Insper

Quantum Computing Applied to Financial Market Optimization Problems

Felipe Schiavinato Borges Souza (felipesbs@al.insper.edu.br)
Matheus Silva Melo de Oliveira (matheussmo@al.insper.edu.br)
Nívea de Abreu Dantas Lima (niveaadl@al.insper.edu.br)

Capstone Project Report
Final Version

Trabalho de Conclusão de Curso

São Paulo – SP – Brazil
March, 2024

Felipe Schiavinato Borges Souza (felipesbs@al.insper.edu.br)
Matheus Silva Melo de Oliveira (matheussmo@al.insper.edu.br)
Nívea de Abreu Dantas Lima (niveaadl@al.insper.edu.br)

Quantum Computing Applied to Financial Market Optimization Problems

Capstone Project Report
Final Version

Trabalho de Conclusão de Curso

Undergraduate Capstone Project Report submitted to the respective (Computer/Mechanical/Mechatronic) Engineering programs or Computer Science program in partial fulfillment of the requirements for the Bachelor's Degree.

Advisor: Prof. Dr. Luciano Silva / Luciano Pereira Soares

General Coordinator TCC/PFE: Prof. Dr. Luciano Pereira Soares

São Paulo – SP – Brazil
March, 2024

Felipe Schiavinato Borges Souza (felipesbs@al.insper.edu.br)

Matheus Silva Melo de Oliveira (matheussmo@al.insper.edu.br)

Nívea de Abreu Dantas Lima (niveaadl@al.insper.edu.br)

Quantum Computing Applied to Financial Market Optimization Problems

Undergraduate Capstone Project Report submitted to the respective (Computer/Mechanical/Mechatronic) Engineering programs or Computer Science program in partial fulfillment of the requirements for the Bachelor's Degree.

Advisor: Prof. Dr. Luciano Silva / Luciano Pereira Soares

Capstone Evaluation Committee

____

Luciano Pereira Soares

Insper

____

Luciano Pereira Soares

Insper

____

Raul Ikeda Gomes da Silva

Insper

____

Rodrigo Carareto

Insper

TABLE OF CONTENTS

Quantum Computing Applied to Financial Market Optimization Problems

Felipe Schiavinato Borges S.
Computer Engineering
Insper Institute of Education and Research
São Paulo, Brazil
felipesbs@al.insper.edu.br

Matheus Silva Melo de Oliveira
Computer Engineering
Insper Institute of Education and Research
São Paulo, Brazil
matheussmo@al.insper.edu.br

Nívea de Abreu Dantas Lima
Computer Engineering
Insper Institute of Education and Research
São Paulo, Brazil
niveaadl@al.insper.edu.br

***Abstract*— This Capstone Project explores the application of Quantum Computing to enhance logistic operations through the optimization of the NP-hard problem of finding the shortest possible tour in the Travelling Salesman Problem (TSP), adapted as the Vehicle Routing Problem (VRP). By harnessing advanced quantum algorithms such as the Quantum Approximate Optimization Algorithm (QAOA) and the Variational Quantum Eigensolver (VQE), the project seeks to develop a flexible VRP solution adaptable to a wide range of logistics challenges, the adapted algorithm will be compared against classical solutions to validate its efficiency. As a case study, we will specifically address a cash distribution network for Bradesco, aiming to optimize routes for cost, time, demand, and security efficiency. This example demonstrates the practical applicability of our approach to real-world logistics problems. The project delivered a comprehensive system, featuring a complete pipeline that integrates data flow from input to optimized routing solutions, presented through an intuitive visual interface. The outcome can be measured against key performance indicators such as operational cost reduction and routing efficiency. This approach not only promises a cutting-edge solution but also sets a benchmark for integrating quantum computing solutions in logistic operations.**

Keywords*— Quantum Computing, Optimization Problems, Vehicle Routing Problem (VRP), Travelling Salesman Problem (TSP), Quantum Algorithms, Quantum Approximate Optimization Algorithm (QAOA), Variational Quantum Eigensolver (VQE), Logistic Operations, Efficiency Validation, Classical Solutions, Real-World Applications

***Resumo*—Este Projeto Final de Engenharia explora a aplicação da Computação Quântica para aprimorar operações logísticas por meio da otimização do problema NP-difícil de encontrar o percurso mais curto no Problema do Caixeiro Viajante (TSP), adaptado como o Problema de Roteamento de Veículos (VRP). Utilizando algoritmos quânticos avançados, como o Quantum Approximate Optimization Algorithm (QAOA) e o Variational Quantum Eigensolver (VQE), o projeto busca desenvolver uma solução flexível para o VRP adaptável a uma ampla gama de desafios logísticos, o algoritmo adaptado será comparado com soluções clássicas para validar sua eficiência. Como estudo de caso, abordaremos especificamente uma rede de distribuição de numerários para o Bradesco, com o objetivo de otimizar rotas para eficiência de custo, tempo, demanda e segurança. Este exemplo demonstrará a aplicabilidade prática de nossa abordagem a problemas logísticos do mundo real. O projeto fornece um sistema abrangente, apresentando um pipeline completo que integra o fluxo de dados desde a entrada até as soluções de roteamento otimizadas, apresentado por meio de uma interface visual intuitiva. O resultado pode medido contra indicadores-chave de desempenho, como redução de custos operacionais e eficiência de roteamento. Esta abordagem não apenas promete uma solução de ponta, mas também estabelece um marco para a integração de soluções de computação quântica em operações logísticas.**

Palavras-chave*— Computação Quântica, Problemas de Otimização, Problema de Roteamento de Veículos (VRP), Problema do Caixeiro Viajante (TSP), Algoritmos Quânticos, Quantum Approximate Optimization Algorithm (QAOA), Variational Quantum Eigensolver (VQE), Operações Logísticas, Validação de Eficiência, Soluções Clássicas, Aplicações no Mundo Real

---

The authors are pursuing a bachelor's in computer engineering, Insper Institute of Education and Research, São Paulo, SP 04546-042, Brazil (e-mail: felipesbs@al.insper.edu.br; matheussmo@al.insper.edu.br; niveaadl@al.insper.edu.br).

I. INTRODUCTION

This Capstone Project of Insper's Engineering course delves into the application of Quantum Computing to improve logistic operations by tackling the NP-hard Vehicle Routing Problem (VRP) [1], an adaptation of the classic Travelling Salesman Problem (TSP). By leveraging state-of-the-art quantum algorithms like the Quantum Approximate Optimization Algorithm (QAOA) and the Variational Quantum Eigensolver (VQE), this study aims to develop a robust and flexible VRP solution that can be adapted to a broad spectrum of logistic challenges. The formal definition of the Travelling Salesman Problem (TSP) is given in the Appendix A.

The Vehicle Routing Problem (VRP) extends the Traveling Salesman Problem (TSP) by incorporating multiple vehicles into the scenario. The VRP aims to determine the optimal set of routes for a fleet of vehicles, starting and ending at a common depot, such that all designated nodes are visited at least once [1]. This adjustment introduces additional complexities, including the distribution of nodes among vehicles to minimize overall travel time or distance. The complete definition of the Vehicle Routing Problem (VRP) for the scope of this document can be found in Section II-B.

The cash supply chain is a concrete example of VRP, it consists of distributing money from a central to adjacent branches, seeking to establish the best route for different vehicles, considering the money-carrying capacity of each car, safety aspects and the shortest route to be covered by each car. Financial institutions use conventional approaches to solve optimization problems, the project aims to use quantum computing to solve an optimization problem for the partner organization, Bradesco S.A.[2].

The process analysis and forecasting of cash demand is essential for ensuring the availability of money and integrity of the supply chain. Once demand and needs of each supply point are defined and calculated using forecast models, which include agencies and ATMs, cash remittances are arranged in multiple banknote groups of 1000, always adjusting in appropriate cases to the immediately higher multiple of a thousand. [2]

After establishing the cash demand for each of the ATMs to be supplied, it is up to the transport sector, whether outsourced or integrated with the bank, to define the route supply, considering factors such as security, distance and even quantity of banknotes that each truck must take to fulfill the established routes, becoming a challenge to create routes that efficiently meet these requirements in a timely manner for supply of the ATM's network

The developed version of VRP, which given a number of nodes (agencies or ATMs), and a number of vehicles, must establish a solution that returns the most appropriate routes for each vehicle, seeking to minimize the cost (distance, car load, or even the number of vehicles if the timespan allow this exchange for cost effectiveness) and increase the agility of logistics team. An optimal solution of VRP, in different contexts, can represent a cost reduction of almost 30% in transportation expenses [3], generating a demand for algorithms capable of bringing this output. Current solutions reside in exact methods, heuristics and meta-heuristics, which are limited to process hundreds[3]

[1] Consider the Travelling Salesman Problem (TSP) and Vehicle Routing Problem (VRP) in their NP-hard or combinatorial versions. We focus on the challenge of, given a list of cities and the distances between each pair of cities, determining the *shortest* possible route that visits each city exactly once and returns to the origin city. The NP-complete or decision version of these problems, which involves deciding whether a tour of at most a given length L exists, is not addressed in this project. [6]

[2] Bradesco is one of the largest banks in Brazil and the world, founded in 1943 by Amador Aguiar. Bradesco has so much influence in Brazilian territory because it is the only private bank in the country to be present in all municipalities in the country in 2010. [42]

[3] Hundreds of variables might seem excessive for a physical logistics problem, such as cash distribution. However, when considering larger-scale applications like internet network traffic or trail paths in microprocessors and other semiconductors, the value of solutions to this problem becomes evident.

of variables [4], not being sufficient for the scale necessary to meet larger scenarios such as cash supply chain.

Therefore, the use and modeling of a Quantum Computing algorithm aims to solve this problem on larger scales (more than thousands of variables) than classical solutions, meeting demands of large volume scenarios, such as being able to serve agencies distributed across a metropolis, or even cities. Our main goal is to generate better distribution routes that minimize costs related to the problem, such as distance, security, and logistics, allowing better agility for logistics teams and better results obtained from heuristics and benchmarks established during the modeling process. [5]

The prototyping of a quantum algorithm to solve the problem described is of paramount importance as it aims to accelerate the determination of the best routes to be followed by vehicles and can directly influence the transportation cost of the financial institution, facilitating decision-making of the associated logistics team.

This project aims to build a solution that integrates data entry as inputs (agency locations and money demands per agency), data processing by quantum algorithms and generation of routes with the lowest estimated cost in optimizations, and construction of an interface for visualizing calculated routes, travel times, associated costs, and other related metrics. We can see an initial planned architecture in Fig. 1.

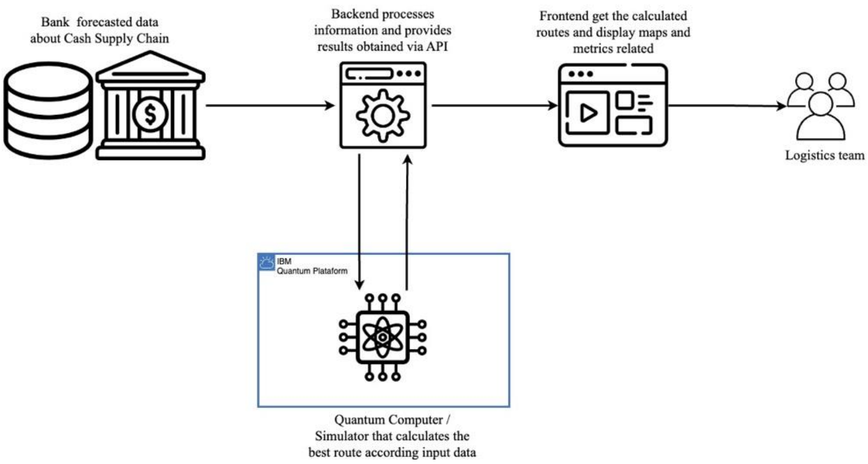

Fig. 1. Planned project overview architecture.

A. *Classical vs Quantum Solutions*

In computational problem-solving, optimization challenges often expose the limitations of classical computing. Classical computation, grounded in binary logic (1s and 0s) and Turing machines, faces inherent complexities. This reality is particularly evident in the computational intractability of problems within the P vs. NP complexity classification.

The class P consists of decision problems that can be solved in polynomial time by a deterministic Turing machine, i.e. a solution can be computed relatively quickly. The class NP (nondeterministic polynomial time) encompasses problems where a solution can be verified in polynomial time but finding that solution might require exponential time with deterministic computation. The assumption $P \neq NP$ suggests that many problems, while their solutions can be verified efficiently, cannot be solved efficiently in the first place. This assumption is widely believed because, despite the best efforts of mathematicians and computer scientists worldwide, no polynomial-time solution has been found for any NP-complete[4] problem. As such, the Clay Mathematics Institute has offered a $1 million prize for anyone who can conclusively resolve the P vs. NP

[4] "If a problem is NP and all other NP problems are polynomial-time reducible to it, the problem is NP-complete." [45]

problem. Given that no one has yet succeeded, there is a strong, though debatable and not proved, consensus among field experts that P $\neq$ NP in classical computation. [6]

As a result, heuristics and approximation algorithms are frequently used to bypass the exhaustive searches required for optimal solutions. However, something must be given when using heuristics or approximation algorithms. To gain time, these methods often sacrifice optimality, which can be frustrating to many mathematicians who strive for precise solutions. Despite their practical utility, approximations inherently imply a compromise: they do not guarantee that the solution is the best possible but instead offer a "good enough" result within a feasible timeframe.

A famous NP problem that highlights this compromise is the Traveling Salesman Problem (TSP). Even the fastest exact algorithms we have, such as branch and bound or branch and cut, which guarantee optimality, still face factorial complexity in the worst case due to the nature of the problem.

To estimate the complexity of the TSP we will reduce our problem to a complete graph[5] of $\boldsymbol{n}$ nodes, (1) gives the number of edges $\boldsymbol{E}$ in a complete graph of $\boldsymbol{n}$ nodes. Simplifying the expression for Big O[6] notation, we can infer that $\boldsymbol{E}$ scales quadratically $\boldsymbol{O(n^2)}$.

$$E = \frac{n(n-1)}{2} \tag{1}$$

where

$\boldsymbol{E}$	number of edges in a compete graph;
$\boldsymbol{n}$	number of nodes in a complete graph.

However, brute force solutions iterate through all possible paths, given by the permutation of edges in the complete graph or the Hamiltonian Cycle [7], the number of permutations $\boldsymbol{N_H}$ can be calculated in (2) given in [8]. Simplifying (2) for Big O notation we can infer that $\boldsymbol{N_H}$ scales factorially $\boldsymbol{O(n!)}$.

$$N_H = \frac{(n-1)!}{2} \tag{2}$$

where

$\boldsymbol{N_H}$	permutations of edges that visits all nodes (all Hamiltonian Cycles);
$\boldsymbol{n}$	number of nodes in a complete graph.

Given the factorial explosion in permutations as the number of cities (nodes) increases, these algorithms must explore all paths or use intricate pruning techniques, still $\boldsymbol{O(n!)}$ in the worst case, to return with certainty the optimum solution.

In response to these challenges, quantum computing offers promising alternatives. By harnessing quantum-mechanical phenomena, such as superposition and entanglement, quantum computers can explore multiple solutions simultaneously. The use of qubits instead of classical bits allows for an exponential scaling of the solution space. Algorithms like Grover's search and Shor's factoring have shown that quantum computing can drastically outperform classical methods under specific circumstances.

For TSP class problems, a quantum algorithm called phase estimation can improve complexity from $\boldsymbol{O(n!)}$ to $\boldsymbol{O(\sqrt{(n-1)!}}$, which is not polynomial, but still a great

[5] "A complete graph is a graph in which each pair of graph vertices is connected by an edge." [43], this modeling represents all paths possible between all cities in the TSP, allowing the deduction of the problem complexity from the complete graph parameters equations.

[6] Big O notation definition with examples in [44].

improvement as seen in Fig. 2. [9] Note that x-axis and y-axis are *not in the same scale* due to rapidly $\boldsymbol{y}$ increase in both complexity curves.

In a theoretically perfect quantum system representing the Traveling Salesman Problem (TSP), each edge of a complete graph can be encoded as a qubit. This encoding allows each edge to function as a binary variable, indicating whether it is included in the optimal solution. Consequently, the space complexity would be $\boldsymbol{O}(\boldsymbol{n}^2)$ qubits, where $\boldsymbol{n}$ represents the number of nodes or cities. The time complexity would reduce to $\boldsymbol{O}(\mathbf{1})$, as the system's measurement would instantly collapse to the solution.

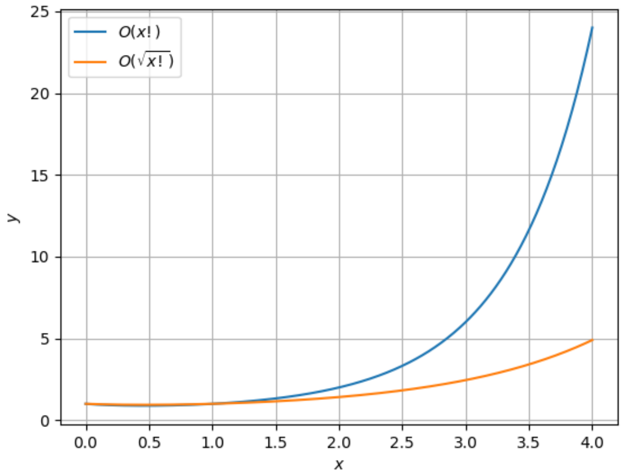

Fig. 2. Factorial and square root factorial over continuous values

In practical terms, the most effective current approaches to tackling the Traveling Salesman Problem (TSP) and other optimization challenges in quantum computing involve hybrid quantum heuristic methods. These algorithms combine quantum state observation with classical optimization techniques, leveraging the ansatz approach commonly used in quantum computing. An ansatz is essentially an initial guess that can be iteratively improved by the algorithm, which measures the system and refines the initial guess by identifying patterns in the wave function. Although these heuristic algorithms do not guarantee optimality, they excel in approximating the ground state (the solution) and can potentially outperform classical heuristic methods when dealing with large problem instances. Their complexity is challenging to estimate due to the dependence on the ansatz. Such algorithms are designed to operate on NISQ (Noisy Intermediate-Scale Quantum) processors, which are suitable for algorithms capable of mitigating the noise inherent in this architecture. [10]

Quantum computing is not without caveats. Current quantum hardware remains prone to errors and limited in the number of qubits it can manipulate effectively (NISQ). The practical implementation of quantum algorithms, such as the Variational Quantum Eigensolver (VQE) and the Quantum Approximate Optimization Algorithm (QAOA), is therefore constrained by these hardware challenges. Despite these limitations, quantum algorithms promise polynomial or even constant-time improvements for specific problems, which can significantly shift the paradigm of computational problem-solving.

As research continues to address these hardware and error challenges, quantum computing emerges as a complementary tool for tackling complex optimization problems. The exploration of quantum-enhanced heuristics, error mitigation techniques, and hybrid classical-quantum algorithms holds the potential to break the complexity barriers of classical computation, providing efficient solutions where traditional methods fall short.

B. Project Scope

In the project focused on enhance logistic operations through the optimization of the VRP using quantum computing, specifically cash supply chain for Bradesco, the objective is to develop, implement, and test quantum algorithms and circuits aimed at enhancing the metrics for money distribution across various locations. This endeavor into financial optimization is driven by the potential of quantum computers to address problems that are currently seen as intractable by classical computing methods, or those that require an impractically long time to solve. The integration of such cutting-edge technologies is anticipated to markedly improve the efficiency and effectiveness of logistics operations, leading to cost reductions and enhanced customer service.

A key aspect of the project involves conducting simulations and tests on quantum computers, with a preference for accessing the advanced resources offered by IBM. This approach enables a direct comparison of the performance of the developed algorithms against classical solutions. The thorough analysis and documentation of these processes, culminating in a detailed technical report, are crucial for assessing the feasibility and efficacy of applying quantum computing in this domain.

The project explicitly excludes the development or modification of quantum hardware, as well as the application of the developed algorithms to areas outside logistic optimization. The project will not provide ongoing technical support for the maintenance or updating of the implemented solutions after its completion and the knowledge transfer to Bradesco, including documentation, source codes, and training.

Costs associated with the project encompass access to specialized quantum computers, human resources (notably the recruitment of experts in quantum computing), and the procurement of software licenses and materials necessary for project development and dissemination. Challenges include limited access to the requisite quantum computing resources, the intrinsic complexity of quantum computing concepts, and reliance on third-party tools for algorithm simulation and testing.

Project Premises:

1) **Collaboration with IBM:** Access to IBM's quantum computers, which are not publicly available, is anticipated.
2) **Team Learning Capability:** The team is expected to acquire necessary quantum computing knowledge within the project's timeframe.
3) **Technical Feasibility:** The selected optimization problems are deemed theoretically solvable through quantum computing, based on currently or imminently available technologies.

C. Resources

The resources expected for a good execution of the development and testing processes:

1) Access to IBM Quantum Computers publicly available and restricted and others restricted to paid access.
2) Anonymized data representative of problem situations requested by Bradesco.
3) Bibliographies and technical references on the development of scalable and adaptable quantum solutions to real contexts, such as articles, books and documentation.
4) Project management platform, such as Trello, for cataloging activities, progress and objectives, facilitating the solution development cycle.

D. Project Timeline

The timeline planned for the project follows the indications foreseen by the group for the development of the Project, as shown in Table I.

E. Stakeholders

Table II categorizes the stakeholders involved in the project, detailing their roles and interests. This overview is essential for understanding the responsibilities and contributions of each party to the project.

TABLE I
SCHEDULE

Activity	Expected Weeks
Delimiting the scope and aligning expectations with the company	1-3
Initial modeling of the algorithm and adaptation to the nuances of the inputs and restrictions provided	4-6
Definition of metrics and benchmarks for model validation and testing in situations close to reality	7-9
Validation of the model against classical approaches and creation of integration of the solution with a real Quantum Computer.	9-11
Creation of an interactive dashboard to visualize the solution's output and integration with database systems with synthetic data provided by the company.	11-13
Final delivery of the solution prototyped and validated.	14

TABLE II
STAKEHOLDERS TABLE

Stakeholder	Position	Role	Expectations
Bradesco	Partner Company	Propose the challenge and carry out validations.	Obtain a quantum algorithm capable of solving the proposed problem.
Insper	College	Integrate students and companies, through the delivery of the project and validation of the knowledge acquired during graduation.	Validate students' technical knowledge in real projects
Luciano Silva / Luciano Soares	Advisor	Direct technical aspects of the project.	Provide maximum guidance for successful project delivery.
Felipe Schiavinato; Matheus Oliveira; Nívea Lima.	Students	Apply knowledge acquired during graduation in practical and real cases, in addition to increasing networking in the job market.	Being able to solve the proposed problem and put knowledge obtained into practice.

F. Risks

Understanding that the solution must handle sensitive information from the partner company is essential. Therefore, implementing robust privacy and security mechanisms is crucial for its proper functioning and successful operation.

Tests must be carried out on different volumes of data in order to indicate success, which in our specific case, refer to several combinatorial scenarios, which, as far as possible, must be simulated to better collect comparative metrics and understanding the limits of the proposed solution.

G. Ethical and Professional

To carry out the project, it is necessary to comply with ethical and professional standards that guarantee its success for the parties involved.

It is necessary to guarantee that data, if provided, by the partner company by the group will not be exposed to unauthorized sources or the internet. Thus, the handling of data provided for the sole and strict use of the group for the development of the project must be followed with caution.

To ensure transparency and avoid bias, the group is committed to documenting and sharing steps followed and decisions taken during the process to better align the parties and avoid disagreements that generate dilemmas or disagreements about ethical and professional aspects.

Considering all the possible ethical and moral implications present and inherent to this project, an approach based on integrity, transparency and responsibility that is supported by legal and social standards must be followed.

H. Standards

When designing the visual interface for displaying calculated routes in interactive maps and presenting metrics that add value to the selected route for users, it is essential to adhere to established ergonomic standards in UI/UX. These standards, provide crucial guidelines for ensuring optimal human-system interaction, usability, and accessibility.

Incorporating standards of this type into the design process will help ensure that the visual interface for displaying calculated routes and associated metrics in interactive maps meets the highest ergonomic, usability, and accessibility standards, ultimately enhancing the overall user experience and satisfaction.

Dealing with highly sensitive information, the application must be compliant with good practices for implementing security controls for the application that guarantee, in addition to the expected performance and results, the security of input and processed data, without exposure inappropriate use of these to third parties.

By following these guidelines, the application follows best practices and technical standards that guarantee greater reliability and less exposure to possible risks.

I. State-of-the-Art Review

For improving cash managements systems in current ATM networks, some analysis and research has already been carried out using classical algorithms. In this scope, in [11], choosing the best strategy for cash managements in ATM networks is necessary to both minimize costs regarded the ecosystem and risks linked to this scope. They built a Python module to deliver cash management solutions based on linear programming, using multi-objective optimizations to process incidences matrices that represents the problem.

Due to the computational complexity of the problem, classical solutions may often not be completely efficient in large scale of the Vehicle Routing Problem (VRP). According to [4] traditional and heuristic methods have the capability of dealing with hundreds of variables in minutes with good quality of output, however expanding it to thousands of variables in reasonable time became a challenge, VRP modelling can deal in a max of 400 nodes, thus not being able to deal with problems in large scenarios, such cash supply chain in an outline of a metropolis like São Paulo.

This limitation provides an opportunity for quantum computing, as underscored in [12], to emerge as a potent tool for revolutionizing cash distribution within ATM networks. With operational costs for cash replenishment constituting a substantial portion of total expenses, quantum algorithms present an opportunity to minimize these expenditures significantly [12]. By analyzing vast permutations of cash amounts and routing options across extensive ATM networks, quantum approaches enable the determination of optimal cash quantities at individual ATMs and the most efficient replenishment routes using specific optimization algorithms, such as Variational Quantum Eigensolver (VQE) [13] and Quantum Approximate Optimization Algorithm (QAOA) [14]. This capability not only promises enhanced profitability but also facilitates superior operational efficiency in managing cash supply chains.

Therefore, even quantum solutions for cash supply chain problems are not yet widely discussed, there are consistent alternatives for resolving the vehicle routing problem, which is basically a simplified version of the premises to be solved by the cash distribution problem and an interesting first path to implementing a solution. The research conducted in [15] sheds light on the effectiveness of Quantum walk-based optimization algorithms. Their findings reveal that these algorithms can produce solutions that outperform those achievable through classical random sampling, even when using the same computational resources [15].

In [16] a practical study focused on the use of quantum computing was conducted to solve supply chain problems in general. They outline the formulation of a quadratic problem for better routes in solving the Vehicle routing problem and translate it into Quadratic Unconstrained Binary Optimization (QUBO), which makes the solution possible using quantum optimizers through the company's computer infrastructure and quantum simulators IBM.

II. METHODOLOGY

To conduct the project, the group adopted two methodologies to improve the collective workflow and also understand the individual objectives and results of each member:

1) Agile Software Development [17], which, among its fundamental principles, were considered for the collaborative workflow:
    a) Satisfy the stakeholder through early and continuous delivery of valuable software, meaning the continuous and integrated construction of a valuable application.
    b) Deliver working software frequently, ensuring continuous improvement of processes and alignments between those involved.
    c) Build projects around motivated individuals, understanding and supporting group members for better integration of the parties and better workflow continuity.
    d) The best architectures, requirements, and designs emerge from self-organizing teams, in which by dividing the team into individual tasks, greater collaborative value and creative diversity are added to the process.
    e) At regular intervals, the team reflects on how to become more effective, allowing constant adjustments between the parties.

2) Kanban Methodology [18], for managing individual tasks for each member, generating good indicators of project development from the perspective of everyone in the group.

By following these frameworks, the first for collective agglutination of project development and the second for individual understanding of progress on tasks, the group was able to adjust appropriately for an effective workflow. In this way, the group was divided into three fronts, based on the principles above, for greater efficiency during the project:

1) Felipe Schiavinato was responsible for the theoretical, mathematical, and analytical review of the algorithms to be used in the project, understanding their respective levels of complexity, foundations, characteristics and starting points for implementing the solution from a theoretical point of view, helping to choose appropriate mechanisms for the project progress and choice metrics to determine its success.
2) Matheus Oliveira oversaw the practical application of algorithms and choices reviewed by the group using quantum computing frameworks such as Qiskit, experimenting and qualitatively validating success metrics through code implementations, guiding the group to an empirical visualization of first results regarding the resolution of the problem. He also established the base architecture of the Backend.
3) Nívea de Abreu was tasked with understanding the nuances of the Cash Supply chain problem today, such as cost and distance metrics associated with the problem and adequately delimit the expected format of expected data inputs (geographical points and data intrinsic to them) up to the present moment and their respective consumption flow between backend and frontend, following usability and security standards. She was also responsible for building the frontend visualization interface.

With the appropriate supervision of our advisor, weekly meetings were held, as highlighted in Table III in the subsequent session, in which alignments were made regarding the continuation of the project, dialogue between the parties for connection and integrity of the group.

A. Communication

Table III presents the schedule of meetings held with the project advisor throughout the duration of the project. The table includes the dates, times, and main objectives discussed in each session to ensure alignment and progress in the project phases.

B. Vehicle Routing Problem (VRP) and Cash Distribution Model

1) Problem Definition: The vehicle routing problem (VRP) is a classic combinatorial optimization problem first defined by *[19]*. The VRP seeks to determine the most efficient routes for a fleet of vehicles to deliver goods or services to a set of clients, starting and ending at a central depot. The objective is to minimize the total distance traveled or the overall cost, subject to constraints such as vehicle capacity, time windows, and the number of vehicles available.

A closely related problem is the distribution of cash to a network of ATMs or bank branches. This can be viewed as a variant of the VRP, where the vehicles are armored vans, and the clients are the ATMs or branches requiring cash replenishment. The goal is to optimize the routes of the vans to ensure timely and cost-effective delivery of cash, while considering constraints similar to those in the VRP, such as the capacity of the vans and the demand at each ATM or branch.

The cash distribution problem can be framed as a VRP, with the added complexity of ensuring the security and integrity of the cash being transported. Both problems involve finding the optimal routes for a fleet of vehicles to service a set of clients, with the aim of minimizing costs and meeting specific operational constraints.

2) MTZ Formulation: Several formulations are possible, extending various formulations of the traveling salesman problem in *[6]*. In this context, we present a formulation known as MTZ *[20]*.

a) Cost Function: Let $\boldsymbol{n}$ be the number of clients, indexed as $\mathbf{1}, \dots, \boldsymbol{n},$ and let $\boldsymbol{K}$ represent the number of available vehicles. We define the binary decision variable $\boldsymbol{x_{ij}} \in \{\mathbf{0}, \mathbf{1}\}$, which activates the segment from node $\boldsymbol{i}$ to node $\boldsymbol{j}$ if it is equal to $\mathbf{1}$. The node index ranges from $\mathbf{0}$ to $\boldsymbol{n}$, where $\mathbf{0}$ is conventionally the depot. Consequently, there are twice as many distinct decision variables as there are edges. For instance, in a fully connected graph, there would be $\boldsymbol{n}(\boldsymbol{n} + \mathbf{1})$ binary decision variables.

If there is a link from node $\boldsymbol{i}$ to node $\boldsymbol{j}$, we denote this as $\boldsymbol{i} \sim \boldsymbol{j}$. Furthermore, we consider continuous variables for all nodes $\boldsymbol{i} = \mathbf{1}, \ldots, \boldsymbol{n}$, denoted by $\boldsymbol{u_i}$. These variables are essential in the MTZ formulation of the problem to eliminate sub-tours between clients.

TABLE III
MEETING WITH ADVISOR

Datetime (DD/MM/YY)	Location	Agenda
08/02/24	Remote	Definition of the scope and expectations of the project, elucidating current scenario and quantum algorithms such as Vehicle Routing.
22/02/24	Remote	Understanding of problem constraints, possible data input and analogous implementations for theoretical and practical knowledge, such as defining the use of VRP in the quadratic formulation of the problem and resolution through Hamiltonian representations or Quadratic unconstrained binary optimization (QUBO).
29/02/24	Remote	Alignment to update the contractual situation with a partner company and review existing articles and implementations in the field and correlation with the desired solution.
07/03/24	Remote	Definition of roles for group members, mathematical formulation of the algorithm, backend structure for connection to a quantum computer and creation of a frontend interface for visualizing the calculated route in the form of a map. The feasibility of each task and expectations were validated.
15/03/24	Remote	Checkpoint between group members with a mentor for validation and definition of next steps.
25/03/24	Remote	Presentation for advisor adjusting points of attention before the intermediate evaluation panel.
11/04/24	Remote	Analysis of feedback received, correction of highlighted aspects and definition of next steps, such as executions to be carried out in simulators, without the need for auxiliary hybrid approaches.
18/04/24	Remote	Elucidation of the operational and executive aspects of the project, such as savings generated, use cases and advantages compared to existing approaches for presentation to a Falconi mentor.
25/04/24	Remote	Analysis of the results obtained from full execution on quantum simulators, and parameterization for execution on quantum computers.
02/05/24	Remote	Verification of results obtained on quantum computers (Osaka and Sherbrooke), and definition of edge weight by distance factor and node demand.
09/05/24	Remote	Inquiry into the results obtained via mixed edge definition, and possible corrections in the next steps, in addition to defining the structure of the final report.

We define a cost function (3) that can be interpreted as finding the decision variables within their respective domains that minimize the sum of all corresponding products of weights $\boldsymbol{w_{ij}}$ and decision variables $\boldsymbol{x_{ij}}$, which represent the total cost of the vehicle routes. Mathematically, this formulation can be visualized through the use of matrices, where $\boldsymbol{w}$ and $\boldsymbol{x}$ are matrices that encapsulate the cost structure and routing decisions, respectively, derived from the graphical representation of the problem.

$$f = \min_{x_{ij_{i\sim j}} \in \{0,1\}, u_{i_{i=1,\dots,n}} \in \mathbb{R}} \sum_{i\sim j} w_{ij} x_{ij} \quad (3)$$

where

$\boldsymbol{f}$ minimum value for the cost function;

$\boldsymbol{x_{ij}}$ binary decision variables weather a path is activated or not;

$\boldsymbol{u_i}$ continuous variables associated with each client node $\boldsymbol{i}$. They are used to represent the position of the node in the tour;

$\boldsymbol{w_{ij}}$ weight variable that represents the cost of path from $\boldsymbol{i}$ to $\boldsymbol{j}$.

We also define $\boldsymbol{\delta(i)^+}$ as the set of nodes to which $\boldsymbol{i}$ has a link, meaning that $\boldsymbol{j} \in \boldsymbol{\delta(i)^+} \Leftrightarrow \boldsymbol{i} \sim \boldsymbol{j}$. Similarly, $\boldsymbol{\delta(i)^-}$ represents the set of nodes that are connected to i, such that $\boldsymbol{j} \in \boldsymbol{\delta(i)^-} \Leftrightarrow \boldsymbol{j} \sim \boldsymbol{i}$ i.e. where

$\boldsymbol{\delta(i)^+}$ set of all nodes containing paths coming from $\boldsymbol{i}$ (outgoing);

$\boldsymbol{\delta(i)^-}$ set of all nodes containing paths leading to $\boldsymbol{i}$ (incoming).

These sets are necessary in order to define constraints to our cost function, e.g. in Fig. 3. The path analysis indicates that $\boldsymbol{\delta(i)^+}$ includes the nodes $\boldsymbol{y}$ and $\boldsymbol{z}$, which are represented by paths colored blue originating from node **i**. Additionally, $\boldsymbol{\delta(i)^-}$ comprises the nodes $\boldsymbol{x}$ and $\boldsymbol{y}$, denoted by paths colored red and leading to node **i**.

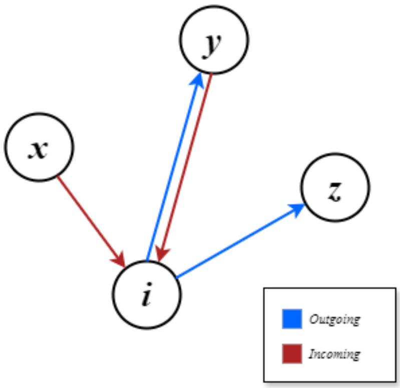

Fig. 3. Graph example illustrating $\delta(i)^+ = y, z$ and $\delta(i)^- = x, y$

b) Node-Visiting: In the VRP, the node-visiting constraint (4) plays a pivotal role in ensuring that each client node is visited exactly once by a vehicle. This constraint is comprised of two equations for every client node $\boldsymbol{i}$ in the set $\{\boldsymbol{1}, \dots, \boldsymbol{n}\}$

$$\sum_{j\in\delta(i)^+} x_{ij} = 1, \quad \sum_{j\in\delta(i)^-} x_{ji} = 1, \quad \forall i \in \{1, \dots, n\} \quad (4)$$

where

$\boldsymbol{\delta(i)^+}$ set of all nodes containing paths coming from $\boldsymbol{i}$ (outgoing);

$\boldsymbol{x_{ij}}$ binary decision variables weather a path is activated or not;

$\boldsymbol{\delta(i)^-}$ set of all nodes containing paths leading to $\boldsymbol{i}$ (incoming).

The outgoing constraint $\sum_{\boldsymbol{j\in\delta(i)^+}} \boldsymbol{x_{ij}} = \boldsymbol{1}$ ensures that there is exactly one outgoing link from each client node $\boldsymbol{i}$ that is part of the solution, indicating that each client is departed from exactly once. The incoming constraint $\sum_{\boldsymbol{j\in\delta(i)^-}} \boldsymbol{x_{ji}} = \boldsymbol{1}$ ensures that there is exactly one incoming link to each client node $\boldsymbol{i}$, signifying that each client arrived at exactly once.

By enforcing the node-visiting constraint, the VRP formulation guarantees that each client is visited exactly once by some vehicle. This is a fundamental requirement in the VRP, as it prevents scenarios where a client is either skipped or visited multiple times, thereby ensuring that the solution is feasible and meets the basic expectations of the routing problem.

c) Depot-Visiting: In the VRP formulation, the depot-visiting constraints (5) are critical for ensuring that the fleet of vehicles is utilized correctly.

$$\sum_{i\in\delta(0)^+} x_{0i} = K, \quad \sum_{j\in\delta(0)^+} x_{j0} = K \tag{5}$$

where

- $\boldsymbol{\delta(0)^+}$ is the set of nodes that can be directly reached from the depot (node 0);
- $\boldsymbol{x_{0i}}$ binary decision variables indicating whether a vehicle travels from the depot to node $\boldsymbol{i}$;
- $\boldsymbol{K}$ is the number of available vehicles in the fleet;
- $\boldsymbol{x_{j0}}$ binary decision variables indicating whether a vehicle travels from node $\boldsymbol{j}$ to the depot.

First Constraint $\sum_{\boldsymbol{i\in\delta(0)^+}} \boldsymbol{x_{0i}} = \boldsymbol{K}$ ensures that exactly $\boldsymbol{K}$ vehicles leave the depot to start their routes. Each vehicle is assigned to a different route, and no vehicle remains idle at the depot. In other words, the total number of routes originating from the depot equals the number of available vehicles.

Second Constraint $\sum_{\boldsymbol{j\in\delta(0)^+}} \boldsymbol{x_{j0}} = \boldsymbol{K}$ ensures that exactly $\boldsymbol{K}$ vehicles return to the depot after completing their routes. It guarantees that every vehicle that leaves the depot must eventually return, ensuring that there are no stranded vehicles or unfinished routes.

The depot-visiting constraints are essential for maintaining the balance between the number of vehicles leaving and returning to the depot. They ensure that the fleet's capacity is fully utilized without exceeding the available resources. These constraints also contribute to the formulation's integrity by preventing scenarios where vehicles are duplicated, or routes are left incomplete.

d) Sub-Tour: The sub-tour elimination constraints (6) and (7), ensures each vehicle's route constitutes a single, continuous tour that encompasses all designated clients. These constraints effectively prevent the emergence of multiple, disjoint sub-tours, which, intrinsically cost less by failing to traverse all required nodes. Integrating these constraints into the formulation guarantees that the solution adheres to the practical demand of complete route coverage for each vehicle.

$$u_i - u_j + Qx_{ij} \leq Q - q_j,\ \forall i \sim j,\ i,j \neq 0 \tag{6}$$

$$q_i \leq u_i \leq Q,\ \forall i,\ i \neq 0 \tag{7}$$

where

- $\boldsymbol{u_i}$ continuous variables associated with each client node $\boldsymbol{i}$. They are used to represent the position of the node in the tour;
- $\boldsymbol{u_j}$ continuous variables associated with each client node $\boldsymbol{j}$. They are used to represent the position of the node in the tour;
- $\boldsymbol{x_{ij}}$ binary decision variable that equals $\mathbf{1}$ if a vehicle travels from node $\boldsymbol{i}$ to node $\boldsymbol{j}$, and $\mathbf{0}$ otherwise;

$\boldsymbol{Q}$ large positive number;

$\boldsymbol{q_j}$ positive numbers representing the demand at each client node $\boldsymbol{j}$;

The main purpose of these constraints is to prevent the formation of sub-tours among the clients. By using the continuous variables $\boldsymbol{u_i}$, the constraints ensure that if there is a route from node $\boldsymbol{i}$ to node $\boldsymbol{j}$ ($\boldsymbol{x_{ij} = 1}$), then the position of node $\boldsymbol{j}$ in the tour ($\boldsymbol{u_j}$) must be greater than the position of node $\boldsymbol{i}$ ($\boldsymbol{u_i}$) by at least the demand at node $\boldsymbol{j}$ ($\boldsymbol{q_j}$). This logic helps in maintaining a single continuous tour.

The constraints $\boldsymbol{q_i \leq u_i \leq Q}$ ensure that the values of the continuous variables are within reasonable bounds, reflecting the position of each node in the tour.

By incorporating the sub-tour elimination constraints into the VRP formulation, the solution is guaranteed to be a single tour that covers all clients, rather than multiple disjointed sub-tours. This is essential for the practical applicability of the solution, as it ensures that each vehicle completes a full tour starting and ending at the depot, without leaving any client unvisited or visiting any client more than once.

3) Classical Solution: The VRP can be effectively addressed through classical optimization methods, with CPLEX being a prominent tool in this domain. CPLEX is a sophisticated solver that applies a branch-and-bound-and-cut algorithm to find approximate solutions to mixed-integer linear programs (MILPs), a category under which the VRP falls when formulated appropriately *[21]* .

a) CPLEX: CPLEX's branch-and-bound-and-cut algorithm is particularly well-suited for tackling MILPs like the VRP. This method systematically explores the solution space by branching on decision variables, bounding the objective function to eliminate suboptimal solutions, and cutting off regions of the solution space that do not contain optimal solutions.

b) Decision Variables Representation: In the context of the VRP, decision variables are binary and indicate whether a particular route between two nodes is part of the solution. To facilitate computational processing, these variables are typically consolidated into a vector as in (8) where $\boldsymbol{z} \in \{\boldsymbol{0}, \boldsymbol{1}\}^{\boldsymbol{N}}$ and $\boldsymbol{N = n(n+1)}$.

$$z = \left[x_{01}, x_{02}, \dots, x_{10}, x_{12}, \dots, x_{n(n-1)}\right]^T \quad (8)$$

where

$\boldsymbol{z}$ vector containing all possible values for decision variable $\boldsymbol{x}$;

$\boldsymbol{x}$ decision variable of all possible edges.

This vectorization of decision variables streamlines their manipulation within the solver and provides a structured representation of the solution space.

c) Problem Scaling: It is noteworthy that the dimensionality of the problem, as represented by the number of decision variables, scales quadratically with the number of nodes (clients plus the depot). This quadratic scaling implies that the complexity of the VRP increases significantly as the number of clients grows, presenting a challenge for classical solvers like CPLEX, particularly for large-scale instances.

d) Optimal Solution and Cost: The outcome of employing CPLEX for solving the VRP is twofold: the optimal solution vector $\boldsymbol{z}^*$ and the associated optimal cost $\boldsymbol{f}^*$. The solution vector delineates the routes that constitute the optimal solution, while the optimal cost represents the minimum total distance or cost achievable under the given constraints.

e) Conclusion: In conclusion, the classical solution to the VRP using CPLEX involves formulating the problem as a MILP, organizing the decision variables into a compact vector, and applying a branch-and-bound-and-cut algorithm to approximate the optimal solution. Although effective for moderate-sized problems, the scalability of this approach becomes a concern for larger instances, underscoring the necessity for advanced techniques or heuristics to efficiently tackle the VRP.

4) Quantum Solution: In this section, we explore a quantum computing approach to solving the Vehicle Routing Problem (VRP), utilizing the Quantum approximate optimization algorithm (QAOA) proposed by [19]. This method involves a combination of classical and quantum computing steps and employs the Variational Quantum Eigensolver (VQE) technique. It is important to note that due to the limited depth of quantum circuits used in this approach, the solution obtained is heuristic, making it challenging to discuss potential speed-ups. Heuristic methods are valuable for investigating certain classes of problems, especially those of significant practical importance.

In Fig. 4, we illustrate our approach to solving the VRP. Initially, the mathematical model established in Section II-B2 (MTZ) is converted to either the Ising Hamiltonian or QUBO (Quadratic Unconstrained Binary Optimization) formulation. These approaches are further developed in Sections II-B4a and II-B4b, respectively. From a mathematical perspective, both formulations are equivalent. However, QUBO is more straightforward for code implementation, and thus, it is used in our final application. On the other hand, the Ising Hamiltonian facilitates the transition from the MTZ model (3), making it easier to understand mathematically. Therefore, this report will focus on the mathematical details of the Ising Hamiltonian. Finally, we can send the data for the quantum algorithms to process.

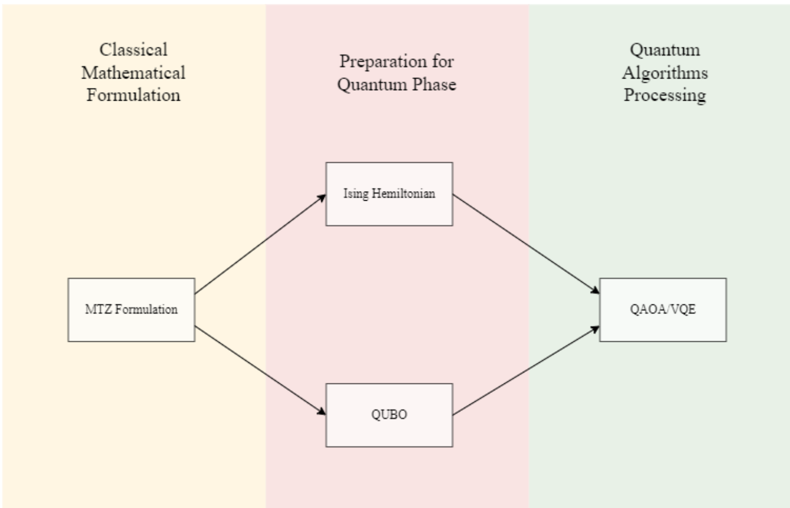

Fig. 4. Quantum solution diagram

The algorithm begins with a series of preparation steps:

1) **Problem Transformation:** The combinatorial VRP is first transformed into a binary polynomial optimization problem with only equality constraints.
2) **Ising Hamiltonian Mapping:** The transformed problem is then mapped into an Ising Hamiltonian ($\boldsymbol{H}$) for variables $\boldsymbol{z}$ and basis $\boldsymbol{Z}$. In this context, $\boldsymbol{z}$ represents the vector of binary variables or spins in the Ising model, where each component can take a value of $+1$ or -1. The basis $\boldsymbol{Z}$ refers to the computational basis in quantum computing, consisting of the eigenstates of the Pauli-Z operator, which correspond to the "spin up" and "spin down" states of the Ising spins. This mapping may involve penalty

methods if necessary to ensure the problem's compatibility with the Ising model. (Section II-B4c)

3) **Quantum Circuit Depth:** The depth of the quantum circuit, denoted by $\boldsymbol{m}$, is chosen. The depth refers to the number of layers of quantum gates applied in the circuit. A deeper circuit can represent more complex functions but may be more challenging to implement on near-term quantum hardware due to noise and decoherence. The depth can be adjusted adaptively based on the problem's complexity and the quantum hardware's capabilities.
4) **Trial Function:** A set of controls $\boldsymbol{\theta}$ is selected, and a trial wavefunction $|\boldsymbol{\psi}(\boldsymbol{\theta})\rangle$ [7] is constructed using a quantum circuit composed of controlled-phase (C-Phase) gates and single-qubit Y rotations. The trial wavefunction is an ansatz, or a guess, for the ground state of the quantum system described by the Ising Hamiltonian. The C-Phase gates are two-qubit gates that introduce entanglement between qubits, and single-qubit Y rotations are gates that rotate qubits around the Y-axis of the Bloch sphere, parameterized by the components of $\boldsymbol{\theta}$. These components $\boldsymbol{\theta}$ are the variational parameters that are optimized during the algorithm to minimize the energy of the trial wavefunction.

In the next steps of the algorithm, the trial wavefunction is used to evaluate the cost function and iteratively update the controls $\boldsymbol{\theta}$ to find the optimal solution to the VRP.

The algorithm proceeds with the following steps:

1) **Expectation Value Evaluation**: The cost function $\boldsymbol{C}(\boldsymbol{\theta}) = \langle\boldsymbol{\psi}(\boldsymbol{\theta})|\boldsymbol{H}|\boldsymbol{\psi}(\boldsymbol{\theta})\rangle$ is evaluated by sampling the outcome of the quantum circuit in the Z-basis and summing the expectation values of the individual Ising terms.
2) **Classical Optimization**: A classical optimizer is used to update the set of controls $\boldsymbol{\theta}$ to minimize the cost function $\boldsymbol{C}(\boldsymbol{\theta})$.
3) **Convergence Check**: The algorithm iterates until $\boldsymbol{C}(\boldsymbol{\theta})$ reaches a minimum, indicating that a solution $\boldsymbol{\theta}^*$ close to the optimal has been found.
4) **Solution Sampling**: The final set of controls $\boldsymbol{\theta}$ is used to generate a set of samples from the distribution $|\langle\boldsymbol{z_i}|\boldsymbol{\psi}(\boldsymbol{\theta})\rangle|^2$ for all qubits $\boldsymbol{i}$, providing the solution to the VRP.

The trial wavefunction used in the VQE is given by (9). The choice of the trial wavefunction is a crucial parameter in the algorithm, as it influences the efficiency and accuracy of the solution.

[7] If you are not familiar with $\langle\boldsymbol{bra}|\boldsymbol{ket}\rangle$ or Dirac notation, for this section you should see [46].

$$|\psi(\theta)\rangle = \left[U_{single}(\theta)U_{entangler}\right]^m |+\rangle \quad (9)$$

where

$\|\boldsymbol{\psi}(\boldsymbol{\theta})\rangle$	trial wavefunction;
$\boldsymbol{U_s}$	$= \prod_{i=1}^{N} \boldsymbol{Y}(\boldsymbol{\theta_i})$;
$\boldsymbol{N}$	number of qubits;
$\boldsymbol{\theta}$	controls or classical parameters;
$\boldsymbol{U_e}$	collection of C-Phase gates that fully entangle the qubits;
$\boldsymbol{m}$	depth of the quantum circuit;
$\|+\rangle$	plus state[8].

a) Ising Hamiltonian: The Ising Hamiltonian is a specific version of the Hamiltonian applied in the Ising model, both concepts are included in Appendixes B and C, respectively. It describes the energy of a system of spins on a lattice, accounting for the interactions between neighboring spins and the influence of an external magnetic field. The Ising Hamiltonian is a crucial tool for examining magnetic properties and phase transitions in statistical mechanics *[22]*.

In order to construct the Ising Hamiltonian for the Vehicle Routing Problem (VRP), we simplify the problem by considering cases where $\boldsymbol{K = n - 1}$. This assumption allows us to ignore the sub-tour elimination constraints, (6) and (7), as each vehicle will visit exactly one client before returning to the depot. Consequently, our problem consists solely of equality constraints, (4) and (5), which are necessary for mapping the problem to an Ising Hamiltonian.[9] From MTZ formulation we model (9) as an Ising Hamiltonian.

$$H = \sum_{i\sim j} w_{ij}x_{ij} + A\sum_{i\in\{1,\dots,n\}}\left(\sum_{j\in\delta(i)^+} x_{ij} - 1\right)^2 + A\sum_{i\in\{1,\dots,n\}}\left(\sum_{j\in\delta(i)^-} x_{ji} - 1\right)^2$$
$$+A\left(\sum_{i\in\delta(0)^+} x_{0i} - K\right)^2 + A\left(\sum_{j\in\delta(0)^+} x_{j0} - K\right)^2 \quad (10)$$

where

$\boldsymbol{x_{ij}}$	binary decision variables weather a path is activated or not;
$\boldsymbol{w_{ij}}$	weight variable that represents the cost of path from $\boldsymbol{i}$ to $\boldsymbol{j}$;
$\boldsymbol{A}$	large positive constant used to enforce the equality constraints;
$\boldsymbol{K}$	is the number of available vehicles in the fleet.

In this formulation:

1) The first term represents the total cost of the routes, (3) without the minimization parameter, with $\boldsymbol{w_{ij}}$ being the cost of traveling from node $\boldsymbol{i}$ to node $\boldsymbol{j}$.

---

[8] Defined as $|+\rangle = \frac{\mathbf{1}}{\sqrt{\mathbf{2}}}(|\mathbf{0}\rangle + |\mathbf{1}\rangle)$

[9] The proposed solution is not limited to scenarios where $\boldsymbol{K = n - 1}$. To formulate the Ising Hamiltonian, it is necessary to convert all inequality constraints into equality constraints. For clarity in the mathematical transitions, we will focus on cases that exclude the inequality constraints, specifically sub-tours (6) and (7). In our implementation, we will address this issue by employing slack variables. The detailed methodology for this approach will be discussed in Sections II-B4b and III-B.

2) The second and third terms are node-visiting constraints for clients (4), ensuring that each client is visited exactly once.
3) The fourth and fifth terms are depot-visiting constraints (5), ensuring that exactly $\boldsymbol{K}$ links connect to and from the depot.
4) By squaring the constraints, we ensure that the penalty is always non-negative and becomes significant if the constraints are violated.

b) Quadratic Unconstrained Binary Optimization (QUBO): QUBO problems are commonly employed, with variables assuming values of 1 (TRUE) and 0 (FALSE). A QUBO problem is characterized by an upper-diagonal matrix $\boldsymbol{Q}$, which is an $\boldsymbol{N} \times \boldsymbol{N}$ upper-triangular matrix of real weights, and a vector of binary variables $\boldsymbol{x}$. The objective is to minimize the function (11), the minimization process is written in (12).

$$f(x) = \sum_{i} Q_{i,i}\, x_i + \sum_{i<j} Q_{i,j}\, x_i x_j \tag{11}$$

where

$\boldsymbol{f}$ cost function;

$\boldsymbol{x}$ binary decision variables weather a path is activated or not;

$\boldsymbol{Q_{i,i}}$ linear coefficients and the nonzero off-diagonal terms;

$\boldsymbol{Q_{i,j}}$ denote the quadratic coefficients.

$$\min_{x \in \{0,1\}^n} x^T\, Qx \tag{12}$$

where

$\boldsymbol{x}$ binary decision variables weather a path is activated or not;

$\boldsymbol{Q}$ denote the quadratic coefficients.

In Fig 5., we can see how the cost function and constraints are written in QUBO for [16] implementation.

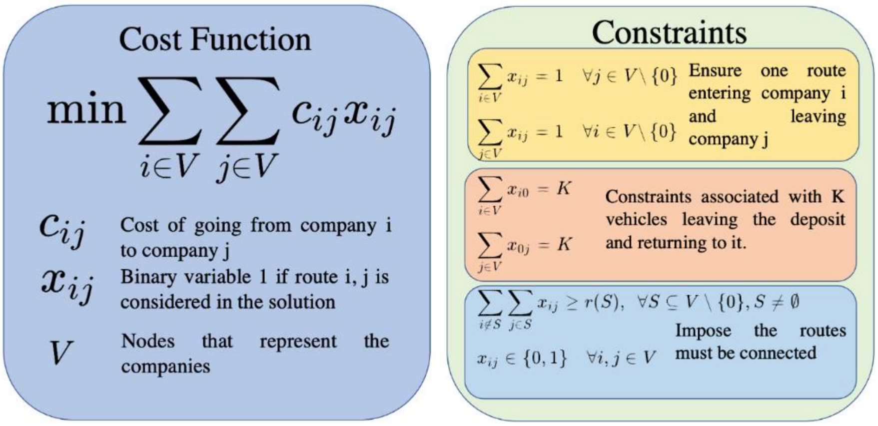

Fig. 5. Cost function and constraints [16]

The constraints of the VRP are incorporated into the QUBO cost function as penalties to ensure that any solution violating these constraints will have a higher cost and thus be less favorable. This approach allows the problem to be solved using quantum computing techniques by finding the configuration of binary variables $\boldsymbol{x_{ij}}$ that minimizes the cost function while satisfying the constraints.

In the context of solving the Vehicle Routing Problem (VRP) via quantum computing, one must first construct a quadratic program that represents the model. Utilizing the Qiskit Optimization library [23], this quadratic program can be translated into a Quadratic Unconstrained Binary Optimization (QUBO) representation. Specifically, when dealing with inequality constraints within the VRP formulation, additional variables are typically introduced to convert inequalities into equalities. These additional variables are known as slack variables, which serve to accommodate any excess or deficiency in the constraints' values [24].

For the VRP in question, the introduction of slack variables to account for inequality constraints has led to a substantial increase in the number of variables needed to represent the problem. The original count of variables has expanded from 52 to 74, demonstrating the complexity introduced when representing the VRP as a QUBO problem suitable for quantum computation. This expansion is necessary to ensure that the QUBO model accurately encapsulates all aspects of the original problem, including the inequality constraints.

c) From Hamiltonian to Quadratic Programming (QP)[10] *Formulation:* In the vector $\boldsymbol{z}$ and for a complete graph ($\boldsymbol{\delta(i)^{+} = \delta(i)^{-} = \{0, 1, \dots, i-1, i+1, \dots, n\}}$) we can imply (13) from previous (10).

$$H = \min_{z \in \{0,1\}^{n(n+1)}} w^T z + A \sum_{i \in \{1,\dots,n\}} (e_i \otimes 1_n^T z - 1)^2 + A \sum_{i \in \{1,\dots,n\}} (v_i^T z - 1)^2 + A((e_0 \otimes 1_n)^{Tz} - K)^2 + A(v_0^{Tz} - K)^2 \tag{13}$$

where

$\boldsymbol{H}$	minimization of the energy (cost) of the system;
$\boldsymbol{z}$	decision variable vector;
$\boldsymbol{w}$	weighs vector;
$\boldsymbol{n}$	number of nodes;
$\boldsymbol{A}$	large positive number;
$\boldsymbol{e_i}$	standard basis vector with all zeros except for a $\mathbf{1}$ in the $\boldsymbol{i-th}$ position;
$\boldsymbol{v}$	auxiliar vector for constraints;
$\boldsymbol{K}$	number of vehicles.

$$\min_{z \in \{0,1\}^{n(n+1)}} z^T Qz + g^T z + c \tag{14}$$

where

$\boldsymbol{z}$	decision variable vector;
$\boldsymbol{Q}$	matrix (will be defined IN (12));
$\boldsymbol{n}$	number of nodes;
$\boldsymbol{g}$	linear function represented by vector defined in (13);
$\boldsymbol{c}$	constant (14).

[10] "Quadratic programming (QP) is minimizing or maximizing an objective function subject to bounds, linear equality, and inequality constraints" [47].

First term is the quadratic term (15). $\boldsymbol{Q}$ is constructed using tensor products. The tensor product $\boldsymbol{e_i} \otimes \boldsymbol{1_n}$ creates a matrix where each row corresponds to a node-visiting constraint for a particular node. The term $\boldsymbol{v_i v_i^T}$ represents the depot-visiting constraints. The matrix $\boldsymbol{Q}$ encapsulates the quadratic penalties for violating these constraints.

$$Q = A \sum_{i\in\{0,1,\dots,n\}} [(e_i \otimes 1_n)(e_i \otimes 1_n)^T + v_i v_i^T] \tag{15}$$

The vector $\boldsymbol{g}$ in (16) combines the weights from the cost function with linear penalties from the constraints. The terms involving $\boldsymbol{e_i} \otimes \boldsymbol{1_n}$ and $\boldsymbol{v_i}$ correspond to penalties for not visiting each client and the depot exactly once, respectively.

$$g = w - 2A \sum_{i\in\{1,\dots,n\}} [(e_i \otimes 1_n) + v_i] - 2AK[(e_0 \otimes 1_n) + v_0] \tag{16}$$

The constant $\boldsymbol{c}$ in (17) accounts for the fixed penalties associated with the number of clients ($\boldsymbol{n}$) and the number of vehicles ($\boldsymbol{K}$).

$$c = 2An + 2AK^2 \tag{17}$$

In this QP formulation, the tensor products are used to construct matrices and vectors that represent the constraints of the VRP in a way that is suitable for optimization using quantum algorithms like VQE. The quadratic term $\boldsymbol{Q}$ ensures that the solution satisfies the constraints, while the linear term $\boldsymbol{g}$ and the constant term $\boldsymbol{c}$ together minimize the total cost of the routes. Now, the model is ready to be processed by the VQE.

d) Variational Quantum Eigensolver (VQE): The Variational Quantum Eigensolver (VQE) is a hybrid quantum-classical algorithm that leverages the variational principle to estimate the ground state energy of a Hamiltonian, a fundamental problem in quantum chemistry and condensed matter physics *[13]*. The VQE algorithm is especially suitable for Noisy Intermediate-Scale Quantum (NISQ) devices due to its resilience to certain types of quantum noise and its ability to provide meaningful results with a limited number of qubits and quantum gates *[25]*

In the VQE framework, a trial wavefunction $|\boldsymbol{\psi}(\boldsymbol{\theta})\rangle$, parameterized by a set of classical parameters $\boldsymbol{\theta}$, is prepared on a quantum computer. The expectation value of the Hamiltonian $\boldsymbol{H}$ with respect to this trial wavefunction, $\langle\boldsymbol{\psi}(\boldsymbol{\theta})|\boldsymbol{H}|\boldsymbol{\psi}(\boldsymbol{\theta})\rangle$, is then measured. According to the variational principle, this expectation value provides an upper bound to the true ground state energy $\boldsymbol{E_0}$ of the Hamiltonian expressed in (18).

$$E_0 \leq \langle\psi(\theta)|H|\psi(\theta)\rangle \tag{18}$$

The objective of the VQE algorithm is to minimize the expectation value $\langle\boldsymbol{\psi}(\boldsymbol{\theta})|\boldsymbol{H}|\boldsymbol{\psi}(\boldsymbol{\theta})\rangle$ with respect to the parameters $\boldsymbol{\theta}$, thereby approximating the ground state energy as closely as possible within the chosen ansatz for the trial wavefunction. This optimization is typically carried out using classical optimization algorithms, with the quantum computer employed to prepare the trial wavefunction and measure the expectation value of the Hamiltonian.

The effectiveness of the VQE algorithm depends on the choice of the trial wavefunction ansatz, the efficiency of measuring the expectation value of the Hamiltonian, and the performance of the classical optimization algorithm. Various strategies have been developed to tackle these challenges, making VQE a promising approach for solving quantum many-body problems on near-term quantum computers [26] [27].

C. Data Collection

For security aspects and greater vastness in experimentation, we decided to use generic and randomized data, specifically points, latitude and longitude, spread across the city of São Paulo, which theoretically represent agencies that have each a demand for money for replenishment. With this guideline, we continued with the project implementation steps.

As can be seen in Fig. 6, the data generated randomly by the city of São Paulo aims to create theoretical problem situations that allow the calculation of appropriate routes according to the problem guidelines. Basically, after we get central coordinates of the city via Python library *Geopandas*, we produce the origin point (simulating the initial stock of trucks leaving) and after, the data points (latitude, longitude and amount of money needed) around the city, simulating the ATM's.

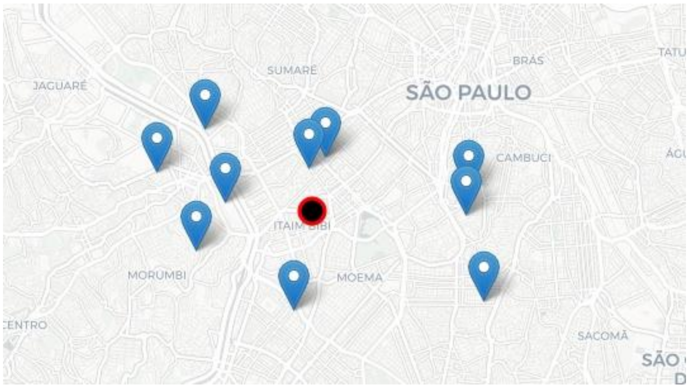

Fig. 6. Visualization of randomized data generated which allowed the group to work on implementations.

D. Data Analysis

With the data generated randomly at runtime, graphs with normalized distance were generated for better later computational efficiency. The following normalization formula was used around the maximum coordinate (represented by a tule of latitude and longitude) in (19).

$$Normalized\ coordinate_{ij} = \frac{Coordinate_{ij} - Origin_j}{max_k\left(Coordinate_{kj}\right)} \tag{19}$$

Where:

$$Coordinate_{ij} = (latitude, longitude)$$

And the distance (or weight), between two generic nodes ***m, n*** were expressed in (20).

$$distance_{mn} = \sqrt{\begin{array}{c}(normalized\ latitude_m - normalized\ latitude_n)^2 + \\ (normalized\ longitude_m - normalized\ longitude_n)^2\end{array}} \tag{20}$$

To form this graph, all geographic points on the map were received as input, and edges were generated following the conditions:

1) There is always an edge between the origin node (represented in red) and any other node in the graph, guaranteeing the possibility of a route, which always starts from the origin, starting at any node.

2) There is an edge between a given node $\boldsymbol{i}$ and a node $\boldsymbol{j}$, where $\boldsymbol{i}, \boldsymbol{j} \neq \mathbf{0}$ (0 representing the origin node), if and only if their weight (or normalized distance, if $\boldsymbol{p} = \mathbf{1}$ in (23)) is not greater than a threshold (generally assumed by the value of 1), ensuring that there are no edges between very distant nodes and avoiding costly routes in advance, which should be ignored in favor of intermediate routes that pass through other nodes instead of this connection.

Based on these principles, we can generate an input data graph with a much lower complexity, and without significant loss of information, since by eliminating costly edges in advance, the quality of the input data is improved by saving algorithms from analyzing subcases. of edges with weight above the established threshold. This process of reducing the complexity of the input set is shown in Fig.7.

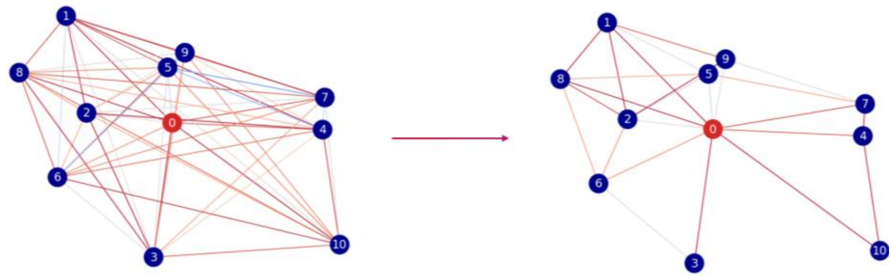

Fig. 7. Graph representing the allowed edges between the geographical points according premises, reducing its complexity.

Each data point consists of a tuple with values of latitude, longitude and cash demand, and is going to be used by the mathematical steps of the algorithm, specially transforming it in a mathematical quadratic program. The first experimental graph consists of 10 points and 26 edges (in this specific case) that represent possible paths allowed to delimit the route to be chosen based on parameters and restrictions.

Other experimentations were made after, with different number of nodes, and with different cost estimations for the edges weights. We utilized hybrid approaches, mixing distance and cash demanded for each node, allowing the generation of different contexts and situations for better analysis and experimentation of the solution. This way, based on the cash demand for each demand, we also normalized its value in (21).

$$Cash\ demand\ normalized_i = \frac{Cash\ demand_i}{max\ (Cash\ demand)} \quad (21)$$

And cash demand between two nodes $\boldsymbol{m}, \boldsymbol{n}$, representing the cash amount necessary to be available in the truck to go from a node $\boldsymbol{m}$ immediately to a node $\boldsymbol{n}$, without the need to return to the origin in (22).

$$Cash\ amount_{mn} = |Cash\ demand\ norm._m - Cash\ demand\ norm._n| \quad (22)$$

Which allowed us to create miscellaneous approaches to the concept of weight between edges in later experimentations (23), with this data in hand, its consumption and processing become more appropriate via studied algorithms and subsequently constructed application structure.

$$weight_{mn} = p \cdot |distance_{mn}| + (1 - p) \cdot |Cash\ amount_{mn}|, for\ p \leq 1 \quad (23)$$

III. EXPERIMENTS AND RESULTS

With the theoretical framework established in terms of algorithms, quadratic formulations and restrictions inherent to the problem, we carried out computational experiments and comparisons between already pre-established implementations such as classical solutions through CPLEX to check its limitations in dealing with complex and larger system as mentioned by [4], and Quantum solutions as [16] and [28].

A. Classical solution limits

To verify the feasibility of traditional algorithms, we initially generate different sets of coordinates with less complexity until larger systems to check the capacity of the solution. Following the conditions established in the session of *Data Analysis* to form the input graph (setting threshold to decide if an edge is feasible or not), simplifying its complexity, it's possible to see the relation between the number of points (coordinates) generated and the binary variables (valid edges) needed to solve the problem in Fig. 8.

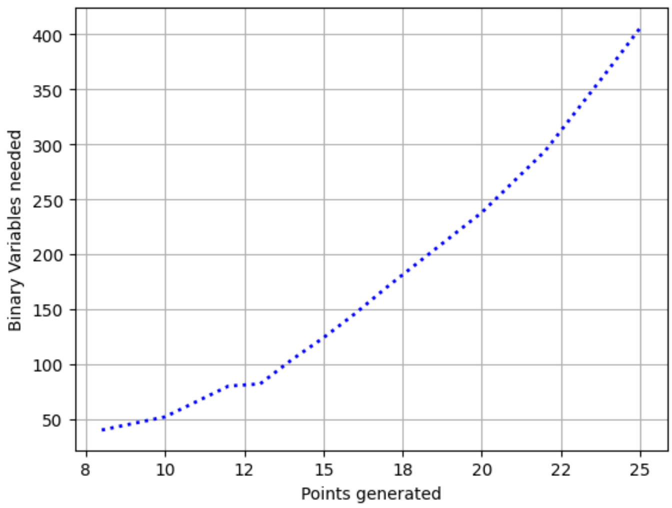

Fig. 8. Plot illustrating the growth of binary variables needed per points in the system.

With different sets of complexity for the input variables, we tried to solve these systems locally. As observed in the literature, as the system grows in terms of variables, the time and space needed to solve the problem grows exponentially, elucidating that in sets of more than 400 nodes, the need for increasingly powerful hardware becomes a notable limitation. In this experiment, a machine with 16 GB of RAM, can deal with a max of 406 variables, extrapolating it's capacity after reaching a number of almost 450 variables in Fig. 9 and Fig 10.

This exponential behavior was also noted for the time needed to solve these sets of nodes, making it clear that for larger scales, the time to obtain optimal solutions tends to become unfeasible.

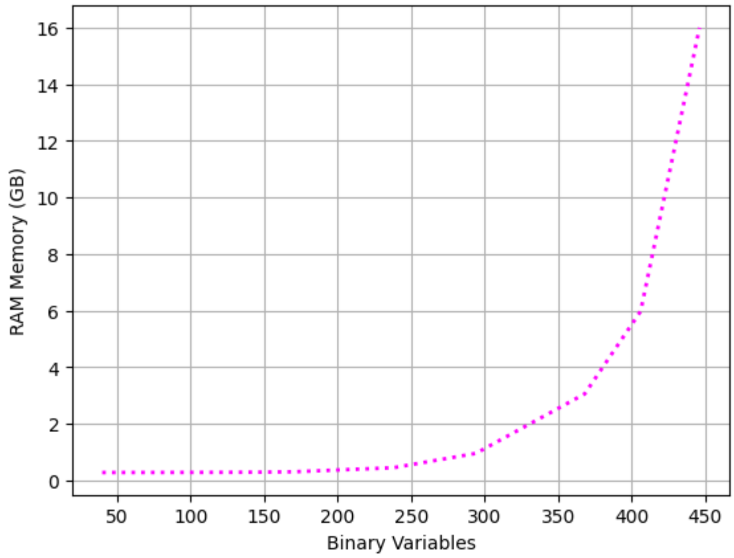

Fig. 9. Plot illustrating the growth RAM memory needed per variables in the system.

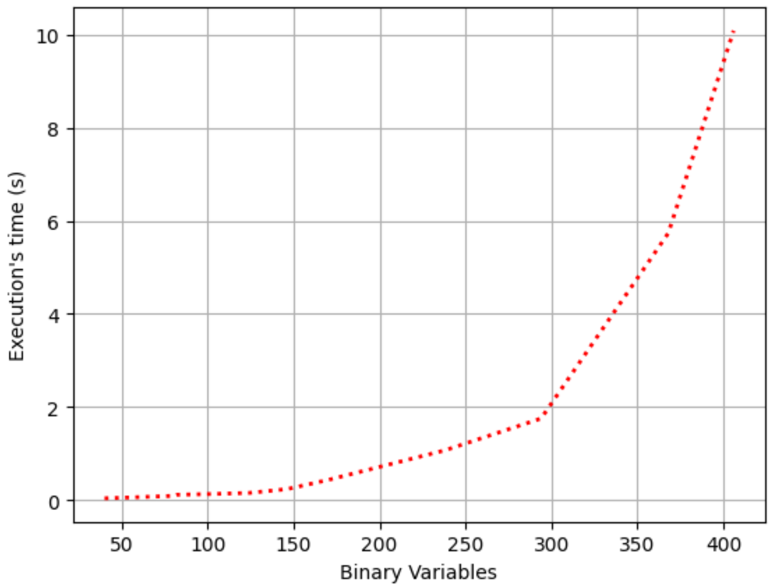

Fig. 10. Plot illustrating the growth of Execution's time needed per variables in the system.

B. Hybrid implementation

Seeking to carry out initial experiments with the input data (subject to the restrictions and processing), a hybrid execution was initially planned. This way, we solve part of the problem with CPLEX and seeking subsets of variables to be solved quantumly to verify the best archetype of quantum algorithms, such as VQE and QAOA, and respective hyperparameters.

To create the Quadratic Problem, we use the Model class from the docplex.mp module, which acts as a factory to create optimization objects, decision variables, and constraints [29]. Due to the representation stated in the graph in Fig. 7 in which we have 10 nodes and 26 edges between them (subject to the conditions elucidated previously and assuming $\boldsymbol{p} = \mathbf{1}$ for (23)), we have around 52 binary variables for this quadratic program, arising from the fact that the edges, so far, they are not directed (movement in both directions freely), allowing the algorithm to have the flexibility to find the lowest cost routes without worrying about the direction from one node to another.

A quantity of 4 trucks was arbitrarily chosen (and the number of routes to be generated as a result). These parameters are used to generate the restrictions established in the previous session of the quadratic program and VRP specifications. As we will use quantum optimization algorithms, we follow the standard input format for these types of problems to be compatible with the algorithms to be used in the experimentation phase, VQE and QAOA. Therefore, we must transform the quadratic representation of the program obtained in the previous step to QUBO, which, as explained previously, helps in the optimization of binary problems. This implementation was carried out by the [23] module, which maps these representations to Ising Hamiltonians, which can later be processed by the algorithms mentioned above [23].

In solving this problem with QUBO, we have inequality constraints that require extra variables, known as slack variables. These arise with the aim of transforming inequality restrictions into equality restrictions. The slack variable $\boldsymbol{S}$ shows how much the left side of the inequality is smaller than the right side. If the left side is smaller, then $\boldsymbol{S}$ will be a positive number, which is the gap between the two sides [30]. We can see it, without loss of generality, by the equation transformation (24), where $\boldsymbol{x_i}$ are binary variables, $\boldsymbol{w_i}$ are arbitrary weights in the optimization problem and $\boldsymbol{K}$ is a reference constant for inequality

$$w_0x_0 + w_1x_1 + w_2x_2 \leq K \longrightarrow w_0x_0 + w_1x_1 + w_2x_2 + S = K \tag{24}$$

where

$\boldsymbol{w_i}$ arbitrary weights;

$\boldsymbol{x_i}$ binary variables;

$\boldsymbol{K}$ reference constant for inequality;

$\boldsymbol{S}$ slack variable $\boldsymbol{0} \leq \boldsymbol{S} \leq \boldsymbol{K}$.

That way, according [30], a slack variable also has to be represented in binary form (25).

$$S = \sum_{m=0}^{N-1} 2^m s_m \tag{25}$$

where

$\boldsymbol{S}$ slack variable

$\boldsymbol{N}$ $= \lfloor \boldsymbol{log_2}(\boldsymbol{max\,S}) \rfloor + \boldsymbol{1}$;

Therefore, generating the need for more N binary variables to the problem, in which replacing equation (25) in (24), give (26).

$$w_0x_0 + w_1x_1 + w_2x_2 + 2^0 s_0 + \cdots + 2^{N-1} s_{N-1} = K \tag{26}$$

Assuming, without loss of generality, that $\boldsymbol{s_0} = \boldsymbol{x_3}$, it means, a new binary variable in the system, this points to the fact that instead of 3 variables as previously in the inequality, we now have $\boldsymbol{N+3}$ variables in the QUBO equality representation, which facilitates the minimization, in our case (27).

$$w_0x_0 + w_1x_1 + w_2x_2 + w_3x_3 + \cdots + w^{N+2}x_{N+2} = K \tag{27}$$

For the case experimented, in which we had 52 binary variables, when transforming to QUBO representation, this number was increased to 74 variables. With this scenario represented, another point of attention to pay attention to is the number of qubits that we can simulate.

As local execution (on members' machines) is costly, the group's computers do not have enough capacity (a fact verified by empirical attempts) to process all 52 variables, not including slack variables, and the default maximum number of qubits allowed for assembling the circuit in the coupling map for IBM QASM is 32 and IBM MPS has memory limitations, the following steps were divided for primary exploration of the performance of quantum algorithms, generating a hybrid approach initially:

1) Solve the problem classically with CPLEX using the QUBO representation and save the optimization weights (solution obtained) of binary and slack variables for later combination with quantum algorithms, with fewer variables to be solved.
2) Reduce the problem to be solved with 7 quantum variables, qubits, to be solved both locally and by quantum simulators, helping to elucidate more suitable algorithms and optimizers, and complete the solution with the 45 binary variables and 22 slack variables, already solved by CPLEX.
3) With the combination of algorithm and optimizer that generates the best results in the previous scenario, remap the problem to 25 quantum variables, qubits, and 27 known classical variables and 22 slack variables for greater use of quantum capacity and remove the bias from the solution arising from CPLEX previously, to be performed only in simulators and take advantage of more of their capabilities.

In this way, two variable mappings were created, in which the quantum variables were randomly selected, and those left unchanged by the classical solution. For the complementary local execution, two simulators available free of charge on the IBM Quantum Platform were used, *simulator_mps* (Fig. 11) and *ibmq_qasm_simulator* (Fig. 12), which have 100 and 32 qubits respectively, with the point of attention that while the first is a type of simulator restricted to the product state matrix solution problems, the second is of a general type for any context, which can generate different behaviors in the outputs, despite the problem being compatible with both approaches.

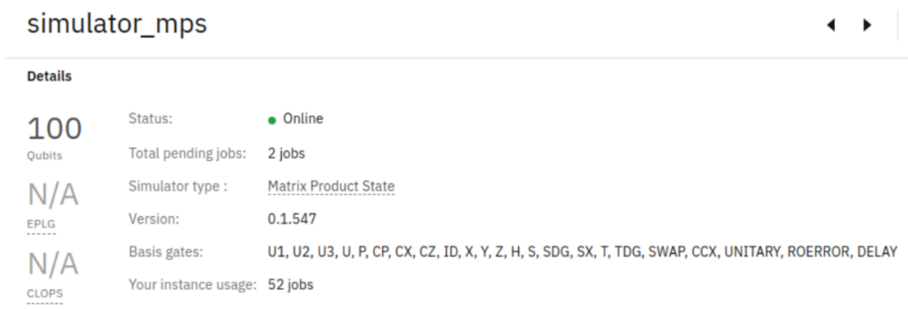

Fig. 11. Simulator MPS properties.

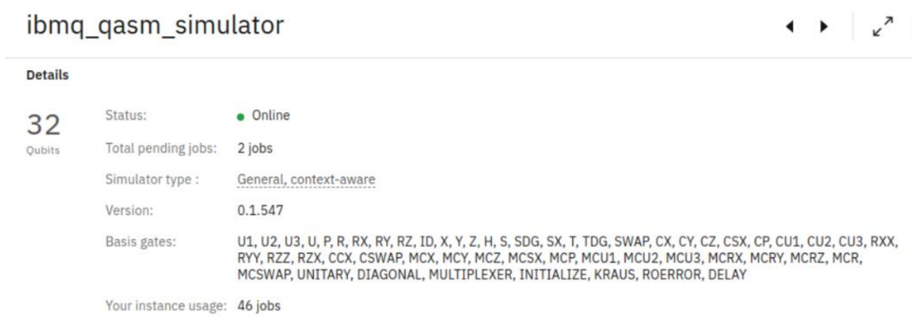

Fig. 12. Simulator QASM properties.

In order to experiment and verify the performance of different algorithms, we tested both the use of the aforementioned VQE and also its analogue, Quantum approximate optimization algorithm (QAOA), which is a hybrid iterative method for solving combinatorial optimization problems [31], as illustrated in Fig. 13.

To continue experimentation, both VQE and QAOA require optimizers, which are mathematical methods for the objective, in our specific case, minimization. For this purpose, two sources made available by the *qiskit.algorithms.optimizers* library were used:

1) COBYLA (Constrained Optimization by Linear Approximation optimizer), a numerical optimization method for constrained problems where the derivative of the objective function is not known [32].
2) SPSA (Simultaneous Perturbation Stochastic Approximation optimizer), an algorithmic method for optimizing systems with multiple unknown parameters. As an optimization method, it is appropriately suited to large-scale population models, adaptive modeling, and simulation optimization [33].

Executions were carried out under the previously mentioned conditions that generated preliminary results, which are described in more detail in the homonymous section. No runs on quantum computers have been carried out to date due to the state of experimentation and decision-making of algorithms and parameters for resolution. Furthermore, there is a restriction on the use of just 10 minutes per month per account of IBM quantum computers, allowing the group to make the best possible use of this resource, especially after better testing and selection in controlled environments such as simulators.

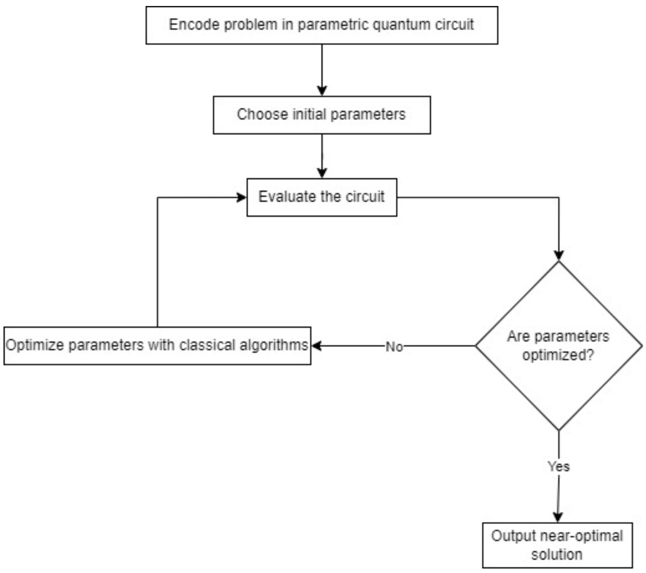

Fig. 13. Overview of QAOA algorithm steps [31]

C. *Results for Hybrid implementation*

From now on, experimenting with algorithms and local executions and simulators, initial results for hybrid approach were obtained that allow a primary understanding of possible choices for future steps in the project. As the first step described in this session was solving the problem via the classic CPLEX algorithm, we can visualize the generated solution in the form of an undirected graph Fig 14. It is noted that although the graph visually appears undirected, there is a delimited order for continuation between the nodes on a given route, which at first was not represented for simplicity purposes.

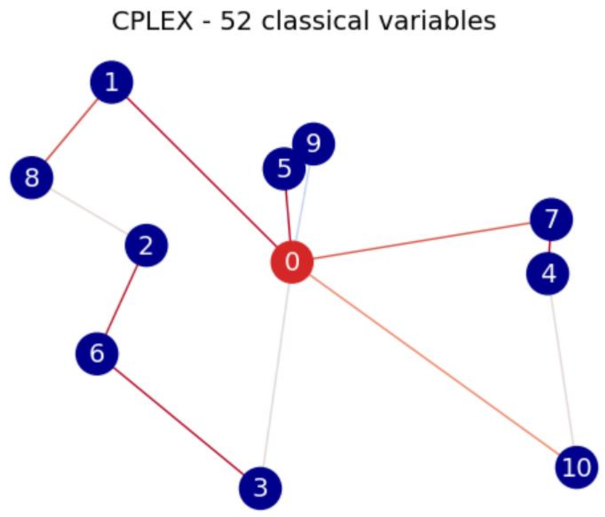

Fig. 14. Solution generated by CPLEX in the form of an undirected graph.

Subsequently, the algorithms VQE and QAOA, each with COBYLA and SPSA optimizers, were executed locally in the condition of 7 variables for the quantum system, and the mapping of 45 variables previously solved by the classical algorithm.

As can be seen from the graphs in Fig. 15 and Fig. 16, while QAOA did not reach convergence for the set of iterations, regardless of the optimizer used, VQE showed a tendency to stabilize in cost (or system energy in this context) minimization, which is an interesting indicator about the use of this last to continue the project.

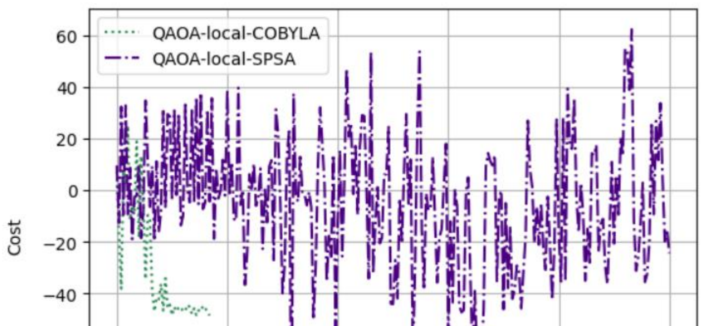

Fig. 15. Cost per iterations plot of QAOA algorithm to solve the VRP problem locally.

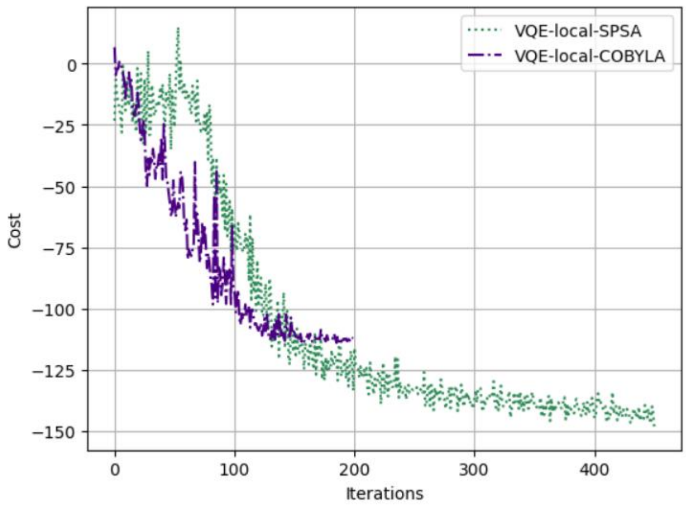

Fig. 16. Cost per iterations plot of VQE algorithm to solve the VRP problem locally.

Then, the problem was run in the ibmq_qasm and mps_simulator simulators under the same conditions with 7 variables. In the ibmq_qasm simulator, the VQE algorithm once again showed clear convergence behavior while QAOA did not register a stabilization trend, as elucidated in Fig. 17 and Fig. 18

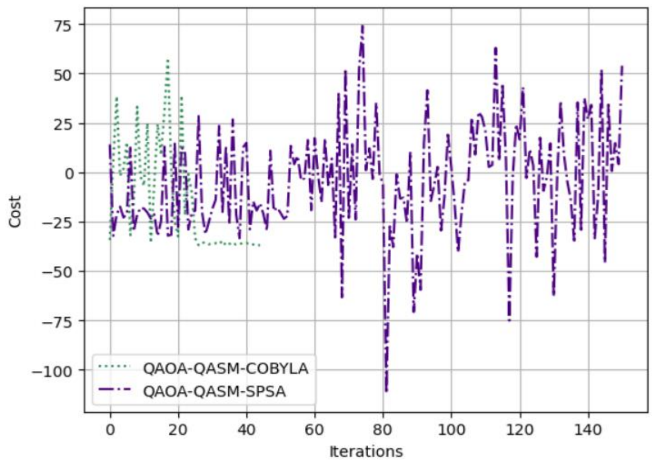

Fig. 17. Cost per iterations plot of QAOA algorithm to solve the VRP problem in IBM QASM

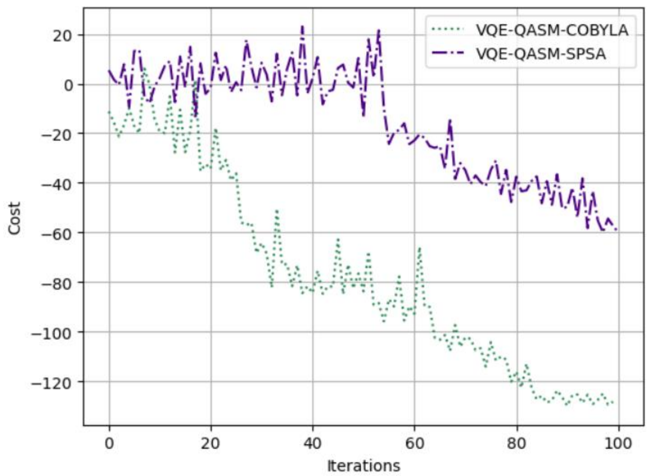

Fig. 18. Cost per iterations plot of VQE algorithm to solve the VRP problem in IBM QASM

The same set with 7 variables to be resolved in the quantum circuit was also run in *mps_simulator*. As verified in other executions, VQE has a much better performance in terms of optimizing the objective function compared to QAOA, as noted in Fig. 19 and Fig. 20.

It is concluded through empirical verification that VQE for the conditions described in the problem presents better results, thus the performance of the VQE algorithm was compared in a local environment, *ibmq_qasm* and *mps_simulator*, to visualize the approaches. As can be seen in Fig. 21, as there were more executions locally, this presented a clear convergence stabilization point, while in the simulators, due to a smaller load of iterations, this behavior was not fully observed despite a clear perception of the process of optimization in search of cost minimization and, possibly, with more iterations, stabilization should probably also be observed.

The fact of using fewer iterations in the simulators is also related to the timeliness of collecting results, since although executions in these were extremely fast, the queuing time due to the shared use of the server often makes rapid experimentation unfeasible due to the time required to wait.

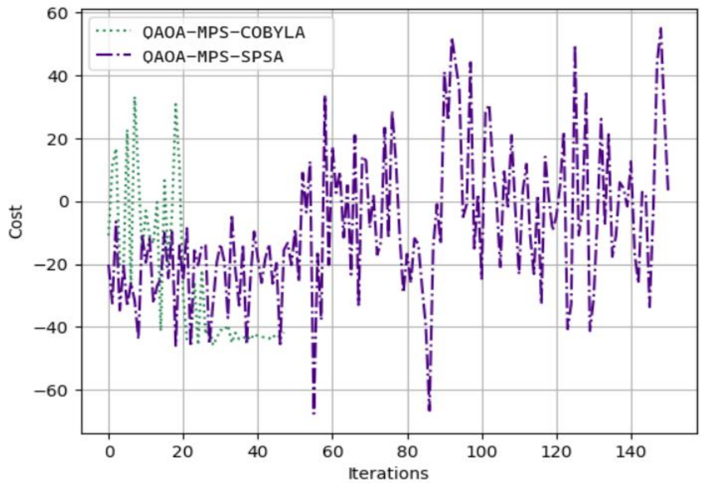

Fig. 19. Cost per iterations plot of QAOA algorithm to solve the VRP problem in IBM MPS

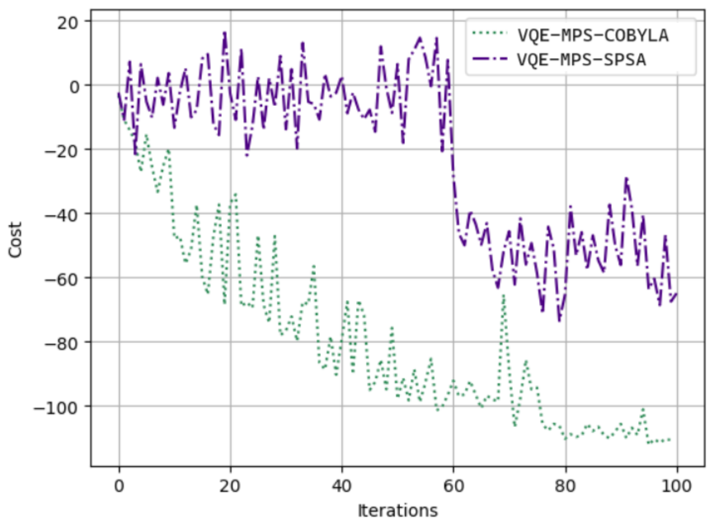

Fig. 20. Cost per iterations plot of VQE algorithm to solve the VRP problem in IBM MPS

It was also noted that the COBYLA optimizer presented better results combined with VQE, having lower accumulated cost values per iterations, being chosen for this reason for comparison between execution environments.

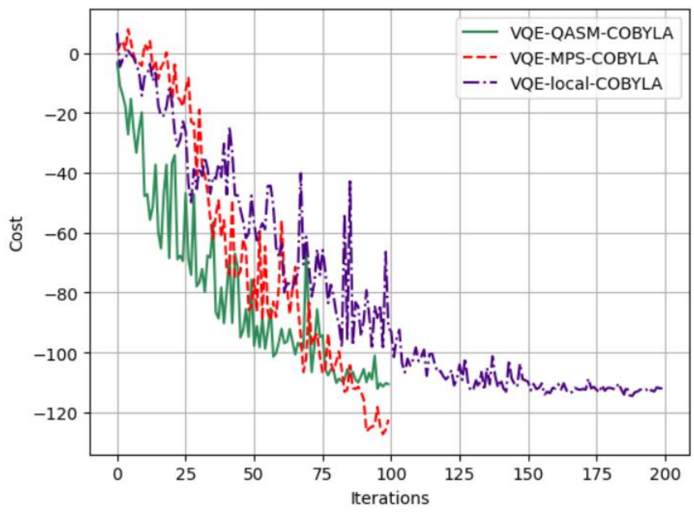

Fig. 21. Cost per iterations plot of VQE algorithm to solve the VRP problem in all environments of execution.

All VQE executions generated solutions practically identical to the one obtained via CPLEX, which leads us to the indication that a hybrid system with a much larger number of variables already pre-defined by the classical solution may not generate different or optimized results with the remaining set of variables for quantum resolution, as seen in Fig. 22.

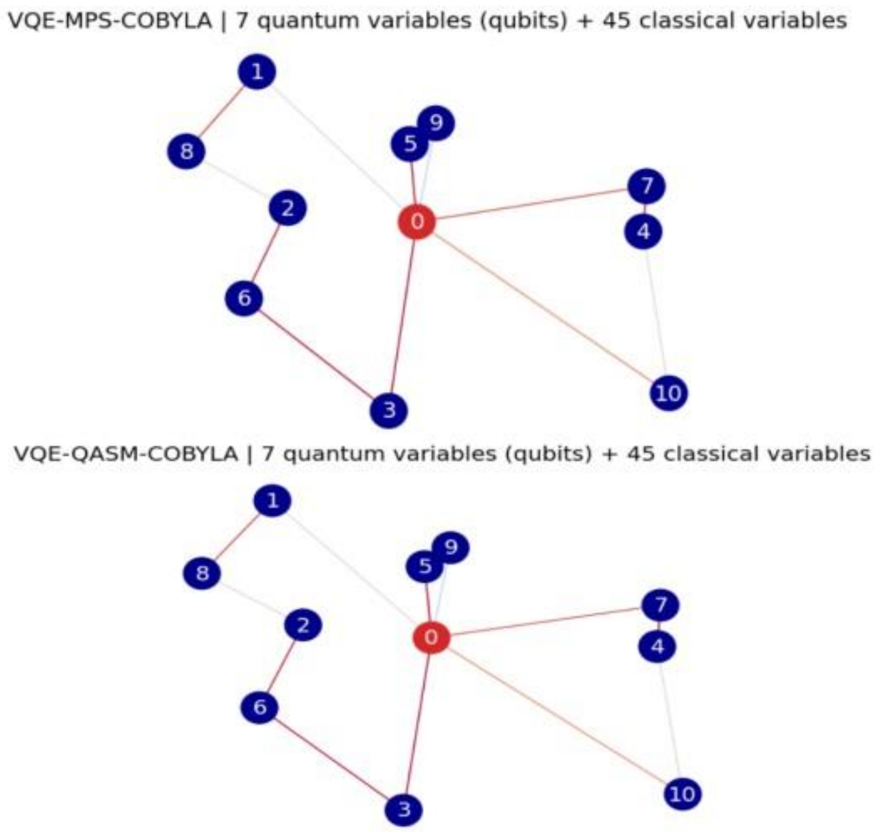

Fig. 22. Solutions obtained by VQE with 7 quantum variables in quantum simulators

Then, as previously established in the experiment session, when delimiting the best set of algorithm and optimizer, VQE and COBYLA respectively, for the 7 variable condition, this was selected for execution in a mapping of 25 quantum variables and 27 already pre-resolved by CPLEX. This context tends to illustrate more of the potential of quantum environments in seeking more efficient solutions that are different from those obtained classically. It is worth noting that this mapping was carried out only in simulators since the local execution took too long for experimentation purposes and would possibly not add interesting enough results for collection. Thus, when remapping the set of variables, it was executed in both *ibmq_qasm* and *mps_simulator* simulators, obtaining the cost (system's energy) graph per iterations of the Fig. 23.

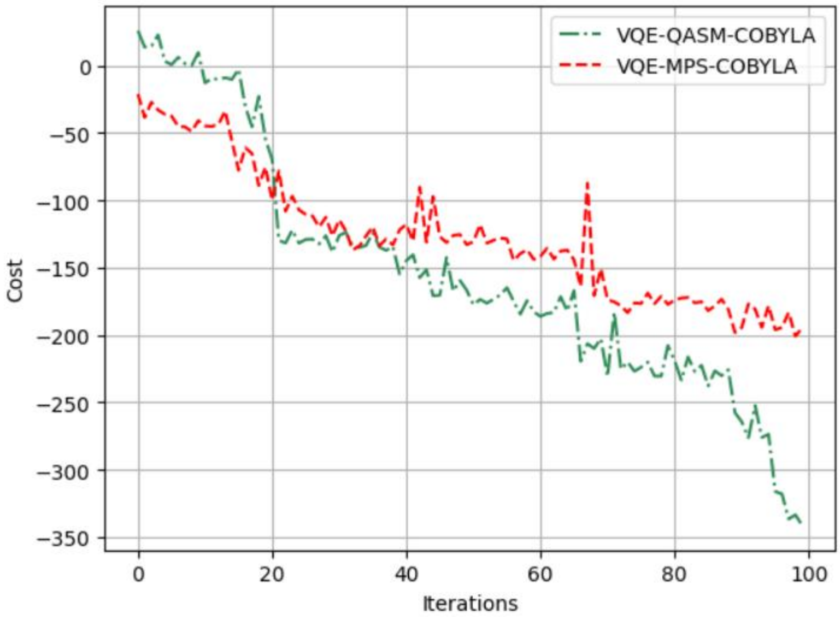

Fig. 23. Cost per iterations plot of VQE algorithm with COBYLA optimizer to solve the VRP problem with 25 variables in both simulators

As can be seen from the cost value, this was minimized by around 3 times in the *ibmq_qasm* simulator, more than in previous steps, which points to good indicators that when using quantum circuits with more qubits they can generate even more suggestive results.

Thus, when representing the solution obtained as output from this simulator in graph format, it already presents different routes, which show the potential of using even more complex quantum circuits (more than 25 variables), which can provide results more favorable and interesting than classical algorithms, as noted in Fig. 24.

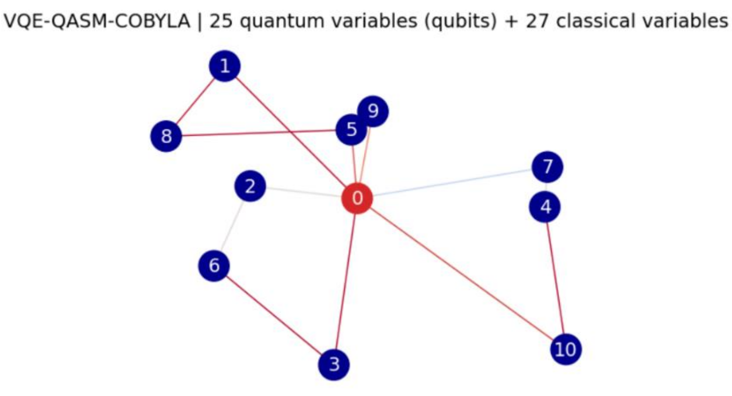

Fig. 24. Solution obtained by VQE with 25 quantum variables in QASM simulator.

D. Full processing in quantum simulators

After seeing that VQE algorithm, with COBYLA as optimizer in the QASM simulator brings better results towards the optimal solution, we conducted experiments now processing entire systems of variables in the quantum simulators and comparing its function objective values with CPLEX solutions, with aim of evaluating how good the solutions generated by this quantum conjuncture are close to solutions considered adequate in terms of optimization.

As the QASM simulator has a limitation regarding the number of qubits for processing, subsets of a maximum of 24 binary variables were explored with the aim of comparing the results in classical and quantum environments.

With this goal, 6 points around São Paulo city were generated following the conditions of the existence of an edge, assuming $p = 1$ again for (23), generating 12 edges between those points and 24 binary variables to be evaluate by algorithms, as illustrated in Fig. 25. Due the less complexity of the subset of data, 3 trucks, and consequently 3 routes, were chosen without loss of generality, adapting to contexts close to real scenarios, in which there is flexibility of the expected number of routes and trucks.

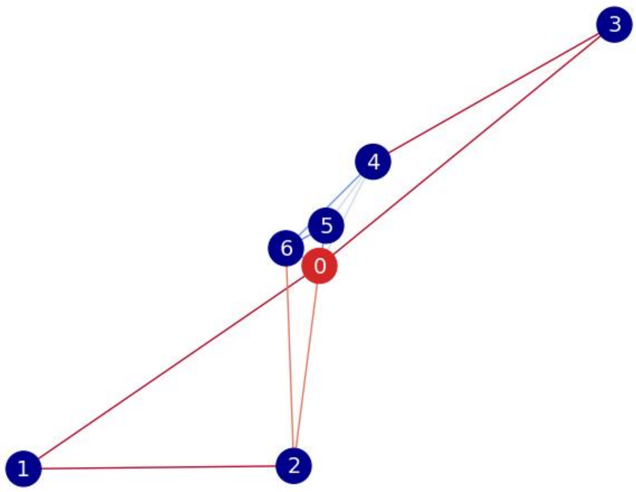

Fig. 25. Input data for 6 coordinates, generating 24 binary variables for analysis.

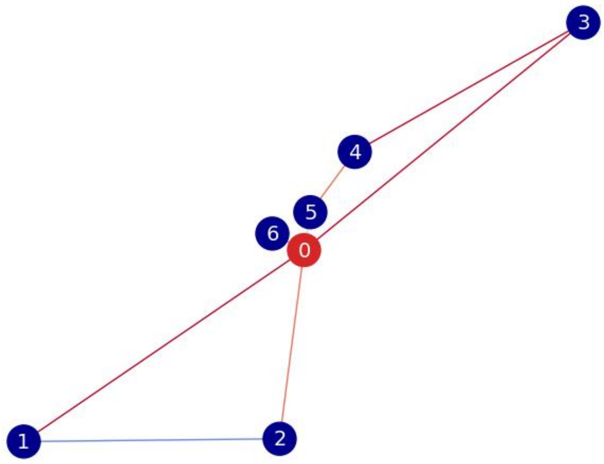

Fig. 26. Solution found by CPLEX for 24 variables.

Then, as performed in previous steps, the quadratic problem was converted to QUBO, and then, using the VQE algorithm, with COBYLA optimizer, in the QASM simulator, a combination that demonstrated the best results among the previous experiments, it was carried out in 100 iterations the quantum system searches for the optimal solution. Outputs generated by both approaches are elucidated in Fig. 26 and Fig. 27.

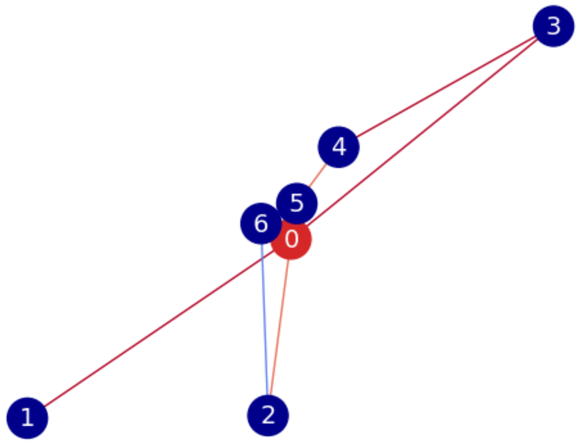

Fig. 27. Solution found by VQE for 24 variables.

It's also clear from the cost per iteration plot that the solution obtained tends to be close to optimal due to the tendency to minimize the objective function used, as noted in Fig. 28.

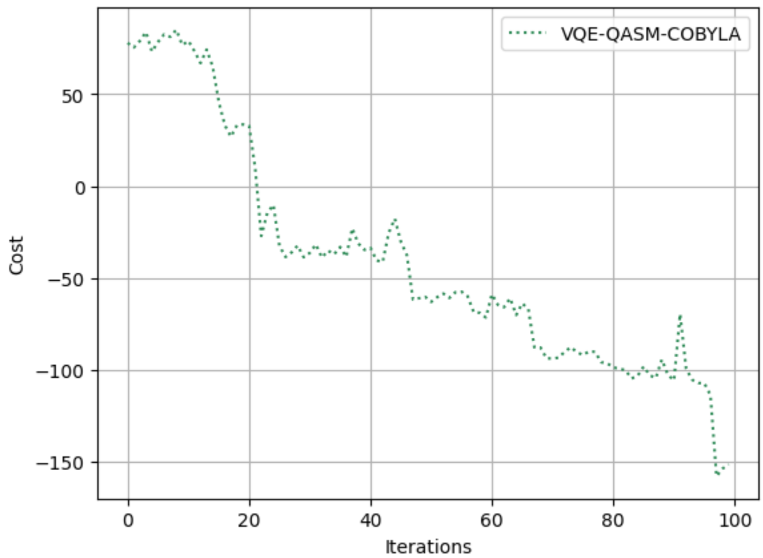

Fig. 28. Plot of cost function per iterations of VQE in QASM with 24 variables.

When comparing the objective function values of each method, in Table IV, classical and quantum, it is noted that even for a few iterations the results obtained by the latter approach are close to the optimal value, which is obtained by minimizing the objective function.

This is an interesting indication, given that given the fact that it can perform well in controlled scenarios, when expanded to larger scale scenarios, the tendency is for the model to seek solutions that are close to optimal without having execution bottlenecks.

TABLE IV

OBJECTIVE FUNCTION VALUES FOR CPLEX AND VQE IN QASM SIMULATOR

CPLEX	VQE-COBYLA-QASM
6.205	7.368

E. Full processing in quantum computers

With very promising results arising from execution on quantum simulators, executions on quantum computers were planned to compare algorithms executed on a quantum machine against the results considered optimal by traditional approaches.

This execution on quantum machines was carried out, like previous experiments, using the IBM ecosystem. However, the use of quantum hardware is limited to 10 minutes of execution per user per month, which does not allow the exploration and validation of a set of larger variables in the sufficient time provided.

In this way, the input set was reduced to around 5 coordinates, excluding the origin, totaling 9 possible edges, according to the pre-processing conditions explained in section II.C of this report, and again for reasons of simplicity adopting the value of $\boldsymbol{p} = \mathbf{1}$ for (23).

Two executions were carried out in these environments, following the conditions previously explained, with the purpose of verifying the efficiency of the VQE algorithm on suitable quantum hardware:

1) Execution on the IBM Sherbrooke 127 qubit computer with 50 iterations.
2) Execution on the IBM Osaka 127 qubit computer with 100 iterations.

1) Execution on the IBM Sherbrooke computer: Sherbrooke is one of the quantum computers available on the platform's free plan. This machine belongs to the Eagle R3 processor family, whose main characteristics are having 127 qubits in its architecture and up to 5K CLOPS (a measure of how quickly processors can run Quantum Volume circuits in series), as shown in Fig. 29. One of its great differences is its low EPLG, error-per-layered gate, the average error for each gate in these layered circuits, which helps to measure the amount of noise obtained during the execution of a circuit *[34]*.

Due to this low percentage and its free availability, the number and duration of jobs on this machine are high, guiding us to carry out testing with fewer iterations to be able to capture a result in a timely manner.

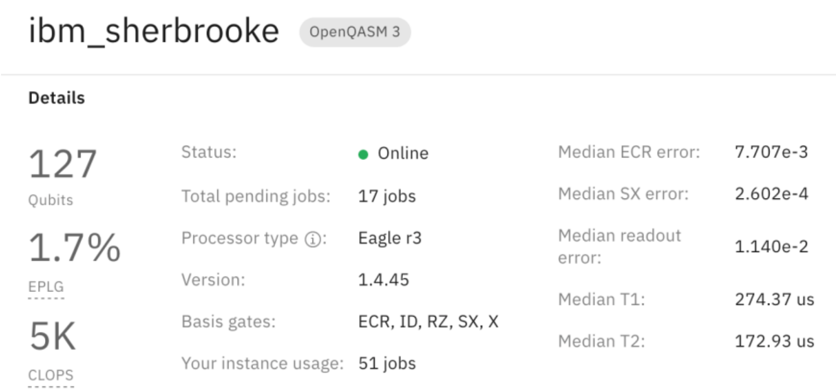

Fig. 29. Sherbrooke Quantum Computer properties

It was processed as an input set, 5 coordinates, which interconnected in routes, generated 9 edges, culminating in 18 binary variables in the system. Arbitrarily, a quantity of 3 trucks was also selected as a parameter for the problem, without evident loss of generality, as viewable in Fig. 30.

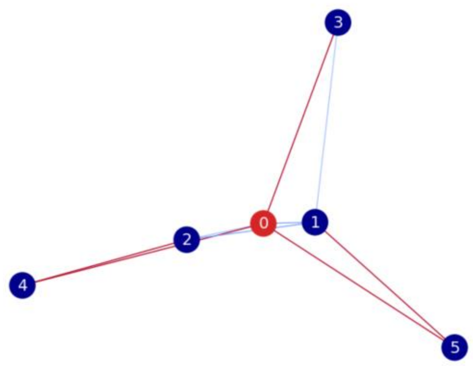

Fig. 30. Input data for 5 coordinates, generating 18 binary variables for analysis.

Using the classic approach with CPLEX, a solution illustrated in Fig. 31 was obtained.

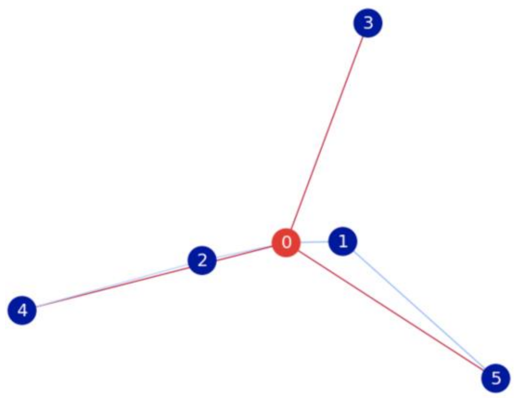

Fig. 31. Solution found by CPLEX for 18 variables.

Following procedures illustrated previously, the input set was prepared for execution in a quantum environment (QUBO) and using the VQE algorithm with COBYLA optimizer, 50 iterations were carried out.

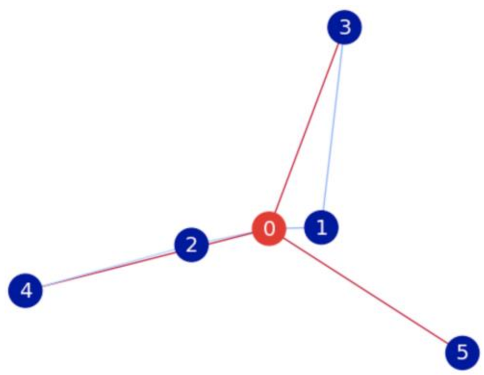

Fig. 32. Solution found by VQE for 18 variables in Sherbrooke Quantum Computer.

The output obtained in Fig. 32, despite not being the same as that obtained classically, is extremely close to the latter. Considering the small number of iterations carried out, the potential of using quantum quantities to obtain solutions close to optimal on extremely large scales is noted. As Table V shows, the percentage difference between the objective function values is only 1.68%.

TABLE V
OBJECTIVE FUNCTION VALUES FOR CPLEX AND VQE IN SHERBROOKE COMPUTER

CPLEX	VQE-COBYLA-SHERBROOKE
6.258	6.364

It is also noted that the plot of cost per iteration, Fig. 33, once again elucidates the trend of convergence of the system's energy (associated cost). Even being on a quantum computer, that is, not respecting ideal conditions as in simulators, the ability to obtain routes close to the optimality plateau is noted.

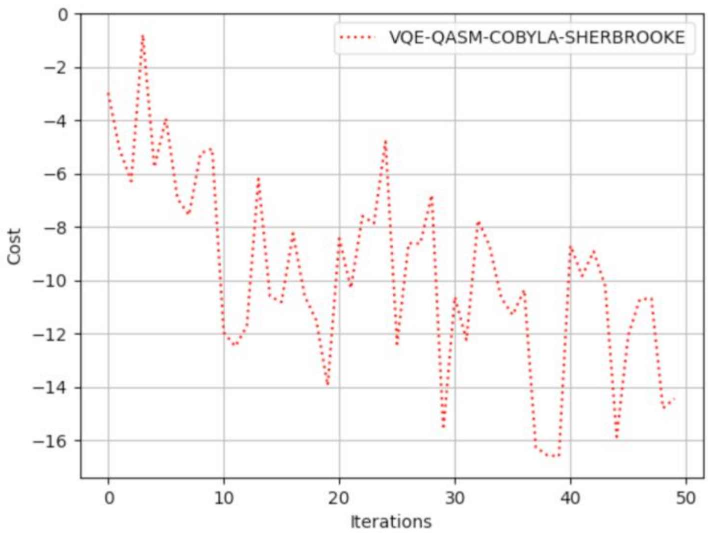

Fig. 33. Plot of cost function per iterations of VQE in Sherbrooke with 18 variables.

2) Execution on the IBM Osaka computer: Next, a larger set of iterations (approximately 100) was tried for a system of also 18 binary variables. To achieve this execution in a timely manner, the Osaka quantum computer was chosen, which, similarly to the Sherbrooke one, also belongs to the Eagle R3 family of processors, noticeable in Fig. 34. Its main difference is that its EPLG is slightly larger when compared to other available hardware, which makes the waiting time in its queues a little shorter, making it easier to execute these 100 iterations.

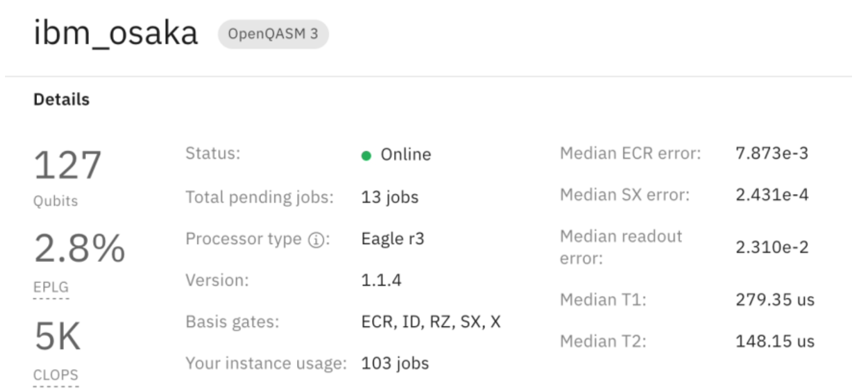

Fig. 34. Osaka Quantum Computer properties

Again, with 5 random coordinates generated, we obtained 18 binary variables for the input system and (Fig. 35), therefore, it was possible to find the solution considered optimal by CPLEX for a number of 3 trucks selected.

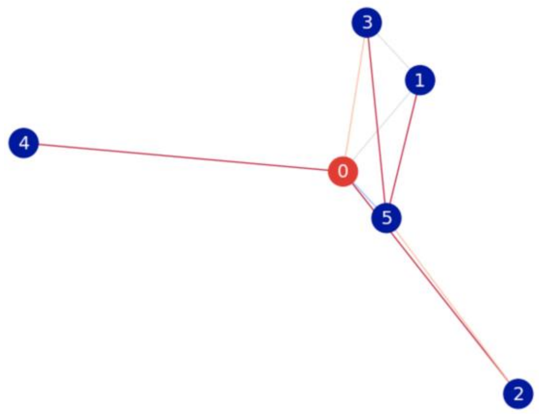

Fig. 35. Input data for 5 coordinates, generating 18 binary variables for analysis.

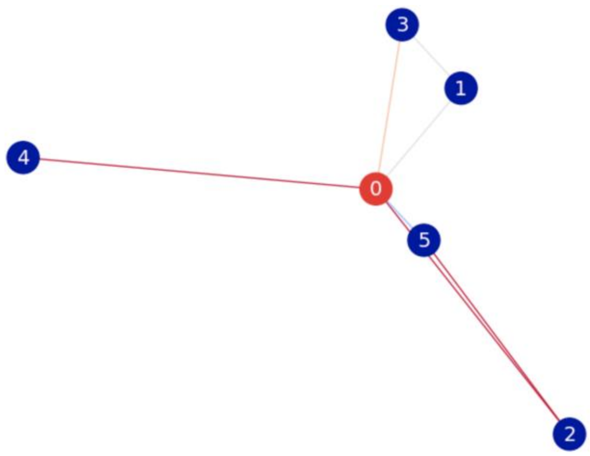

Fig. 36. Solution found by CPLEX for 18 variables.

Due to the greater number of iterations, the algorithm run on a quantum machine was able to find a practically identical result to that obtained via CPLEX, as viewable in Fig. 36 and Fig. 37. The difference between the generated solutions only occurs in the order to be followed in some sub-routes, but this does not have a major variation impact.

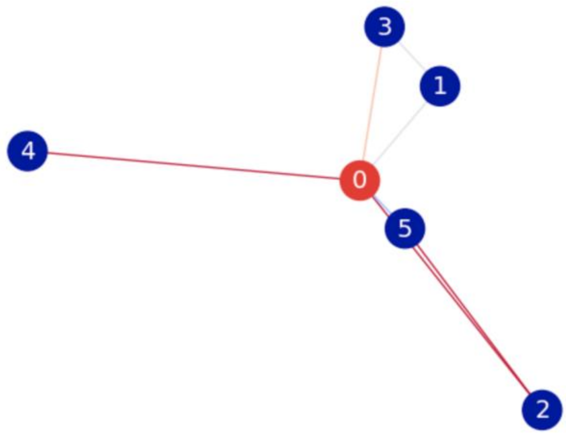

Fig. 37. Solution found by VQE in Osaka Computer for 18 variables.

This proximity between the solutions is also perceived in their objective function values (Table VI), which are extremely similar, illustrating the ability to find optimal solutions using optimization algorithms.

TABLE VI
OBJECTIVE FUNCTION VALUES FOR CPLEX AND VQE IN OSAKA COMPUTER

CPLEX	VQE-COBYLA-OSAKA
5.757	5.896

Due to a higher error rate per layer, it is expected that the results obtained in this execution environment may have noise during its execution. This fact can be seen in the cost per iteration plot (Fig. 38), which shows that even reaching the solution considered classically optimal, the system did not even show convergence, which can be an important decision for selecting the environment and parameters in eventual executions that require more complexity.

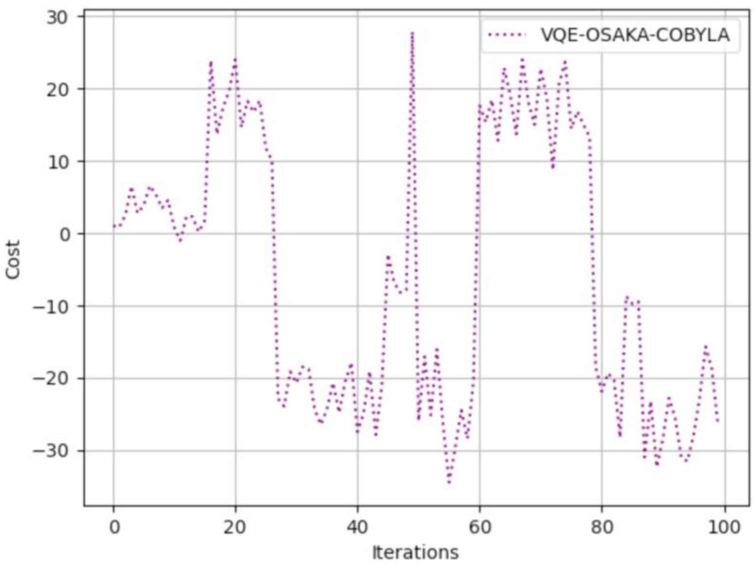

Fig. 38. Plot of cost function per iterations of VQE in Osaka with 18 variables.

F. Cash demand of each node as an influential factor in the edge weight

To explore other scenarios adjacent to solving supply chain problems via VRP, other factors can be considered such as traffic, security, and specific demand for each node (money, products, deliveries), in addition to the distance between coordinates, already considered.

As explained in (23), one way to incorporate more parameters into the formulation of the problem is to weigh up some of these described factors. In the case of supply chain problems, the specific demand between us, that is, what should be taken from one point to another, is a discriminating factor, in addition to the distance to define a good route. Specifically, in the case of cash supply chain, the concept of demand between points can be understood as the amount of money that each ATM needs, corresponding, for example, to a physical limitation of how much a truck can carry on a given supply sub-route.

To evaluate the efficiency of solving these problems with mixed weights, the definition of weight between edges previously presented was combined, and different values of p were experimented with, to visualize possible solutions between the execution environments:

1) Assuming $p = 0.6$, validating a greater weight for cash demand.
2) Assuming $p = 0.8$, validating a relatively smaller weight for cash demand.

These values, although arbitrary, also elucidate closer choices to be made in logistical contexts, in which the main factor is distance, and therefore it is not expected that this will have a lower weight than other variables for logical reasons. In both experiments carried out, a number of 6 generated coordinates and 3 trucks were considered as problem parameters.

1) Assuming $\boldsymbol{p = 0.6}$

In this context, the weight equation takes the following form:

$$weight_{mn} = 0.6 \cdot |distance_{mn}| \ + 0.4 \cdot |Cash\ amount_{mn}|$$

In this case, there is a greater equalization between the factors, that is, the possibility of searching for solutions that include a balance between shorter distance routes and that transport a smaller amount of money per truck.

The input set was generated again (Fig. 39), still subject to the established pre-processing conditions, however now, comparing the threshold in relation to the mixed weight of each edge.

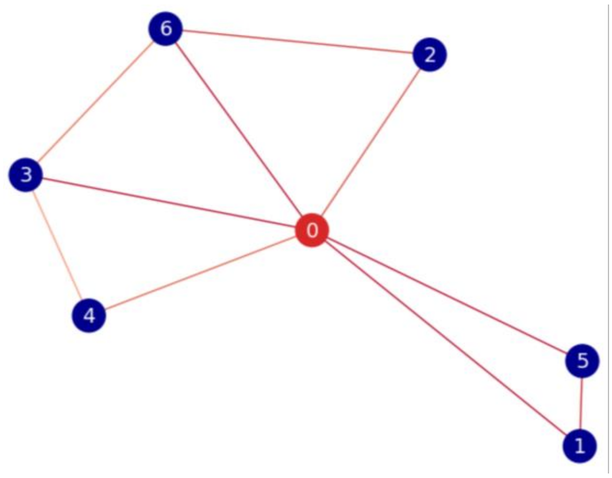

Fig. 39. Input data for 6 coordinates, generating 20 binary variables.

In this way, classical executions via CPLEX were compared, analogously to previous procedures, with VQE in a QASM environment, as noted in Fig. 40.

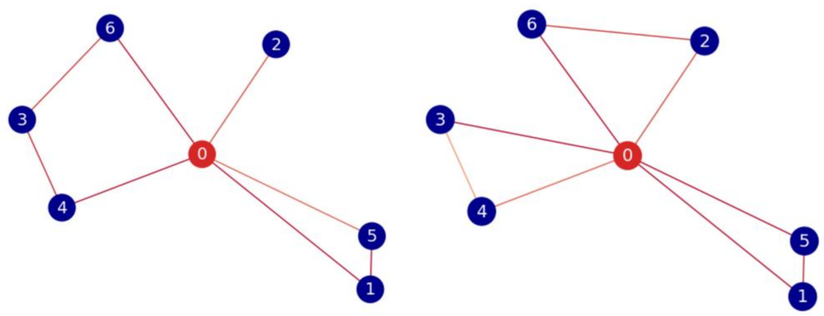

Fig. 40. Solution found by CPLEX on left, and solution found by VQE-QASM on right.

As noted, the outputs found are not similar, and present notable divergences in their objective function values (Table VII). This indicates that possible treatments for better equalization and aggregation of variables in the edge weight composition can be taken, with the aim of facilitating the search for optimal solutions through optimizers, as performed by VQE.

TABLE VII
OBJECTIVE FUNCTION VALUES FOR CPLEX AND VQE IN QASM SIMULATOR FOR $p = 0.6$

CPLEX	VQE-COBYLA-QASM
6.765	36.191

It is also interesting to note that even with a percentage difference of approximately 137% between the objective values, during the optimizer iterations, it registered a convergence behavior (Fig. 41), which elucidates that with slightly more precise formulations, better results could be obtained for this case.

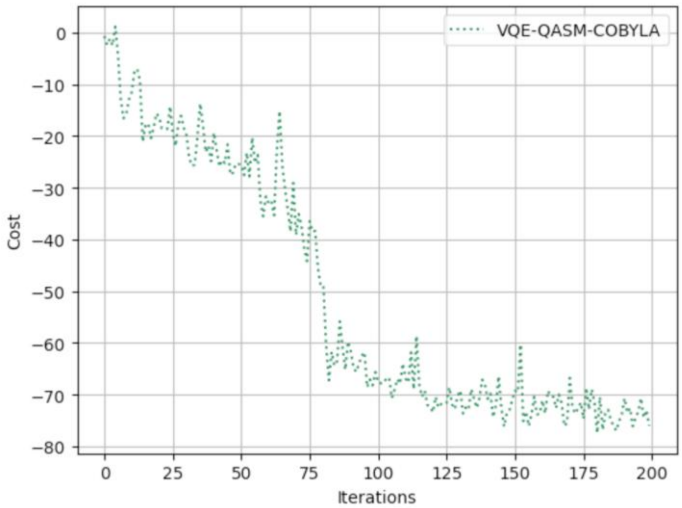

Fig. 41. Plot of cost function per iterations of VQE in QASM with 20 variables and p = 0.6.

2) Assuming $\boldsymbol{p = 0.8}$

Now assuming a less favorable approach to the influence of cash demand on the conception of weight, the effects of this parameterization were also tested in a simulator.

$$weight_{mn} = 0.8 \cdot |distance_{mn}| + 0.2 \cdot |Cash\ amount_{mn}|$$

Still following the processing conditions of the previous session, the input graph for the problem was generated. Given a set of 6 generated points, 24 binary variables were considered to set up the quadratic problem, as illustrated in Fig. 42.

Similar to what was observed in the previous experiment, it was noted that although the quantum approach finds a possible solution within the established restrictions (Fig. 43), its objective values differ greatly from that found by CPLEX (Table VIII), which generates evidence that the weight formulation can be incorporated in a robust to improve the achievement of results in scenarios that reveal greater complexity of associated factors. For example, changing the way the data is normalized (to a logarithmic scale) or even changing the weighting factor.

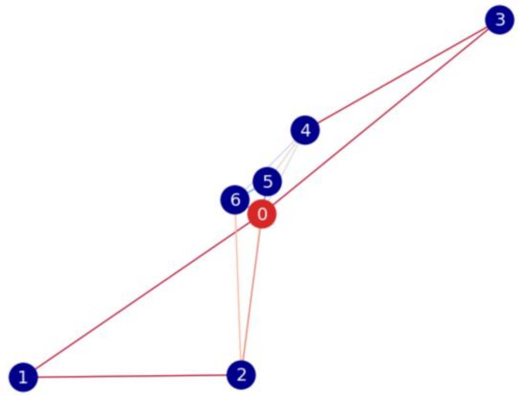

Fig. 42. Input data for 6 coordinates, generating 24 binary variables.

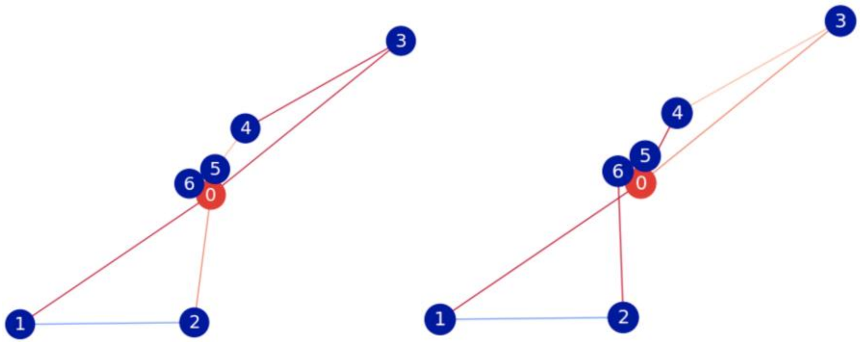

Fig. 43. Solution found by CPLEX on left, and solution found by VQE-QASM on right.

TABLE VIII

OBJECTIVE FUNCTION VALUES FOR CPLEX AND VQE IN QASM SIMULATOR FOR $p = 0.8$

CPLEX	VQE-COBYLA-QASM
5.761	36.495

Even with a percentage difference of almost 145% between the objective values obtained, the cost per iteration plot (Fig. 44) illustrates the tendency towards stabilization of an optimal solution, so with more appropriate formulations, the tendency may be to obtain even better solutions. close to those considered optimal.

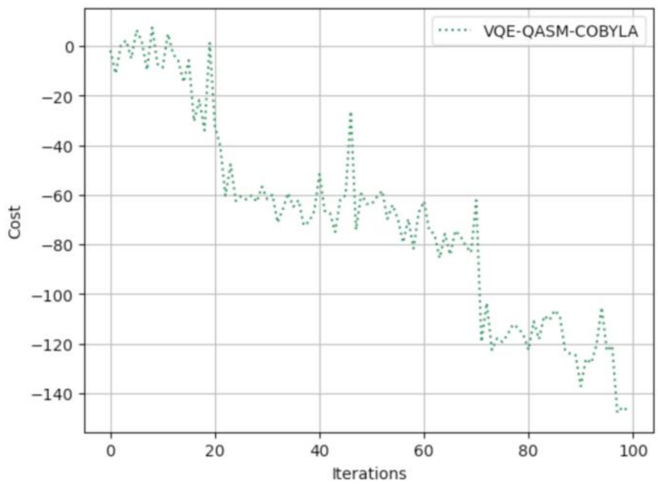

Fig. 44. Plot of cost function per iterations of VQE in QASM with 24 variables and p = 0.8.

G. Proposed solution architecture

After the experimentation process was completed, and we had parameters and indicators in hand to compare approaches, the solution architecture was created. To follow good practices to ensure the solution works properly, it was divided into two scopes:

1) Backend, in which classical or quantum processing occurs through the selection of user parameters and returns the set of routes closest to the optimal solution.
2) Frontend, an interactive dashboard that allows the user to select search parameters that facilitate the visualization of insights about the chosen routes.

1) Backend architecture: In order to build a scalable and secure application, the FastAPI web framework for Python was chosen for the backend, which is known for its modernity, scalability and ease of use for building APIs.

The architecture consisted of given the parameters selected by the user such as points to be frosted, quantum algorithm and optimizer, a pre-processing to define the number of trucks and calculation of respective best routes will be carried out.

With the aim of facilitating experimentation and allowing comparisons between approaches, two endpoints were implemented:

1) An endpoint named *classical*, which will process the input data locally using CPLEX.
2) An endpoint named *quantum*, which, when pre-processing the input data, will seek a solution using a local simulator or IBM quantum computer, depending on the user's choice.

This archetype can be seen in greater detail in the diagram of the Fig. 45.

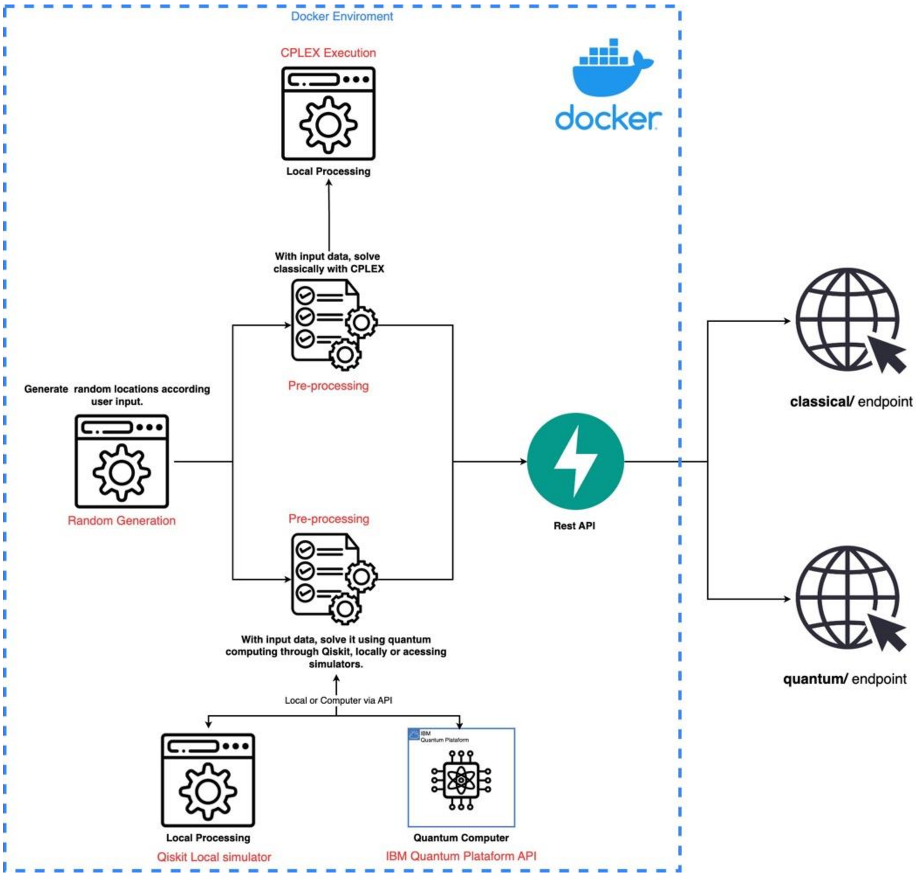

Fig. 45. Overview of backend architecture.

This REST API was implemented with the aim of consumption in the frontend architecture and to provide a prototype solution using the analyzes conducted so far, to provide input for dashboard visualizations and integrative maps.

2) Frontend Architecture: In order to provide an adequate visualization adjusted to real routes, a frontend interface was prototyped with the aim of providing an interactive visualization of the chosen routes and other associated metrics, such as travel time and total values that trucks should carry to supply the ATMs during the route.

The *React.js* framework was used for JavaScript language, due to its compatibility with reusable components, its use of virtual DOM for greater responsiveness and real-time data processing generating better user experience, and adherence to external modules that allow robust construction of the platform, such as visualization of maps or construction of a metrics dashboard. Similarly, to consume the data available through the endpoints described in the backend architecture, we use the *Axios* HTTP client, for local requests and in the future, from external servers.

The publicly available Google Maps API was used to render the routes calculated in real time with the advantage of using real routes through the city of São Paulo and approximate travel times that improve the user experience and usefulness of the interface. The prototyped architecture is shown in Fig. 46:

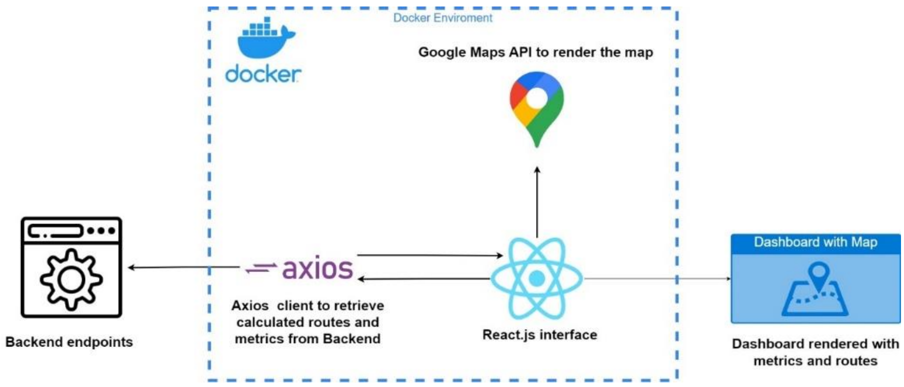

Fig. 46. Overview of frontend architecture.

As the main purpose is to be a proof of concept for visualizing the routes closest to optimality, the user was allowed to initially select to see results by classical or quantum algorithm, and choose customization parameters, such as points to be supplied, quantum algorithm (if applicable) and others. Thus, the initial interface, as shown in Fig. 47, aims to make it easier for the user to differentiate parameter selection between solutions.

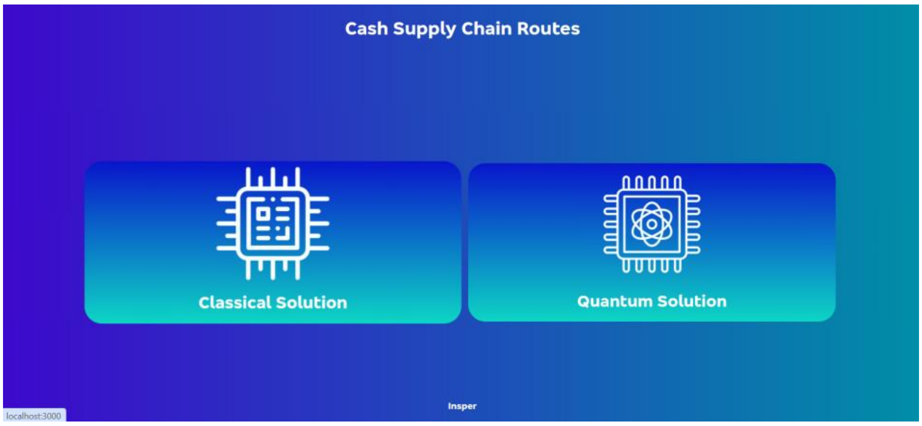

Fig. 47. First screen of proof of concept.

After selecting the chosen solution, and its respective parameters, the user will initially have access to a general view of all routes to be pursued in the city of São Paulo (Fig. 48), already incorporated into urban directions and limitations, facilitating a general overview of the solution within the limited scope of the city of São Paulo.

Fig. 48. Visualization of all calculated routes in São Paulo city.

To facilitate and direct each route individually, a screen was also created that illustrates each specific route (Fig. 49), with details about its duration, route, and financial demand to be transported by the truck.

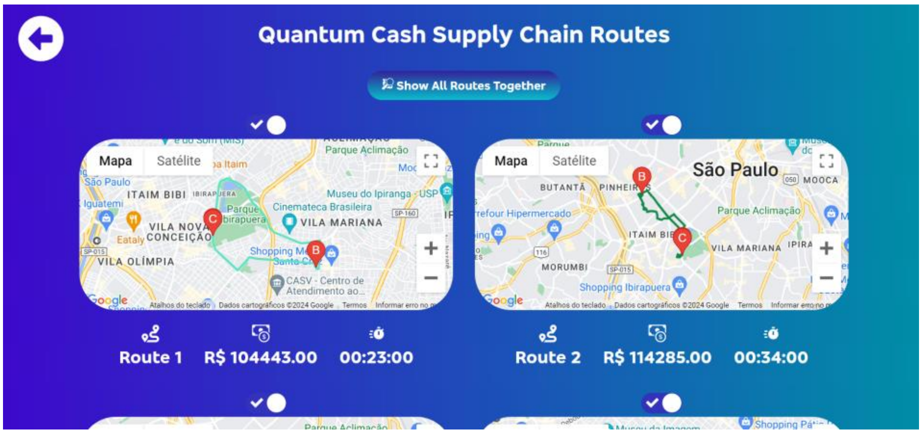

Fig. 49. Visualization of all calculated routes in São Paulo city.

In this way, the dashboard, in addition to intuitively providing the best routes calculated by the algorithm, facilitates understanding and directing the logistics team in an integrated and safe manner, facilitating, and accelerating the supply process.

IV. Conclusion

From now on, the possibilities of scale in solving supply chain problems allowed by quantum computing, that is, being able to solve problems previously limited to hundreds of variables, expanding to thousands, there is great potential in using this technology to solve not only cash supply chain problems, but also related to GPS route calculations, marketplace distribution and many others.

In this scope, it is noted that as a product of this project, not only a proof of concept was obtained regarding the use of quantum computing in classical computing bottleneck scenarios, but also a glimpse of the potential of this topology applied to extremely complex problems. computational, as discussed during this report.

In a real scenario, the advantages of a solution close to optimality on larger scales can be objectively measured in financial terms. As already mentioned, appropriate VRP solutions, and their dissent, can reduce up to 30% [3] of costs related to urban transport. In the context of cash supply chain, it is noted that armored car companies can charge up to 1% [35] of the value transported as a transport fee.

In this regard, assuming a massive supply context for approximately 300 agencies in a metropolis, each with a need of R$ 50,000.00, would lead to a total transported value of R$ 15,000,000.00, and therefore almost R$ $150,000.00 in shipping cost. According to empirical verifications, this problem resides around thousands of variables (approximately 3800 according to the proportion in Fig.7), which would take, in a hypothetical quantum machine with sufficient qubits for processing, approximately 3800 seconds (following patterns experimentally verified during executions of the group) for processing the problem (excluding waiting times in queues). Therefore, considering the prices of IBM's pay-as-you-go plan [36], of $1.60 for each second of execution, the search for an optimal solution would cost approximately R$ 31,372.80 (dollar exchange rate at R$ 5.16).

With this solution reducing around 30% of associated costs, and assuming that these can be deducted from the total transport value, we would have a discount of approximately R$ 45,000.00, which when subtracted from the cost associated with execution, generates savings of approximately R$ 13,628.80.

When considering that savings like these can be obtained for supplies that can occur on a weekly schedule, it becomes an excellent strategy for reducing costs and with wide application in several other areas that deal with highly complex problems and require high processing rates.

Therefore, it is noted that the horizon of quantum computing is still very vast and susceptible to innovations that can improve challenges currently faced computationally. Specifically, IBM continues to invest in quantum machine processors, recently announcing new features such as the 133-qubit Heron that reduces instability and noise [37], and the development of a processor that surpasses the 1000-qubit barrier, the Condor [38], which promises to revolutionize this field.

Not only has IBM been innovating in this scope, Nvidia recently revealed a major innovation effort, creating NVIDIA Quantum Cloud [39], allowing easier access to simulation platforms that accelerate research and implementations. It is also noted that other big techs also have latent initiatives in this technology, such as Amazon Braket from Amazon Web services, Azure Quantum from Microsoft, and Google Quantum AI from Alphabet.

It is well known that even with physical impediments to some quantum solutions, the possibilities that this technology opens for solving logistical and highly complex problems are immense and could bring new inflection points in what we claim to be a technological revolution.

Therefore, the main product of this capstone project is not only to deliver a solution regarding the feasibility and possibility of using quantum computing in the context of cash supply chain, but also to illustrate that this topology can meet and solve today's problems on a larger scale limited, and could establish a new technological revolution in the current standard of society.

ACKNOWLEDGMENT

We extend our deepest gratitude to our project advisor, Prof. Luciano Silva, whose expertise and guidance were instrumental in the conceptualization and execution of this project. We are also thankful to Prof. Luciano Soares, not only for his role as an advisor but also for his invaluable input as a member of the Evaluation Committee and as the Engineering Capstone Project Coordinator at Insper Institute of Education and Research. His insights and oversight have significantly contributed to the rigor and success of our work.

Our appreciation also goes to Prof. Raul Ikeda and Prof. Rodrigo Carareto, whose evaluations were crucial in refining our project objectives and methodologies. Their feedback has been a cornerstone of our project's development and success.

We are grateful to Insper Institute of Education and Research for providing us with the resources and environment necessary for conducting this research. Their support has been essential in our academic and professional growth.

Lastly, we would like to acknowledge the unwavering support and encouragement from the families of all members of this capstone project. Their support has been a constant source of motivation throughout this journey.

APPENDIX A

The following is the quotation from the first three paragraphs in [6]

"Given a set of cities along with the cost of travel between each pair of them, the traveling salesman problem, or TSP for short, is to find the cheapest way of visiting all the cities and returning to the starting point. The "way of visiting all the cities" is simply the order in which the cities are visited; the ordering is called a tour or circuit through the cities.

This modest-sounding exercise is in fact one of the most intensely investigated problems in computational mathematics. It has inspired studies by mathematicians, computer scientists, chemists, physicists, psychologists, and a host of nonprofessional researchers. Educators use the TSP to introduce discrete mathematics in elementary, middle, and high schools, as well as in universities and professional schools. The TSP has seen applications in the areas of logistics, genetics, manufacturing, telecommunications, and neuroscience, to name just a few.

The appeal of the TSP has lifted it to one of the few contemporary problems in mathematics to become part of the popular culture. Its snappy name has surely played a role, but the primary reason for the wide interest is the fact that this easily understood model still eludes a general solution. The simplicity of the TSP, coupled with its apparent intractability, makes it an ideal platform for developing ideas and techniques to attack computational problems in general." [6]

APPENDIX B

The Ising model is a fundamental concept in statistical mechanics, traditionally used to study magnetic systems. In this model, variables represent spins that can be in one of two states: "spin up" (↑) or "spin down" (↓), corresponding to $\mathbf{+1}$ and $\mathbf{-1}$ values, respectively. These spins are arranged on a lattice, and their interactions are described by couplings, which can be either correlations or anti-correlations between the spins.

The energy of a system described by the Ising model, also known as the objective function, is given by (28).

$$E(s) = \sum_{i=1}^{N} h_i s_i + \sum_{i=1}^{N} \sum_{j=i+1}^{N} J_{i,j} s_i s_j \tag{28}$$

where

$\boldsymbol{E}(\boldsymbol{s})$	energy of a system given the lattice $\boldsymbol{s}$;
$\boldsymbol{N}$	continuous variables associated with each client node $\boldsymbol{j}$. They are used to represent the position of the node in the tour;
$\boldsymbol{h_i}$	linear coefficients corresponding to qubit biases;
$\boldsymbol{J_{i,j}}$	quadratic coefficients corresponding to the coupling strengths between spins;
$\boldsymbol{s_i}$	spin corresponding to $\boldsymbol{i}$;
$\boldsymbol{s_j}$	spin corresponding to $\boldsymbol{j}$.

Here, $\boldsymbol{h_i}$ represents the linear coefficients corresponding to qubit biases, and $\boldsymbol{J_{i,j}}$ represents the quadratic coefficients corresponding to the coupling strengths between spins $\boldsymbol{s_i}$ and $\boldsymbol{s_j}$. The first sum accounts for the individual contributions of each spin, while the second sum accounts for the interactions between pairs of spins.

The Ising model has been instrumental in understanding the behavior of magnetic materials and has been extended to study a variety of phenomena, including phase transitions, optimization problems, and neural networks [40]

APPENDIX C

The Hamiltonian is a foundational concept in physics, particularly in quantum mechanics, representing the total energy of a system. It serves as an operator describing the system's evolution over time. In classical mechanics, the Hamiltonian encompasses the sum of kinetic and potential energies. In quantum mechanics, it is portrayed as an operator in the Schrödinger equation, governing the dynamics of quantum systems [41].

REFERENCES

[1] P. Toth and D. Vigo, in *The Vehicle Routing Problem*, Eds., Ed., Philadelphia, PA, USA: SIAM, 2002.

[2] R. d. C. Paiva and M. A. Mesquita, "Modelo de previsão e reposição de numerário em uma rede de caixas eletrônicos," EPUSP, São Paulo, 2006.

[3] H. N. Psaraftis, "Dynamic Vehicle Routing Problems," in *Vehicle Routing: Methods and Studies*, North-Holland, 1988, pp. 223-248.

[4] S. -Y. Tan and W. -C. Yeh, "The Vehicle Routing Problem: State-of-the-Art Classification and Review," *Applied Sciences*, vol. 11, no. 21, 2 November 2021.

[5] Q. O. D. Team, "Vehicle Routing," Qskit, [Online]. Available: https://qiskit-community.github.io/qiskit-optimization/tutorials/07_examples_vehicle_routing.html. [Accessed May 5 2024].

[6] D. L. Applegate, R. E. Bixby, V. Chvátal and W. J. Cook, "The Problem," in *The Traveling Salesman Problem: a Computational Study*, vol. 17, Princeton, NJ, USA: Princeton Univ. Press, 2006, pp. 1-50.

[7] E. W. Weisstein, "Hamiltonian Cycle," MathWorld--A Wolfram Web Resource, [Online]. Available: https://mathworld.wolfram.com/HamiltonianCycle.html. [Accessed May 8 2024].

[8] "Hamiltonian Circuits," Lumen Learning, [Online]. Available: https://courses.lumenlearning.com/waymakermath4libarts/chapter/hamiltonian-circuits/#:~:text=Number%20of%20Possible%20Circuits,3%E2%8B%852%E2%8B%851. [Accessed May 8 2024].

[9] V. Boskov, "Traveling Salesman Problem with Quantum Optimization. Solutions and Perspectives.," Medium, June 17 2022. [Online]. Available: https://medium.com/@vasilybokov/traveling-salesman-problem-with-quantum-optimization-solutions-and-perspectives-59137f3241cd. [Accessed May 10 2024].

[10] M. Li, "The Variational Quantum Eigensolver, Implemented with IBM's Qiskit," Medium, May 23 2020. [Online]. Available: https://lzylili.medium.com/the-variational-quantum-eigensolver-c473c6dcd46. [Accessed May 11 2024].

[11] International Organization for Standardization, "ISO 9241-110:2020," May 2020. [Online]. Available: https://www.iso.org/standard/75258.html.

[12] International Organization for Standardization, "ISO 9241-112:2017," March 2017. [Online]. Available: https://www.iso.org/standard/64840.html.

[13] International Organization for Standardization, "ISO 9241-11:2018," March 2018. [Online]. Available: https://www.iso.org/standard/63500.html.

[14] International Organization for Standardization, "ISO 9241-171:2008," July 2008. [Online]. Available: https://www.iso.org/standard/39080.html.

[15] International Organization for Standardization, "ISO/IEC 27002:2022," March 2022. [Online]. Available: https://www.iso.org/standard/75652.html.

[16] F. Salas-Molina, . J. . A. Rodríguez-Aguilar and P. Díaz-García, "PyCaMa: Python for cash management," *RePEc: Research Papers in Economics.*, p. 9, February 2017.

[17] Fujitsu, " The case for quantum and quantum-inspired computing in financial services," 2019. [Online]. Available: https://www.fujitsu.com/global/digitalannealer/pdf/wp-da-financialsector-ww-en.pdf.

[18] A. Peruzzo, J. McClean, P. Shadbolt, M.-H. Yung, X.-Q. Zhou, P. J. Love, A. Aspuru-Guzik and J. L. O'Brien, "A variational eigenvalue solver on a photonic quantum processor," *Nature Communications*, vol. 5, 2014.

[19] E. Farhi, J. Goldstone and S. Gutmann, "A Quantum Approximate Optimization Algorithm," *arXiv*, 2014.

[20] T. Bennett, E. Matwiejew and J. B. Wang, "Quantum walk-based vehicle routing optimisation," *Frontiers in Physics*, p. 9, 2021.

[21] A. Maldonado and A. Montanez, "Quantum supply chain manager [Source code]," 2022. [Online]. Available: https://github.com/alejomonbar/Quantum-Supply-Chain-Manager.

[22] Manifesto for Agile Software Development, "Manifesto for Agile Software Development," 2001. [Online]. Available: https://agilemanifesto.org/iso/en/manifesto.html.

[23] Kissflow, "Kanban Methodology: The Simplest Agile Framework," 2023. [Online]. Available: https://kissflow.com/project/agile/kanban-methodology/.

[24] G. B. Dantzig and J. H. Ramser, "The Truck Dispatching Problem," *Management Science,* vol. 5, no. 1, pp. 80-91, 1959.

[25] C. E. Miller, A. W. Tucker and R. A. Zemlin, "Integer Programming Formulation of Traveling Salesman Problems," *J. ACM,* vol. 7, no. 4, p. 326–329, October 1 1960.

[26] IBM, " IBM ILOG CPLEX Optimizer," [Online]. Available: https://www.ibm.com/br-pt/products/ilog-cplex-optimization-studio/cplex-optimizer.

[27] R. J. Baxter, "Critical antiferromagnetic square-lattice Potts model," *R. Soc. Lond,* pp. 43-54, 1982.

[28] Qiskit optimization, "Converters for Quadratic Programs - Qiskit Optimization 0.6.1," 2024. [Online]. Available: https://qiskit-community.github.io/qiskit-optimization/tutorials/02_converters_for_quadratic_programs.html.

[29] G. Kochenberger, J.-K. Hao, F. Glover and e. al., "The unconstrained binary quadratic programming problem: a survey," *Journal of Combinatorial Optimization,* vol. 28, no. 1, pp. 58-81, 2014.

[30] J. Preskill, "Quantum Computing in the NISQ era and beyond," p. 79, 2018.

[31] A. Kandala, A. Mezzacapo, K. Temme, M. Takita, M. Brink, J. M. Chow and J. M. Gambetta , "Hardware-efficient variational quantum eigensolver for small molecules and quantum magnets," *Nature,* 2017.

[32] Y. Cao, J. Romero, J. P. Olson, M. Degroote, P. D. Johnson, . M. Kieferová, I. D. Kivlichan, T. Menke, . B. Peropadre, N. P. D. Sawaya and S. Sim, "Quantum Chemistry in the Age of Quantum Computing," *Chem. Rev.,* p. 119(19), 30 August 2019.

[33] Qiskit community, "Vehicle Routing - Qiskit Optimization 0.6.1," [Online]. Available: https://qiskit-community.github.io/qiskit-optimization/tutorials/07_examples_vehicle_routing.html.

[34] IBM, "Docplex.mp.model module — Docplex.MP: Mathematical Programming Modeling for Python V2.25 documentation," [Online]. Available: https://ibmdecisionoptimization.github.io/docplex-doc/mp/docplex.mp.model.html.

[35] A. Montanez, " Quadratic unconstrained binary optimization (QUBO)," February 2024. [Online]. Available: https://pennylane.ai/qml/demos/tutorial_QUBO/.

[36] OpenQAOA, "What is the QAOA? - OpenQAOA," [Online]. Available: https://openqaoa.entropicalabs.com/what-is-the-qaoa/#description-of-the-algorithm.

[37] Qiskit algorithms, "COBYLA | IBM Quantum Documentation," [Online]. Available: https://docs.quantum.ibm.com/api/qiskit/0.26/qiskit.algorithms.optimizers.COBYLA.

[38] Qiskit algorithms, "SPSA | IBM Quantum Documentation," [Online]. Available: https://docs.quantum.ibm.com/api/qiskit/0.25/qiskit.algorithms.optimizers.SPSA.

[39] A. Wack and D. McKay, "Updating how we measure quantum quality and speed," 2023. [Online]. Available: https://www.ibm.com/quantum/blog/quantum-metric-layer-fidelity.

[40] UOL-Economia, "Carro-forte," UOL, 2015. [Online]. Available: https://economia.uol.com.br/noticias/redacao/2015/08/19/carro-forte-leva-ate-jogador-de-futebol-viagra-e-bolinhas-da-mega-sena.htm.

[41] IBM Quantum Plataform, "Access quantum," [Online]. Available: https://www.ibm.com/quantum/pricing.

[42] IBM Quantum Documentation, "Processor types," [Online]. Available: https://docs.quantum.ibm.com/run/processor-types.

[43] IBM Quantum Research, "The hardware and software for the era of quantum utility is here," 2023. [Online]. Available: https://www.ibm.com/quantum/blog/quantum-roadmap-2033.

[44] NVIDIA, "NVIDIA Quantum," [Online]. Available: https://www.nvidia.com/pt-br/solutions/quantum-computing/.

[45] "Solving Problems with Quantum Samplers," D-Wave Systems Inc., [Online]. Available: https://docs.dwavesys.com/docs/latest/c_gs_3.html. [Accessed May 13 2024].

[46] D. J. Griffiths and D. F. Schroeter, Introduction to Quantum Mechanics, 3rd ed., Cambridge: Cambridge University Press, 2018.

[47] D. Redação, "Bradesco garante presença em 100% das cidades brasileiras," Exame, October 10, 2010. [Online]. Available: https://exame.com/negocios/bradesco-garante-presenca-100-cidades-brasileiras-513123/. [Accessed May 4, 2024].

[48] E. W. Weisstein, "Complete Graph," MathWorld--A Wolfram Web Resource, January 18 2024. [Online]. Available: https://mathworld.wolfram.com/CompleteGraph.html. [Accessed May 8 2024].

[49] "Big O Notation Explained with Examples," freeCodeCamp, February 1 2020. [Online]. Available: https://www.freecodecamp.org/news/big-o-notation-explained-with-examples/. [Accessed May 8 2024].

[50] T. E. o. E. Britannica, "NP-complete problem," Britannica, April 4 2024. [Online]. Available: https://www.britannica.com/science/NP-complete-problem. [Accessed May 10 2024].

[51] "Bra-Ket Notation," MathsIsFun, [Online]. Available: https://www.mathsisfun.com/physics/bra-ket-notation.html. [Accessed May 14 2024].

[52] "What Is Quadratic Programming?," MathWorks, [Online]. Available: https://www.mathworks.com/discovery/quadratic-programming.html#:~: [Accessed May 14 2024].

Observação: Projeto foi realizado em parceria com o Bradesco, contudo não foi feita nenhuma validação do Bradesco na solução apresentada.