

# **Currency Arbitrage Optimization using QAOA in Qiskit**

and Comparative Study with Classical Solvers

Author: Bhuvanesh

Date: January 2026

# Currency Arbitrage Optimization using QAOA in Qiskit

## Abstract

This project formulates currency arbitrage detection as a Quadratic Unconstrained Binary Optimization (QUBO) problem and solves it using QAOA implemented in Qiskit. Exchange rates are modeled as a directed graph, and profitable cycles are identified by minimizing a cost Hamiltonian. Classical methods such as Bellman-Ford and brute-force search are used as baselines. The Qiskit-based QAOA implementation is validated through sampling, constraint satisfaction, and comparison with classical results.

## 1. Introduction

Currency arbitrage exploits inconsistencies in exchange rates to generate profit through cyclic trades. This can be modeled as a graph where nodes are currencies and edges are exchange rates. A profitable cycle exists if the product of rates exceeds one. By applying  $-\log(\text{rate})$ , the problem becomes additive and suitable for QUBO formulation.

## 2. Motivation

Classical algorithms like Bellman-Ford are efficient for small graphs but scale poorly. QAOA in Qiskit offers a quantum-classical hybrid approach to explore large solution spaces using parameterized quantum circuits. This project evaluates QAOA's effectiveness in modeling arbitrage cycles and compares it with classical solvers.

## 3. Related Work

Previous QAOA implementations in Qiskit and Cirq have demonstrated feasibility but faced issues with constraint satisfaction and parameter tuning. Classical methods like simulated annealing and brute-force search provide accurate baselines. This project builds on these by implementing a Qiskit-based QAOA with improved diagnostics.

## 4. Problem Statement and Formulation

Each currency exchange is a binary variable  $b_{ij}$  in  $\{0,1\}$ . The objective is to minimize:

$$C = \sum -\log(r_{ij}) * b_{ij}$$

Subject to:

## Currency Arbitrage Optimization using QAOA in Qiskit

- One incoming and one outgoing edge per currency (cycle constraint)

These constraints are encoded as quadratic penalties, resulting in a QUBO suitable for Ising conversion.

## 5. Methodology: QAOA in Qiskit

The QUBO is converted to an Ising Hamiltonian using Qiskit's QuadraticProgram and QAOA is executed using SamplerQAOA. The key steps include:

- Constructing the QUBO
- Converting to Ising form
- Building the QAOA circuit
- Sampling and evaluating bitstrings

Working QAOA Cell (Cell 8):

```
from qiskit.algorithms import QAOA
from qiskit.primitives import Sampler
from qiskit_optimization.algorithms import MinimumEigenOptimizer

qaoa = QAOA(sampler=Sampler(), reps=1)
optimizer = MinimumEigenOptimizer(qaoa)
result = optimizer.solve(problem)
print(result)
```

## 6. Experimental Details

Dataset: Data-sheet.csv with 18 currency exchange rates.

Data loading uses upward directory search to locate the file.

Mapping table is generated using Option A to preserve CSV row order.

Mapping Code:

```
edges = list(zip(df["Source"].tolist(), df["Target"].tolist(), df["Rate"].tolist()))
mapping = []
for i, (src, dst, rate) in enumerate(edges):
    mapping.append((i, f"x{i}", f"{src} -> {dst}", rate))
```

## 7. Results and Discussion

## Currency Arbitrage Optimization using QAOA in Qiskit

Ising Operator:

$0.5 * \text{IIZIZZ} + 0.5 * \text{IZZIIZ} + \dots$

Offset: -2.5

QAOA sampling produced valid bitstrings with low energy. Classical Bellman-Ford detected a profitable cycle:

USD -> CAD -> EUR -> USD with product 1.015 and profit 1.5%.

Break-even transaction fee: 1.48%

## 8. Comparison and Analysis

Classical methods are exact and fast for small graphs. QAOA offers scalability and quantum readiness. Constraint handling is explicit in classical solvers and Hamiltonian-based in QAOA. Qiskit QAOA showed correctness and feasibility for small instances.

## 9. Future Directions

Future work includes scaling to larger currency networks, using deeper QAOA layers, integrating hybrid classical-quantum workflows, and deploying on real quantum hardware.

## 10. Conclusion

This project demonstrates that Qiskit-based QAOA can model currency arbitrage as a QUBO problem and reproduce classical results. While classical solvers remain efficient for small cases, QAOA provides a scalable and interpretable quantum alternative.

## Appendix: Key Code Snippets

Data Loading Function:

```
def find_data_file(filename="Data-sheet.csv", folder_name="Data", max_up=5):
    p = Path.cwd()
    for _ in range(max_up + 1):
        candidate = p / folder_name / filename
        if candidate.exists():
            return candidate
    p = p.parent
    return None
```