VRP - QInnovision 2025

Iago Leal de Freitas Anurag Ramesh David E. Bernal Neira

Davidson School of Chemical Engineering, Purdue University
November 30, 2024

1 Introduction

The Vehicle Routing Problem (VRP) is a cornerstone of logistics and supply chain optimization, addressing the question of how to design optimal routes for a fleet of vehicles delivering goods to a set of customers. Its complexity increases significantly when considering constraints such as vehicle capacities, time windows, and other operational limitations. In the maritime sector, this problem arises in the transportation of liquefied natural gas (LNG) from supply ports to demand nodes. This challenge is critical for ensuring efficient and cost-effective operations in the global energy industry. Efficient solutions to this problem have far-reaching implications, including reduced operational costs, improved delivery reliability, and significant environmental benefits.

Traditionally, the VRP has been tackled using Mixed-Integer Programming (MIP) formulations, often coupled with tailored decomposition techniques to handle its combinatorial complexity. While these methods have been highly influential in solving many VRP instances, they face significant limitations in scaling to larger problem sizes due to the computational intensity required. Moreover, MIP solvers have not been able to fully exploit the capabilities of modern GPU hardware, such as that provided by NVIDIA, which offers immense parallel computing power.

In previous work, we explored the mapping of VRP instances to Quadratic Unconstrained Binary Optimization (QUBO) problems [2]. This reformulation enables the direct application of quantum and quantum-inspired discrete optimization approaches, opening the door to novel methodologies that leverage the computational advantages of hybrid quantum-classical algorithms. By representing VRP constraints and objectives as QUBO problems, we can integrate emerging quantum algorithms such as the Quantum Approximate Optimization Algorithm (QAOA) and Variational Quantum Eigensolver (VQE) to solve VRP instances more efficiently.

Building on this foundation, we present a novel approach that combines quantum-inspired methods with classical acceleration techniques. NVIDIA's CUDA-Q framework facilitates the implementation of QAOA on GPUs, offering a hybrid approach to solving QUBO problems. However, unlike existing QAOA-based solutions, we propose a custom-built package that solves QUBO problems using tensor network mappings and contractions. This method diverges from traditional quantum algorithms by leveraging tensor networks to encode QUBOs compactly and efficiently perform operations on GPUs. By exploiting the inherent parallelism of tensor contractions, our approach fully utilizes the computational power of modern GPU architectures, achieving significant efficiency improvements compared to standard methods.

The remainder of this document is structured as follows: Section 2 discusses the application of VRP to the maritime routing of LNG, highlighting its economic and environmental importance. Section 3 details the transformation of VRP formulations into QUBO problems, while Section 4 describes their solution using NVIDIA's CUDA-Q framework and QAOA. Section 5 introduces our tensor network-based solver for QUBO instances, followed by a comparative analysis of results in Section 6. Finally, Section 7 concludes with insights and future directions.

2 Application

The maritime routing problem involves determining the optimal set of routes for a fleet of vessels to deliver liquefied natural gas (LNG) from supply ports to demand nodes globally. This problem is critical for ensuring efficient and cost-effective transportation in the energy sector. In particular, this variant of the Vehicle Routing Problem (VRP) considers operational constraints such as fleet capacity, alternating visits between supply and demand nodes, and adherence to strict time windows.

The maritime routing problem can be formulated as a specific instance of the Capacitated Vehicle Routing Problem with Time Windows (CVRPTW). The problem is modeled as a Quadratic Unconstrained Binary Optimization (QUBO) problem, which is particularly well-suited for quantum computing algorithms [2].

Our approach investigates three mathematical formulations, each offering unique advantages for modeling the routing problem:

1. Route-Based Formulation: Defines decisions at the route level, focusing on whether a specific route is traveled. This formulation is compact but relies on pre-generated feasible routes.

2. Arc-Based Formulation: Models decisions at the arc level, representing whether a vehicle travels between specific nodes at particular times. It provides flexibility in modeling timing but requires more decision variables.

3. Sequence-Based Formulation with Continuous Time: Captures routes as sequences of visited nodes and includes continuous variables for arrival times, enabling explicit modeling of time windows and minimizing route durations.

These formulations are designed to be compatible with quantum algorithms such as the Quantum Approximate Optimization Algorithm (QAOA) and Variational Quantum Eigensolver (VQE), allowing direct mapping to the Max-Cut problem required for this competition.

Efficient maritime routing of liquefied natural gas (LNG) is critical for global energy security and supply chain optimization. LNG transportation accounts for a significant share of the world's seaborne trade, with over 75% of bulk commodities transported by sea in 2017. As global energy demands rise and supply chains grow increasingly interconnected, optimizing maritime logistics can deliver considerable economic benefits. The LNG market alone is projected to grow from $129.8 billion in 2023 to $255.2 billion by 2030, driven by the global transition to cleaner energy sources [1]. Effective routing strategies can reduce operational costs, improve fleet utilization, and enhance delivery reliability—factors that are pivotal for competitiveness in the energy sector.

The environmental implications of optimizing LNG shipping routes are equally significant. Maritime transport contributes approximately 2.5% of global greenhouse gas emissions, representing around 1,000 million tons of CO_2 annually [4]. Optimized routing can help reduce fuel consumption and emissions by minimizing travel distances and idle times at ports. For example, a 5% reduction in average vessel travel distances could lead to annual fuel savings of over 1 billion USD globally while avoiding approximately 13 million metric tons of CO_2 emissions. Such gains align with international sustainability goals, such as the International Maritime Organization's (IMO) target to cut shipping emissions by 50% by 2050 [1].

Furthermore, the societal impact of efficient LNG transportation extends beyond economic and environmental factors. LNG plays a vital role in providing cleaner energy alternatives for countries transitioning from coal and oil-based energy systems. Timely and cost-effective delivery of LNG to markets can directly support energy affordability, access, and security in regions heavily reliant on imports. These advances not only strengthen the resilience of global supply chains but also bolster the integration of renewable energy sources by ensuring a stable energy mix during the transition phase. Thus, solving the LNG routing problem at scale contributes to both global economic stability and environmental sustainability, emphasizing its importance for industries and policymakers alike [1].

The formulations and methodologies used are based on our prior research, which explores the application of quantum computing to VRP variants and demonstrates their potential on quantum and hybrid systems [2].

3 QUBO formulation for VRP

To facilitate reproducibility and experimentation, we provide an open-source repository for generating VRP instances tailored to the maritime routing problem. The repository accompanying our previous work in [2] is available at `https://github.com/smharwood/vrp-as-qubo`, includes tools for instance generation and transformation to QUBO formulations.

Based on our previous paper [2], we provide a summary of the formulations and their reformulations into Quadratic Unconstrained Binary Optimization (QUBO) problems.

3.1 Route-Based Formulation

The route-based formulation defines decisions at the route level. Let x_r be a binary variable indicating whether a route $r \in \mathcal{R}$ is traveled ($x_r = 1$) or not ($x_r = 0$). The VRP can be expressed as:

$$\min_x \sum_{r \in \mathcal{R}} c_r x_r$$

subject to:

$$\sum_{r \in \mathcal{R}} \delta_{i,r} x_r = 1, \quad \forall i \in \mathcal{N},$$

$$x_r \in \{0,1\}, \quad \forall r \in \mathcal{R},$$

where c_r is the cost of route r, $\delta_{i,r}$ is an indicator that equals 1 if route r visits customer i, and 0 otherwise.

3.2 Arc-Based Formulation

The arc-based formulation models decisions at the arc level, where $x_{i,s,j,t}$ is a binary variable indicating whether a vehicle travels from node i at time s to node j at time t. The objective is:

$$\min_x \sum_{i,s,j,t} c_{i,j} x_{i,s,j,t},$$

subject to:

$$\sum_{i,s,t} x_{i,s,j,t} = 1, \quad \forall j \in \mathcal{N},$$

$$\sum_{j,t} x_{j,t,i,s} = \sum_{j,t} x_{i,s,j,t}, \quad \forall (i,s) \in \mathcal{N} \times \mathcal{T},$$

$$x_{i,s,j,t} \in \{0,1\}, \quad \forall (i,s,j,t).$$

3.3 Sequence-Based Formulation with Continuous Time

This formulation tracks the sequence of visited nodes and the arrival time at each node. Let $x_{v,p,i}$ be a binary variable that equals one if vehicle v visits node i at position p in its sequence, and s_i be the continuous variable representing the arrival time at node i. The objective is:

$$\min_{x,s} \sum_v \sum_{p<P} \sum_{(i,j)\in\mathcal{A}} c_{i,j} x_{v,p,i} x_{v,p+1,j},$$

subject to:

$$\sum_{v\in\mathcal{V}} \sum_{p=1}^{P} x_{v,p,i} = 1, \quad \forall i \in \mathcal{N},$$

$$\sum_{i\in\mathcal{N}\cup\{d\}} x_{v,p,i} = 1, \quad \forall v \in \mathcal{V}, p \in \{1,\ldots,P\},$$

$$a_i \le s_i \le b_i, \quad \forall i \in \mathcal{N},$$

$$x_{v,p,i} \in \{0,1\}, \quad s_i \in \mathbb{R}, \quad \forall (v,p,i).$$

3.4 Mapping to QUBO

Each of these formulations can be transformed into a QUBO problem by penalizing constraint violations in the objective function:

$$\min_x H(x) = \text{original objective} + \rho \sum (\text{constraint violations}),$$

where ρ is a penalty parameter sufficiently large to ensure equivalence between the penalized formulation and the constrained problem.

For the formulations mentioned above, we found a bound for ρ that guarantees that the reformulation is valid. Yet, it does not over-penalize the constraints, harming the numerical stability of the problem.

Following the results in [2], we used the path-based formulation of the VRP problems in the remainder of this report. This formulation proved to be the most compact in terms of the number of variables required in the QUBO and, thus, the number of qubits to be executed or simulated. This does not preclude the methodology herein from being applied to the other formulations; we used it to scale in larger instances of the original VRP models.

4 Solution via QAOA and CUDA-Q

4.1 Transforming QUBO Formulations from VRP into Max-Cut Instances

The QUBO formulations derived from the Vehicle Routing Problem (VRP) can be transformed into equivalent Max-Cut problem instances. Specifically, a QUBO problem with n variables is mapped to a Max-Cut problem on a complete graph K_{n+1} with $n+1$ vertices labeled as $\{0, 1, \ldots, n\}$. For an edge $ij \in K_{n+1}$, where $i, j > 0$, we define the weight of the edge as:

$$w_{ij} = q_{ij} + q_{ji},$$

where q_{ij} are coefficients from the original QUBO formulation. Additionally, for an edge $i0$ (connecting vertex $i > 0$ with vertex 0), the weight is set as:

$$w_{i0} = \sum_{j=1}^{n} q_{ij} + q_{ji}.$$

Given this transformation, finding a cut $\delta(W)$ in K_{n+1} with weight W is equivalent to solving the QUBO problem with a solution value:

$$Q = -\frac{W}{2} + C,$$

where C is defined as:

$$C = \frac{1}{4}\left(\sum_{e \in E} w_e + 2\sum_{i} q_{ii} + \sum_{i<j} q_{ij} + q_{ji}\right).$$

The QUBO solution can then be directly interpreted from the Max-Cut solution by setting $x_i = 0$ if node i belongs to the set $\delta(W)$, and $x_i = 1$ otherwise.

By applying this transformation, the QUBO formulations from the VRP can leverage advanced quantum and hybrid quantum-classical algorithms designed for Max-Cut problems, such as the Quantum Approximate Optimization Algorithm (QAOA). This enables solving complex routing problems efficiently while preserving the structure of the original optimization model.

4.2 Implementation of Multi-Level QAOA via CUDA-Q

To solve the Max-Cut instances derived from the Vehicle Routing Problem (VRP), we implemented a generalized multi-level Quantum Approximate Optimization Algorithm (QAOA) using NVIDIA's CUDA-Q framework. CUDA-Q provides a unified environment for hybrid quantum-classical computations, leveraging GPU acceleration to scale QAOA for larger graph instances.

Our implementation, based on the tutorial available in the CUDA-Q academic repository, was extended to handle weighted Max-Cut instances. These instances required additional considerations in both the problem formulation and result evaluation stages.

Key Implementation Features

1. **Hamiltonian Construction:** The Max-Cut problem for a weighted graph $G = (V, E)$ with vertex set V and edge set E is represented by the Hamiltonian:

$$H = \sum_{(u,v) \in E} w_{uv} \left(Z_u Z_v - I_u I_v\right),$$

   where w_{uv} is the weight of edge (u, v), and Z and I denote Pauli operators. This generalization allowed the algorithm to solve weighted Max-Cut instances, which are closer to real-world scenarios.

2. **Parameterized QAOA Layers:**

   - *Problem Layer:* Encodes the Hamiltonian evolution, penalizing configurations based on edge weights and the cut value.
   - *Mixer Layer:* Introduces variational flexibility by rotating qubits in superposition states, enabling exploration of the solution space.

3. **Optimization:** Using the CUDA-Q VQE module, we optimized the variational parameters (α and β) to minimize the expectation value of the Hamiltonian. The COBYLA optimizer was employed for this task, initialized with random parameter values to ensure broad exploration.

4. **Result Selection:** Unlike traditional QAOA implementations, which select the most probable sample from the final measurement distribution, our approach was extended to select the sample that maximizes the cut value (or, equivalently, minimizes the QUBO objective function). This refinement ensures the algorithm's robustness and effectiveness in solving weighted instances.

5. **Multi-Level Scaling:** For larger graphs, a divide-and-conquer strategy was employed:

   - Graphs were partitioned into subgraphs using community detection algorithms.
   - Independent QAOA runs computed the Max-Cut solution for each subgraph.
   - A merging stage combined the subgraph solutions into a global solution, using an additional QAOA layer for the border graph.

6. **Potential Distributed GPU Execution:** The CUDA-Q framework's integration with MPI enabled the distribution of subgraph computations across multiple GPUs. This parallelization significantly enhanced scalability and resource efficiency, allowing the solution of large instances.

Generalization and Results

The implemented framework was successfully generalized for weighted Max-Cut instances. This generalization can potentially improve the algorithm's applicability to real-world problems and ensure accurate solutions by selecting the best sample based on the cut value.

Complete implementation details, including source code and results, are available in the CUDA-Q academic repository. The source code used to solve the VRP problem for small instances can be found in: VRP-Challenge.py. The decomposition technique used for larger instances can be found in: QAOA-cudaQ.py.

We note that after experimenting with this implementation, we were not able to make it scale beyond the sizes that were relevant to our application. In particular, it was impractical to simulate a single layer of QAOA with more than 15 nodes in the graph to find a maximum cut. Although the multi-level approach would enhance the sizes of instances that could be solved, we were unable to make problems with sizes larger than 50 variables converge to the optimal solution using CPUs. These challenges motivated us to find an alternative, quantum(-inspired) method that could take advantage of GPU resources and scale to sizes relevant to our application. That is how we developed the tensor network construction solver for discrete optimization, `TenSolver`.

5 Tensor Network Solver

Our Tensor network solver, which we called `TenSolver`, maps the QUBO instance to a Hamiltonian that, on its own, can be approximated by a tensor network. The bond dimension of the tensor network can be used to control the level of approximation, which also strongly affects its solvability. In particular, we make use of the well-developed methods to contract tensor networks using GPUs, some of them implemented in the software `iTensor` or the library CUDA-TN to reach scales of practical interest. We have developed a fully open-source implementation of our solver in the language Julia, in a package that we called `TenSolver.jl`, found in the repository `https://github.com/SECQUOIA/TenSolver.jl`.

5.1 Hamiltonian Representation of a QUBO

Consider a Quadratic Unconstrained Binary Optimization (BUBO) problem,

$$\begin{aligned} \min_x \quad & x^\top Q x \\ \text{s.t.} \quad & x \in \{0,1\}^N. \end{aligned}$$

Finding the optimal solution to an arbitrary such problem is NP-hard [3].

To solve this problem, we convert it from a discrete problem in N variables to a quantum system with N particles taking values on 2-dimensional Hilbert spaces (qubits). First, introduce the matrix

$$D = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}.$$

D is a Hermitian operator acting on $\mathbb{C}^2$. Also notice that it can be generated from Pauli matrices as $D = (\mathrm{I} - \sigma_Z)/2$. For a multiparticle system, we denote by D_i the operator that applies D to the i-th particle and the identity to all others.

From the matrix Q on a QUBO objective function, we form a Hamiltonian by substituting each variable x_i with the respective operator D_i. Since these are single-particle operators, they commute with each other, i.e., $D_iD_j = D_jD_i$, and no ambiguity arises.

$$H = \sum_{i,j} Q_{ij} D_i D_j.$$

The Hamiltonian H establishes a quantum system whose ground state has energy, by construction, equal to the minimum value of the original QUBO. Additionally, any eigenstate with this energy will be a probability distribution over the optimal binary strings for the QUBO. This way, solving this quantum system also solves the original problem.

5.2 Tensor Networks

A general state in a N qubits system requires 2^N degrees of freedom. For any reasonably large N, this is too much, even for the best classical hardware available. In many cases, however, these states are sparse and do not actually require all such information. To take advantage of this, we can rewrite sparse tensors in a more amenable representation.

A *tensor network* is an unevaluated tensor contraction. It consists of a graph in which each degree k node is a rank k tensor, and each edge represents a contraction between respective indices. See Figure 1 for the basic operations one can perform on a tensor network together with their graph representations.

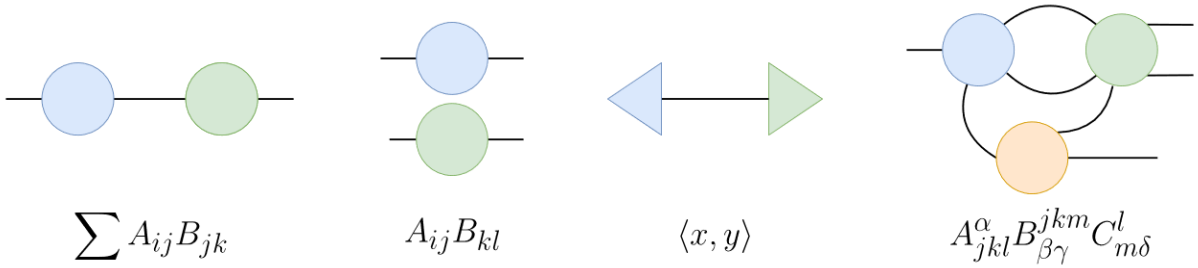


Figure 1: A tensor network is a graph where each node represents a tensor and each edge represents a contraction. The rank of a tensor network is the number of "free" (uncontracted) wires. We arrive at their tensor product by juxtaposing two networks without any connecting wires. In the figure, you can see how networks represent some basic operations.

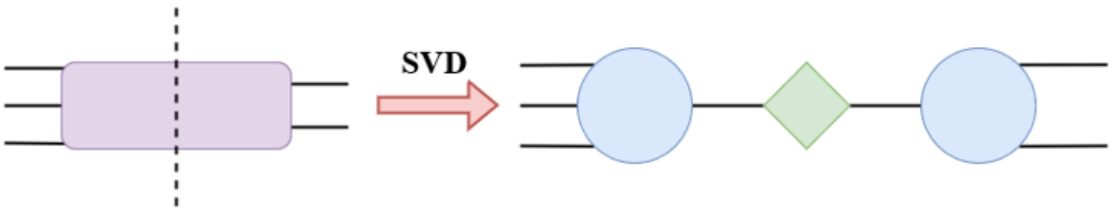


Figure 2: A tensor can be decomposed into a network by reshaping it into a matrix and applying a decomposition, such as the SVD.

Usually, a network representation of a tensor allows for better taking advantage of hidden topological properties that might not be straightforward when looking at it as a single tensor. As an example, large but sparse tensors can often be written as contractions of small tensors, enabling significant reductions in their storage requirements and related computational costs.

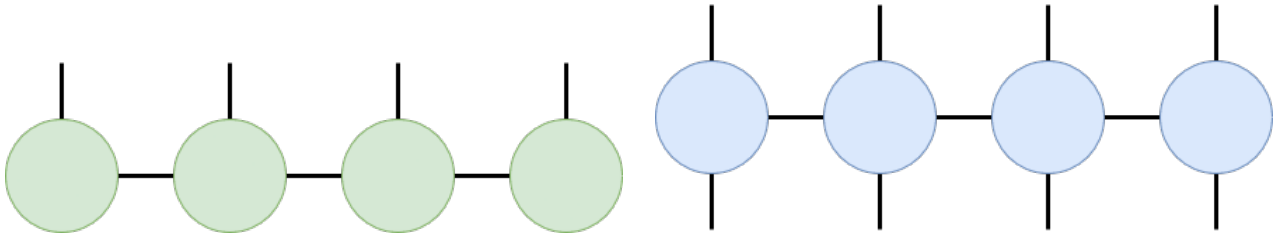
Even when a tensor is not sparse, it can be approximated by a contraction through its Singular Value Decomposition (SVD). See Figure 2 for a graphical representation of the process. The tensor T is factored into a contraction where the middle matrix Σ contains its singular values. By throwing away all singular values below a specific cutoff, this factorization approximates a tensor into a contraction with an arbitrarily small linking dimension.

5.3 Matrix Product States

In general, for a d-dimensional vector space, a rank k tensor contains d^k degrees of freedom. Using the SVD factorization discussed above, it is possible to approximate it by a network containing $O(kd)$ degrees of freedom.

In a quantum setting, this means passing from exponential to linear complexity on the number of particles.

A *Matrix Product State* (MPS) is a sequential tensor network contraction of rank 3 tensors. They usually represent quantum states. A *Matrix Product Operator* (MPO) is a sequential tensor network contraction of rank 4 tensors. They usually represent quantum operators. See Figure 3 for the tensor network representation of such operators.



(a) Matrix Product State (b) Matrix Product Operator

Figure 3: Matrix Product States and Operators are tensors represented by linear graphs. When the dimensions of the contractions are bounded, they only have a number of degrees of freedom that is polynomial to the number of open wires.

5.3.1 DMRG—Finding Ground States

For a tensor network in MPO form, the method *Density Matrix Renormalization Group* [5, 6] is a variational algorithm that finds an MPS approximating its lowest eigenstate in polynomial time.

5.4 Optimization Methodology

The tensor network-based method finds the optimal solution to a QUBO by converting it into a quantum system using the procedure from Section 5.1, representing it by an MPO tensor network, and finding its ground state through DMRG. The steps are represented on Algorithm 1.

Algorithm 1 Steps for a Tensor Network-based QUBO solver.

$H \leftarrow \text{Quantize}(Q)$
$T \leftarrow \text{MPO}(H)$
$E, \psi \leftarrow \text{DMRG}(T)$
Return a sample from ψ

Notice in the algorithm that while the quantization process is exact, the MPO representation is an approximation, and the DMRG is an iterative algorithm that might not yield the exact solution. Generally, there is a controllable trade-off between accuracy and speed when choosing how much these approximations should simplify the original problem.

Additionally, for reasons of numerical stability, the algorithm performs better for full-rank matrices. Therefore, we add a small noise to the Q matrix to improve its stability. The implemented algorithm minimizes the quadratic form defined by $\varepsilon I + Q$.

6 Results

This section presents the computational results of our proposed methods for solving QUBO instances derived from the Vehicle Routing Problem (VRP). The comparison includes the tensor network-based solver (`TenSolver.jl`) executed on GPU and CPU and classical MIP solvers (Gurobi and HiGHS) applied to the QUBO instances. All methods were evaluated with a timeout of 1000 seconds.

We considered also including a CUDA-Q implementation of the Quantum Approximate Optimization Algorithm (QAOA), but faced a complication leading to suboptimal solutions for relatively limited sized instances. For that reason, we only include those limits corresponding to a single level and the multi-level decomposition of Max-Cut instances and instead center on the performance of our proposed `TenSolver.jl`, which still relies heavily on CUDA Tensor contraction libraries.

6.1 Performance Comparison

Figure 4 illustrates the solving time (in seconds) as a function of the number of variables in the QUBO problem. The top x-axis corresponds to the time horizons for the maritime routing problems generated to test the implementations. For simplicity, consider that only the path-based formulation results are included. The results reveal the following:

1. **Tensor Network Solver (`TenSolver.jl`):** The CPU implementation of `TenSolver.jl` demonstrates the best performance for larger instances, solving problems with up to 794 variables in approximately 230 seconds. In comparison, the GPU implementation is slower (365 seconds for the largest instance) but still outperforms classical solvers for instances beyond 217 variables. More importantly, from preliminary results, we notice that the scaling of the GPU implementation is less pronounced, leading to speedups at larger scales.

2. **Classical Solvers (Gurobi and HiGHS):** Classical solvers perform well for smaller problem sizes. Gurobi achieves competitive solving times for instances with up to 96 variables but begins to time out for larger instances (beyond 217 variables). HiGHS demonstrates similar performance, showing its limitations as the problem size increases.

3. **CUDA-Q QAOA Implementations:** The single-layer and multi-level QAOA implementations using CUDA-Q exhibit scalability limits, as shown by the dashed lines in Figure 4. The single-layer implementation becomes computationally expensive beyond 50 variables, while the multi-level approach slightly extends the solvable range but remains constrained by the complexity of larger QUBO instances.

6.2 Scalability and Hardware Utilization

The implementation of `TenSolver.jl` showcases advantages in scalability by leveraging the parallelism inherent in tensor contractions. This approach enables the efficient handling of large-scale QUBO instances, outperforming both classical solvers and CUDA-Q implementations for large problem sizes. The results highlight the importance of hardware-aware optimization techniques in achieving superior performance for quantum-inspired solvers.

6.3 Insights for Maritime Routing

These results have critical implications for solving real-world maritime routing problems. The ability to efficiently solve large QUBO instances could allow for the optimization of complex routing scenarios involving multiple vessels, supply ports, and demand nodes. This is particularly important in time-sensitive and cost-driven industries such as LNG transportation, where operational efficiency directly impacts economic and environmental outcomes.

The tensor network-based solver provides a robust framework for tackling VRP-derived optimization problems, demonstrating the potential of quantum-inspired methods to outperform classical techniques in solving complex real-world problems.

6.4 Implementation details

The results here presented were generated with Julia 1.10.5, `TenSolver.jl v0.1.1`, Gurobi 12.0, Highs 1.7.1. The CPU results were run on an AMD Epyc Milan node, and the GPU results were obtained in an NVidia A100; both provided by the Anvil supercomputer at Purdue University. Preliminary results for CPU were run on a laptop with a Intel Core i7 CPU with 32Gb or RAM.

The code to generate the instances is found on `https://github.com/smharwood/vrp-as-qubo` based on our paper [2], the Tensolver is found in the repository `https://github.com/SECQUOIA/TenSolver.jl` and the code supporting this project is found at `https://github.com/SECQUOIA/vrp-qinnovision`.

7 Conclusion

In this work, we presented a comprehensive approach to solving QUBO instances derived from the Vehicle Routing Problem (VRP) using quantum-inspired and classical methods. Our tensor network-based solver, `TenSolver.jl`, demonstrated superior scalability compared to classical solvers like Gurobi and HiGHS applied to QUBO instances, particularly for larger problem sizes. By leveraging tensor contractions, our approach reduced solving times, enabling the solution of instances with up to 794 variables within practical time limits. Additionally, the custom tensor network implementation outperformed existing CUDA-Q QAOA methods

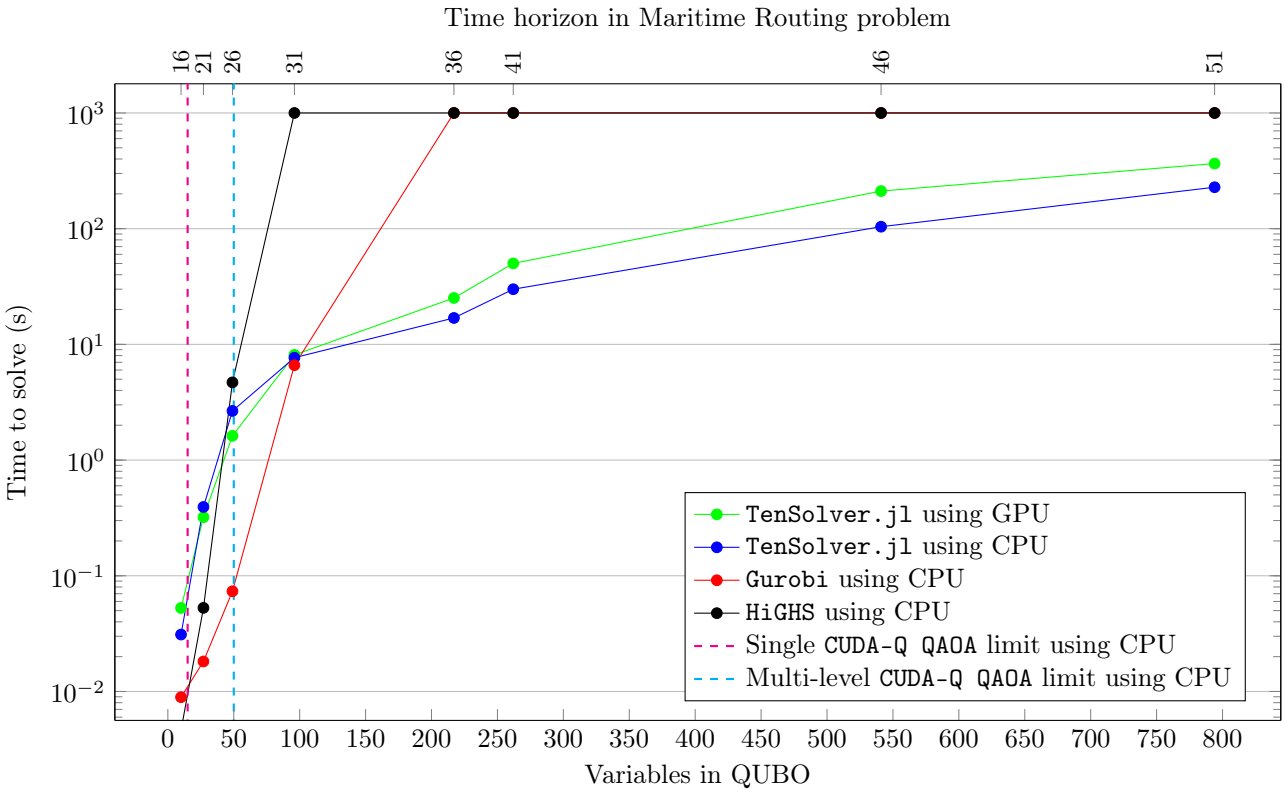


Figure 4: QUBO Solving time comparison between `TenSolver.jl` and a classical solver (`Gurobi`). All instances had a timeout of 1000s.

in terms of both scalability and efficiency, highlighting the benefits of alternative representations for QUBO problems.

These results underscore the potential of quantum(-inspired) optimization techniques to address complex real-world problems. We focused on maritime routing in liquefied natural gas (LNG) transportation, as it was related to our previous paper [2] but understand that there are plenty of applications of VRP, particularly relevant for the chemical industry which we specialize in. Our methods efficiently solve discrete optimization problems, providing actionable insights for industries reliant on efficient logistics and supply chain operations. One key issue that our limited time has prevented us from completing is the comparison against the traditional methods used in practice to tackle VRP problems, namely mixed integer and constraint programming-based algorithms. We acknowledge that these methods have been developed over decades, and if a quantum(—inspired) solution procedure is to be proposed for VRP problems, it needs to be competitive with the state-of-the-art. We remain motivated that our approach based on network contractions and our open-source solver `TenSolver.jl` will inspire others to integrate these solution methods into finding optimal solutions to VRP and other combinatorial optimization problems. Our approach seems to be a way toward unlocking the potential of GPU processing and NVidia hardware for discrete optimization.

References

[1] Maricruz A Fun-sang Cepeda, Newton Narciso Pereira, Suzana Kahn, and Jean-David Caprace. A review of the use of lng versus hfo in maritime industry. *Marine Systems & Ocean Technology*, 14(2):75–84, 2019.

[2] Stuart Harwood, Claudio Gambella, Dimitar Trenev, Andrea Simonetto, David Bernal, and Donny Greenberg. Formulating and solving routing problems on quantum computers. *IEEE Transactions on Quantum Engineering*, 2:1–17, 2021.

[3] Abraham P Punnen. The quadratic unconstrained binary optimization problem. *Springer International Publishing*, 10:978–3, 2022.

[4] Jack Sharples. Lng supply chains and the development of lng as a shipping fuel in northern europe. 2019.

[5] Steven R. White. Density matrix formulation for quantum renormalization groups. *Phys. Rev. Lett.*, 69:2863–2866, Nov 1992.

[6] Steven R. White. Density-matrix algorithms for quantum renormalization groups. *Phys. Rev. B*, 48:10345–10356, Oct 1993.