Applying QAOA to the Oversubscribed Satellite Scheduling Problem

Marco Baioletti*
Dipartimento di Matematica e Informatica, Università degli studi di Perugia, Via Vanvitelli, 1, Perugia, 06123, Italy

Fabrizio Fagiolo † and Angelo Oddi ‡ and Riccardo Rasconi §
Instituto di Scienze e Tecnologie della Cognizione, CNR - Consiglio Nazionale delle Ricerche, Via Gian Domenico Romagnosi 18A, Roma, 00196, Italy

This paper explores the application of the Quantum Approximate Optimization Algorithm (QAOA) to the Oversubscribed Satellite Scheduling Problem (OSSP), a critical challenge in optimizing communication tasks between satellites and ground stations. The OSSP involves assigning tasks within predefined time windows while minimizing conflicts from overlapping assignments, a problem inherently combinatorial in nature. By formulating OSSP as a Quadratic Unconstrained Binary Optimization (QUBO) problem and translating it into a quantum circuit suitable for QAOA, we present an algorithm for the optimization of the QAOA's parameters as the number of circuit levels p increases, by retaining the best parameter values found at each iteration. Experimental evaluations on quantum simulators demonstrate the effectiveness of our approach, thus realizing the QAOA's potential in addressing complex scheduling tasks efficiently, providing insights into its scalability and applicability in real-world satellite operations.

I. Introduction

Quantum computing represents a paradigm shift in computation, leveraging the principles of quantum mechanics to perform tasks that would be infeasible or highly inefficient on classical systems; this idea is also known as *quantum supremacy* [1]. At the core of quantum computing lies the *qubit*, the quantum analogue of the classical bit. Unlike a bit, which can only exist in binary states 0 or 1, a qubit can exist in a *superposition* of these states, allowing it to represent both 0 and 1 simultaneously. This property exponentially increases the computational space that quantum systems can explore. In addition to superposition, quantum computing exploits two other fundamental properties of quantum mechanics: *entanglement* and *interference*. Entanglement allows qubits to be correlated in such a way that the state of

*Prof., Dipartimento di Matematica e Informatica, Università degli studi di Perugia, Via Vanvitelli, 1, Perugia, 06123, Italy. *marco.baioletti@unipg.it* (corresponding author)

†Dr., Instituto di Scienze e Tecnologie della Cognizione, CNR - Consiglio Nazionale delle Ricerche, Via Gian Domenico Romagnosi 18A, Roma, 00196, Italy. *fabrizio.fagiolo@istc.cnr.it* (corresponding author)

‡Senior Researcher, Instituto di Scienze e Tecnologie della Cognizione, CNR - Consiglio Nazionale delle Ricerche, Via Gian Domenico Romagnosi 18A, Roma, 00196, Italy. *angelo.oddi@cnr.it* (corresponding author)

§Senior Researcher, Instituto di Scienze e Tecnologie della Cognizione, CNR - Consiglio Nazionale delle Ricerche, Via Gian Domenico Romagnosi 18A, Roma, 00196, Italy. *riccardo.rasconi@istc.cnr.it* (corresponding author)

one qubit is intrinsically linked to the state of another, even at large distances. This allows for highly coordinated and efficient computations [2]. Quantum interference, on the other hand, is used to manipulate the probability amplitudes of quantum states, amplifying paths leading to correct solutions and eliminating erroneous ones. These unique properties make quantum computing particularly suitable for solving problems involving large combinatorial spaces. Unlike *classical* systems, which must evaluate each possible solution sequentially, quantum computers can process many potential solutions simultaneously, taking advantage of their parallelism to explore the solution space efficiently. In fact, quantum computing has the potential to revolutionize fields such as logistics, finance, materials science, and telecommunications, opening new frontiers for solving problems beyond the reach of classical computation.

According to the literature, two prominent paradigms in quantum computing are the *gate based* model, which enables universal quantum computation, and *quantum annealing*, which is primarily used for solving optimization problems [3]. We will consider the gate-based approach, which performs quantum computation by applying a sequence of quantum gates to a set of qubits. A sequence of quantum gates is a *circuit* and represents a quantum algorithm. Although promising, quantum gate-based computers are still in the so-called *noisy intermediate-scale quantum* (NISQ) era [4], characterized by limited qubit counts and susceptibility to errors. However, even in this era, quantum-classical hybrid algorithms such as the Quantum Approximate Optimization Algorithm (QAOA) are emerging as powerful tools to address complex optimization problems [5]. QAOA belongs to the Variational Quantum Algorithms (VQA) family, a class of hybrid quantum-classical algorithms designed to take advantage of the unique capabilities of gate-based quantum devices in the NISQ era [6]. The VQA combine the use of quantum computers to perform specific computations with classical computers to optimize variational parameters, offering a powerful and versatile approach to address optimization problems and other computational challenges [7, 8].

In this paper, we explore the opportunities offered by QAOA to solve a given satellite scheduling problem and discuss the potential impact of QAOA in this field, analyzing its performance in a prototypical scenario. In particular, we consider the oversubscribed satellite scheduling problem (OSSP), a critical novel challenge in optimizing communication tasks between satellites and ground stations by using a QAOA approach. The OSSP involves assigning tasks within predefined time windows while minimizing conflicts from overlapping assignments.

The paper is structured as follows. Section II describes the background literature, while Section III proposes a formal definition of the given satellite scheduling problem. Next, Section IV introduces the QAOA approach and Section V gives the details of the proposed quantum circuit to solve the given satellite problem. Finally, Section VII proposes an extensive empirical validation of the proposed approach and Section VIII traces the same conclusions and future line of work.

II. Related Works

In this paper, we focus on addressing planning and scheduling problems that are particularly relevant in the space domain, for example, see [9, 10]. Specifically, we consider the Oversubscribed Satellite Scheduling Problem (OSSP) using quantum resolution techniques. The potential ability of quantum algorithms to efficiently explore large solution spaces has led to significant progress in solving combinatorial optimization problems. Despite the limited size of the available quantum hardware, in the literature there exist a number of works highlighting how quantum computing has been used to solve difficult combinatorial problems in space domains [11], such as the Mars Lander Problem, consisting in the problem of scheduling the activities of a lander equipped with multiple scientific instruments and a robotic arm capable of interacting with its surroundings. In the latter case, the tasks include: (1) conducting scientific research to fulfill mission objectives, (2) transmitting collected data, and (3) performing operations to ensure its functionality (see [12]).

In recent years, as shown in [13], quantum computation has been established as a promising method to solve graph coloring problems[14] *, particularity through algorithms such as the Quantum Approximate Optimization Algorithm (QAOA)[5]. In addition, there exist a number of works in the literature showing that scheduling problems can generally be reduced to the Graph Coloring (GC) problem[15], which allows to leverage well-known techniques from graph theory (see next Section III, about the reduction of Graph Coloring to the given OSSP). In general, this reduction proves to be particularly effective in representing constraints as conflicts between graph vertices, a method widely studied in classical scheduling problems [16]. For example, recent studies have shown how graph coloring can model scenarios such as resource allocation, frequency assignment [17], or time slot allocation, highlighting its versatility in the application of combinatorial challenges [18]. The authors in [19] provide an example of a satellite range scheduling problem where a set of communication jobs must be assigned to a set of ground stations with the objective of minimizing the number of conflicting jobs, where heuristics based on efficient graph coloring methods are used.

In terms of works more strictly related to the contents of this paper, recent developments have applied quantum algorithms for solving Satellite Mission Planning Problems (SMPPs). For example, in [20] a quantum annealer is used to solve satellite scheduling problems for Earth observation, showing that a tuned quantum annealer approach can outperform the classical exact solver in some instances with sizes compatible with the used quantum hardware. As a further example of Earth observation scheduling, in [21] quantum algorithms have been applied, significantly outperforming basic greedy methods and paving the way to quantum-enabled solutions for future applications in the space industry. In [22] the authors use a quantum optimization method to solve SMPPs, evaluating both Quantum Annealing (QA) and Quantum Approximate Optimization Algorithm (QAOA) on realistic datasets, identifying how key factors like graph connectivity and constraint structure can influence performance. Another relevant work is [23],

*The decision version of the graph coloring problem asks whether for a given graph $G = (V, E)$ and integer k, a *k-coloring* exists, that is, a mapping that assigns to each vertex $v \in V$ a color in the set $\{1 \dots k\}$ such that the endpoints of every edge in $(u, v) \in E$ are assigned different colors.

where a hybrid classical-quantum approach is proposed to solve SMPPs, using QAOA approaches in conjunction with classical optimization methods. In particular, the work [23] highlights the potential of hybrid strategies to obtain higher quality and efficient solutions in complex satellite planning scenarios, further emphasizing the applicability of quantum techniques to the future of real-world problems.

III. Problem definition

A set of n satellites $SAT = \{S_1, S_2, \ldots, S_n\}$ is required to communicate with a set of k ground stations $GS = \{G_1, G_2, \ldots, G_k\}$ having unary capacity (i.e., each ground station G_i can communicate only with one satellite at a time). Each satellite S_t should perform a series of m_t communication tasks CT_t using a subset $GS_t \subseteq GS$ of the available ground stations GS. Each communication task c_i should be executed in a specific time window $[s_i, e_i]$. To simplify the notation, we denote by L_i the set of ground stations that can be used to perform the communication task c_i and by n_i the cardinality of L_i. Clearly, for $c_i \in CT_t$, L_i coincides with GS_t.

Denoting by $CT = CT_1 \cup CT_2 \cup \ldots \cup CT_n$ the set of all communication tasks and by m its cardinality $|CT|$, the problem is to assign a ground station $g(c_i) \in GS$ to each communication task $c_i \in CT$, such that $g(c_i) \in L_i$ and there are no conflicts. A *conflict* arises between two communication tasks c_i and c_j when they overlap in time and $g(c_i) = g(c_j)$, i.e. they are assigned the same ground station.

When the set of satellite communication tasks CT exceeds the capacity of ground stations, it is impossible to find non-conflicting assignments to each element in CT. In such cases, it becomes necessary to give up some communication tasks. In the *Oversubscribed Satellite Scheduling Problem* (OSSP), a *feasible* solution is defined as an assignment $g : CT \to GS \cup \{\perp\}$, where $\perp$ represents the decision to not assign a specific communication task. The feasibility condition requires that, for any pair of communication tasks $c_i, c_j \in CT$ overlapping in time, at least one of the following holds: $g(c_i) \neq g(c_j)$, is satisfied if c_i and c_j are assigned to different ground stations; $g(c_i) = \perp$ or $g(c_j) = \perp$, meaning that at least one task c_i or c_j is not assigned to any ground station. An *optimal* solution $g^\star$ is a feasible solution that maximizes the number of communication tasks assigned to the ground stations.

About the complexity of the considered scheduling problem, as shown in [24–26], the *list coloring* problem on *interval graphs*[27] [†] - an *NP-complete problem* - can be reduced to the satellite scheduling problem without oversubscription, which is a particular case of the OSSP. List coloring is a variation of the well-known *graph coloring* problem on undirected graphs $G(V, E)$, where each vertex $v \in V$ has a list of allowed colors $L(v)$. We can restrict the graph $G(V, E)$ to the set of *interval graphs* such that: (i) V is a set of intervals on the real line and each vertex $v \in V$ is an interval $v = [s_v, e_v]$; (ii) two vertices i and j are adjacent $(i, j) \in E$ if and only if they have a nonempty intersection $i \cap j \neq \emptyset$. Given a list coloring problem on an interval graph G, a corresponding OSSP can be generated as

[†] Interval graphs are perfect graphs generated from a set of intervals on the real line, with a vertex for each interval and an edge between vertices whose intervals intersect.

it follows: (i) the set $CT = V$; (ii) each satellite S_t performs a single communication task $c_t \in CT$, such that $CT_t = \{c_t\}$, $t = 1 \ldots m$; (iii) $GS_i = L(i)$ and $GS = GS_1 \cup GS_2 \cup \ldots \cup GS_m$. Hence, as the list coloring problem is *NP-complete*, the same holds for the OSSP.

IV. QAOA Algorithm

The Quantum Approximate Optimization Algorithm (QAOA) is a hybrid quantum-classical algorithm introduced in [5] in 2014. The algorithm is designed to solve combinatorial optimization problems, which are typically hard for classical computers. It is inspired by the adiabatic theorem in quantum mechanics, which states that a quantum system remains in its ground state if the Hamiltonian changes slowly enough. QAOA discretizes this evolution, using variational parameters to approximate the adiabatic path. QAOA leverages the principles of quantum mechanics, particularly quantum superposition and entanglement, to explore the solution space more efficiently than classical algorithms.

It is worth noting that this paper addresses the potential of using the QAOA algorithm to solve the OSSP scheduling problem. Indeed, the so-called *quantum supremacy* of the QAOA algorithm over the best classical algorithm to solve the OSSP will be probably achieved when the number of available qubits is on the order of thousands or tens of thousands. To support this estimation, we cite a well-known paper [28], such that for solving the Max-Cut problem[‡], a particular case of Graph Coloring, the authors estimates the crossover value to be between several hundreds and a few thousands qubits for QAOA over one of the best classical algorithm for solving the Max-Cut. Indeed, as the current biggest quantum processors have only a few hundreds of qubits, our study is only preparatory for the future applications of quantum computing.

QAOA works by encoding the optimization problem in a problem Hamiltonian (H_C), a mathematical object that describes the total energy of a quantum system. The algorithm iteratively refines a set of parameters to guide the quantum system towards states that represent good solutions of the optimization problem. It operates in two main stages: the initialization stage and the iterative application of quantum operators, guided by classical optimization.

More precisely, the problem Hamiltonian H_C encodes the cost function of the optimization problem; it is typically a diagonal operator in the computational basis, where each eigenvalue corresponds to the cost of a classical solution. The goal of quantum optimization is to find the ground state (the state with the lowest energy) of H_C, which corresponds to the optimal solution of the problem.

For a combinatorial optimization problem, H_C can be constructed according to the following steps:

1) Definition of the cost function: the cost function $C(z)$ is defined over binary variables $z \in \{0, 1\}$. For example, in a Max-Cut problem, $C(z)$ represents the number of edges between two sets of vertices;

2) Translation to Quantum Operators: as mentioned above, the cost function is translated into a diagonal operator in

[‡]The Max-Cut problem: given a graph $G = (V, E)$, find a subset S of the vertex set V such that the number of edges between S and the complementary subset $(V - S)$ is maximized.

the computational basis. If $z = (z_1, z_2, \ldots, z_n)$ represents a string of bits, then

$$H_C = \sum_{ij} J_{ij} Z_i Z_j + \sum_i h_i Z_i$$

where Z_i are the Pauli-Z operators that act on the i-th qubit, J_{ij} represents the interaction coefficients between qubits i and j, and h_i are local fields.

The Mixing Hamiltonian, denoted as H_B, promotes transitions between different states in the solution space, allowing the algorithm to explore various configurations. A common choice for H_B is:

$$H_B = \sum_i X_i$$

where X_i is the Pauli-X operator acting on the i-th qubit. The Pauli-X operator, also known as the bit-flip operator, changes the state of a qubit from $|0\rangle$ to $|1\rangle$ and vice versa. This Hamiltonian encourages the system to move away from local minima and explore a broader region of the solution space.

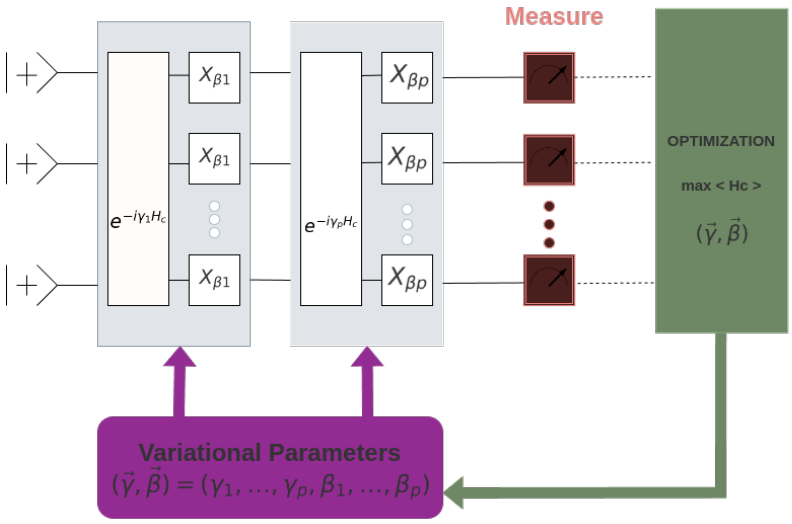

Figure 1 Scheme of a QAOA with p levels.

The QAOA, whose scheme is graphically presented in Figure 1), operates through the following steps:

- **Initialization**: preparation of the initial quantum state as an equal superposition of all possible states:

$$|\psi_0\rangle = H^{\otimes n} |0\rangle^{\otimes n}$$

where H is the Hadamard gate applied to each qubit, thus creating the state:

$$|\psi_0\rangle = \frac{1}{\sqrt{(2^n)}} \sum_{z \in \{0,1\}^n} |z\rangle$$

- **Alternating Application of Operators**: application of the problem Hamiltonian H_C and the Mixing Hamiltonian H_B in an alternating fashion for p layers (or steps), with parameters γ and β that are to be optimized. The quantum state after p layers is:

$$|\psi_p(\vec{\gamma}, \vec{\beta})\rangle = U_B(\beta_p)U_C(\gamma_p)\ldots U_B(\beta_1)U_C(\gamma_1)|\psi_0\rangle$$

  where both units are defined as:

$$U_C(\gamma) = e^{-i\gamma H_C}, U_B(\beta) = e^{-i\beta H_B}$$

- **Parameter optimization**: the two sets of parameters $\vec{\gamma} = (\gamma_1, \gamma_2, \ldots, \gamma_p)$ and $\vec{\beta} = (\beta_1, \beta_2, \ldots, \beta_p)$ are optimized using a classical optimizer. The objective is to maximize the expected value of the cost function:

$$\langle C \rangle = \langle \psi_p(\vec{\gamma}, \vec{\beta})|H_C|\psi_p(\vec{\gamma}, \vec{\beta})\rangle$$

- **Measurement and Solution Extraction**: after evolving the quantum state with the optimal parameters, the final state $|\psi_p(\vec{\gamma^*}, \vec{\beta^*})\rangle$ is measured in the computational basis. The measurement results provide candidate solutions to the optimization problem. The bit string corresponding to the lowest measured cost function value is selected as the approximate solution.

As explained in the following sections, the QAOA algorithm relies on classical optimization (e.g., COBYLA, SPSA) to minimize the expected value of the cost Hamiltonian. While convergence is not guaranteed in general, empirical results show good performance for many problems. Generally, performance improves with the circuit's depth[§], though practical limitations, such as noise, constrain scalability.

V. Circuit description

A. Mathematical formulation

Many combinatorial optimization problems are particular cases of the Quadratic Constrained Binary Optimization (QUBO) problem, that aim at minimizing the following function

$$g(x_1, \ldots, x_n) = \sum_{i=1}^{n} \sum_{j=1}^{n} B_{ij} x_i x_j \tag{1}$$

[§]The *depth* of a quantum circuit is the number of time steps (layers) required to execute all gates, assuming gates on different qubits can run in parallel.

where $x_1, \ldots, x_n$ are binary variables and $\mathbf{B}$ is a symmetric matrix.

The Oversubscribed Satellite Scheduling problem can be formulated as a QUBO problem in the following way.

Following up on the problem description provided in Section III, for each communication task $c_j \in CT$, $n_j + 1$ binary variables $x_{j,0}, x_{j,1}, \ldots, x_{j,n_j}$ are introduced where, for each $x_{j,l}$ variable: (i) $j = 1, \ldots, m$; (ii) $l = 0 \ldots, n_j$ with n_j corresponding to the cardinality $|L_j|$ and where L_j is the set of ground stations available for the communication task c_j. In each feasible solution exactly one variable among $x_{j,0}, x_{j,1}, \ldots, x_{j,n_j}$ is 1 and the other are 0.

If $x_{j,l} = 1$, for an index $l = 1, \ldots, n_j$, it means that c_j is assigned to j–th ground station in L_j. If $x_{j,0} = 1$, then c_j is not performed.

The total number of binary variables is $m + \sum_{j=1}^{n} n_j$, which is at most $m(1 + k)$.

Given two overlapping communication tasks $c_i, c_j \in CT$, let NG_{ij} be the set of indices (h, l), where $h = 1, \ldots, n_i$, and $l = 1, \ldots, n_j$, such that the h–th ground station of L_i is equal to the l–th ground station of L_j.

Hence, in that situation, $x_{ih} = x_{jl} = 1$ means a conflict because the two communication tasks overlap and are assigned to the same ground station.

It is easy to see that the sets NG_{ij} can be computed in polynomial time and the size $|NG_{ij}|$ is always less than or equal to the number of ground stations k. Let E denote the set of pairs (i, j) such that $NG_{ij} \neq \emptyset$.

Since the OSSP is a constrained problem, we encode the constraint as a penalization term, which forces the solution that minimizes the objective function to be a feasible solution. Hence, the objective function to be minimized is composed by two terms. The first term is the number of communication tasks not performed, while the second term is the penalization function.

$$f(\vec{x}) = \sum_{j=1}^{n} x_{j0} + A \cdot \sum_{(i,j)\in E} \sum_{(h,l)\in NG_{ij}} x_{ih} x_{jl} \tag{2}$$

The parameter A is used to force all the feasible solutions to have no conflicts, while they can have some communication tasks not executed. In our experiments we set $A = \frac{3}{2}$.

B. Satellite Scheduling: quantum circuit

The OSSP can be solved using a variant of the QAOA, called the Quantum Alternating Optimization Ansatz [29], which optimizes a parametric circuit similar to the ordinary QAOA.

The main difference between the Quantum Alternating Optimization Ansatz and the Quantum Approximate Optimization Algorithm is that the latter can be used only with problems where all the bit strings are feasible solutions, while the former is able to encode constrained problems, such as OSSP.

Let x_{jl} be a binary variable associated with a qubit Q_{jl}. The qubits Q_{jl} are organized into m registers, denoted as $\mathcal{R}_1, \ldots, \mathcal{R}_m$, where each register $\mathcal{R}_j$ consists of the qubits $Q_{j0}, \ldots, Q_{j,n_j}$.

From the objective function (2) it is easy to create the cost Hamiltonian H_C by applying to f the transformations

$$x_{jl} \to \frac{1}{2}(I_{jl} - Z_{jl}), \tag{3}$$

where I_{jl} and Z_{jl} are, respectively, the identity gate and the Z gate acting on the qubit Q_{jl}. This transformation replaces the binary variables of the QUBO formulation with the Z operators, whose eigenvalues are ± 1.

The preparation of the initial state creates in each register $\mathcal{R}_j$ a superposition of all binary strings of size $n_j + 1$ with 1 one and n_j zeros. This can be done by using the W_N gate described in [30].

The generic Phase-Shifting block for the level p corresponds to $\exp(-i\gamma_p H_C)$, where H_C is the cost Hamiltonian. The block is composed by the following gates:

- A gate $R_Z(\gamma_p)$ acting on each qubit Q_{i0}, for $i = 1, \ldots, m$;
- A gate $R_{ZZ}(A\gamma_p)$ acting on each pair of qubits Q_{ih} and Q_{jl}, for $(i, j) \in E$ and for $(h, l) \in NG_{ij}$;
- Two gates $R_Z(A\gamma_p)$ acting on the qubit Q_{ih} and Q_{jl}, for $(i, j) \in E$ and for $(h, l) \in NG_{ij}$.

It is important to note that all the gates in the PS block commute with each other, being rotations on the z-axis and hence represented as diagonal matrices.

The Mixing block for level p corresponds to $\exp(-i\beta_p H_M)$, where H_M is the mixing Hamiltonian. Here we adopt the same scheme used for the Graph Coloring problem in [29].

Namely, in each register $\mathcal{R}_j$, the values $0, 1, \ldots, n_j$ are arranged in the ring

$$0 \to 1 \to \cdots \to (n_j - 1) \to n_j \to 0.$$

For each qubit Q_{jl}, with $l = 0, 1, \ldots, n_j$, let h the successor of l, i.e. $h = (l + 1) \mod (n_j + 1)$, then the Mixing block comprises the two gates $R_{XX}(\beta_p)$ and $R_{YY}(\beta_p)$ that act on the qubits Q_{jl} and Q_{jh}.

Since the gates $R_{XX}(\beta_p)$ and $R_{YY}(\beta_p)$ do not commute, a specific order has to be defined. We employ the same strategy used in [29] ("*Parity single-qudit ring mixer*")[¶].

Specifically, the unitary transformation

$$U_j(\beta_p) = U_{j,\text{last}}(\beta_p) U_{j,\text{even}}(\beta_p) U_{j,\text{odd}}(\beta_p)$$

is applied to each register R_j. Transformation $U_{j,\text{even}}(\beta_p)$ is defined as

$$U_{j,\text{even}}(\beta_p) = \prod_{l \text{ even}} \exp\left(-i\beta_p (X_{j,l} X_{j,l+1} + Y_{j,l} Y_{j,l+1})\right)$$

[¶]As opposed to the qubit, a *qudit* is a quantum system capable of existing in and being measured in more than two distinct states.

where $X_{j,l}$ and $Y_{j,l}$ are the gates X and Y applied to l–th qubit of $\mathcal{R}_j$, respectively. The unitary transformation $U_{j,\text{odd}}$ is analogously defined.

Finally, for n_j even

$$U_{j,\text{last}} = \exp\left(-i\beta_p (X_{j,n_j} X_{j,0} + Y_{j,n_j} Y_{j,0})\right),$$

otherwise it is the identity operator.

It is important to notice due to the use of the W_N gates in the initialization phase and the particular structure of the mixing blocks, the circuit produces only solutions in which each communication task is assigned to at most one ground station.

The final step of the circuit comprises the measurement of the registers R_j, from which it is possible to extract a solution of the OSSP instance.

VI. Optimization Process

Algorithm 1 presents the pseudo-code of our resolution process. This process deals with the optimization of the parameters γ and β for a multilevel quantum circuit, progressively minimizing the energy calculated using the Hamiltonian H_C. It is indeed known from the paper that firstly introduced the QAOA[5] that the returned solutions improve as the levels p of the quantum circuits are iteratively increased [31, 32]. Therefore, the performance of the algorithm will be evaluated as the value of p iteratively increases, by optimizing the γ and β values at each iteration. As will be clarified in the following, the central idea of the algorithm is the utilization of the best γ and β values found at level k as the baseline for the optimization computed at the subsequent level $k + 1$.

Algorithm 1 Optimization Algorithm

Require: Problem Hamiltonian H_C, number of levels *nLevels*, number of initial samples *nInitSamples*, number of best pairs *nBestPairs*, number of new pairs *nNewPairs*

Output: Array of best $\langle \gamma^*, \beta^* \rangle$ pairs *bestPairs*

Optimize(H_C, *nLevels*, *nInitSamples*, *nBestPairs*, *nNewPairs*)

bestPairs ← ∅, *currentPairs* ← ∅
for $p \leftarrow 1$ **to** *nLevels* **do**
  if $p = 1$ **then**
    currentPairs ← randomInit(*nInitSamples*, $[0, \pi]$)
  else
    currentPairs ← randomUpdate(*bestPairs*, *nNewPairs*, $[0, \pi]$)
  end if
  energies ← evaluateEnergy(*Hc*, *currentPairs*)
  sortedPairs ← sortByEnergy(*currentPairs*, *energies*)
  bestPairs ← minEnergyPairs(*sortedPairs*, *nBestPairs*)
  bestPairs ← runOptimizer(*Hc*, *bestPairs*, *COBYLA*)
end for
return *bestPairs*

The optimization procedure starts with the analysis of the first level of the circuit ($p = 1$), and proceeds by iteratively

considering the subsequent levels ($1 < p \leq nLevels$). Relatively to the level $p = 1$ (lines 3-4), the process begins with the generation of *nInitSamples* random $\langle \gamma, \beta \rangle$ pairs through the RANDOMINIT procedure, where we sample $\langle \gamma, \beta \rangle$ in ($[0, \pi]$) to provide several starting points for the classical optimizer; moreover, because QAOA layers are π periodic in γ and β, $[0, \pi]$ serves as fundamental domain; sampling beyond it would only introduce redundant and dynamically equivalent points. The generated pairs are then stored in the $currentPairs$ matrix (line 4). When $p = 1$ the *currentPairs* matrix is basically a column vector of dimensions ($nInitSamples \times p$) whose rows are composed of a single $\langle \gamma, \beta \rangle$ pair. As the value of p increases during the execution of Algorithm 1, new columns are added to *currentPairs*, and each row of such matrix will be treated as a $\langle \gamma, \beta \rangle$ *pair sequence* that retains the values of the γ and β parameters of the current p-level quantum circuit.

Subsequently (line 8 of the algorithm), each $\langle \gamma, \beta \rangle$ pair sequence corresponding to the rows of the *currentPairs* matrix is individually evaluated by the EVALUATEENERGY function. Basically, the EVALUATEENERGY function performs *nInitSamples* executions of a QAOA algorithm (see Section IV) using each pair sequence contained in the *currentPairs* matrix as a starting point. The main goal of this function is to estimate the expected value of the Hamiltonian H_C associated with the state $|\psi_p(\gamma, \beta)\rangle$ returned by the quantum circuit, as an estimate of the energy of the associated quantum system. In other words, such a value is an indicator of the quality of each $\langle \gamma, \beta \rangle$ pair sequence: the lower the assessed value of the energy, the better the sequence. At the end of its execution, the function EVALUATEENERGY returns the *energies* array (line 8), whose elements will store the energy values associated with each original pair sequence.

The *energies* array is then used by the SORTBYENERGY function (line 9) to sort all the pair sequences belonging to the *currentPairs* matrix in ascending order of energy, and saving them in the $sortedPairs$ matrix.

In the next step (line 10), the MINENERGYPAIRS procedure selects the first *nBestPairs* rows of the *sortedPairs* matrix and saves them in the $bestPais$ matrix, for further refinement. Indeed, these pairs sequences are a promising starting point for the subsequent optimization process, which is performed by the RUNOPTIMIZER procedure (line 11). The RUNOPTIMIZER function works as follows. It applies a local optimizer (i.e., COBYLA[‖]) to each row of the *bestPairs* matrix, further reducing the energy function associated with the Hamiltonian H_C, and returning an improved set of $\langle \gamma, \beta \rangle$ pair sequences that will substitute the previous values of the *bestPairs* matrix (line 11).

Once the $p = 1$ iteration of the optimization loop is complete, the process continues with the execution of the RANDOMUPDATE function (line 6). The RANDOMUPDATE function operates in a similar way as the RANDOMINIT function at line 4: it accepts the $bestPairs$ matrix computed at the previous iteration as well as the $nNewPairs$ parameter, and returns an updated version of the *currentPairs* matrix of dimensions ($[nBestPairs \cdot nNewPairs] \times p$), constructed as follows. Each of the $nBestPairs$ rows of the original *bestPairs* matrix will be copied $nNewPairs$ times into the new *currentPairs* matrix, ultimately resulting in a matrix of dimensions ($[nBestPairs \cdot nNewPairs] \times [p - 1]$).

[‖] *Constrained Optimization BY Linear Approximations*, a derivative-free optimization algorithm commonly used in variational quantum algorithms for parameter tuning.

Subsequently, a new pair $\langle \gamma, \beta \rangle$, whose values will be randomly chosen in $[0, \pi]$, will be added to each of these rows, ultimately returning a new *currentPairs* matrix of dimensions $([nBestPairs \cdot nNewPairs] \times p])$.

At the end of the optimization loop (line 13), the *bestPairs* matrix of dimensions $(nBestPairs \times nLevels)$ containing the best $\langle \gamma, \beta \rangle$ pair sequences is finally returned.

VII. Experiments

For the experimental part of this study, we generated a set of test instances using a satellite scheduling problem generator, a tool designed to synthesize different scenarios related to communications between satellites and ground stations.

Each scenario consists of multiple satellites, where each satellite generates a single communication task that must reach a ground station within a specific time window. The time windows are randomly generated to better reflect the real-world conditions. The generator returns problem instances expressed in terms of (*conflict graphs*) where each node represents a satellite and each satellite performs a communication task to one of the available ground stations. The edges of a conflict graph connect pairs of nodes representing ground stations whose communication requests on behalf of the satellites may overlap in time. This overlap indicates a possible conflict, which occurs when two satellites request two communications that should be (even partially) served by the same ground station.

Each generated conflict graph is saved in a file whose name reflects the number of ground stations, satellites, and possible conflicts that characterize the instance. For example, the instance *Problem_**x**Sat_**y**Gs_**c**_**n*** is characterized by **x** satellites, **y** ground stations, and **c** conflict(s); finally, **n** is the instance's index.

A. Instances Description

In particular, we focused primarily on two types of problem instances: those where a complete assignment that resolves all conflicts can be found (type 0 instances), and those where such a complete assignment does not exist, and therefore it is necessary to give up one communication request (type 1 instances). As a measure of hardness, we calculated for each problem instance the probability P_{opt} of finding an optimal solution and the probability P_{unf} of finding an unfeasible solution, using a random search. Since the instances are small, these probability values can be computed with an exhaustive search; in particular, P_{opt} is obtained as the ratio between the number of optimal solutions found and the number of all possible solutions, while P_{unf} is obtained by dividing the number of unfeasible solutions by the number of all possible solutions. Clearly, the greater the difference between P_{opt} and P_{unf}, the harder the problem instance. The problem instances for our experimental campaign are selected by choosing 3 instances of type 0 and 3 instances of type 1 characterized by a probability $P_{\text{opt}} \leq 3\%$.

A detailed overview of the selected problem instances is presented in Table 1 below.

Table 1 Instances

Instance Name	Qubits	Instance type	P_{opt}	P_{unf}
Problem_5Sat3Gs_0_4	16	0	0.025	0.481
Problem_5Sat3Gs_0_6	16	0	0.019	0.537
Problem_5Sat3Gs_0_12	16	0	0.012	0.568
Problem_5Sat3Gs_1_8	16	1	0.025	0.733
Problem_5Sat3Gs_1_5	16	1	0.019	0.741
Problem_5Sat3Gs_1_4	16	1	0.012	0.796

Each instance represents a unique configuration of the satellite planning problem, with specific characteristics that influence the complexity of the optimization process. In the table, the column *Instance type* reports the instance type (0 or 1), whose value coincides with the minimal number of conflicts present in the related optimal solution. We limited our analysis to instances of type 0 and 1, because a larger number of conflicts would produce unrealistic instances, with at least 40% of not performed tasks.

Indeed, it should be noted that all the instances listed in the table require 16 qubits; simulating a larger number of qubits would result in an excessive use of computational resources, making the process computationally impractical [33]. The limit of 16 qubits was then decided as a reasonable compromise between resolution feasibility and significance of the tackled instances.

Finally, the columns P_{opt} and P_{unf}, respectively, describe the probability of finding an optimal solution and an unfeasible solution, which also determines the difficulty of each instance. In our case, all instances in Table 1 are listed in increasing order of difficulty (P_{opt}).

B. System and Simulation Environment

The simulations were performed on a system equipped with an Intel(R) Xeon(R) E5-2620 v4 processor running at 2.10 GHz, with a total of 32 threads and 32 GB of RAM. The operating system used was Ubuntu 20.04.5 LTS with the Linux kernel version *5.15.0-60-generic*. The simulation environment was based on Python 3.10.12 and the Qiskit library version 1.1.0.

1. Algorithm Parameter Settings

The experimental campaign was executed based on the following setup. The number of initial samples was set to $nInitSample = 100$; the maximum number of levels for each quantum circuit was set to $nLevels = 8$, the number of best pair sequences selected at each iteration was set to $nBestPairs = 10$. In addition, the number of new pairs is set to $nNewPairs = 10$. It should be emphasized that the penalty factor identified as A in the objective function (2) plays a critical role in balancing the feasibility and quality of the solution. Setting A to a low value allows the algorithm to find many optimal solutions, but also increases the production of unfeasible solutions. In contrast, a high value for A significantly decreases the production of unfeasible solutions, at the cost of reducing the probability of finding optimal solutions. After the execution of several preliminary tests, a suitable value for A was determined to be 1.5. This value

ensures that unfeasible solutions are sufficiently penalized to be worse than the best feasible solution (especially for instances whose optimal value is 1). This approach reflects a commonly used strategy in optimization problems to appropriately balance the trade-off between feasibility and solution quality.

Another key component to configure in the QAOA algorithm is the classical optimizer, which is responsible for tuning the variational parameters. Among the optimizers we evaluated including derivative-free methods such as Nelder-Mead and Powell, as well as gradient-based techniques like SPSA (Simultaneous Perturbation Stochastic Approximation) a stochastic optimization scheme that estimates search directions via simultaneous random perturbations of all parameters, we selected COBYLA (Constrained Optimization BY Linear Approximations) as the most effective choice. As shown in Table 2, which reports the average results over 1000 iterations, COBYLA consistently outperformed SPSA in all instances of the benchmark. In particular, COBYLA achieved significantly higher mean optimal solution probabilities (mean P_{opt}), significantly lower mean infeasibility rates (mean P_{unf}), and better mean energy values (mean E). In our experiments, COBYLA consistently outperformed all other candidates. Given its superior performance and its ability to handle constraints without requiring gradient information a particularly relevant feature for the OSSP problem addressed by QAOA, we adopt COBYLA as the primary optimizer for all experiments presented in this paper.

Table 2 Comparison between SPSA and COBYLA

Instance	SPSA			COBYLA		
	mean P_{opt}	mean P_{unf}	mean E	mean P_{opt}	mean P_{unf}	mean E
0_4	0.317	0.279	-3.595	0.930	0.014	-4.670
0_6	0.268	0.421	-4.049	0.887	0.014	-5.360
0_12	0.215	0.355	-4.005	0.906	0.023	-5.364
1_4	0.041	0.575	-5.682	0.299	0.155	-7.005
1_5	0.072	0.602	-4.634	0.744	0.103	-6.434
1_8	0.092	0.584	-5.363	0.787	0.162	-7.315

C. Experimental results

In our experiments, we applied our algorithm to a set of optimization problems to evaluate its effectiveness in finding optimal solutions. In this section, we will describe the best experimental results obtained and compare them with the performance obtained using a random approach. The results show how well the algorithm performs in terms of its ability to locate the optimal solution (P_{opt}) and avoid unfeasible solutions (P_{unf}). Furthermore, we assessed the energy values (E) associated with the solutions found, providing further insight into the quality of optimization.

1. Best results obtained with Algorithm 1

Table 3 reports the results obtained in the three instances of type 0 we selected for our experimentation, namely **5Sat3Gs_0_4**, **5Sat3Gs_0_6** and **5Sat3Gs_0_12**, presented in increasing order of difficulty. It should be emphasized that these results are obtained by allowing the COBYLA optimizer used in line 11 of Algorithm 1 with a maximum iteration limit $maxiter = 3000$. Parallel experiments have been carried out using values $maxiter = 1000$ and $maxiter = 2000$,

Table 3 Results for the instances of type 0

p	mean P_{opt}	mean P_{unf}	mean E	best P_{opt}	best P_{unf}	best E
			Instance: 5Sat3Gs_0_4			
1	0.300	0.115	-3.693	0.306	0.112	-3.693
2	0.617	0.098	-4.216	0.630	0.091	-4.216
3	0.833	0.029	-4.549	0.848	0.027	-4.549
4	0.862	0.018	-4.590	0.875	0.012	-4.595
5	0.879	0.016	-4.616	0.890	0.014	-4.616
6	0.907	0.009	-4.643	0.915	0.008	-4.643
7	0.925	0.008	-4.669	0.951	0.006	-4.687
8	**0.961**	**0.007**	**-4.702**	**0.970**	**0.005**	**-4.702**
			Instance: 5Sat3Gs_0_6			
1	0.146	0.218	-3.976	0.161	0.197	-4.020
2	0.444	0.111	-4.722	0.458	0.107	-4.722
3	0.663	0.074	-5.061	0.672	0.077	-5.061
4	0.685	0.060	-5.087	0.694	0.057	-5.087
5	0.735	0.043	-5.186	0.750	0.042	-5.186
6	0.789	0.026	-5.257	0.803	0.024	-5.257
7	0.872	0.015	-5.342	0.880	0.015	-5.342
8	**0.904**	**0.011**	**-5.387**	**0.912**	**0.008**	**-5.387**
			Instance: 5Sat3Gs_0_12			
1	0.098	0.209	-3.889	0.107	0.218	-3.889
2	0.304	0.125	-4.527	0.316	0.119	-4.527
3	0.530	0.130	-4.859	0.547	0.116	-4.859
4	0.613	0.104	-4.981	0.663	0.092	-5.026
5	0.681	0.071	-5.084	0.743	0.046	-5.156
6	0.841	0.031	-5.284	0.849	0.032	-5.284
7	0.857	0.030	-5.298	0.892	0.028	-5.347
8	**0.907**	**0.021**	**-5.371**	**0.924**	**0.019**	**-5.376**

conceptually obtaining results that are qualitatively identical to those of Table 3 and characterized by improvements of mere quantitative nature. For this reason, the complete result sets have been omitted for reasons of space; some examples will be however presented for comparison at the end of the current section.

Relatively to the **5Sat3Gs_0_4** instance, the optimizer demonstrates a steady improvement in performance as p increases. The mean probability to find the optimal solution increases from 0.300 in $p = 1$ to 0.961 in $p = 8$, while the mean probability to find unfeasible solutions decreases significantly from 0.115 to 0.007. The mean energy values show a consistent improvement, starting from -3.693 at $p = 1$ and reaching the value of -4.702 at $p = 8$. This suggests that the optimizer is effectively converging with higher values of p, taking advantage of the increased number of iterations. It is also interesting to observe how the evolution of the best values (columns 5, 6 and 7 in the table) follows the dynamics of the average values.

A similar pattern is also observed in the case of the **5Sat3Gs_0_6** instance, although the lower initial values indicate the presence of a more challenging optimization space. At $p = 1$, the mean P_{opt} starts at 0.146 and increases to 0.904 at $p = 8$, while the probability of unfeasible outcomes drops from 0.218 to 0.011. We also observe that the energy values undergo a steady improvement, passing from -3.976 to -5.387. These results confirm the role of additional iterations in facilitating the convergence towards higher-quality solutions.

The **5Sat3Gs_0_12** instance's results start with the lowest mean probability of finding optimal solutions, reflecting the higher complexity, and representing the most complex scenario among the three. Indeed, the mean P_{opt} starts with a modest 0.098 at $p = 1$, but finally reaches the value of 0.907 as p is increases to $p = 8$. On the other hand, the probability of unfeasible solutions decreases from 0.209 to 0.021, and the energy value improves significantly from -3.889 to -5.371. Despite the greater complexity, our optimization procedure manages to achieve substantial progress

provided a sufficient number of iterations.

Table 4 Results for the instances of type 1

p	mean P_{opt}	mean P_{unf}	mean E	best P_{opt}	best P_{unf}	best E
			Instance: 5Sat3Gs_1_8			
1	0.109	0.507	-5.723	0.118	0.500	-5.723
2	0.289	0.277	-6.304	0.299	0.295	-6.304
3	0.431	0.204	-6.632	0.445	0.203	-6.632
4	0.527	0.190	-6.856	0.537	0.186	-6.856
5	0.687	0.191	-7.078	0.708	0.180	-7.078
6	0.733	0.169	-7.207	0.750	0.160	-7.207
7	0.737	0.166	-7.208	0.754	0.155	-7.208
8	**0.790**	**0.158**	**-7.326**	**0.803**	**0.145**	**-7.326**
			Instance: 5Sat3Gs_1_5			
1	0.092	0.472	-5.080	0.098	0.478	-5.080
2	0.245	0.255	-5.612	0.264	0.254	-5.612
3	0.387	0.161	-5.886	0.402	0.147	-5.886
4	0.502	0.123	-6.108	0.522	0.124	-6.108
5	0.606	0.112	-6.260	0.617	0.104	-6.260
6	0.675	0.113	-6.349	0.687	0.106	-6.349
7	0.740	0.107	-6.433	0.752	0.106	-6.433
8	**0.742**	**0.101**	**-6.438**	**0.757**	**0.091**	**-6.438**
			Instance: 5Sat3Gs_1_4			
1	0.031	0.454	-5.925	0.036	0.430	-5.925
2	0.041	0.334	-6.096	0.045	0.344	-6.096
3	0.071	0.360	-6.387	0.081	0.361	-6.387
4	0.324	0.186	-7.092	0.342	0.188	-7.092
5	0.462	0.206	-7.385	0.478	0.200	-7.385
6	0.517	0.149	-7.567	0.537	0.143	-7.567
7	0.607	0.157	-7.702	0.624	0.146	-7.702
8	**0.658**	**0.170**	**-7.815**	**0.673**	**0.166**	**-7.815**

The results in Table 4 report the performance of our algorithm for the problem instances of type 1, again in increasing order of difficulty. As explained in Section VII.A, these are the instances for which a complete assignment does not exist, and whose resolution involves renunciation of one communication request.

Relatively to the **5Sat3Gs_1_8** instance, the optimizer achieves the most consistent performance gains. The mean P_{opt} increases from 0.109 at $p = 1$ to 0.790 at $p = 8$, while the mean P_{unf} decreases from 0.507 to 0.158. Currently, the energy improves from -5.723 to -7.326.

The **5Sat3Gs_1_5** instance shows slightly weaker improvements compared to the previous instance. At $p = 1$, the mean P_{opt} is 0.092, and increases to 0.742 at $p = 8$. Similarly, P_{unf} decreases from 0.472 to 0.101, and the energy reaches -6.438. However, this instance still exhibits a clear trend in which the increase of p yields a steady and significant improvement in all performance metrics.

Continuing with the **5Sat3Gs_1_4** instance, the results still indicate a gradual progress as p increases, despite the instance is the most difficult among the three. The mean probability of finding the optimal solution starts at 0.031 for $p = 1$ and increases to 0.658 at $p = 8$. Consequently, the mean probability of unfeasible solutions reduced from 0.454 to 0.170, while the energy improves significantly from -5.925 to -7.815. These values suggest that, although the convergence is slower compared to previous instances, the optimizer still benefits significantly from higher p values.

These results highlight the optimizer's effectiveness in tackling more challenging instances, achieving stable

convergence and high-quality solutions as p increases. To have a more concrete assessment of the efficacy of our algorithm, we compared the previous performance with a completely random version of Algorithm 1. Basically, the random version of our algorithm can be obtained by eliminating the RANDOMUPDATE function (line 6) from the Algorithm 1 (and consequently eliminating the *if-then-else* statement). In this way, the $\langle \gamma, \beta \rangle$ pairs are randomly selected at each iteration of the algorithm for all the p levels of the quantum circuit, instead of being updated based on the best values retained from the previous iteration.

The obtained results relative to all the 6 instances are reported in Table 5. Observing these results, the following conclusions can be drawn: (i) our algorithm performs significantly better in terms of overall performance with respect to the random case; (ii) the results obtained from the random approach exhibit a more *noisy* behavior, as the performance seems to be much more independent from the p level whereas, according to the Adiabatic Theorem (in the ideal case of absence of noise) the performance should steadily improve with the number of levels p[**]

2. Using different values of maxIter

It should be noted that in our experimentation, Algorithm 1 was executed 3 times allowing COBYLA to exploit a different number of max iterations ($maxIter \in \{1000, 2000, 3000\}$). The obtained results for all the values of *maxIter* relatively to the 3 instances of Type 0 are reported in Table 6 while the results for the 3 instances of Type 1 are reported in Table 7. It should be noted that despite such experimentation was carried out in all the 6 instances and for all levels of p ranging from 1 to 8, the tables report only the results for the $p = 8$ case, in order not to make the description too heavy; indeed, the results followed the same trend for all other p values as well. The results shown in Tables 6 and 7 tell us that further increasing the value of *maxIter* would not provide any further significant improvement, so we stopped at $maxIter = 3000$.

Table 6 Impact of number of iterations of COBYLA: better results for instances of type 0

Number	mean P_{opt}	mean P_{unf}	mean E	best P_{opt}	best P_{unf}	best E
Best result for 1000 iterations						
0_4	0.930	0.014	-4.670	0.939	0.011	-4.670
0_6	0.887	0.014	-5.360	0.899	0.012	-5.360
0_12	0.906	0.023	-5.364	0.914	0.023	-5.364
Best result for 2000 iterations						
0_4	0.944	0.009	-4.685	0.954	0.008	-4.685
0_6	0.909	0.011	-5.391	0.916	0.007	-5.391
0_12	0.897	0.024	-5.361	0.925	0.018	-5.377
Best result for 3000 iterations						
0_4	0.961	0.007	-4.702	0.970	0.005	-4.702
0_6	0.904	0.011	-5.387	0.912	0.008	-5.387
0_12	0.907	0.021	-5.371	0.924	0.019	-5.376

[**]Ideally, as $p \to \infty$, QAOA effectively becomes equivalent to the continuous adiabatic evolution.

Table 5 Results from the Randomic Approach

p	mean P_{opt}	mean P_{unf}	mean E	best P_{opt}	best P_{unf}	best E
			Instance: 5Sat3Gs_0_4			
1	0.196	0.353	-3.313	0.332	0.533	-3.548
2	0.300	0.353	-3.586	0.406	0.366	-3.725
3	0.499	0.216	-3.964	0.714	0.177	-4.293
4	0.501	0.198	-3.984	0.716	0.202	-4.307
5	0.487	0.229	-3.956	0.701	0.131	-4.347
6	0.535	0.194	-4.057	**0.793**	**0.115**	**-4.454**
7	0.573	0.166	-4.110	0.739	0.126	-4.333
8	**0.573**	**0.134**	**-4.138**	0.782	0.058	-4.444
			Instance: 5Sat3Gs_0_6			
1	0.122	0.464	-3.638	0.190	0.598	-3.776
2	0.220	0.450	-3.978	0.463	0.476	-4.499
3	0.278	0.344	-4.164	0.399	0.326	-4.510
4	0.295	0.368	-4.219	0.426	0.415	-4.493
5	0.345	0.351	-4.291	0.540	0.349	-4.730
6	0.354	**0.263**	**-4.396**	0.512	**0.080**	-4.812
7	**0.357**	0.282	-4.366	**0.687**	0.263	**-4.993**
8	0.346	0.290	-4.391	0.443	0.231	-4.637
			Instance: 5Sat3Gs_0_12			
1	0.110	0.504	-3.624	0.169	0.673	-3.784
2	0.184	0.510	-3.876	0.339	0.512	-4.303
3	0.227	0.373	-4.064	0.457	0.480	-4.560
4	0.248	0.348	-4.122	0.431	**0.344**	-4.587
5	0.247	0.354	-4.147	0.344	0.372	-4.409
6	0.240	0.342	-4.112	0.327	0.424	-4.323
7	0.249	0.314	-4.149	**0.469**	0.387	**-4.613**
8	**0.310**	**0.295**	**-4.309**	0.413	0.364	-4.460
			Instance: 5Sat3Gs_1_8			
1	0.075	0.567	-5.309	0.134	0.629	-4.994
2	0.077	0.626	-5.409	0.229	0.381	-6.034
3	0.106	0.570	-5.582	0.192	0.516	-5.984
4	0.083	0.572	-5.398	0.258	0.326	-5.900
5	0.120	0.591	-5.537	0.301	0.453	-6.245
6	0.138	**0.520**	-5.627	0.283	**0.178**	-6.259
7	**0.154**	0.551	**-5.650**	0.363	0.434	**-6.456**
8	0.132	0.561	-5.649	**0.367**	0.289	-6.455
			Instance: 5Sat3Gs_1_5			
1	0.065	0.553	-4.763	0.098	0.476	-5.080
2	0.073	0.567	-4.857	0.137	0.493	-5.212
3	0.080	0.591	-4.854	0.125	0.281	-5.100
4	0.102	0.443	-5.004	0.202	0.447	-5.247
5	0.106	0.511	-4.952	0.205	0.246	-5.481
6	0.127	0.447	-5.016	**0.320**	0.247	**-5.726**
7	0.133	**0.434**	**-5.094**	0.234	**0.213**	-5.527
8	**0.138**	0.561	-4.976	0.311	0.449	-5.532
			Instance: 5Sat3Gs_1_4			
1	0.024	**0.475**	-5.780	0.056	0.589	-6.176
2	0.036	0.567	-5.909	0.078	0.667	-6.244
3	0.048	0.528	**-6.056**	0.116	0.636	-6.020
4	0.047	0.592	-5.975	0.129	0.352	-6.354
5	0.049	0.600	-5.902	0.080	0.421	-6.316
6	0.065	0.598	-5.911	0.129	**0.311**	-6.572
7	**0.090**	0.553	-6.008	**0.230**	0.343	**-6.573**
8	0.068	0.592	-6.019	0.131	0.518	-6.557

Table 7 Impact of number of iterations of COBYLA: better results for instances of type 1

Number	mean P_{opt}	mean P_{unf}	mean E	best P_{opt}	best P_{unf}	best E
			Best result for 1000 iterations			
1_8	0.787	0.162	-7.315	0.802	0.156	-7.315
1_5	0.744	0.103	-6.434	0.759	0.100	-6.434
1_4	0.299	0.155	-7.005	0.313	0.155	-7.005
			Best result for 2000 iterations			
1_8	0.798	0.156	-7.326	0.808	0.151	-7.326
1_5	0.781	0.099	-6.503	0.796	0.091	-6.503
1_4	0.662	0.160	-7.787	0.677	0.160	-7.787
			Best result for 3000 iterations			
1_8	0.790	0.158	-7.326	0.803	0.145	-7.326
1_5	0.742	0.101	-6.438	0.757	0.091	-6.438
1_4	0.658	0.170	-7.815	0.673	0.166	-7.815

3. Boxplot Analysis of Algorithm Performance

In this section, we present the box plots for the selected instances **5Sat3Gs_0_6**, **5Sat3Gs_1_8**, and **5Sat3Gs_1_5**, based on the results obtained with our algorithm, in case $maxIter$ = 3000, listed in order of increasing difficulty (see Table 1). We have selected these instances because the they exhibited the improvement of P_{opt} with the increase of the p levels in a particularly clear way. This allows us to assess the algorithm's performance under varying conditions and provide a comprehensive view of its effectiveness in addressing problems with increasing complexity.

Problem_5Sat3Gs_0_6. This instance belongs to the category of type 0 and was chosen because it represents an ideal case for evaluating the performance of the algorithm under less constrained conditions. Although the problem can be solved without renouncing to perform any communication tasks, it still presents a non-trivial challenge as it requires an effective exploration of the space of solutions to maximize the probability of finding the optimum.

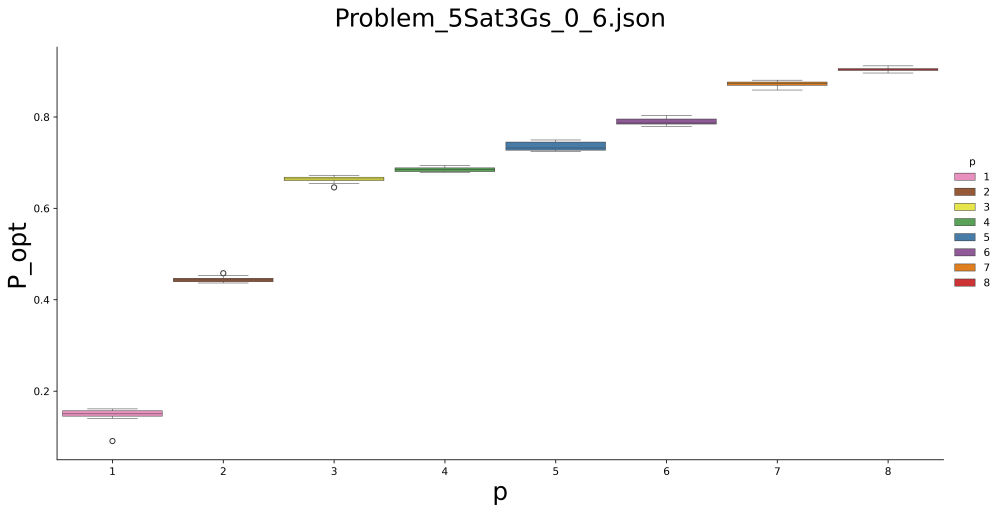

Figure 2 The values of P_{opt} as a function of the level p.

As the plot in Fig. 2 shows, the values of P_{opt} show very narrow distributions, indicating that the algorithm consistently finds solutions close to the optimal. The high median of the results reflects the algorithm's effectiveness in solving instances of Type 0, where the variability between different runs is minimal. This suggests that the algorithm performs reliably in simpler structural scenarios, maintaining consistent performance.

Problem_5Sat3Gs_1_8. This instance represents an instance of type 1 of intermediate difficulty.

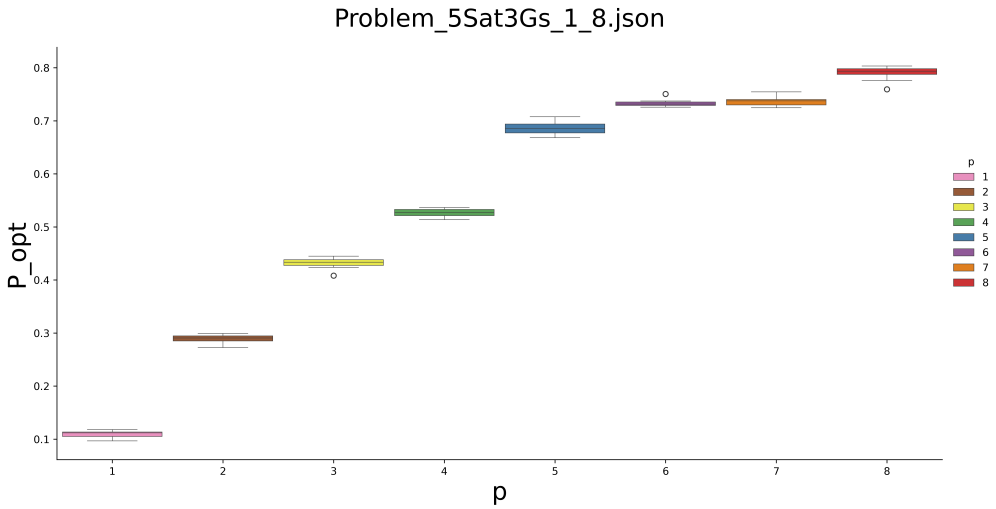

Figure 3 The values of P_{opt} as a function of the level p.

The results obtained for this instance, displayed on Fig. 3, show a distribution of P_{opt} that stands in between the previous and the next solution's distributions, making this an ideal case for evaluating performance in intermediate conditions. The variability is greater than that for the instance **5Sat3Gs_0_6** but less pronounced than that for the next instance **5Sat3Gs_1_5**. A steady improvement in the median can still be observed as p increases, indicating that the algorithm effectively converges even in scenarios characterized by a higher complexity.

Problem_5Sat3Gs_1_5. This is the most complex case among the instances considered in this analysis. It was selected to illustrate the algorithm's ability to handle highly constrained scenarios, confirming its capability to increase the probability of finding optimal solutions.

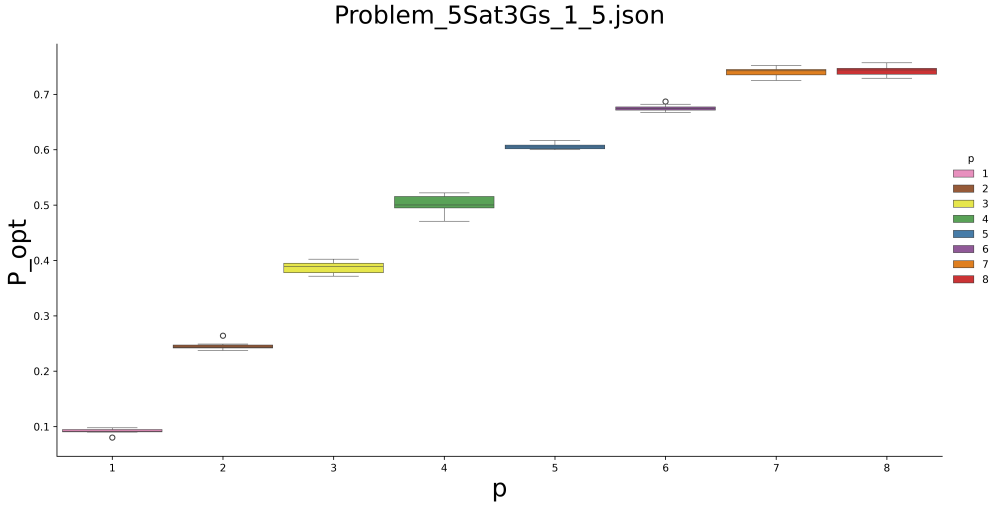

Figure 4 The values of P_{opt} as a function of the level p.

For this instance, representing a highly complex scenario with significant overlaps, the results generally exhibit greater variability. This behavior highlights the challenges posed by the presence of overlapping constraints, which make optimization more demanding. However, as shown in Fig. 4, the probability of finding optimal solutions improves, as p increases, even in this case, demonstrating the algorithm's ability to effectively explore a complex solution space and converge towards high-quality configurations.

VIII. Conclusions and Future Work

This study explored the application of the Quantum Approximate Optimization Algorithm (QAOA) to the problem of Oversubscribed Satellite Scheduling, highlighting its potential in addressing complex combinatorial optimization problems. By formulating the problem as a QUBO and implementing it through quantum circuits, we demonstrated how QAOA progressively improves the quality of the solution as the number of levels and iterations increases. Key results include the ability of QAOA to effectively handle overlapping time windows, reduce conflicts, and maximize valid communications. In simple cases, the algorithm often achieves optimal solutions, whereas in more complex cases, it significantly reduces invalid solutions, proving its robustness even in difficult configurations.

The optimization of parameters such as the initial $\langle \gamma, \beta \rangle$ values depending on the number of iterations, was crucial for improving the solutions. Increasing the number of circuit levels (p) tends to improve the results, as well as increasing the value of COBYLA's iterations (*maxIter*), though the latter no longer provides any improvement beyond a certain threshold.

Although in this work no real quantum hardware was used but only a simulator, the conducted experimental campaign was limited to instances of limited size. Yet, the results obtained confirm that QAOA may represent a promising approach for addressing real-world problems such as the Oversubscribed Satellite Scheduling Problem (OSSP) tackled in this paper, if translated to real quantum hardware characterized by hundreds of qubits. The integration of classical and quantum techniques proved essential in dealing with complexity, confirming the value of hybrid strategies. In addition, the algorithm effectively handled the constraints, reducing infeasible solutions even in complex scenarios.

In the future, it will be important to explore new research directions to expand the potential of this methodology. The first goal will be the application of other methods to optimize QAOA parameters, using different techniques from those adopted in this work. For example, advanced initialization strategies or different optimization approaches could be explored. Also, it might be interesting to study other types of variational algorithms for solving the OSSP problem, or consider the use of quantum annealers as well as other hybrid architectures.

In addition, we are planning to extend the approach to additional satellite-related problems, such as resource allocation or antenna planning, to demonstrate the flexibility and effectiveness of our approach in different scenarios. Another interesting direction will be to implement the approach on real quantum hardware, to evaluate its performance on real Noisy Intermediate-Scale Quantum (NISQ) devices. Finally, it will be crucial to study the scalability of the

methodology in larger and more complex problem instances to validate its applicability in real operational contexts. These developments may contribute significantly to the integration of Quantum Computing into practical problem solving, opening up new opportunities in the field of satellite planning and optimization.

References

[1] Preskill, J., "Quantum computing and the entanglement frontier," , 2012. URL https://arxiv.org/abs/1203.5813.

[2] Parmar, S. J., Parmar, V. R., Verma, J., Roy, S., and Bhattacharya, P., "Quantum Computing: Exploring Superposition and Entanglement for Cutting-Edge Applications," *2023 16th International Conference on Security of Information and Networks (SIN)*, 2023, pp. 1–6. https://doi.org/10.1109/SIN60469.2023.10474803.

[3] Farhi, E., Goldstone, J., Gutmann, S., Lapan, J., Lundgren, A., and Preda, D., "A Quantum Adiabatic Evolution Algorithm Applied to Random Instances of an NP-Complete Problem," *Science*, Vol. 292, No. 5516, 2001, pp. 472–475. https://doi.org/10.1126/science.1057726, URL https://www.science.org/doi/abs/10.1126/science.1057726.

[4] Preskill, J., "Quantum Computing in the NISQ era and beyond," *Quantum*, Vol. 2, 2018, p. 79. https://doi.org/10.22331/q-2018-08-06-79, URL https://doi.org/10.22331/q-2018-08-06-79.

[5] Farhi, E., Goldstone, J., and Gutmann, S., "A Quantum Approximate Optimization Algorithm," , 2014. URL https://arxiv.org/abs/1411.4028.

[6] Blekos, K., Brand, D., Ceschini, A., Chou, C.-H., Li, R.-H., Pandya, K., and Summer, A., "A review on Quantum Approximate Optimization Algorithm and its variants," *Physics Reports*, Vol. 1068, 2024, pp. 1–66. https://doi.org/https://doi.org/10.1016/j.physrep.2024.03.002, URL https://www.sciencedirect.com/science/article/pii/S0370157324001078, a review on Quantum Approximate Optimization Algorithm and its variants.

[7] Cerezo, M., Arrasmith, A., Babbush, R., Benjamin, S. C., Endo, S., Fujii, K., McClean, J. R., Mitarai, K., Yuan, X., Cincio, L., and Coles, P. J., "Variational quantum algorithms," *Nature Reviews Physics*, Vol. 3, 2020, pp. 625 – 644. URL https://api.semanticscholar.org/CorpusID:229297850.

[8] Zhou, L., Wang, S.-T., Choi, S., Pichler, H., and Lukin, M. D., "Quantum Approximate Optimization Algorithm: Performance, Mechanism, and Implementation on Near-Term Devices," *Physical Review X*, 2018. URL https://api.semanticscholar.org/CorpusID:119061693.

[9] Ferrari, B., Cordeau, J.-F., Delorme, M., Iori, M., and Orosei, R., "Satellite Scheduling Problems: A survey of applications in Earth and outer space observation," *Computers & Operations Research*, Vol. 173, 2025, p. 106875. https://doi.org/https://doi.org/10.1016/j.cor.2024.106875, URL https://www.sciencedirect.com/science/article/pii/S0305054824003472.

[10] Ceballos, A., Bensalem, S., Cesta, A., Silva, L. D., Fratini, S., Ingrand, F., Ocòn, J., Orlandini, A., Py, F., Rajan, K., Rasconi, R., and Winnendael, M. V., "A Goal-Oriented Autonomous Controller for Space Exploration," *11th Symposium on Advanced Space*

Technologies in Robotics and Automation. Proceedings (ASTRA 2011), pp. 1–8, Noordwijk, the Netherlands, 12-14 April 2011, 2011.

[11] Torta, P., Casati, R., Bruni, S., Mandarino, A., and Prati, E., "Quantum computing for space applications: a selective review and perspectives," *EPJ Quantum Technology*, Vol. 12, No. 1, 2025, p. 66. https://doi.org/10.1140/epjqt/s40507-025-00369-8, URL https://doi.org/10.1140/epjqt/s40507-025-00369-8.

[12] Tran, T. T., Wang, Z., Do, M. B., Rieffel, E. G., Frank, J. D., O'Gorman, B., Venturelli, D., and Beck, J. C., "Explorations of Quantum-Classical Approaches to Scheduling a Mars Lander Activity Problem," *AAAI Workshop: Planning for Hybrid Systems*, 2016. URL https://api.semanticscholar.org/CorpusID:19855109.

[13] Stollenwerk, T., Hadfield, S., and Wang, Z., "Toward Quantum Gate-Model Heuristics for Real-World Planning Problems," *IEEE Transactions on Quantum Engineering*, Vol. 1, 2020, pp. 1–16. https://doi.org/10.1109/TQE.2020.3030609.

[14] Lewis, R. M. R., *Guide to Graph Colouring: Algorithms and Applications*, Springer, 2021.

[15] Bentert, M., Bevern, R., and Niedermeier, R., "Inductive k-independent graphs and c-colorable subgraphs in scheduling: a review," *Journal of Scheduling*, Vol. 22, No. 1, 2019, pp. 3–20. https://doi.org/10.1007/s10951-018-0595-8, URL https://ideas.repec.org/a/spr/jsched/v22y2019i1d10.1007_s10951-018-0595-8.html.

[16] Kuryatnikova, O., Sotirov, R., and Vera, J. C., "The Maximum k-Colorable Subgraph Problem and Related Problems," *INFORMS Journal on Computing*, Vol. 34, 2021, pp. 56–669. https://doi.org/10.1287/ijoc.2021.1086.

[17] Araujo, J., Bermond, J.-C., Giroire, F., Havet, F., Mazauric, D., and Modrzejewski, R., "Weighted improper colouring," *Journal of Discrete Algorithms*, Vol. 16, 2012, pp. 53–66. https://doi.org/https://doi.org/10.1016/j.jda.2012.07.001, URL https://www.sciencedirect.com/science/article/pii/S1570866712001049, selected papers from the 22nd International Workshop on Combinatorial Algorithms (IWOCA 2011).

[18] Marx, D., "Graph Colouring Problems and their Applications in Scheduling," *Periodica Polytechnica Electrical Engineering*, Vol. 48, No. 1-2, 2004, p. 11–16. https://doi.org/N/A, URL https://pp.bme.hu/ee/article/view/926.

[19] Zufferey, N., Amstutz, P., and Giaccari, P., "Graph colouring approaches for a satellite range scheduling problem," , 2008. https://doi.org/10.1007/s10951-008-0066-8, URL https://infoscience.epfl.ch/handle/20.500.14299/61114.

[20] Stollenwerk, T., Michaud, V., Lobe, E., Picard, M., Basermann, A., and Botter, T., "Agile Earth Observation Satellite Scheduling With a Quantum Annealer," *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 57, No. 5, 2021, pp. 3520–3528. https://doi.org/10.1109/TAES.2021.3088490.

[21] Rainjonneau, S., Tokarev, I., Iudin, S., Rayaprolu, S., Pinto, K., Lemtiuzhnikova, D., Koblan, M., Barashov, E., Kordzanganeh, M., Pflitsch, M., and Melnikov, A., "Quantum Algorithms Applied to Satellite Mission Planning for Earth Observation," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, Vol. 16, 2023, pp. 7062–7075. https://doi.org/10.1109/JSTARS.2023.3287154.

[22] Makarov, A., Pérez-Herradón, C., Franceschetto, G., Taddei, M. M., Osaba, E., del Barrio Cabello, P., Villar-Rodriguez, E., and Oregi, I., "Quantum Optimization Methods for Satellite Mission Planning," *IEEE Access*, Vol. 12, 2024, p. 71808–71820. https://doi.org/10.1109/access.2024.3402990, URL http://dx.doi.org/10.1109/ACCESS.2024.3402990.

[23] Quetschlich, N., Koch, V., Burgholzer, L., and Wille, R., " A Hybrid Classical Quantum Computing Approach to the Satellite Mission Planning Problem ," *2023 IEEE International Conference on Quantum Computing and Engineering (QCE)*, IEEE Computer Society, Los Alamitos, CA, USA, 2023, pp. 642–647. https://doi.org/10.1109/QCE57702.2023.00079, URL https://doi.ieeecomputersociety.org/10.1109/QCE57702.2023.00079.

[24] Bonomo, F., Durán, G., and Marenco, J., "Exploring the complexity boundary between coloring and list-coloring," *Electronic Notes in Discrete Mathematics*, Vol. 25, 2006, pp. 41–47. https://doi.org/https://doi.org/10.1016/j.endm.2006.06.064, URL https://www.sciencedirect.com/science/article/pii/S1571065306000655, cTW2006 - Cologne-Twente Workshop on Graphs and Combinatorial Optimization.

[25] Biró, M., Hujter, M., and Tuza, Z., "Precoloring extension. I. Interval graphs," *Discrete Mathematics*, Vol. 100, No. 1, 1992, pp. 267–279. https://doi.org/https://doi.org/10.1016/0012-365X(92)90646-W, URL https://www.sciencedirect.com/science/article/pii/0012365X9290646W.

[26] Enright, J., Stewart, L., and Tardos, G., "On List Coloring and List Homomorphism of Permutation and Interval Graphs," *SIAM Journal on Discrete Mathematics*, Vol. 28, No. 4, 2014, pp. 1675–1685. https://doi.org/10.1137/13090465X, URL https://doi.org/10.1137/13090465X.

[27] Grötschel, M., Lovász, L., and Schrijver, A., "Polynomial Algorithms for Perfect Graphs," *Topics on Perfect Graphs*, North-Holland Mathematics Studies, Vol. 88, edited by C. Berge and V. Chvátal, North-Holland, 1984, pp. 325–356. https://doi.org/https://doi.org/10.1016/S0304-0208(08)72943-8, URL https://www.sciencedirect.com/science/article/pii/S0304020808729438.

[28] Guerreschi, G. G., and Matsuura, A. Y., "QAOA for Max-Cut requires hundreds of qubits for quantum speed-up," *Scientific Reports*, Vol. 9, No. 1, 2019, p. 6903. https://doi.org/10.1038/s41598-019-43176-9, URL https://doi.org/10.1038/s41598-019-43176-9.

[29] Hadfield, S., Wang, Z., O'Gorman, B., Rieffel, E., Venturelli, D., and Biswas, R., "From the Quantum Approximate Optimization Algorithm to a Quantum Alternating Operator Ansatz," *Algorithms*, Vol. 12, No. 2, 2019, p. 34. https://doi.org/10.3390/a12020034, URL http://dx.doi.org/10.3390/a12020034.

[30] Cruz, D., Fournier, R., Gremion, F., Jeannerot, A., Komagata, K., Tosic, T., Thiesbrummel, J., Chan, C. L., Macris, N., Dupertuis, M.-A., and Javerzac-Galy, C., "Efficient Quantum Algorithms for GHZ and W States, and Implementation on the IBM Quantum Computer," *Advanced Quantum Technologies*, Vol. 2, No. 5-6, 2019, p. 1900015. https://doi.org/https://doi.org/10.1002/qute.201900015, URL https://onlinelibrary.wiley.com/doi/abs/10.1002/qute.201900015.

[31] Farhi, E., Gamarnik, D., and Gutmann, S., "The Quantum Approximate Optimization Algorithm Needs to See the Whole Graph: A Typical Case," , 2020. URL https://arxiv.org/abs/2004.09002.

[32] Gaidai, I., and Herrman, R., "Performance analysis of multi-angle QAOA for $p > 1$," *Scientific Reports*, Vol. 14, No. 1, 2024, p. 18911. https://doi.org/10.1038/s41598-024-69643-6.

[33] Zhou, Y., Stoudenmire, E. M., and Waintal, X., "What Limits the Simulation of Quantum Computers?" *Physical Review X*, Vol. 10, No. 4, 2020. https://doi.org/10.1103/physrevx.10.041038, URL http://dx.doi.org/10.1103/PhysRevX.10.041038.