automatically calculated based on the size of the problem (see Appendix C). For visual representation in Figures 3 and 4, we depict the average results from these multiple runs.

The CSPS is implemented such that if no solution is found for the fleet size assumed in the original definition of a given task, the fleet is expanded. Knowing the optimal solutions to the CVRP formulated as described in Section III-B, we know that adding new vehicles does not improve the solution for the specific instances we considered. Thus, allowing the CSPS to employ more than the minimum number of vehicles specified in the problem definition does not create new possibilities for obtaining a better solution.

In the experiments, we first coarsened the graph with different ratios, then applied the solver to find a good solution, and finally reconstructed the solution for the original graph using the inflation method. As demonstrated in Figures 3 and 4, the more we coarsen the graph, the closer the solution found is to the optimal solution reported in the Christofides dataset. Coarsening significantly improves the performance of the tested algorithms, even when the number of vehicles exceeds the original fleet size, since it only increases the potential optimal cost without undermining the effectiveness of the approach.

3) SOLVER OUTCOMES FOR DIFFERENT PROBLEM INSTANCES

The performance of the CSPS was evaluated for different instances of the Christofides dataset. When analyzing the results obtained by the CSPS, it is important to consider not only parameters such as the graph size and coarsening rate, but also the structure of the graph.

The thumbnails of the graphs used in the experiments, which provide an overview of the spatial distribution of the vertices, can be found in Appendix A. Particularly noticeable are instances CMT11 and CMT12, where the vertices form distinct groups — ten groups in the case of CMT12 (which corresponds exactly to the number of vehicles specified in the definition of this CVRP task), and six groups in the case of CMT11.

Figure 5 presents the mean relative cost achieved by the CSPS for the `leap` and `neal` backends for individual instances of the CVRP from the analyzed subset.

4) SOLVING TIME VARIATIONS

When analyzing the computation time for both backends, it is important not to directly compare the times achieved for the

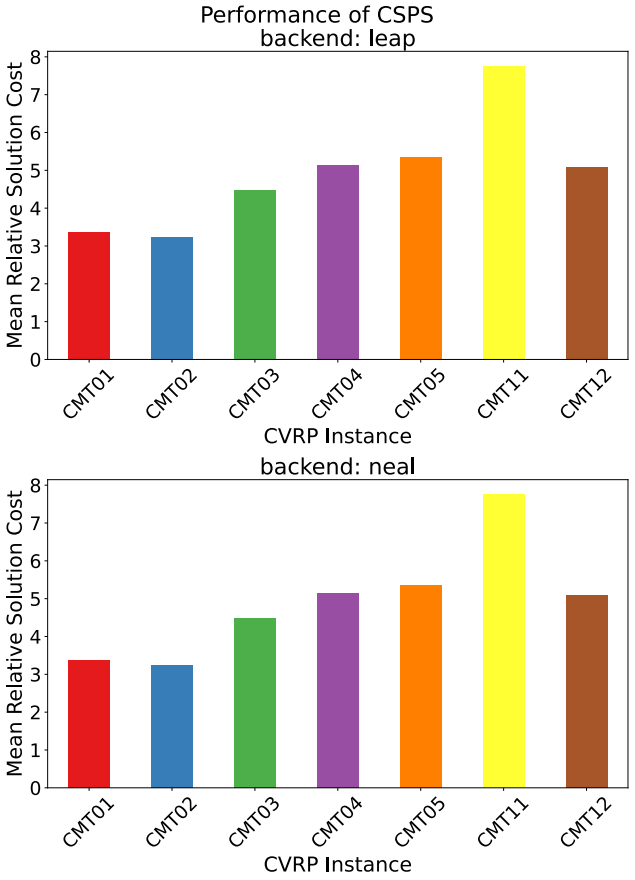


FIGURE 5. Relative cost averaged over all runs of experiments for all values for the coarsening rate obtained by CSPS for `leap` and `neal` backends. For each problem instance calculations were performed three times for all examined coarsening rate and all these results were averaged.

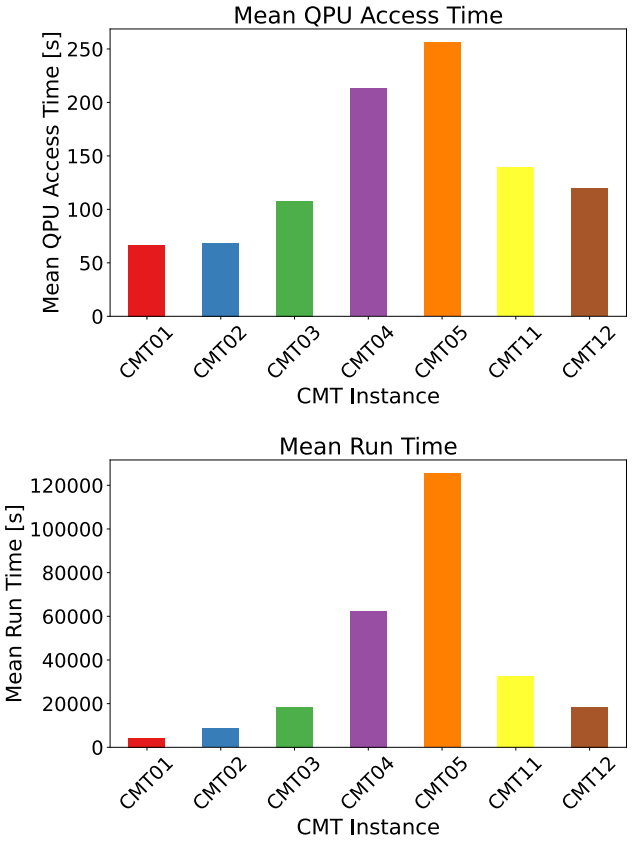


FIGURE 6. Mean QPU access time and run time (over multiple runs of experiments) for CSPS using `leap` backend for each instance of Christofides subset.

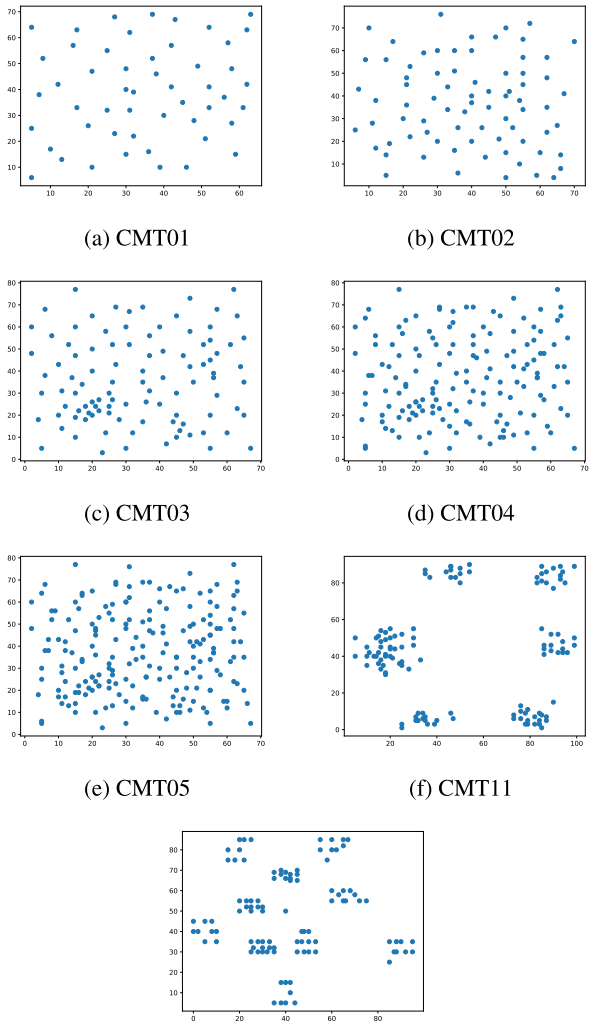
`leap` and `neal` backends. This is because of the presence of the time limit parameter in the first case. Each time it was set to the minimum, which is the time calculated by the Ocean SDK [31] framework for a given problem instance, taking into account its size. Additionally, for the `leap` backend, there is an added time overhead for network communication, in this case, between the server in the Google Colab Pro infrastructure and the Leap Cloud Service.

Figure 6 presents a precise comparison of the time taken for computations for individual CVRP instances in the Leap Cloud Service infrastructure, considering both the entirety of hybrid computations and the times that the Leap Hybrid Sampler spent solely on the QPU within the framework of hybrid computations. We can see that the most time (in terms of calculations on the QPU itself and overall, i.e., CPU plus QPU) was spent finding solutions for CMT04 and CMT05 instances. This outcome is not unexpected — notably, these two represent the largest graphs among those evaluated in this investigation.

In the case of the `neal` backend, we do not have a time limit imposed in advance conditioning the need to stop the calculations. For both backends (`leap` and `neal`), a straightforward correlation can be observed between the computation time required by the simulated annealing algorithm and the magnitude of the problem / coarsening ratio (cf. Figures 3 and 4).

IV. DISCUSSION OF RESULTS

Analyzing the solution quality trends, we considered the relative cost — the cost of the solution obtained, which is the sum of the distances covered by individual vehicles, calculated in relation to the optimal solution in the Christofides dataset (see Appendix A) — and processing time for both backends. The results are shown in Figures 2 to 6. In terms of the relative cost, both solvers considered for the BQM formulation performed comparably, although the `leap` backend had a slight advantage. Crucially, from the perspective of this work, we observe that as the coarsening rate decreased, the results obtained by the CSPS solver (for both backends) improved. This trend is in contrast to that observed for the ILP-based formulation powered by Gurobi, as discussed in Section III-c. A deeper analysis reveals that the performance of the coarsening method itself remains consistent, as it significantly reduces the computational time across both problem formulations. However, the results differed among



(a) CMT01 (b) CMT02 (c) CMT03 (d) CMT04 (e) CMT05 (f) CMT11 (g) CMT12

FIGURE 7. A subset of instances of the Christofides [23] dataset used for the experiments. For the sake of clarity, the edges have been omitted when drawing the graphs.

the formulations in terms of the deviation of the cost function. The ILP formulation, when combined with the Gurobi solver, already achieves results that are very close to the known optimum when using the entire network. Consequently, there is limited room for further improvement in this context, and it is expected to deviate from the optimum when adding an approximation method. In contrast, the BQM formulation benefits more substantially from the coarsening method. The results clearly show an improvement in the value deviation of the cost function when using the BQM formulation. This indicates that the coarsening algorithm effectively enhances the performance of the BQM formulation, making it a promising candidate for use in combination with other algorithms that target quantum devices. This improvement underscores the potential of the coarsening method to facilitate more efficient and effective solutions in the realm of quantum computing.

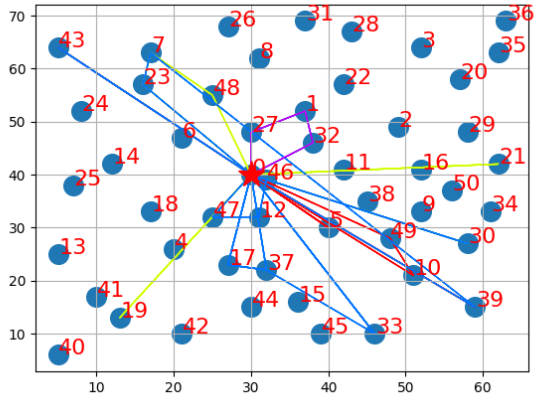
TABLE 1. Detailed information about instances from the Christofides [23] dataset.

Graph	Optimal solution		Coarsening rate	No. vertices
	Cost	No. of trucks		
CMT01	524.611	5	1.0	51
			0.9	45
			0.7	35
			0.5	25
			0.3	15
CMT02	835.262	10	1.0	76
			0.9	68
			0.7	53
			0.5	38
			0.3	22
CMT03	826.137	8	1.0	101
			0.9	90
			0.7	70
			0.5	50
			0.3	30
CMT04	1028.42	12	1.0	151
			0.9	135
			0.7	105
			0.5	75
			0.3	45
CMT05	1291.29	16	1.0	200
			0.9	180
			0.7	140
			0.5	100
			0.3	60
CMT11	1042.12	7	1.0	121
			0.9	108
			0.7	84
			0.5	60
			0.3	36
CMT12	819.56	10	1.0	100
			0.9	90
			0.7	70
			0.5	50
			0.3	30

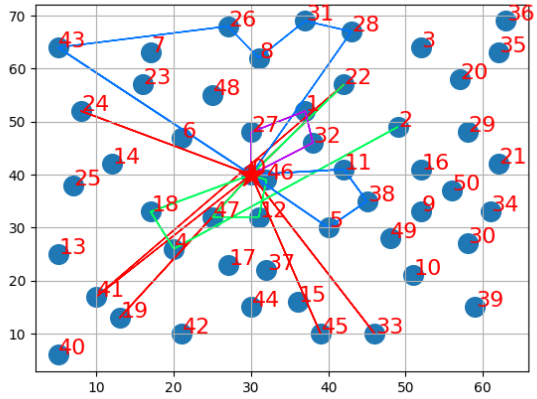
V. CONCLUSION

In this study, we successfully implemented the graph coarsening algorithm and utilized it in conjunction with both ILP-based and BQM-based formulations to tackle the CVRP. As anticipated, the ILP-based formulation powered by Gurobi Sovler, which operates directly on a model using the ILP's characteristic objective function and constraints, exhibited superior performance when applied to the fully connected network compared to graphs where the coarsening rate was lower than 1.0. This formulation achieved results very close to the optimal when the entire graph was utilized. Providing a coarsened version of the graph served as an approximation that had a minimal impact on the cost function value but significantly reduced the computation time.

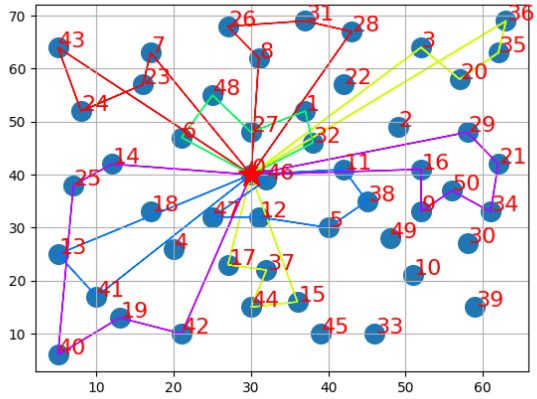
In conclusion, our study demonstrates that employing BQM-based formulations for coarsened graphs yields better results compared to the original network, unlike ILP-based formulations. This improvement is attributed to the reduced complexity of the BQM model when applied to smaller, coarsened graphs, which alleviates the computational burden of the solver. The findings underscore the potential benefits of integrating graph coarsening techniques with BQM-based formulations to enhance the efficiency of solv-



(a) coarsening rate = 0.7

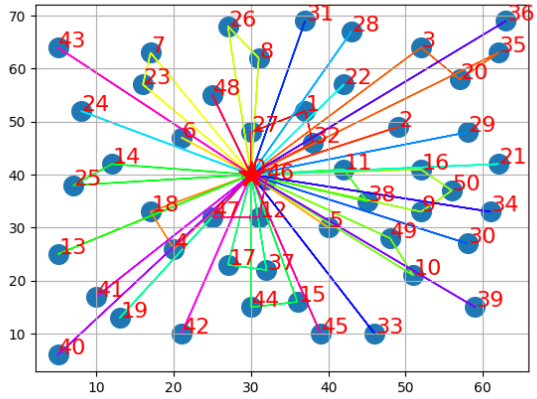


(b) coarsening rate = 0.5

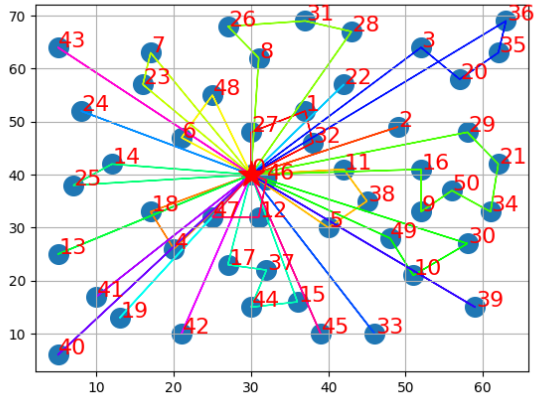


(c) coarsening rate = 0.3

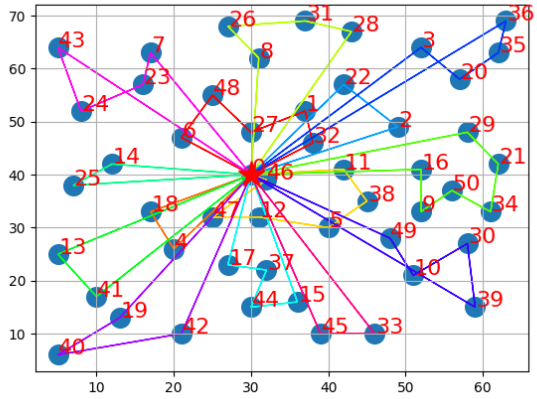
FIGURE 8. Exemplary results of CFQS (`leap` backend) for CMT01 instance and different coarsening rate values.



(a) coarsening rate = 0.7



(b) coarsening rate = 0.5



(c) coarsening rate = 0.3

FIGURE 9. Exemplary results of CSPS (`leap` backend) for CMT01 instance and different coarsening rate values.

ing complex optimization problems, particularly with hybrid and quantum solvers. Additionally, a comparative analysis between ILP-based solvers and QUBO formulations using annealing algorithms reveals that the problem formulation significantly influences performance, as evidenced by the

TABLE 2. Minimum time limit values (in seconds) for the `leap` backend for individual CVRP instances from the Christofides set.

Graph	Coarsening rate	Min. time limit for `leap`
CMT01	1.0	6.36
	0.9	5.08
	0.7	3.30
	0.5	3.00
	0.3	3.00
CMT02	1.0	17.77
	0.9	12.00
	0.7	6.83
	0.5	3.79
	0.3	3.00
CMT03	1.0	40.00
	0.9	29.44
	0.7	13.38
	0.5	6.14
	0.3	3.00
CMT04	1.0	140.00
	0.9	103.65
	0.7	46.53
	0.5	17.01
	0.3	5.08
CMT05	1.0	254.86
	0.9	211.66
	0.7	114.57
	0.5	38.99
	0.3	8.60
CMT11	1.0	75.2
	0.9	51.59
	0.7	24.19
	0.5	8.60
	0.3	3.46
CMT12	1.0	40.00
	0.9	29.44
	0.7	13.38
	0.5	6.14
	0.3	3.00

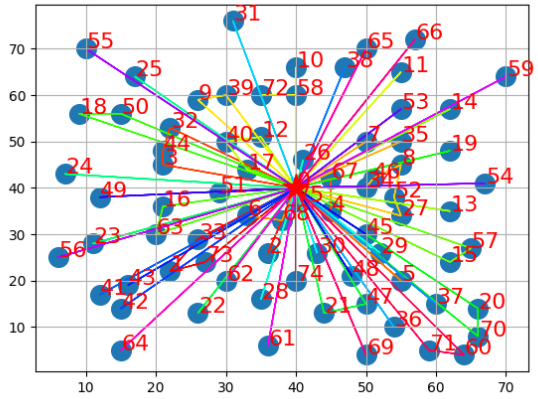
similar outcomes of the classically computed `neal` backend and the quantum-based `leap`.

Future research will focus on exploring advanced optimization techniques that leverage the strengths of both classical and quantum algorithms to develop more efficient and effective solvers for complex combinatorial optimization problems such as the VRP and its variants. Additionally, we aim to investigate whether the advantage of Gurobi and ILP formulations is due to the inherent superiority of ILP over BQM formulations, or if it is a result of the limitations of the `neal` and `leap` solvers. Specifically, we plan to test other classical optimization algorithms applied to the BQM formulation and explore alternative BQM (QUBO) formulations for the CVRP.

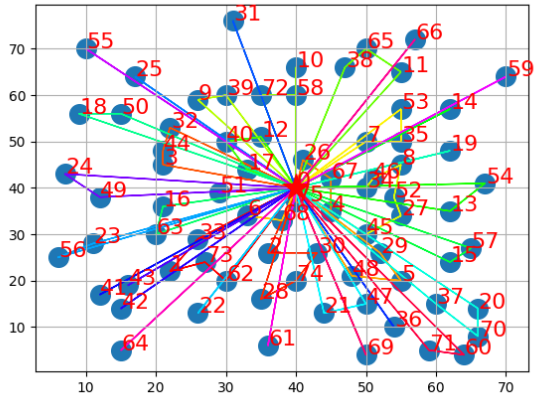
.

APPENDIX A CVRP INSTANCES FROM CHRISTOFIDES

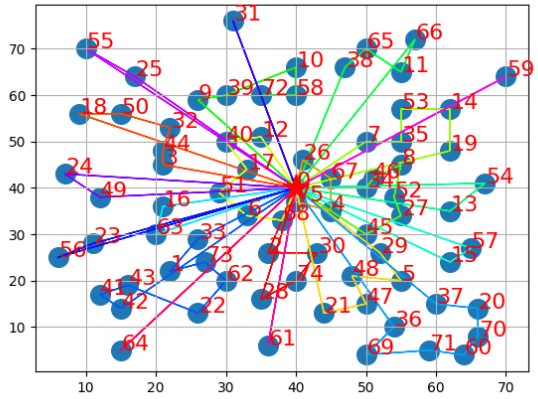
Figure 7 presents the thumbnails of graphs derived from the Christofides dataset employed in our experiments, providing insights into the spatial distribution of vertices.



(a) coarsening rate = 0.7



(b) coarsening rate = 0.5

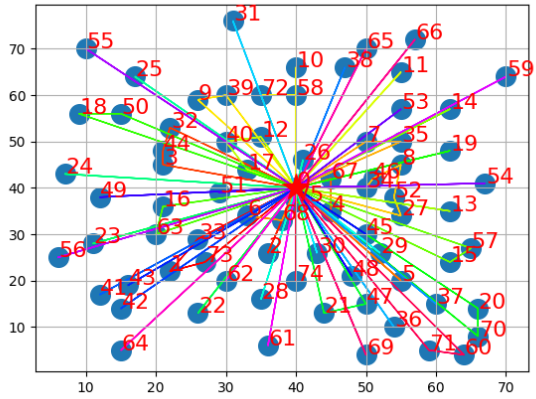


(c) coarsening rate = 0.3

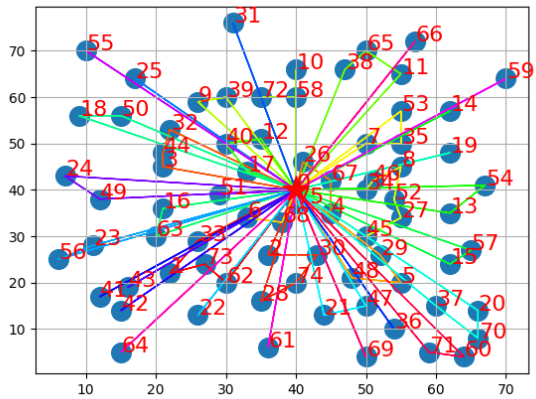
FIGURE 10. Exemplary results of CFQS (`neal` backend) for CMT02 instance and different coarsening rate values.

APPENDIX B PRELIMINARY EXPERIMENTS FOR BQM-BASED SOLVERS

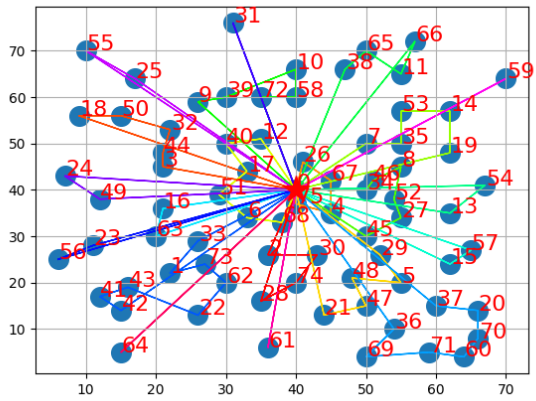
Based on the information in the "Vehicle Routing Problem" notebook [27], specifically the "Capacitated Solvers"



(a) coarsening rate = 0.7

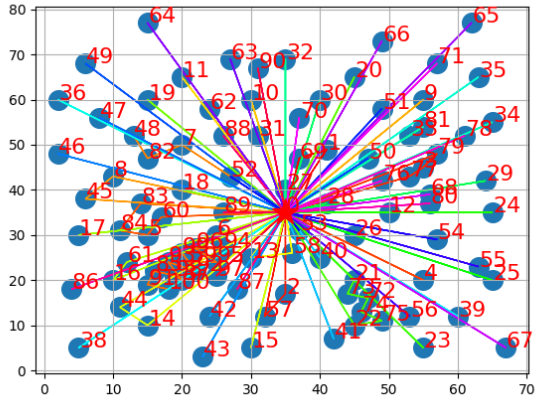


(b) coarsening rate = 0.5

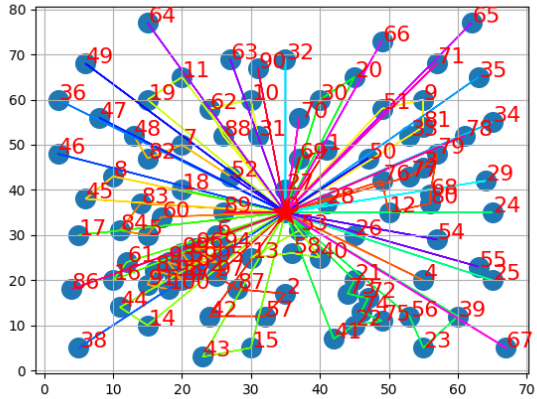


(c) coarsening rate = 0.3

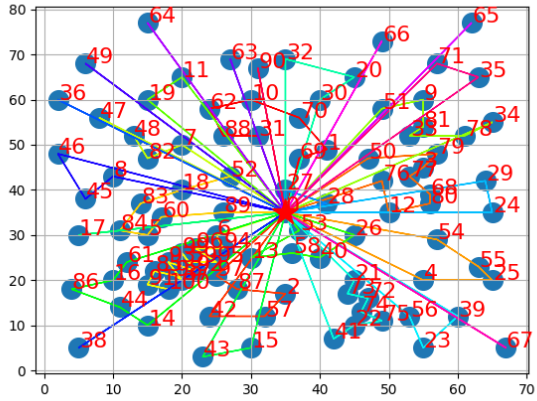
FIGURE 11. Exemplary results of CFQS solver (`leap` backend) for CMT02 instance and different coarsening rate values.



(a) coarsening rate = 0.7



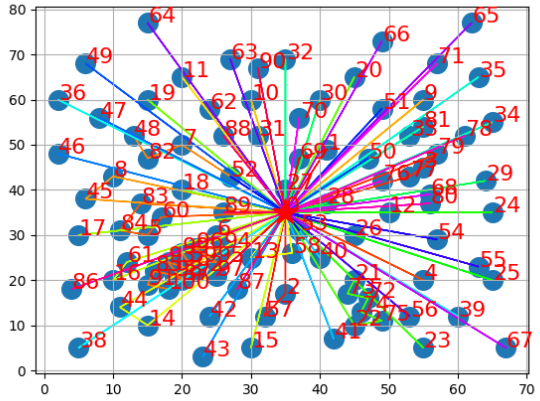
(b) coarsening rate = 0.5



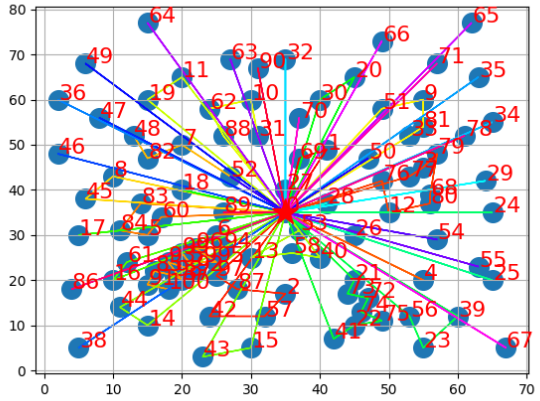
(c) coarsening rate = 0.3

FIGURE 12. Exemplary results of CFQS solver (`neal` backend) for CMT03 instance and different coarsening rate values.

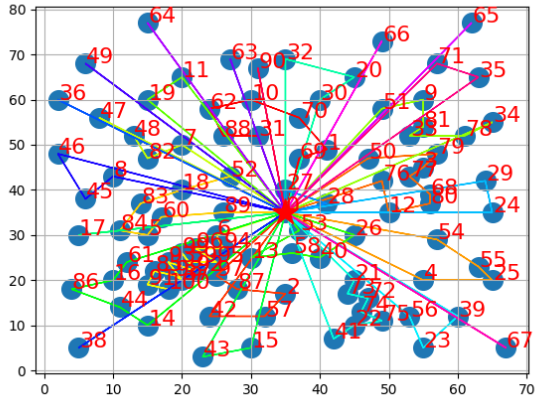
section, we expected that only the CSPS would have a real chance to handle such large instances of the CVRP found in the Christofides set. Preliminary experiments conducted for the CFQS, CAPS, CRAS, CCTS, and CSPS solvers confirmed these assumptions. Three of the solvers, CAPS, CRAS, and CCTS, in the initial phase of calculations—



(a) coarsening rate = 0.7



(b) coarsening rate = 0.5



(c) coarsening rate = 0.3

FIGURE 13. Exemplary results of CFQS solver (`leap` backend) for CMT03 instance and different coarsening rate values.

specifically, when constructing the BQM model based on the input graph—exceeded the available RAM in the computing environment. Consequently, these solvers could

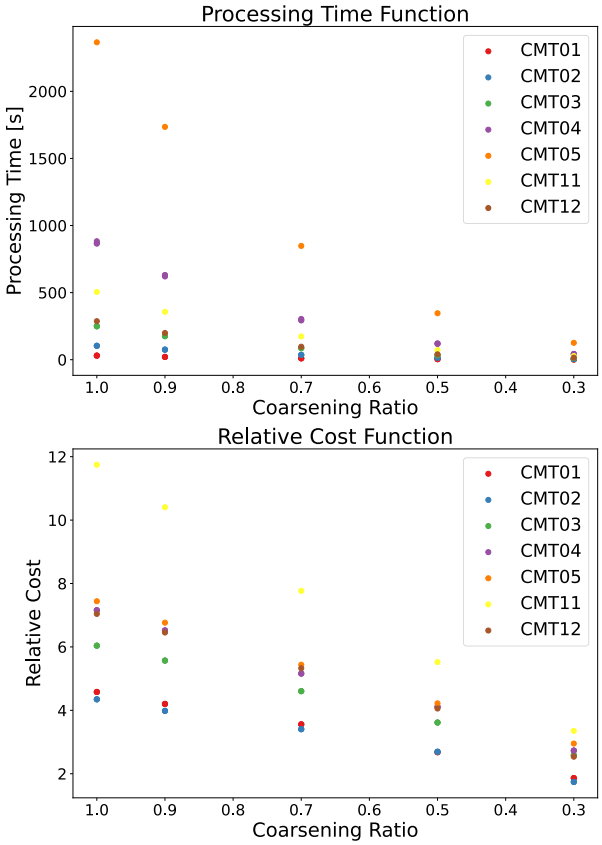


FIGURE 14. Summary of experiment results (detailing individual instances) for CSPS solver and `neal` backend.

not find a solution to the CVRP even for the smallest input graph, CMT01. Ultimately, the results were achieved only with the CFQS and CSPS solvers. Figures 8 and 9 depict a comparison of their effectiveness for the CMT01 instance using the `leap` backend and various coarsening rate values.

The Figures 8 and 9 indicate that the solutions proposed by the CFQS solver were infeasible; they failed to satisfy all the conditions of the CVRP task — not every "city" was visited. This is particularly evident for larger graphs with higher coarsening rates.

APPENDIX C MINIMUM TIME LIMIT VALUES FOR `LEAP` BACKEND

See Table 2.

APPENDIX D BQM-BASED SOLVER RESULTS FOR SELECTED CVRP INSTANCES FROM THE CHRISTOFIDES DATASET

Figures 10 to 13 present sample results obtained for the CMT02 and CMT03 instances using CFQS. The routes of individual vehicles are marked with different colors. All solutions obtained for individual instances of CMT01-05, 11, 12 for various coarsening rates and backends can be found in [32].

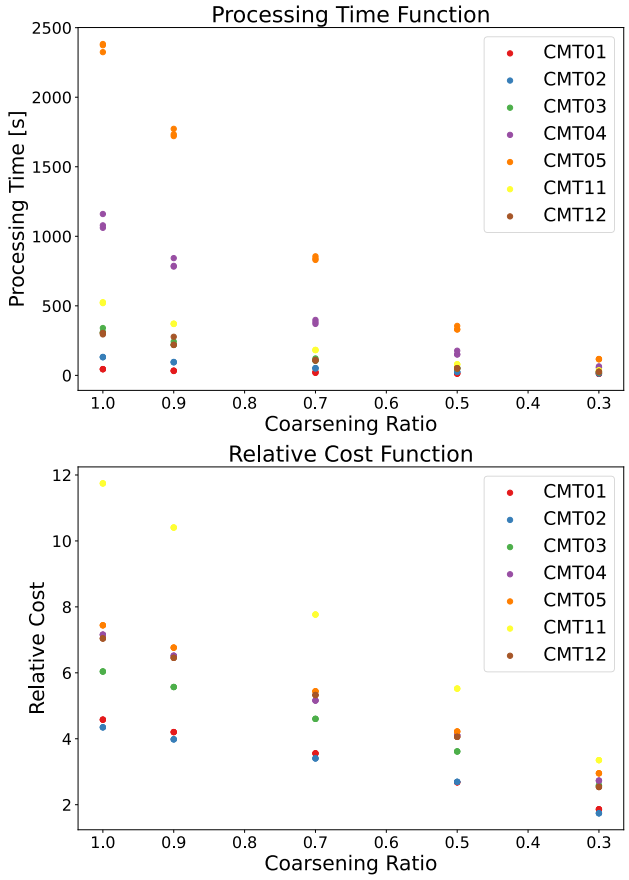


FIGURE 15. Summary of experiment results (detailing individual instances) for CSPS solver and `leap` backend.

APPENDIX E COMPARISON OF CSPS PERFORMANCE ON CHRISTOFIDES DATASET USING `NEAL` AND `LEAP` BACKENDS

Figures 14 and 15 further highlight how the CSPS performed for each instance from the Christofides dataset, both for the `neal` and `leap` backends. It is clear that in both scenarios the CMT11 instance mentioned above proved to be the most challenging. Although smaller than, for example, CMT04 or CMT05, it is specific due to the existence of clear clusters of vertices in the graph. Perhaps the strategy adopted by the CSPS (solving the TSP and dividing the final route into individual vehicles) does not work efficiently for this type of graph. However, this effect was not visible in the case of CMT12, which is another instance with vertex clusters present.

Additionally, for the `leap` backend, there is an added time overhead for network communication, in this case between the server in the Google Colab Pro infrastructure and the Leap Cloud Service. Therefore, for the `leap` backend, the computation times displayed in Figure 15 only provide an approximate idea of the differences in computation times for individual problem instances, which are directly dependent on their size.

AUTHOR CONTRIBUTIONS STATEMENT

All authors contributed to the conception, design, and implementation of the study. The experiments were performed by Katarzyna Nałęcz-Charkiewicz and Carlos C. N. Kuhn. The first draft of the manuscript was written by Joshua Keene and Carlos C. N. Kuhn. The second, revised version was prepared by Katarzyna Nałęcz-Charkiewicz, Carlos C. N. Kuhn, and Paweł Gora and all authors commented on the previous versions of the manuscript. All authors read and approved the final manuscript.

COMPETING INTERESTS

The authors declare no competing interests.

REFERENCES

[1] G. B. Dantzig and J. H. Ramser, "The truck dispatching problem," *Manage. Sci.*, vol. 6, no. 1, pp. 80–91, Oct. 1959.
[2] J. K. Lenstra and A. H. G. R. Kan, "Complexity of vehicle routing and scheduling problems," *Networks*, vol. 11, no. 2, pp. 221–227, Jun. 1981.
[3] L. K. Grover, "A fast quantum mechanical algorithm for database search," in *Proc. 28th Annu. ACM Symp. Theory Comput.-STOC*, 1996, pp. 212–219.
[4] D. J. Moylett, N. Linden, and A. Montanaro, "Quantum speedup of the traveling-salesman problem for bounded-degree graphs," *Phys. Rev. A, Gen. Phys.*, vol. 95, no. 3, Mar. 2017, Art. no. 032323.
[5] C. V. Raj and M. S. Shivakumar, "Applying quantum algorithm to speed up the solution of Hamiltonian cycle problems," in *Proc. Int. Conf. Intell. Inf. Process.*, Jan. 2006, pp. 53–61.
[6] C. D. B. Bentley, S. Marsh, A. R. R. Carvalho, P. Kilby, and M. J. Biercuk, "Quantum computing for transport optimization," 2022, *arXiv:2206.07313*.
[7] J. Preskill, "Quantum computing in the NISQ era and beyond," *Quantum*, vol. 2, p. 79, Aug. 2018.
[8] J. F. Benders, "Partitioning procedures for solving mixed-variables programming problems," *Numerische Math.*, vol. 4, no. 1, pp. 238–252, Dec. 1962.
[9] G. B. Dantzig and P. Wolfe, "Decomposition principle for linear programs," *Oper. Res.*, vol. 8, no. 1, pp. 101–111, Feb. 1960.
[10] N. Franco, T. Wollschläger, N. Gao, J. M. Lorenz, and S. Günnemann, "Quantum robustness verification: A hybrid quantum-classical neural network certification algorithm," in *Proc. IEEE Int. Conf. Quantum Comput. Eng. (QCE)*, Sep. 2022, pp. 142–153.
[11] J. Ossorio-Castillo and F. Pena-Brage, "Optimization of a refinery scheduling process with column generation and a quantum annealer," *Optim. Eng.*, vol. 23, no. 3, pp. 1471–1488, Sep. 2022.
[12] N. Franco, T. Wollschläger, B. Poggel, S. Günnemann, and J. M. Lorenz, "Efficient MILP decomposition in quantum computing for ReLU network robustness," 2023, *arXiv:2305.00472*.
[13] C. Gambella and A. Simonetto, "Multiblock ADMM heuristics for mixed-binary optimization on classical and quantum computers," *IEEE Trans. Quantum Eng.*, vol. 1, pp. 1–22, 2020.
[14] M. Ponce, R. Herrman, P. C. Lotshaw, S. Powers, G. Siopsis, T. Humble, and J. Ostrowski, "Graph decomposition techniques for solving combinatorial optimization problems with variational quantum algorithms," 2023, *arXiv:2306.00494*.
[15] M. S. Gilbert, S. Acer, E. G. Boman, K. Madduri, and S. Rajamanickam, "Performance-portable graph coarsening for efficient multilevel graph analysis," in *Proc. IEEE Int. Parallel Distrib. Process. Symp. (IPDPS)*, May 2021, pp. 213–222.
[16] J. Ma and V. M. Zavala, "Solution of large-scale supply chain models using graph sampling & coarsening," *Comput. Chem. Eng.*, vol. 163, Jul. 2022, Art. no. 107832.
[17] R. Shaydulin, H. Ushijima-Mwesigwa, C. F. A. Negre, I. Safro, S. M. Mniszewski, and Y. Alexeev, "A hybrid approach for solving optimization problems on small quantum computers," *Computer*, vol. 52, no. 6, pp. 18–26, Jun. 2019.
[18] R. Shaydulin, "Quantum and classical multilevel algorithms for (Hyper) graphs," Ph.D. thesis, Dept. Computer Science, Clemson Univ., Clemson, SC, USA, 2020.

[19] C. E. Miller, A. W. Tucker, and R. A. Zemlin, "Integer programming formulation of traveling salesman problems," *J. ACM*, vol. 7, no. 4, pp. 326–329, Oct. 1960.
[20] *Vehicle Routing—Qiskit Optimization 0.4.0 Documentation*, Version 0.4.0, 2022. [Online]. Available: https://qiskit-community.github.io/qiskit-optimization/tutorials/07_examples_vehicle_routing.html
[21] N. Christofides and S. Eilon, "An algorithm for the vehicle-dispatching problem," *J. Oper. Res. Soc.*, vol. 20, pp. 309–318, Sep. 1969.
[22] A. Subramanian, E. Uchoa, and L. S. Ochi, "A hybrid algorithm for a class of vehicle routing problems," *Comput. Oper. Res.*, vol. 40, no. 10, pp. 2519–2531, Oct. 2013.
[23] N. Christofides, *Combinatorial Optimization*. Hoboken, NJ, USA: Wiley, 1979.
[24] *Gurobi Optimizer Reference Manual*, Gurobi Optim., LLC, Beaverton, OR, USA, 2023.
[25] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, no. 4598, pp. 671–680, May 1983.
[26] T. Kadowaki and H. Nishimori, "Quantum annealing in the transverse Ising model," *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 58, no. 5, pp. 5355–5363, Nov. 1998.
[27] A. K. Mandoi, A. Bhatta, and V. G. Gueorguiev. (2023). *VRP-Explorations*. [Online]. Available: https://github.com/AsishMandoi/VRP-explorations
[28] (2023). *QWorld*. [Online]. Available: https://qworld.net/research-projects-projectRP-009
[29] *D-Wave Systems*, D-Wave Syst. Inc., Palo Alto, CA, USA, 2023.
[30] M. Borowski, P. Góra, K. Karnas, M. Błajda, K. Król, A. Matyjasek, D. Burczyk, M. Szewczyk, and M. Kutwin, "New hybrid quantum annealing algorithms for solving vehicle routing problem," in *Proc. Int. Conf. Comput. Sci.* Cham, Switzerland: Springer, Jan. 2020, pp. 546–561.
[31] D-Wave Syst. (2023). *D-Wave Ocean*. [Online]. Available: https://docs.ocean.dwavesys.com/en/stable/
[32] QWorld, "Graph coarsening approach to the vehicle routing problem: Source code," Zenodo, 2024. [Online]. Available: https://doi.org/10.5281/zenodo.10074989

KATARZYNA NAŁĘCZ-CHARKIEWICZ received the M.Sc. degree in computer science from Warsaw University of Technology, Warsaw, Poland, in 2014, where she is currently pursuing the Ph.D. degree in computer science. Since 2023, she has been an Research Assistant with the Artificial Intelligence Division, Institute of Computer Science, Faculty of Electronics and Information Technology, Warsaw University of Technology. Her primary research interests include bioinformatics, quantum computing, and explainable AI.

ARNAV DAS received the M.Sc. degree in computer science from St. Xavier's University, Kolkata. He is currently a Research Associate with India Internet Foundation. His primary research interests include machine learning, network security, network protocols, quantum circuit cutting, and quantum circuit simulation.

TURBASU CHATTERJEE received the bachelor's degree in computer science and engineering from the Maulana Abul Kalam Azad University of Technology, West Bengal, India. He is currently a graduate student at Virginia Tech, Blacksburg, USA. His primary research interests include quantum learning, quantum algorithms, and quantum logic synthesis.

JOSHUA KEENE received the B.E. degree (Hons.) in mechanical and aerospace engineering from the University of Queensland, in 2020. He is currently pursuing the Ph.D. degree in control theory with the University of Melbourne. His current research aims to explore and develop distributed control algorithms for collective motion coordination applications. More broadly, he is interested in developing effective control techniques within the general multi-agent setting and in the fields of optimization, control theory, and robotics and autonomous systems.

PAWEŁ GORA received the Ph.D. degree in computer science from the University of Warsaw, Poland, in 2024. He is currently the CEO of Fundacja Quantum AI. His primary research interests include machine learning, combinatorial optimization, complex processes, intelligent transportation systems, and quantum computing.

CARLOS C. N. KUHN received the bachelor's degree in physics from the Universidade Federal do Rio Grande do Sul (UFRGS), Brazil, in 2007, and the master's and Ph.D. degrees in theoretical quantum physics from UFRGS. During the Ph.D. studies, he received the prestigious scholarship to study in Australia as an Occupational Trainee. From March 2013 to September 2017, he transitioned from theoretical to experimental physics with ultracold atoms, as a Postdoctoral Researcher with the Atom Laser and Quantum Sensor Group, Australian National University (ANU). From September 2017 to February 2020, he was with the Swinburne University of Technology (SUT), as a Postdoctoral Research Fellow, focusing on ultracold atoms, topological phases, and superfluidity within the ARC Centre of Excellence in Future Low Energy Electronics Technologies (FLEET). In February 2020, he moved to the Department of Defence, Science and Technology, as a Research Scientist, applying his mathematical skills to solve real-world problems. In January 2024, he returned to academia as the Research Chair in open source technologies and an Associate Professor with the University of Canberra. He is also a board member, helping to shape the ACT Quantum Hub, guiding strategic decisions to foster collaboration and innovation. His academic research spans explainable AI and quantum machine learning.