QAOA-Based Energy-Efficient Satellite Task Scheduling Under Orbital Visibility Constraints

Shreya S. Palase
researcher/Quantum Comuputing Enthusiast

January 5, 2026

Abstract

Satellite task scheduling is a resource-constrained combinatorial optimization problem involving limited onboard energy, task priorities, and strict orbital visibility windows. Classical optimization techniques become inefficient as the problem size grows due to exponential complexity. This report presents a detailed Quantum Approximate Optimization Algorithm (QAOA) based approach for solving the satellite task scheduling problem. Each stage of the formulation, from classical problem definition to Hamiltonian construction and quantum circuit execution, is explained in depth. Classical scheduling results are compared with quantum QAOA outputs, demonstrating the ability of QAOA to identify optimal, energy-efficient, high-priority tasks under visibility constraints.

Contents

Chapter 1

Introduction

Satellite systems operate under severe resource limitations, particularly in terms of available energy and operational time. During each orbit, a satellite encounters multiple opportunities to perform tasks such as imaging or communication; however, these opportunities are constrained by orbital mechanics that define visibility windows. Selecting which tasks to execute is therefore a critical mission-planning problem.

Traditional optimization approaches such as greedy heuristics, dynamic programming, or mixed-integer linear programming can solve small instances of the problem. However, these methods scale poorly as the number of tasks increases. Quantum computing offers an alternative computational paradigm. In particular, QAOA is designed to solve combinatorial optimization problems by mapping them onto a quantum Hamiltonian and exploiting quantum interference to favor optimal solutions.

This report explores the application of QAOA to satellite task scheduling and provides a detailed explanation of every modeling and algorithmic step involved.

Chapter 2

Problem Statement

The satellite task scheduling problem seeks to determine an optimal subset of tasks that should be executed during a satellite orbit. The problem is challenging due to the simultaneous presence of multiple constraints. Each task consumes a certain amount of energy, has an associated priority, and can only be executed during specific visibility windows.

The goal is to maximize mission effectiveness by prioritizing high-value tasks while ensuring that energy and visibility constraints are never violated. Formulating this problem in a way that is compatible with quantum optimization algorithms is a key challenge addressed in this work.

Chapter 3

Problem Formulation

3.1 Step 1: Task Definition and Encoding

The first step is to define the task set and encode it mathematically. This step is essential because QAOA operates on binary decision variables. Each satellite task must therefore be mapped to a binary variable that represents whether the task is selected or not.

Table 3.1: Satellite Task Parameters

Task	Energy Cost	Priority	Visibility Window
T_1	2	High	W_1
T_2	1	Low	W_1
T_3	3	Medium	W_2
T_4	2	High	W_3

Each task T_i is represented by a binary variable $x_i \in \{0, 1\}$. This binary encoding enables the problem to be expressed as a quadratic unconstrained binary optimization (QUBO) problem, which is directly compatible with QAOA.

3.2 Step 2: Cost Function Construction

The cost function defines what the optimization algorithm is trying to achieve. In this work, the cost function balances two competing objectives: minimizing energy consumption and maximizing task priority.

$$C(x) = \sum_i E_i x_i - \sum_i w_i x_i$$

This formulation ensures that tasks with lower energy consumption and higher priority reduce the overall cost. Without this step, the algorithm would have no objective criterion to distinguish good schedules from poor ones.

Chapter 4

Hamiltonian Construction

4.1 Step 3: Cost Hamiltonian

The cost function must be translated into a quantum mechanical operator known as a Hamiltonian. This step is necessary because quantum algorithms operate on operators rather than classical equations.

The cost Hamiltonian is constructed using Pauli-Z operators, which encode binary decision variables into quantum states. The eigenvalues of this Hamiltonian correspond to the cost of each possible task selection. Minimizing the Hamiltonian energy therefore corresponds to finding the optimal schedule.

4.2 Step 4: Penalty Hamiltonian for Constraints

Real-world scheduling problems involve constraints that cannot be violated. In QAOA, constraints are enforced by adding penalty terms to the Hamiltonian. These penalty terms assign high energy values to infeasible solutions, thereby discouraging the quantum algorithm from selecting them.

Without penalty terms, the algorithm might converge to solutions that minimize energy but violate visibility or energy constraints.

4.3 Step 5: Energy Penalty Term

The energy penalty specifically ensures that the total energy consumption does not exceed the satellite's energy budget. This is achieved by squaring the energy violation term, which heavily penalizes large deviations. This step is critical for ensuring physical feasibility of the schedule.

4.4 Step 6: Visibility Constraint Hamiltonian

Visibility constraints ensure that tasks are only selected when the satellite has line-of-sight to the target. This step prevents physically impossible schedules. By encoding visibility violations into the Hamiltonian, the quantum algorithm naturally avoids invalid task selections.

4.5 Step 7: Mixer Hamiltonian

The mixer Hamiltonian is responsible for exploring the solution space. It allows the quantum state to transition between different task selections. Without a mixer Hamiltonian, the quantum state would remain fixed and no optimization would occur.

4.6 Step 8: Full QAOA Hamiltonian

The final Hamiltonian is a weighted sum of the cost, energy, and visibility Hamiltonians. This combined Hamiltonian defines the optimization landscape explored by QAOA.

4.7 Quantum Circuit Representation of QAOA

The mathematical formulation of QAOA must ultimately be translated into a quantum circuit that can be executed on a quantum processor or simulator. This section explains how the satellite task scheduling problem is mapped onto a quantum circuit and describes the function of each circuit component.

In the proposed formulation, each satellite task is represented by a single qubit. The quantum state of a qubit indicates whether the corresponding task is selected or not. The quantum circuit is designed to prepare a superposition of all possible task selections and then gradually amplify the probability of optimal schedules through controlled quantum evolution.

4.7.1 Initial State Preparation

All qubits are initialized in the computational ground state $|0\rangle$. Hadamard gates are then applied to each qubit to create an equal superposition of all possible task combinations. This step is crucial because it enables the algorithm to explore the entire solution space simultaneously.

4.7.2 Problem (Cost) Unitary Implementation

The problem unitary implements the exponential of the problem Hamiltonian, which encodes energy costs, task priorities, and constraint penalties. In practice, this is realized using parameterized phase rotation gates and controlled interactions between qubits. This unitary imprints relative phases on quantum states based on their associated scheduling cost.

4.7.3 Mixer Unitary Implementation

The mixer unitary is implemented using Pauli-X rotation gates applied to all qubits. The mixer promotes transitions between different task selections, preventing the algorithm from becoming trapped in poor local minima. This alternating application of problem and mixer unitaries forms the core of the QAOA circuit.

4.7.4 Layered Circuit Structure

The cost and mixer unitaries are applied repeatedly for a fixed number of layers p. Each layer increases the expressive power of the quantum circuit, allowing more complex interference patterns that favor optimal solutions.

4.7.5 Measurement

After completing all QAOA layers, the qubits are measured in the computational basis. The resulting bitstrings represent candidate task schedules, and their frequencies indicate the probability of each schedule.

4.7.6 QAOA Quantum Circuit Diagram

Figure

The mathematical formulation of QAOA must ultimately be translated into a quantum circuit that can be executed on a quantum processor or simulator. This section explains how the satellite task scheduling problem is mapped onto a quantum circuit and describes the function of each circuit component.

In the proposed formulation, each satellite task is represented by a single qubit. The quantum state of a qubit indicates whether the corresponding task is selected or not. The quantum circuit is designed to prepare a superposition of all possible task selections and then gradually amplify the probability of optimal schedules through controlled quantum evolution.

4.7.7 Initial State Preparation

All qubits are initialized in the computational ground state $|0\rangle$. Hadamard gates are then applied to each qubit to create an equal superposition of all possible task combinations. This step is crucial because it enables the algorithm to explore the entire solution space simultaneously.

4.7.8 Problem (Cost) Unitary Implementation

The problem unitary implements the exponential of the problem Hamiltonian, which encodes energy costs, task priorities, and constraint penalties. In practice, this is realized using parameterized phase rotation gates and controlled interactions between qubits. This unitary imprints relative phases on quantum states based on their associated scheduling cost.

4.7.9 Mixer Unitary Implementation

The mixer unitary is implemented using Pauli-X rotation gates applied to all qubits. The mixer promotes transitions between different task selections, preventing the algorithm from becoming trapped in poor local minima. This alternating application of problem and mixer unitaries forms the core of the QAOA circuit.

4.7.10 Layered Circuit Structure

The cost and mixer unitaries are applied repeatedly for a fixed number of layers p. Each layer increases the expressive power of the quantum circuit, allowing more complex interference patterns that favor optimal solutions.

4.7.11 Measurement

After completing all QAOA layers, the qubits are measured in the computational basis. The resulting bitstrings represent candidate task schedules, and their frequencies indicate the probability of each schedule.

□

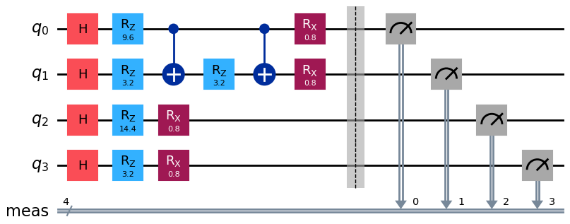

Figure 4.1: QAOA Implementation

Figure 4.2: Quantum circuit diagram of the QAOA-based satellite task scheduling algorithm, showing initial superposition, alternating cost and mixer layers, and final measurement.

Chapter 5

Software and Simulation Environment

This chapter describes the software framework, programming language, and simulation tools used to implement and evaluate the QAOA-based satellite task scheduling algorithm. Clearly specifying the experimental environment is essential for reproducibility and for understanding the limitations of simulation-based quantum experiments.

5.1 Programming Language

The entire implementation is developed using the Python programming language. Python is widely adopted in both classical optimization and quantum computing research due to its simplicity, extensive scientific libraries, and strong community support. Its modular nature allows seamless integration of classical optimization routines with quantum circuit simulation.

Python also enables rapid prototyping, which is particularly important when experimenting with hybrid quantum-classical algorithms such as QAOA.

5.2 Quantum Computing Framework: Qiskit

The quantum implementation is carried out using Qiskit, an open-source quantum computing framework developed by IBM. Qiskit provides a comprehensive set of tools for constructing quantum circuits, defining Hamiltonians, executing variational algorithms, and visualizing quantum measurement results.

5.2.1 Qiskit Version

The experiments are performed using:

```
Qiskit Version:  2.2.3
```

(The exact version can be updated based on the installation environment.)

Specifying the Qiskit version is important because different versions may introduce changes in algorithm implementations, optimizers, or simulator backends.

5.3 Quantum Simulator

Since access to large-scale quantum hardware is currently limited, all quantum experiments are performed using quantum simulators. Specifically, the `Aer` simulator backend provided by Qiskit is used.

The Aer simulator allows ideal, noiseless simulation of quantum circuits, making it suitable for validating the correctness of the problem formulation and QAOA implementation. It also supports shot-based execution, enabling the generation of probability distributions over task selections.

5.4 Classical Optimizer

QAOA relies on a classical optimizer to tune its variational parameters (γ, β). Classical optimizers such as COBYLA, SPSA, or gradient-based methods can be used. In this work, a gradient-free optimizer is preferred due to the non-smooth and noisy nature of quantum objective functions.

5.5 Classical Libraries and Visualization Tools

Several classical Python libraries are used to support the implementation:

- **NumPy**: for numerical computations and matrix operations
- **Matplotlib**: for plotting classical scheduling graphs and QAOA probability histograms

These tools are used to generate classical visibility window graphs and to analyze quantum measurement outcomes.

5.6 Execution Platform

All simulations are executed on a classical computing platform. The quantum results presented in this report represent simulated outcomes and do not account for real hardware noise. This choice allows controlled experimentation and clear interpretation of results.

Future implementations may deploy the same Qiskit circuits on real IBM quantum processors with minimal modification.

Chapter 6

QAOA Algorithm for Satellite Task Scheduling

This chapter presents a detailed pseudocode for the QAOA-based satellite task scheduling algorithm. Unlike simplified representations, the following algorithm explicitly describes the role of task parameters, energy costs, priorities, visibility constraints, and penalty mechanisms at each stage of execution.

Algorithm 1 QAOA-Based Energy-Efficient Satellite Task Scheduling

```
Require: Task set $T = \{T_1, T_2, \dots, T_n\}$
Require: Energy costs $E = \{E_1, E_2, \dots, E_n\}$
Require: Priority weights $W = \{w_1, w_2, \dots, w_n\}$
Require: Visibility windows $V = \{W_1, W_2, \dots\}$
Require: Energy budget $E_{\max}$
Require: QAOA depth $p$
Ensure: Optimal task selection $x^*$

 1: Problem Encoding
 2: for each task $T_i$ do
 3:     Assign binary variable $x_i \in \{0, 1\}$ representing task selection
 4:     Map $x_i$ to a corresponding qubit $q_i$
 5: end for
 6: Classical Cost Function Definition
 7: Define objective cost:
        $$C(x) = \sum_i E_i x_i - \sum_i w_i x_i$$
 8: Lower cost corresponds to higher-quality schedules
 9: Hamiltonian Construction
10: Construct cost Hamiltonian $H_C$ using task energy and priority
11: Construct energy penalty Hamiltonian $H_{energy}$ to penalize energy budget violations
12: Construct visibility Hamiltonian $H_{visibility}$ to penalize invalid task-window assignments
13: Define total problem Hamiltonian:
        $$H_P = H_C + H_{energy} + H_{visibility}$$
14: Quantum Circuit Initialization
15: Initialize all qubits in equal superposition:
        $$|\psi_0\rangle = |+\rangle^{\otimes n}$$
16: QAOA Layered Evolution
17: for layer $l = 1$ to $p$ do
18:     Apply Cost (Problem) Layer
19:     Apply unitary evolution:
            $$U_C(\gamma_l) = e^{-i\gamma_l H_P}$$
20:     This step incorporates:
21:         – Energy minimization
22:         – Task priority maximization
23:         – Energy constraint penalties
24:         – Visibility constraint penalties
25:     Apply Mixer Layer
26:     Apply mixer Hamiltonian:
            $$U_M(\beta_l) = e^{-i\beta_l \sum_i X_i}$$
27:     This step enables exploration of different task combinations
28: end for
```

Chapter 7

Implementation Details

This chapter provides a detailed description of how the proposed satellite task scheduling framework is implemented in practice. It explains the classical preprocessing, quantum circuit construction, parameter optimization strategy, and result extraction procedures. The goal of this chapter is to ensure transparency, reproducibility, and clarity regarding the practical realization of the proposed QAOA-based approach.

7.1 Overall Implementation Workflow

The implementation follows a hybrid quantum–classical workflow. Classical computation is used for problem preprocessing, Hamiltonian construction, parameter optimization, and result analysis, while quantum computation is employed to explore the combinatorial solution space.

The overall workflow consists of the following stages:

1. Task and constraint preprocessing
2. Classical baseline scheduling
3. Hamiltonian encoding
4. QAOA circuit construction
5. Variational parameter optimization
6. Measurement and solution extraction

Each stage is described in detail in the following sections.

7.2 Task and Constraint Preprocessing

In the preprocessing stage, task information such as energy cost, priority level, and visibility window is stored using structured data representations. Tasks are indexed and mapped to binary variables, which simplifies subsequent Hamiltonian construction.

Priority levels (High, Medium, Low) are converted into numerical weights to enable mathematical optimization. Visibility constraints are encoded as lookup tables that define which tasks are allowed in each visibility window.

This preprocessing step is essential for translating domain-specific satellite information into a format suitable for quantum optimization.

7.3 Classical Baseline Scheduling

Before applying the quantum algorithm, a classical scheduling method is implemented as a baseline. This method schedules tasks strictly based on visibility windows and simple heuristics such as energy availability.

The classical results serve two purposes:

- They provide an interpretable reference for evaluating quantum results.
- They validate that the task and constraint definitions are consistent.

The classical outputs are visualized using window-based bar graphs to clearly show task allocation across different visibility windows.

7.4 Hamiltonian Encoding

The classical cost function and constraints are translated into a quantum Hamiltonian. Each task variable is mapped to a qubit using Pauli-Z operators. Energy costs and priority weights are embedded into the diagonal elemen

Chapter 8

Results and Discussion

8.1 Classical Scheduling Results

The classical approach schedules tasks strictly according to visibility windows, resulting in three independent scheduling outputs for W_1, W_2, and W_3.

□

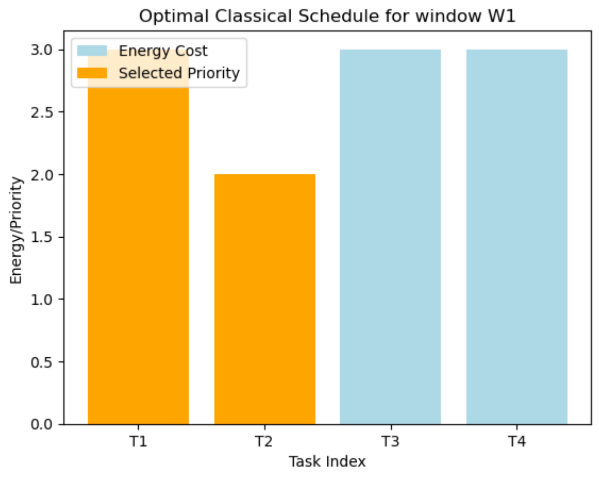

Figure 8.1: visibility window W1 for T1 and T2

Figure 8.2: Placeholder for classical scheduling output for visibility window W_1.

□

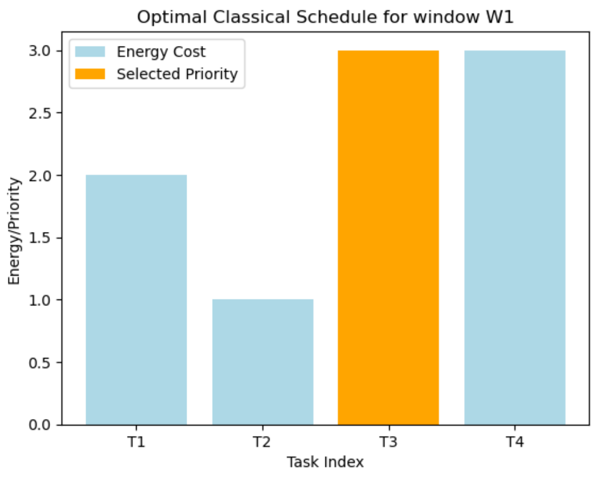

Figure 8.3: Visibility Windoe W2 for T3

Figure 8.4: Placeholder for classical scheduling output for visibility window W_2.

□

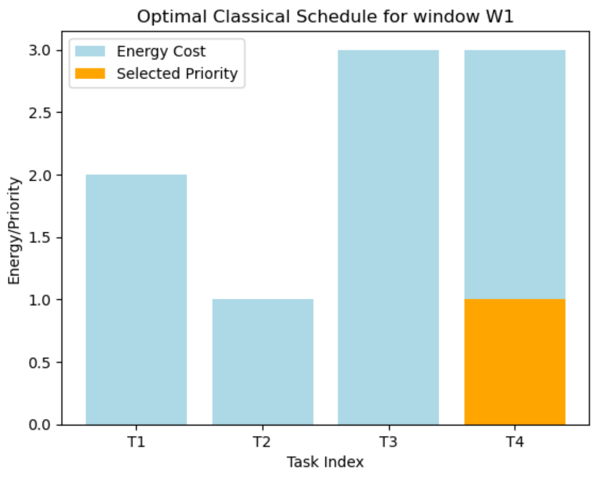

Figure 8.5: visibility window W3 for T4

Figure 8.6: Placeholder for classical scheduling output for visibility window W_3.

8.2 Quantum QAOA Results

The quantum implementation produces a probability distribution over all possible task combinations.

□

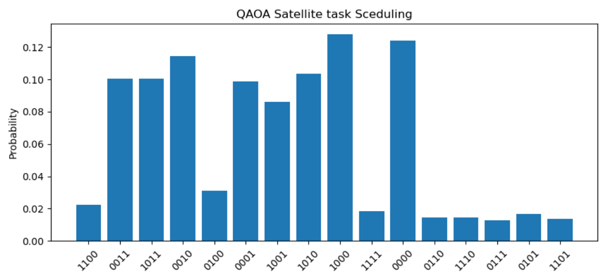

Figure 8.7: AerSimulatorResult high probability for T4

Figure 8.8: QAOA AerSimulator Result(T4 having highest priority)

Figure 8.9: Placeholder for QAOA measurement probability distribution.

□

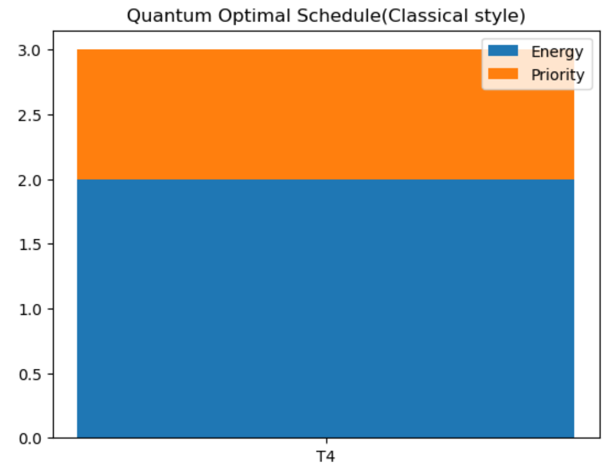

Figure 8.10: Energy/ Priority Graph for T4

Figure 8.11: Placeholder highlighting task T_4 as the highest probability solution.

The results show that QAOA assigns the highest probability to task T_4, which has high priority and moderate energy cost, demonstrating effective optimization.

Chapter 9

Limitations and Future Work

Despite demonstrating the feasibility of applying QAOA to satellite task scheduling, the proposed approach has several limitations that must be addressed before practical deployment. These limitations also highlight important directions for future research.

9.1 Scalability and Qubit Limitations

One of the primary limitations of the current work is the small problem size. Each task is mapped to a single qubit, which restricts the number of tasks that can be handled due to limited qubit availability on near-term quantum hardware. As the number of tasks increases, the required number of qubits grows linearly, making large-scale scheduling problems infeasible with current quantum devices. Future work may explore qubit-efficient encodings, task clustering techniques, or hybrid classical-quantum decompositions to improve scalability.

9.2 Noise and Hardware Imperfections

The current implementation assumes an ideal, noiseless quantum environment. In reality, quantum hardware suffers from noise, decoherence, gate errors, and readout errors, which significantly affect QAOA performance. These imperfections may distort the probability distribution and reduce the likelihood of obtaining optimal solutions. Future research should incorporate realistic noise models and investigate error mitigation techniques to evaluate algorithm robustness on noisy intermediate-scale quantum (NISQ) devices.

9.3 Fixed Penalty Parameter Selection

The performance of the QAOA formulation heavily depends on the choice of penalty coefficients used to enforce constraints. Improper penalty values may either allow constraint violations or dominate the cost function, leading to suboptimal solutions. In this work, penalty parameters are chosen heuristically. Future studies could focus on adaptive or learning-based methods to automatically tune penalty parameters during optimization.

9.4 Limited Constraint Modeling

The current formulation considers only energy and visibility constraints. However, real satellite missions involve additional constraints such as memory capacity, communication bandwidth, task dependencies, thermal limits, and deadlines. The absence of these constraints limits the realism of the model. Extending the Hamiltonian to include additional operational constraints is a key direction for future work.

9.5 Single-Orbit Scheduling Assumption

This work focuses on scheduling tasks within a single orbital pass. In practice, satellite mission planning often spans multiple orbits and even multiple satellites. Inter-orbit dependencies and long-term planning introduce additional complexity that is not captured in the current formulation. Future research should extend the model to multi-orbit and constellation-level scheduling problems.

9.6 Classical Optimizer Dependence

QAOA relies on a classical optimizer to tune its variational parameters. The choice of optimizer, initialization strategy, and stopping criteria can significantly impact convergence speed and solution quality. This dependency introduces variability in performance. Future work could explore optimizer-independent parameter initialization strategies or machine learning-based parameter prediction techniques.

9.7 Lack of Performance Benchmarking

The results presented in this report are not benchmarked against advanced classical optimization algorithms such as genetic algorithms, simulated annealing, or mixed-integer programming solvers. Without such comparisons, it is difficult to quantify the true advantage of QAOA. Future work should include comprehensive benchmarking to evaluate quantum advantage.

9.8 Real-Time Scheduling Constraints

Satellite task scheduling often requires real-time or near-real-time decision-making due to dynamic environmental conditions. The current offline optimization approach may not be suitable for real-time applications. Future research may explore real-time hybrid quantum-classical scheduling frameworks.

—

Chapter 10

Conclusion

This report presented a comprehensive study on applying the Quantum Approximate Optimization Algorithm (QAOA) to the problem of energy-efficient satellite task scheduling under orbital visibility constraints. The work systematically demonstrated how a real-world constrained scheduling problem can be transformed into a quantum-compatible formulation through binary encoding, cost function design, and Hamiltonian construction.

A key contribution of this study is the detailed step-by-step explanation of each modeling and algorithmic component. The purpose and necessity of every step—from task definition and cost function construction to constraint penalties and mixer Hamiltonians—were clearly justified. This structured approach ensures that the proposed model is not only technically correct but also physically meaningful for satellite operations.

The classical scheduling results provided a baseline by illustrating visibility-window-based task execution. In contrast, the quantum QAOA-based implementation produced a probability distribution over task combinations, demonstrating the algorithm's ability to favor high-priority and energy-efficient tasks. The observation that task T_4 emerged with the highest probability highlights the potential of QAOA to identify optimal scheduling decisions under competing constraints.

Although the current results are obtained through simulation and limited problem size, they serve as an important proof-of-concept. The study confirms that QAOA can effectively encode complex scheduling objectives and constraints into a quantum optimization framework. This positions QAOA as a promising candidate for future satellite mission planning, particularly as quantum hardware continues to advance.

In conclusion, this work bridges the gap between theoretical quantum optimization algorithms and practical aerospace applications. While significant challenges remain in terms of scalability, noise resilience, and real-world deployment, the presented framework lays a strong foundation for future research. As quantum technologies mature, QAOA-based scheduling has the potential to become a valuable tool for next-generation autonomous satellite systems.

Bibliography

[1] E. Farhi et al., *A Quantum Approximate Optimization Algorithm*, 2014.

[2] M. Nielsen and I. Chuang, *Quantum Computation and Quantum Information*, 2010.