RESEARCH ARTICLE

Enhancing Knapsack-Based Financial Portfolio Optimization Using Quantum Approximate Optimization Algorithm

CHANSREYNICH HUOT[ID][1]**, KIMLEANG KEA**[1]**, TAE-KYUNG KIM**[ID][2]**,
AND YOUNGSUN HAN**[ID][1]**, (Member, IEEE)**
[1]Department of AI Convergence, Pukyong National University, Nam-gu, Busan 48513, South Korea
[2]Department of Management Information Systems, Chungbuk National University, Seowon-gu, Cheongju-si, Chungcheongbuk-do 28644, South Korea

Corresponding author: Youngsun Han (youngsun@pknu.ac.kr)

ABSTRACT Portfolio optimization is a primary component of the decision-making process in finance, aiming to tactfully allocate assets to achieve optimal returns while considering various constraints. Herein, we proposed a method that uses the knapsack-based portfolio optimization problem and incorporates the quantum computing capabilities of the quantum walk mixer with the quantum approximate optimization algorithm (QAOA) to address the challenges presented by the NP-hard problem. Additionally, we present the sequential procedure of our suggested approach and demonstrate empirical proof to illustrate the effectiveness of the proposed method in finding the optimal asset allocations across various constraints and asset choices. Moreover, we discuss the effectiveness of the QAOA components in relation to our proposed method. Consequently, our study successfully achieves the approximate ratio of the portfolio optimization technique using a circuit layer of $p \geqslant 3$, compared to the classical best-known solution of the knapsack problem. Our proposed methods potentially contribute to the growing field of quantum finance by offering insights into the potential benefits of employing quantum algorithms for complex optimization tasks in financial portfolio management.

INDEX TERMS Best-known solution (BKS), portfolio optimization, knapsack problem, quantum approximate optimization algorithm (QAOA).

I. INTRODUCTION

The financial industry plays a remarkable role in the economic health and growth of every country. In the constantly evolving world of financial markets, technological advancements are shaping traditional portfolio optimization methods, presenting new opportunities and challenges; this process involves enlarging assets to reduce risk by offsetting individual risk profiles crucial to the investment process [1]. Although there are many portfolio optimization models, they possibly have limitations. For example, the mean-variance model, which provides solutions as percentages of the total budget, can result in fractional allocations of non-feasible

The associate editor coordinating the review of this manuscript and approving it for publication was Huaqing Li[ID].

assets [2]. In the constant pursuit of optimization, some studies have explored extensions to portfolio optimization [3]. Among these, a technique that reformulates the financial portfolio optimization problem as a knapsack-based problem has been proposed [4], [5].

Knapsack problem [2] was adapted for portfolio optimization by treating all assets included in the portfolio as items. The value of each asset is represented by its expected return, usually estimated using historical data or forecasting techniques. The weight of each item corresponds to a risky asset, typically measured by its standard deviation or the variance-covariance matrix of the assets. The knapsack capacity corresponds to the budget or available capital that can be invested in the portfolio. By reformulating the knapsack problem, the primary objective is to maximize

the total value of the portfolio while satisfying the capacity constraint and potentially other constraints, such as a target return or a minimum number of assets. However, similar to [6], the knapsack problem is considered NP-hard even with polynomial-bounded weights and values. Therefore, exploring new paradigms for optimization owing to the computational complexity of solving this problem is crucial [7], [8].

Similarly, the paradigm of quantum computing, leveraging the properties of quantum mechanics, has been developed to solve complex problems intractable for classical computers [9], [10]. The financial domain is a primary aspect of quantum computing, with applications spanning price derivation, risk modeling, portfolio optimization, and fraud detection [11]. As technology redefines problem-solving in finance, the quantum approximate optimization algorithm (QAOA), rooted in quantum computing, offers the promise of efficiently finding approximate solutions for computationally demanding problems within the polynomial-bounded NP optimization complexity class [12]. In [13] and [14], the application of QAOA to the knapsack problem is explored for optimization purposes.

In this paper, we reformulate the classical portfolio optimization problem as a knapsack problem that allows for an efficient mapping onto quantum algorithms, particularly advantageous for NP-hard financial optimization. By treating each asset's expected return as value and its risk as weight, the knapsack model aligns portfolio constraints with total risk, permitting optimization under capacity constraints. Our approach introduces a novel Quantum Walk Mixer (QWM) combined with the Quantum Approximate Optimization Algorithm (QAOA), integrating the Quantum Fourier Transform (QFT) within the oracle. This QWM-QAOA framework ensures feasible states are targeted while reducing computational complexity. Empirical results demonstrate that QWM-QAOA enhances solution space exploration, providing robust portfolio configurations with high approximation ratios even under various noise conditions, thus showcasing a significant advancement in leveraging quantum algorithms for practical finance applications. The contributions of our study can be summarized as follows:

- We propose a portfolio selection approach that formulates the portfolio problem as a knapsack concern by incorporating expected returns from the Markowitz model [15] and setting the capacity according to the knapsack framework.
- We present a Quantum Walk Mixer to Quantum Approximate Optimization Algorithm (QWM-QAOA) for the knapsack problem, integrating a shallow circuit layer to decrease computational complexity and improve solution quality.
- We demonstrate that our model consistently enhances the identification of optimal solutions for the knapsack problem, achieving an impressive approximation ratio ranging from 100% to 95% in scenarios involving the selection of 2 to 8 stocks.

The paper is organized as follows: Section II provides the background knowledge. Section III explores the related work. Section IV details our proposed method. Section V demonstrates the experimental setup and performance evaluation. Section VI dissects the findings of the proposed method, underscoring its achievements and limitations. Finally, we conclude our study in Section VII.

II. BACKGROUND
In this section, we briefly describe the background knowledge of portfolio optimization, the knapsack problem, and the quantum approximate optimization algorithm (QAOA).

A. PORTFOLIO OPTIMIZATION
Portfolio optimization is a mathematical framework that maximizes returns while minimizing risks through strategically selecting assets within an investment portfolio [16]. This is typically achieved by strategically allocating the proportion of each asset in the portfolio to optimize the risk-return tradeoff by considering the specified risk tolerance. The process involves four steps: i) identifying suitable assets, ii) projecting anticipated yields based on historical data for future forecasts, iii) quantifying the risk by assessing the uncertainty of each asset, and iv) selecting the optimal portfolio that maximizes the expected yield for a given risk level [17]. One of the various models used in this study was the Markowitz model developed by Harry Markowitz in 1952 [15]. This analysis is in conjunction with the variance of the rate of return, providing a significant assessment of portfolio risk under a rational framework of assumptions. The Markowitz model represented the maximum expected yield by allocating funds into stocks as follows [18]:

$$R_i = \sum_{t=1}^{\infty} d_{it} r_{it}, \qquad (1)$$

where r_{it} indicates the anticipated return at time t per stock invested in, and d_{it} is the rate at which return in the i_{th} security where time t is discounted back to the presents. The standard deviation—the variance of return—is a statistical measure used as an indicator of the uncertainty or risk linked to return. These statistical indicators effectively measure the extent to which returns deviate unpredictably from the average value over a specific period. The variance represents the degree of variation exhibited by the return R_i concerning the expected return $[E(R_i)]$, as illustrated:

$$\sigma_i^2 = \frac{1}{N} \sum_{i=1}^{N} [R_i - E(R_i)]^2. \qquad (2)$$

The covariance of returns measures the relative riskiness of a security within a portfolio of securities. For two securities, denoted as i and j, the covariance of their returns as

$$\sigma_{ij} = E\{[R_i - E(R_i)][R_j - E(R)]\}. \qquad (3)$$

Furthermore, covariance can be measured depending on the variability of the two individual return series:

$$\rho_{ij} = \frac{\sigma_{ij}}{\sigma_i \sigma_j}, \tag{4}$$

where ρ_{ij} is the correlation coefficient of returns, and σ_i and σ_j are the standard deviation of R_{it} and R_{jt}. As previously stated, an efficient portfolio is characterized by selecting individual assets within the portfolio and weighting each asset. Therefore, the portfolio return is calculated as a weighted average of the returns of the individual investments within the portfolio. Next, X_i denoted as the weight and applied to each return of the portfolio takes form as follows:

$$\begin{aligned} R_p &= \sum_{t=1}^{\infty} \sum_{i=0}^{N} d_{it} r_{it} X_i \\ &= \sum_{i=0}^{N} X_i \left(\sum_{t=1}^{\infty} d_{it} r_{it} \right) \\ &= \sum_{i=0}^{N} X_i R_i, \end{aligned} \tag{5}$$

where R_i is independent of X_i. The simplified version of the variance of a portfolio can be written as

$$\sigma_p^2 = \sum_{i=0}^{N} X_i^2 \sigma_i^2 + \sum_{i=0}^{N} \sum_{j=1}^{N} X_i X_j \sigma_{ij}, \tag{6}$$

where σ_p^2 is the variance of the portfolio, X_i is the percentage of the investor's assets that are allocated to the i_{th} asset, and the σ_i^2 represents the variance of the asset j and the covariance between the returns for assets i and j denoted as σ_{ij}. Traditional asset allocation methods, such as the Markowitz theorem, reportedly provide solutions in percentages. This approach can suggest allocating half of a market share, which is often impractical [2]. Therefore, proposing a method for determining the number of shares for each asset is crucial; this involves the conversion of expected returns, prices, and budget into interval values and determines the priority and importance of each share by framing it within a knapsack-based model.

B. KNAPSACK PROBLEM

In this context, a given set of items with known sizes is selected and packed into a knapsack with a fixed capacity [19]. This problem is one of the simpler NP-hard problems in combinatorial optimization because it focuses on maximizing an objective function while adhering to a single resource constraint. To find the exact solutions, some techniques have been employed, such as relaxations, bounds, reductions, and other algorithmic approaches [20]. These techniques include genetic algorithms [21], dynamic programming [22], simulated annealing [23], Tabu search [24], and greedy algorithm [25]. These classical approaches are instrumental in providing optimal solutions. Although these conventional approaches are successful and valuable, they are limitations

in computation in the classical domain. Furthermore, the new paradigm of quantum computing introduces optimization techniques, including QAOA, to tackle problems such as the knapsack problem, yielding better solutions than classical computation techniques [26], [27].

C. QUANTUM APPROXIMATE OPTIMIZATION ALGORITHM (QAOA)

In combinatorial optimization, QAOA excels as solutions tailored for quantum computing while leveraging the strengths of classical computing [12], [28]. QAOA, a hybrid quantum-classical algorithm, has demonstrated remarkable effectiveness in addressing recent NP-hard problems, including Max-Cut [29], traveling salesman problem [30], and quadratic unconstrained binary optimization (QUBO) [31]. Consider a combinatorial optimization problem involving an N-bit binary string represented as $z = z_1 \cdots z_N$, with a classical objective function $f(z) : \{0, 1\}^N \rightarrow \mathbb{R}$ is maximized. The goal is to find a solution z that provides a high approximation to the maximum values of $f(z)$ [32]. QAOA encodes this classical objective function into a quantum Hamiltonian H_c,

$$H_c |z\rangle = f(z)|z\rangle. \tag{7}$$

Furthermore, H_c operates diagonally on the computational basis states of 2^N dimensional Hilbert space (n-qubit space). Ideally, the performance of the p-level QAOA improves with increasing p. For the p-level QAOA, the state $|+\rangle^{\otimes N}$ is initialized, while the H_c and a mixing Hamiltonian:

$$B = \sum_{j=1}^{N} \sigma_j^x, \tag{8}$$

are applied alternately with controlled duration, resulting in a wave function:

$$|\psi_p(\vec{\gamma}, \vec{\beta})\rangle = e^{-i\beta_p B} e^{-i\gamma_p H_c} \cdots e^{-i\beta_1 B} e^{-i\gamma_1 H_c} |+\rangle^{\otimes N}. \tag{9}$$

This variational wave function is characterized by $2p$ variational parameters, γ and β. The expected value of H_c in this state is determined through repeated measurements on a computational basis:

$$f_p(\vec{\gamma}, \vec{\beta}) = \langle \psi_p(\vec{\gamma}, \vec{\beta})| H_c |\psi_p(\vec{\gamma}, \vec{\beta})\rangle. \tag{10}$$

Furthermore, a classical computer is used to search for the optimal parameters (γ^*, β^*) and maximize the averaged output $f(\gamma^*, \beta^*)$:

$$\left(\vec{\gamma^*}, \vec{\beta^*} \right) = arg \max_{\vec{\gamma}, \vec{\beta}} f_p(\vec{\gamma}, \vec{\beta}). \tag{11}$$

Next, the approximation is assessed by the ratio between the optimized values $f_p(\vec{\gamma^*}, \vec{\beta^*})$ and the maximum possible value $f_{max} = max_z f(z)$, as the approximate ratio:

$$r = \frac{f_p \left(\vec{\gamma^*}, \vec{\beta^*} \right)}{f_{max}}. \tag{12}$$

This approximation ratio r reflects how close the QAOA solution is to the optimal classical solution. Typically, the

search for this approximate ratio begins with a random initial estimate of the parameter and performing gradient-based optimization [33].

III. RELATED WORK

Recent studies have explored hybrid approaches to enhance different quantum algorithms for knapsack-based portfolio optimization problems. For example, [27] introduces Quantum-Inspired Evolutionary Algorithm (QIEAs) for difficult knapsack problems. Similarly, [8] used Quantum-inspired meta-heuristic approaches for constrained portfolio optimization problems. These advancements highlight the potential of hybrid quantum-classical approaches for tackling real-world optimization challenges.

Our study builds upon several key pieces of research that delve into the application of QAOA to the knapsack problem concerning portfolio optimization. QAOA has been employed as a portfolio optimization method. In this approach, the portfolio optimization problem is transformed into a binary version. Accordingly, the weight vector is discretized, with the element taking values of either 0 or 1. In [34], a comprehensive study of quantum computing approaches for multi-knapsack problems is proposed by investigating some of the most prominent and state-of-the-art quantum algorithms using different quantum software and hardware tools. Consequently, quantum computing offers the potential for good and fast solutions to multi-knapsack optimization problems in various fields, such as logistics (allocating goods to containers), resource allocation in computing (distributing tasks among different servers), and financial portfolio optimization (allocating assets among different investment opportunities). In [35], the Markowitz models of portfolio optimization were converted into binary knapsacks. A hard constraint model was employed by incorporating hard constraints into the quantum algorithm, involving designing mixing operators based on the constraint conditions. Additionally, a combination of *XY*- and *XYY*-mixers was used to encode the constraints in the quantum circuit. *XY*-mixers were used to mix the quantum state and generate a superposition of feasible solutions, whereas *XYY*-mixers were used to enforce the hard constraints. Moreover, using a hard constraint model ensured that the quantum state evolved between feasible solutions satisfying the constraints while allowing for a high degree of flexibility in opting parameters. Another intriguing study highlights the strengths of the quantum walk optimization algorithm (QWOA) [36] compared to other quantum optimization algorithms while highlighting the challenges posed by the complex and large solution space associated with its lattice structure. Moreover, the quantum mixer optimization algorithm (QMOA) was introduced as an extension of QWOA. It enhanced the efficiency of quantum optimization algorithms for portfolio optimization by reducing the number of iterations required for computations [37].

IV. KNAPSACK-BASED PORTFOLIO OPTIMIZATION

In this section, we explain our overall architecture, knapsack-based portfolio formulation, and QAOA for the knapsack problem.

A. OVERALL ARCHITECTURE

This section outlines the complete process from stock data preprocessing to determining the optimal portfolio configuration using a quantum-enhanced approach. As shown in Figure 1, the workflow begins by processing historical stock prices to compute the daily returns using the Markowitz-based portfolio optimization model. Algorithm 1 calculates the expected returns (E(R)), which are crucial inputs for formulating the portfolio selection as a knapsack problem. This step maps the expected returns, stock weights, and capacity constraints into a knapsack framework to define the feasible portfolio configurations. In the next phase, Algorithm 2 solves the knapsack-based portfolio optimization problem. Here, we first construct the feasible oracle using Quantum Fourier Transform (QFT) [38] to ensure that only valid solutions are considered for further optimization. Once a feasible solution is identified, the Quantum Walk Mixer (QWM) is applied within the Quantum Approximate Optimization Algorithm (QAOA), following the approach of [36]. The combination of QWM and QAOA (QWM-QAOA) efficiently explores the solution space, minimizing computational complexity while improving the accuracy of the optimization. The final output is a binary string, where each bit represents whether a stock is included or excluded from the portfolio. For instance, a result of (1,0) for two stocks indicates that Stock A is selected, while Stock B is excluded, thus guiding the final portfolio selection and investment strategy.

B. KNAPSACK-BASED PORTFOLIO FORMULATION

Our methodology is based on the fundamental principles of mean-variance optimization, focusing on the Markowitz model. This model historically maximizes the anticipated yield of a portfolio while considering a predetermined level of risk, quantified by the variance of portfolio returns. We denoted the expected return $E(R_i)$ of stock i by using the mean return calculation. Next, we transformed the expected return into a knapsack problem. Subsequently, we extended this model by reinterpreting the expected return as a component of a knapsack problem, thereby aligning portfolio optimization with knapsack problem dynamics; this adaptation involves redefining the expected return of each stock as a value optimized under the constraints of total portfolio risk. This mathematical transformation effectively converts portfolio optimization into a knapsack problem:

$$\text{maximize} \quad R(x) = \sum_{i=1}^{n} x_i E(R_i)$$

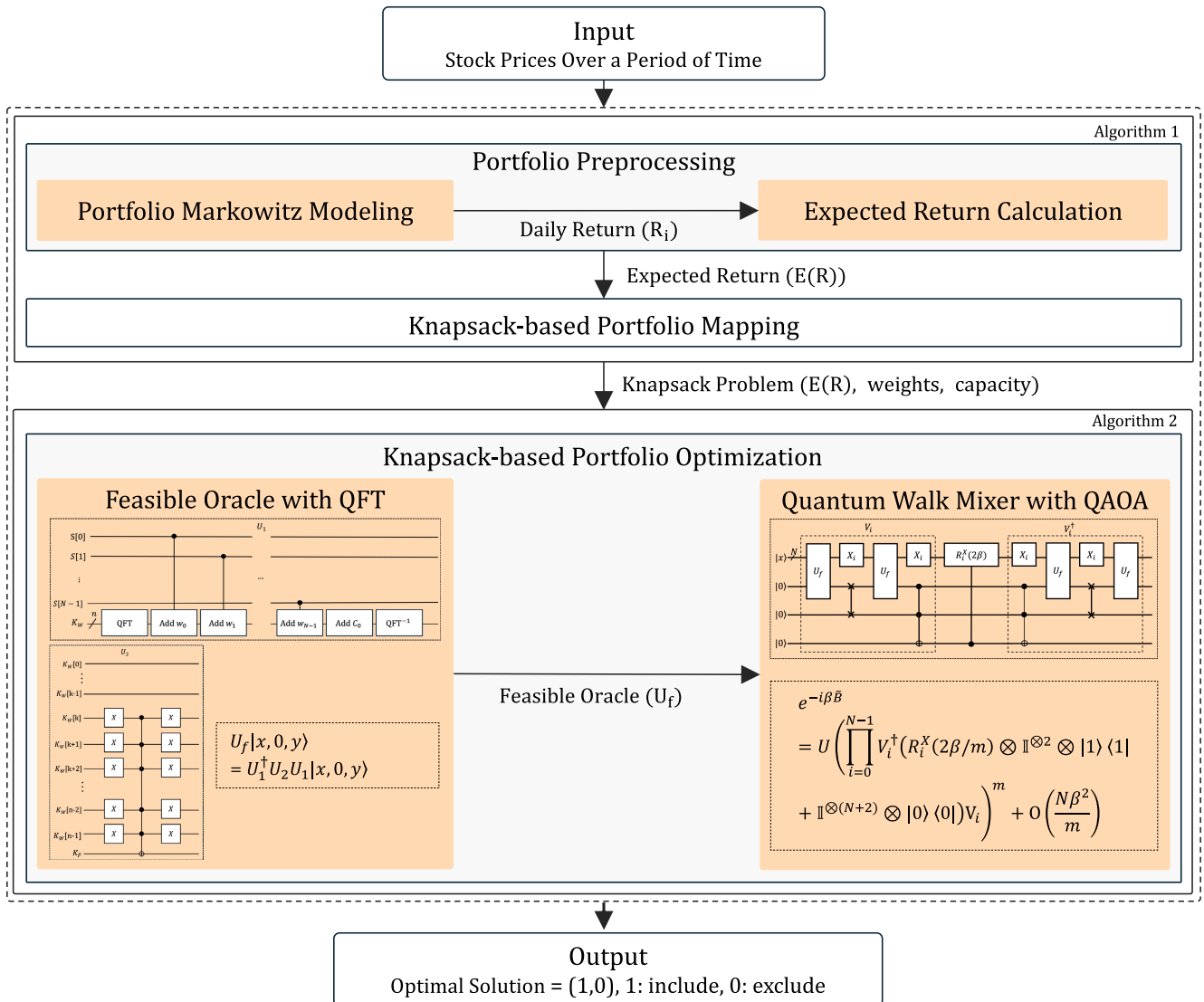$$\text{subject to} \quad \sum_{i=1}^{n} x_i w_i \leq C, \tag{13}$$

FIGURE 1. Workflow diagram of the quantum walk mixer with QAOA for knapsack-based portfolio optimization. The process starts with Algorithm 1, where a list of stock prices is collected over time to prepare historical stock prices using the Markowitz model to calculate the expected return. This expected return is crucial in mapping the knapsack problem by considering the values, weights, and capacity of the portfolio. The optimization process in Algorithm 2 involves employing a quantum walk mixer with QAOA, including the preparation of a feasible oracle. This is followed by executing the quantum walk mixer with QAOA to enhance the search for the optimal portfolio configuration. The final output is a binary string, with 1 indicating the inclusion and 0 indicating the exclusion of stocks from the portfolio, resulting in the optimal solution.

where the risk of including stock i in the portfolio, denoted as w_i, and the total risk tolerance, denoted by C, serves as the knapsack capacity. The binary variable x_i indicates whether to include stock i in the portfolio, and $E(R_i)$ represents its expected return values.

This process can be summarized as Algorithm 1. It begins by extracting the historical returns of each stock within a specified time range, from the start to the end date, using the *HistoricalReturn* function. From line $5 - 12$, these historical returns are then passed into the Markowitz model through the *ExpectReturn* function, which calculates the expected returns, denoted as $E(R)$. These expected returns provide key insights

into the potential profitability of the selected stocks in the portfolio problem. Specifically, the expected returns $(E(R))$ are assigned as the values in the knapsack problem, and the weights for each stock are uniformly set to 1, representing equal weighting for all assets. Afterward, the algorithm also defines the capacity constraint (C) of the knapsack problem, which is set to half the number of stocks, calculated as $Length(E(R))/2$. This framework effectively transforms the portfolio optimization problem into a knapsack problem formulation. This transformation allows for a more structured handling of assets and makes it possible to apply powerful combinatorial optimization techniques.

Algorithm 1 Formulate Portfolio Optimization as a Binary Knapsack Problem

---

INPUT: *start*: start date of stock prices, *end*: end date of stock prices, *stocks*: list of stocks

OUTPUT: *problem*: portfolio optimization modeled as a knapsack problem

1: **procedure** ENCODETOKNAPSACK(*start*, *end*, *stocks*)
2:     // Extract historical return
3:     *prices* ← HistoricalReturn(*start*, *end*, *stocks*)
4:     // Calculate the expected return
5:     $E(R)$ ← ExpectReturn(*prices*)
6:     // Set a weight of 1 to each stock
7:     **for** i ← 0 to Length($E(R)$) **do**
8:         *weights*[i] ← 1
9:     **end for**
10:     // Set the capacity of knapsack
11:     C = Length($E(R)$) / 2
12:     // Formulate the problem
13:     *problem* ← ($E(R)$, *weights*, C)
14:     **return** *problem*
15: **end procedure**

Algorithm 2 Optimize Knapsack Problem Using the QWM-QAOA Algorithm

---

INPUT *problem*: knapsack problem, *p*: circuit layer, *m*: trotter step count for QAOA

OUTPUT *ar*: approximation ratio as optimal choice

1: **procedure** OPTIMIZEKNAPSACK(*problem*, *l*, *m*)
2:     // Determine problem choices
3:     *choices* = Length(*problem.weights*)
4:     // Calculate the total weight
5:     *total_weights* = TotalWeight(*problem.weights*)
6:     // Setup quantum registers
7:     c_{reg} = QuantumRegister(*choices*)
8:     w_{reg} = QuantumRegister(*total_weights*)
9:     // Initialize quantum register
10:     a_{reg} = QuantumRegister(3)
11:     // Define quantum mixing
12:     H_m = QWalkMixer(*problem*,m, c_{reg}, w_{reg}, a_{reg})
13:     // Define phase operations
14:     H_c = PhaseHamiltonian(c_{reg}, *problem*)
15:     // Start with superposition state
16:     $|\psi_0\rangle = |+\rangle^{\otimes choices}$
17:     **for** k = 1 to p **do**
18:         **if** k ← 1 **then**
19:             γ_k, β_k = Initialize()
20:         **else**
21:             $(\gamma_k, \beta_k) = (\gamma_{opt}, \beta_{opt})$
22:         **end if**
23:         // Apply the unitary operators
24:         $U_B(\beta_k) = e^{-i\beta_k H_M}$
25:         $U_c(\gamma_k) = e^{-i\gamma_k H_c}$
26:         $|\psi_k\rangle = U_c(\gamma_k)U_B(\beta_k)|\psi_{k-1}\rangle$
27:         // Define the QAOA cost function
28:         $cost(\gamma, \beta) = \langle\psi_k|H_c|\psi_k\rangle$
29:         // Optimize the cost function
30:         $(\gamma_{opt}, \beta_{opt})$ = ShgoOptimizer($cost(\gamma, \beta)$)
31:     **end for**
32:     // Measure the final state
33:     *probabilities* = MeasureProbabilities($|\psi_p\rangle$)
34:     // Obtain the optimal probability
35:     *ar* = GetOptimalSolution(*probabilities*)
36:     **return** *ar*
37: **end procedure**

C. QAOA FOR THE KNAPSACK PROBLEM

After formulating the portfolio optimization as a knapsack problem, we optimized it using Algorithm 2. We employed the QWM-QAOA to prepare the mixing Hamiltonian for the QAOA process. In lines $2 - 10$, we set up the quantum registers and other necessary preparations for QAOA using the *QuantumRegister* function. Line 12 focuses on constructing the Mixing Hamiltonian H_m using the QWM-QAOA approach, which is achieved via the *QWalkMixer* function. This step requires parameters such as a predefined trotter step *m*, the number of quantum registers, weights, and ancillaries. Next, the algorithm prepares the phase Hamiltonian H_c using *PhaseHamiltonian* (line 13), which plays a crucial role in the construction of the QAOA. More details on these concepts will be provided in the following sections. At each circuit layer *p*, the algorithm computed the expectation value (line 28), facilitating the determination of a new set of optimal 2*p* parameter ($\gamma_{opt}, \beta_{opt}$). These optimal parameters were identified using the SHGO optimization method and implemented as *ShgoOptimizer* line 29. Finally, the algorithm measured probabilities (line 31) through a measurement process. These probabilities were used to determine the optimal solution using the *OptimalChoice* function to choose the best selection of items represented as *ar* as the optimal choice of approximation ratio.

1) FEASIBILITY ORACLE

The feasibility oracle is used as a hypothetical subroutine that instantly determines whether a proposed solution to the knapsack problem violates any constraints [39]. We can explore solutions within a well-defined space of bitstrings by representing our portfolio model as a binary knapsack problem. We defined $K(N) = (0, 1)^N$ as the set of all possible bitstrings of length N representing potential portfolio choices. Furthermore, each possible choice of any of the N items is represented by a bitstring $x \in K(N)$.). Thus, the subset feasible solution was denoted as F for the knapsack problem, and the feasibility function was defined as

$$f : K(N) \rightarrow \{0, 1\}, \quad x \mapsto f(x) = \begin{cases} 1, & \text{if } x \in F, \\ 0, & \text{otherwise.} \end{cases} \quad (14)$$

Considerably, the feasibility oracle to be unitary U_f as:

$$U_f |x, y\rangle = |x, y \oplus w(x)\rangle, \quad \forall x \in K(N), \quad y \in K(1) \quad (15)$$

In portfolio optimization, a state $|x\rangle$ symbolizes a specific stock allocation, which is deemed feasible (i.e., $x \in F$) if the total weight $w(x)$ does not exceed the capacity C. This oracle toggles a flag qubit $|y\rangle$ based on the feasibility of the state $|x\rangle$, representing a possible portfolio configuration.

Next, we allocated qubits for storage as follows: S to record the formulated knapsack choices, K_w to hold the weight of the item choice, and K_F as the flag qubit indicating the feasibility of the state S. Remarkably, the number of qubits required for S, K_w, and K_F are deonted by Q_S, Q_w, and Q_F respectively. Upon that, the total number of qubits that are required is $Q_S + Q_w + Q_F$.

Furthermore, the total weight $w(x)$ is calculated by adding the weight of each item to register K_w, controlled by the corresponding bit in register S. We also compare the computed weight $w(x)$ to the capacity C using an inequality check facilitated by a multiple-controlled NOT gate.

For suitable $W_0 \in \mathbb{N}$, the inequality can be verified using the following condition:

$$w(x) \leq C \Leftrightarrow w(x) + C_0 < C + C_0 + 1 \quad (16)$$

This condition ensures that the binary representation of $w(x) + C_0$ has zeros in all positions beyond k. The feasibility oracle U_f employs two primary unitary operations- U_1 and U_2. U_1 augments K_w with a predetermined offset C_0, facilitating the binary representation required for the feasibility check; U_2 is applied conditionally based on U_1's outcome.

We begin by illustrating how U_1 is prepared. Quantum Fourier Transform (QFT) [38] is employed within the unitary operation U_1 to facilitate the process of adding weights in the knapsack problem.

This process starts with the initialization of the ancillary weight register K_w to $|0\rangle$. Next, QFT is applied to K_w to prepare the register for quantum addition, even though the QFT does not change the state for state $|0\rangle$. This step is necessary for nonzero initial states K_w. Post-state transformation of K_w from the computational basis to the Fourier basis, the weight representation was distributed across the amplitude of the quantum state:

$$QFT |K_w\rangle = \frac{1}{\sqrt{2^n}} \sum_{k=0}^{2^n-1} e^{2\pi i \cdot 0 \cdot k / 2^n} |k\rangle \quad (17)$$

where we use the notaion $e(t) = exp(2\pi i t)$, and $k \in \mathbb{Z}$ is the binary representation of $k_n - 1, \cdots, k_0$. Furthermore, weights were added to $QFT |K_w\rangle$ using controlled operations applying phase rotations. These operations are conditioned on the bits of the bitstring x indicating the selection of items. Each bit in x determines whether the corresponding weight w_i should be added to K_w. The addition in the Fourier space was related to phase rotation and implemented by a sequence

of controlled phase gates. The magnitude of the rotation was determined by w_i and the position of the bit controlling the operation. The implementation of controlled additions can be described as follows: for each selected item (where the corresponding bit in x is $|1\rangle$, a controlled phase rotation is applied to K_w. The phase added to each computational basis state $|k\rangle$ within K_w is proportional to w_i:

$$|K_w\rangle = \begin{cases} |K_w + w_i\rangle, & \text{if } x_i = 1, \\ |K_w\rangle, & \text{otherwise.} \end{cases} \quad (18)$$

This phase rotation effectively encodes the addition of w_i into the quantum state. After completing this process, the step transformed the register back to the computational basis required, where

$$QFT^{-1} |K_w\rangle \quad (19)$$

The inverse QFT decodes the phase information back into a computational state representing the total weight of the selected items. If QFT encodes the weight as a superposition of phases, the inverse QFT converts these phases back into a binary number the sum of the weights. The final state of the register K_w after applying U_1 was the binary representation of the total weight of the items:

$$U_1 |x, 0, y\rangle = |x, w(x) + C_0, y\rangle \quad (20)$$

Figure 2 shows how U_1 is implemented using an algorithm based on QFT. Following the implementation of the unitary operation U_1, U_2 is described in Figure 3, which comprises a quantum circuit that conditionally modifies the state of an ancillary qubit concerning the sum of weights represented in the register K_w. Referring to Equation 16, the ancillary qubit $|y\rangle$ underwent a state flip to signal a valid configuration when the total weight $w(x)$, augmented by a constant offset C_0 is less than the predefined threshold $C + C_0 + 1$. This transformation was executed through several multi-controlled-NOT gates, where each gate was influenced by a qubit distinct from the weight register. The successful transition of the ancillary qubit's state post U_2 signified a feasible solution, adhering to the knapsack's capacity, which effectively segregates the solution space into permissible and impermissible weights.
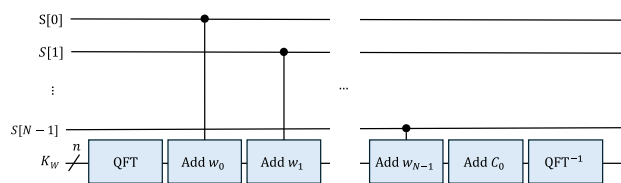

FIGURE 2. Circuit implementation of U_1. It encodes the total weights of item selections into the ancillary register K_W using quantum Fourier transform-based arithmetic. A predefined offset C_0 is added to the register to facilitate subsequent feasibility checking.
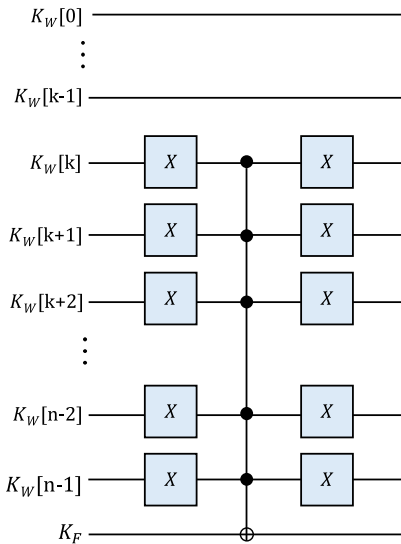
FIGURE 3. Circuit implementation of U_2. It executes a conditional check based on the total weight calculated by U_1 using several multi-controlled NOT gates to flip the ancillary qubit if the weight condition is satisfied.

2) QUANTUM WALK MIXER FOR ENFORCING CONSTRAINTS

This process relies on the mixing operator $U_B(\beta)$ generated by mixing Hamiltonian B from Equation 8 as follows:

$$U_B(\beta) = e^{-i\beta B} \tag{21}$$

Alternatively, by leveraging the feasible function f in Equation 14 and neighboring states $n_i(x)$ for exploring the solution space, B can be described as

$$B\,|x\rangle = \sum_{i=0}^{N-1} f_i(x)\,|n_i(x)\rangle, \quad \forall x \in K(N) \tag{22}$$

where $n_i(x)$ is the i_{th} neighbor of x (x with i_{th} bit flipped). Using this alternative representation, it is easy to observe that $\forall x, x' \in K(N)$.

$$\langle x|\,B\,|x'\rangle = \begin{cases} 1, & \text{if } \text{Ham}(x, x') = 1, \\ 0, & \text{otherwise} \end{cases} \tag{23}$$

where Ham (x, x') represents the Hamming distance between two binary strings x and x', which is the count of positions where the corresponding bits differ. The exact implementation of the desired mixing operator $U_B(\beta)$ presents challenges due to its Hamiltonian, requiring additional resources and non-commuting elements. To address this, we employ an alternative operator $\tilde{B}$, closely resembling B. This operator is constructed from V_i, and the inverse $V_i^\dagger$ for encoding feasibility information of neighboring states into auxiliary qubits that enable the controlled state manipulation. Additionally, the operator includes single-qubit X_i gates for creating neighboring state exploration. Moreover, U_f the feasible oracle is included to determine the feasibility of states by ensuring that only valid states are mixed. Finally, $R_i^X(2\beta)$ aids in adjusting the amplitudes of states based on
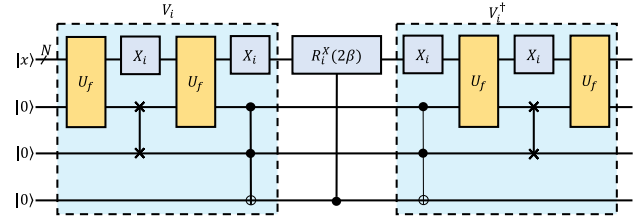

FIGURE 4. General circuit for a single-qubit quantum walk mixer. It illustrates the core operations involved in mixing a quantum state with its feasible neighbor, guided by feasibility oracles and single-qubit rotations, as part of a constrained optimization algorithm.

feasibility and further refining the mixing process, whereas auxiliary qubits are employed to store feasibility information.

As shown in Figure 4, the process starts from:

$$V_i V_i^\dagger\,|x, 0, 0, 0\rangle = |x, 0, 0, 0\rangle \tag{24}$$

which implies that the behavior of V_i when acting on a state $|x, 0, 0, 0\rangle$ is to transform it into a new state with additional information encoded in the auxiliary qubits, dependent on the feasibility function f and its value at neighboring points. Therefore, the state of V_j can be described as

$$V_i\,|x, 0, 0, 0\rangle = |x, f(n_i(x)), f(x), f_i(x)\rangle \tag{25}$$

Next, neighboring state mapping is performed using $V_i^\dagger$ to determine whether the original and neighboring states have the same feasibility. Thus, Equation 24 can be written as

$$\begin{aligned} V_i^\dagger\,&|x, f(n_i(x)), f(x), f_i(x)\rangle \\ &= |n_i(x), f(x) \oplus f(n_i(x)), f(x) \oplus f(n_i(x)), f_i(x) \oplus f_i(x)\rangle \\ &= \begin{cases} |n_i(x), 0, 0, 0\rangle, & \text{if } f(x) = f(n_i(x)), \\ |n_i(x), 1, 1, 0\rangle, & \text{otherwise} \end{cases} \end{aligned} \tag{26}$$

Then, the bit flip of the $i - th$ qubit acts as:

$$X_i\,|x\rangle = |n_i(x)\rangle \tag{27}$$

Therefore, we can describe the encoded feasibility information after applying V_i and based on Equation 25 as follows:

$$\begin{aligned} (X_i \otimes \mathbb{I}^{\otimes 2} \otimes |1\rangle\,\langle 1|) V_i\,&|x, 0, 0, 0\rangle \\ &= f_i(x)\,|n_i(x), f(n_i(x)), f(x), f_i(x)\rangle \end{aligned} \tag{28}$$

Next,

$$\begin{aligned} V_i^\dagger\left(X_i \otimes \mathbb{I}^{\otimes 2} \otimes |1\rangle\,\langle 1|\right) V_i\,&|x, 0, 0, 0\rangle \\ &= f_i(x)\,|n_i(x), 0, 0, 0\rangle \end{aligned} \tag{29}$$

Thus, for a given $\tilde{B}$ operator, which approximates the original B operator, we have

$$\tilde{B} = \sum_{i=0}^{N-1} V_i^\dagger\left(X_i \otimes \mathbb{I}^{\otimes 2} \otimes |1\rangle\,\langle 1|\right) V_i \tag{30}$$

Moreover, $\tilde{B}$ implements B given three ancillary qubits, or more precisely, it is represented as:

$$\tilde{B}|x,0,0,0\rangle = \sum_{i=0}^{N-1} f_i(x)|n_i(x),0,0,0\rangle$$
$$= (B \otimes \mathbb{I}^{\otimes 3})|x,0,0,0\rangle, \quad \forall x \in K(N). \quad (31)$$

It follows that

$$e^{-i\beta\tilde{B}}|x,0,0,0\rangle = e^{-i\beta(B \otimes \mathbb{I}^{\otimes 3})}|x,0,0,0\rangle$$
$$= \left(U_B(\beta) \otimes \mathbb{I}^{\otimes 3}\right)|x,0,0,0\rangle \quad (32)$$

where $\tilde{B}$ is used to generate U_B. Moreover, $\tilde{B}$ is not commute and should use three ancilla qubits. Therefore, the approximate implementation of $e^{-i\beta\tilde{B}}$ needs to use the trotter product formula. Meanwhile, using the identity $R_i^X(2\beta)$, the state representing the rest of the circuit in Figure 4.

$$e^{-i\beta V_i^\dagger (X_i \otimes \mathbb{I}^{\otimes 2} \otimes |1\rangle\langle 1|)V_i}$$
$$= V_i^\dagger \left(R_i^X(2\beta) \otimes \mathbb{I}^{\otimes 2} \otimes |1\rangle\langle 1| + \mathbb{I}^{\otimes(N+2)} \otimes |0\rangle\langle 0| \right)V_i \quad (33)$$

where $R_i^X(2\beta) = cos(\beta)\mathbb{I}^{\otimes N} - isin(\beta)X_i$. Furthermore, we optimized $U_B(\beta) \otimes \mathbb{I}^{\otimes 3}$ with the trotter product formula as

$$e^{-\beta\tilde{B}} = \left(\prod_{i=0}^{N-1} V_j^\dagger \left(R_i^X\left(\frac{2\beta}{m}\right) \otimes \mathbb{I}^{\otimes 2} \otimes |1\rangle\langle 1| \right. \right.$$
$$\left. \left. + \mathbb{I}^{\otimes(N+2)} \otimes |0\rangle\langle 0| \right)V_j \right)^m$$
$$+ O\left(\frac{N\beta^2}{m}\right) \quad (34)$$

Moreover, to maintain β the validity of the approximation within the new parameter range, the β range needs to be adjusted accordingly to $\beta \in [0, m\pi)$.

As an objective function, we use the value function v. Therefore, the corresponding phase separation can be described as follows:

$$U_C(\gamma)|x\rangle = e^{-i\gamma v(x)}|x\rangle \quad (35)$$

3) NUMBER OF QUBIT REQUIREMENTS

The presented illustration implies that using a feasibility oracle with n_f ancilla qubits requires $N + n_f + 3$ qubits. In contrast, there is an exigency for an additional three qubits, specifically for generating the unitary U_B (Equation 32).

V. EVALUATION

This section presents the numerical results by evaluating the effectiveness of the proposed algorithm across different asset numbers along with the impact of parameter configuration.

A. EXPERIMENTAL SETUP

We present the experimental setup for evaluation, specifically the datasets, hyperparameter configuration, and model setup used to evaluate our proposed approach.

1) HYPERPARAMETER CONFIGURATION

Initially, we implemented the proposed method using the Qiskit library [40], a widely used open-source quantum computing framework. We run the algorithms on the QASM simulator, a tool provided by Qiskit for simulating quantum circuits, which helped us gain insights into the behavior and performance of the algorithms before moving on to the real quantum device. Moreover, we used a fake backend noise from 127 qubits of IBM FakeWashington. While using the real device, we used IBM Nazca, a 127-qubit quantum device, for quantum computations, which offers the opportunity to test the algorithms in a real-world quantum computing environment. The experimental phase involves conducting several tests in which we select varying stocks from *yahoofinance*, such as Apple Inc. (AAPL), Amazon.com Inc. (AMZN), Alphabet Inc. (GOOGL), Microsoft Corporation (MSFT), NVIDIA Corporation (NVDA), and Tesla, Inc. (TSLA), Netflix, Inc. (NFLX), and Visa Inc. (N). Notably, the selection of these stocks was based on different scenarios, including all cases from two to eight stocks. Postselection, we proceeded with the experimental setup of the proposed method. During the optimization process, we focused on optimizing QAOA, which was achieved by optimizing the $2p$ angle parameters, i.e., β and γ, using the classical optimizer SHGO [41]. Additionally, we integrated quantum walk to boost optimization by leveraging its benefits for refining the process and achieving superior results.

TABLE 1. Hyperparameter Configuration.

Hyperparameter	Value
Number of circuit layers	3
Number of trotter steps	3
Financial data provider	yahoo finance
Portfolio valuation module	PyPortfolioOpt
Classical optimizer	SHGO
Framework for architecture implementation	Qiskit

2) PROBLEM SETUP

To evaluate our proposed method's performance, we defined our algorithm's problem based on the number of assets to be optimized (Table 2). First, we selected 2–8 subsets of stocks using the capabilities of PyPortfoliopt [42]. The values represent the expected return of stocks counting from January 1, 2018, to January 1, 2023. The knapsack capacity Cap is the maximum weight, where in our case we set it to be half of the total number of assets. This step clarifies the problem and enables us to tailor our algorithm accordingly. We aimed to evaluate the effectiveness of the QAOA optimization technique in portfolio optimization. To achieve this, we specifically evaluated the performance of

TABLE 2. The optimal stock selection and corresponding values for various portfolio sizes under the binary knapsack constraint. The table displays the number of selected stocks (#Ss), stock types (Type), their respective values, the knapsack capacity (Cap), the Best Known Solution (BKS), and the number of qubits used (#Qs). The BKS column lists the binary string representation of the selected stocks, reflecting the optimal solution under the given capacity constraint.

#Ss	Type	Values	Cap	BKS	#Qs
2	MSFT	0.2430	1	(0,1)	7
	AAPL	0.2602			
3	MSFT	0.2430	1	(0,1,0)	8
	AAPL	0.2602			
	NVDA	0.2430			
4	MSFT	0.2430	2	(1,1,0,0)	10
	AAPL	0.2602			
	GOOGL	0.1047			
	NVDA	0.2430			
5	MSFT	0.2430	2	(0,1,0,0,1)	11
	AAPL	0.2602			
	GOOGL	0.1047			
	AMZN	0.0716			
	NVDA	0.2430			
6	MSFT	0.2430	3	(0,1,0,0,1,1)	12
	AAPL	0.2602			
	GOOGL	0.1047			
	AMZN	0.0716			
	NVDA	0.2430			
	TSLA	0.4203			
7	MSFT	0.2430	3	(0,1,0,0,1,1,0)	13
	AAPL	0.1870			
	GOOGL	0.1749			
	AMZN	0.1898			
	NVDA	0.2856			
	TSLA	0.4203			
	NFLX	0.0797			
8	MSFT	0.2430	4	(1,1,0,0,1,1,0,0)	15
	AAPL	0.1899			
	GOOGL	0.1780			
	AMZN	0.1903			
	NVDA	0.2874			
	TSLA	0.4203			
	NFLX	0.0797			
	V	0.1341			

the algorithm using the best-known solution (BKS), a result of using a classical algorithm [43], where 1 signifies selection and 0 denotes exclusion. This technique compared the results obtained from our algorithm with those of the classical solution, providing a benchmark for assessing its performance and effectiveness. By undertaking this evaluation, we can gain valuable insights into the capabilities and limitations of the QAOA optimization technique in portfolio optimization.

B. PERFORMANCE EVALUATION

In this section, we present our model performance and its sensitivity analysis.

1) MODEL PERFORMANCE

By meticulously exploring the variable settings within our QWM-QAOA, we identified that a circuit layer depth of $p = 3$ and a Trotter step value of $m = 3$ yielded the best results. Figure 5 illustrates the approximation ratios for knapsack-based portfolio optimization with varying numbers of stocks (ranging from 2 to 8 stocks) under three

different environmental conditions: noise-free (simulator), noisy (using FakeWashington as a simulated backend), and noisy (real device: IBM Nazca with 127 qubits).

- Noise-free (Simulator): The approximation ratios remain very high across all stock configurations, ranging from 100% (for 2 and 3 stocks) to 96% (8 stocks), indicating that the algorithm performs near-perfectly in an ideal, noise-free environment.
- Noisy (Fake Backend): The introduction of noise through the simulated FakeWashington backend results in a decline in performance. The approximation ratios range from 98% to 70%, with lower values observed for larger portfolios. For example, 2 stocks achieve an approximation ratio of 98%, while 6 and 7 stocks drop to 70%. Interestingly, for 8 stocks, the approximation ratio slightly improves to 78%, indicating a non-linear impact of noise for larger portfolios.
- Noisy (Real Device): The experiment is executed on the real IBM Nazca device with the approximation ratios dropping further, ranging from 96% (for 2 stocks) to 51% (for 6 stocks). Surprisingly, for 7 and 8 stocks, the approximation ratios increased to 52% and 55%, respectively.

This performance reduction is due to real-world factors, such as gate errors and decoherence, which exert a more significant impact as the number of stocks increases. This unexpected result hints at the potential for further improvements, even as the number of stocks increases in a noisy environment [44]. These issues highlight the importance of advanced error mitigation techniques, though such methods are beyond the scope of this study [45], [46].
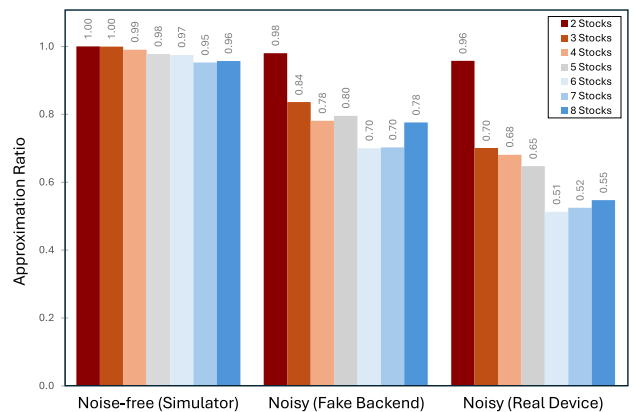

FIGURE 5. Approximation ratio for different scenarios involving various numbers of stocks under three distinct environmental conditions: noise-free with the simulator, noisy with Fake Washington, and noisy with the IBM Nazca device.

2) SENSITIVITY ANALYSIS

We conducted a sensitivity analysis of key parameters using our QWM–QAOA algorithms, such as the circuit layer (p) and the number of trotter steps (m). These parameters are crucial for shaping the effectiveness of our
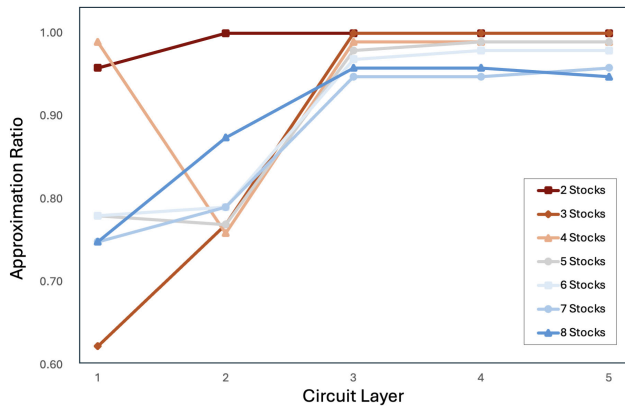
FIGURE 6. Approximation ratio of the circuit layer (p) is 1–5 across all problems. Remarkably, when $p \geqslant 3$, the approximation ratio reaches 100%, indicating that the proposed method is efficient even at the lower circuit layer.

proposed algorithm. Through a methodical examination and evaluation, we present details of the relationship between these parameters and how much they influence the overall efficiency and resilience of our QWM–QAOA.

- **Circuit Layer:** Figure 6 shows that as the circuit layer increases beyond the initial value of $p \geqslant 3$, the approximation ratio of the diverse portfolio optimization result consistently reaches an impressive result ranging from 100% to 95%, categorically based on the setup cases of portfolio optimization. This empirical evidence emphatically establishes that our performance consistently produces an efficient result even at some circuit layers.

- **Trotter Step in QWM-QAOA:** We aimed to find the parameter m, representing the trotter step counts in the QWM–QAOA algorithm while maintaining a constant circuit layer of $p = 3$ and optimize β, γ for the resulting circuit of 1–5. Table 3 shows that the approximation ratio for various stock selections increasingly exhibits a better result starting from $m = 3$, ranging from 100% to 95%. When $m = 5$, we obtained efficient optimization results from our proposed method ranging from 100% to 96%. The approximation ratios demonstrate how increasing the trotter steps improves the portfolio optimization results, particularly when $m \geqslant 3$, yielding consistently higher performance across different portfolio problem sizes.

TABLE 3. Performance of QWM-QAOA with varying Trotter steps across different stock selection scenarios. The table presents the approximation ratios for various numbers of stocks, ranging from 2 to 8, using the QWM-QAOA algorithm with different Trotter step values (1 to 5).

#Stocks	$m = 1$	$m = 2$	$m = 3$	$m = 4$	$m = 5$
2	1.00	0.96	1.00	1.00	1.00
3	1.00	0.60	1.00	1.00	1.00
4	0.77	0.99	0.99	0.99	0.99
5	0.96	0.96	0.98	0.98	0.98
6	0.96	0.97	0.98	0.98	0.98
7	0.96	0.96	0.95	0.96	0.96
8	0.96	0.96	0.96	0.96	0.96

VI. DISCUSSION AND LIMITATION

our research provides a novel application of quantum computing, specifically through the Quantum Walk Mixer combined with a Quantum Approximate Optimization Algorithm (QWM-QAOA) integrating with Quantum Fourier Transform (QFT), to address the knapsack-based financial portfolio optimization problem. By transforming the traditional portfolio optimization problem into a binary knapsack problem formulation, we were able to leverage quantum computing to provide efficient and effective optimization strategies, particularly for NP-hard problems. The results from our experiment on noise-free simulation indicate that the QWM-QAOA approach can yield high approximation ratios in both fake backend and real quantum devices, demonstrating the potential of quantum computing in financial optimization tasks. Our study also highlights the importance of various QAOA parameters, such as circuit layer p and trotter step m, in optimizing the performance of our proposed algorithm. Through sensitivity analysis, we found that a circuit layer of $p \geqslant 3$ and the number of trotter steps $m \geqslant 3$ provided the most converged results, with approximation ratios consistently reaching between 100% and 96% in the best cases. This empirical evidence supports the notion that QAOA with the proper configuration, can outperform classical optimization approaches in the context of financial portfolio optimization problems. However, our experiments on real quantum hardware revealed some limitations. The approximation ratio dropped significantly due to unavoidable hardware noise such as gate errors and decoherence errors, with real-device performance yielding ratios as low as 50%. Despite these current limitations of quantum hardware, this finding emphasizes the pressing need for continued research in quantum error mitigation (QEM) techniques and hardware advancements to unlock the full potential of our proposed QWM-QAOA method on portfolio optimization problems in real-world applications.

A. NOISE AND HARDWARE CONSTRAINTS
The most significant limitation encountered in this study was the impact of overall gate errors when using real quantum devices [47]. While the QAOA algorithm performed well in simulators and fake backend environments, real hardware results showed a substantial drop in approximation ratios (below 50%). This limitation reflects the current state of NISQ devices, which are not yet fully capable of handling complex optimization problems without significant fidelity loss.

B. SCALABILITY
While our approach worked well for smaller portfolios, scaling QWM-QAOA to larger datasets faces challenges due to the increasing number of required qubits and the limitations of current quantum hardware, such as limited qubit availability and short coherence times. These constraints hinder its applicability to larger portfolio optimization problems.

Advancements in quantum hardware and quantum error mitigation (QEM) techniques along with hybrid approaches are needed to extend QWM-QAOA to more complex real-world scenarios.

C. LIMITED DATASET

The experimental evaluations were conducted on a small selection of stocks, ranging from two to eight, providing a controlled setting to demonstrate the algorithm's effectiveness. However, to determine its practical value, future evaluations must incorporate historical data, transaction costs, and market dynamics. Expanding the tests to larger and more diverse datasets will be crucial for assessing the broader applicability of the QWM-QAOA approach in more complex, real-world financial scenarios.

VII. CONCLUSION

In this research, we proposed a novel approach to financial portfolio optimization by leveraging the capabilities of quantum computing, specifically through the Quantum Walk Mixer integrated with a Quantum Approximate Optimization Algorithm (QWM-QAOA). We reformulated the traditional portfolio optimization problem into a knapsack problem, enabling us to address the NP-hard nature of this financial decision-making challenge. Our experimental evaluations, conducted across simulated, fake, and real quantum devices, demonstrated the effectiveness of our approach in yielding high approximation ratios, particularly in noise-free environments. The results of our study reveal that the QWM-QAOA approach is capable of producing efficient portfolio selections, with approximation ratios ranging from 100% to 95% in noise-free conditions and from 98% to 70% on noisy fake devices. However, when tested on real quantum hardware, the results showed a decline in performance due to noise and gate errors, reducing the approximation ratios to approximately 50%. These findings highlight the current limitations of quantum hardware, particularly in dealing with noisy intermediate-scale quantum (NISQ) devices. While our proposed method showed promise, especially in noise-free environments, future work is needed to address the noise in real quantum devices. Error mitigation techniques with further exploration of quantum algorithms for optimization will be crucial in enhancing the robustness and scalability of this approach in real-world applications. Our study contributes to the growing field of quantum finance, offering insights into how quantum algorithms can be applied to complex financial problems such as portfolio optimization.

CODE AVAILABILITY

The code that supports the findings of this study is openly available in the Github repository, QWM-QAOA.

REFERENCES

[1] L. S. Speidell, D. H. Miller, and J. R. Ullman, "Portfolio optimization: A primer," *Financial Analysts J.*, vol. 45, no. 1, pp. 22–30, Jan. 1989.

[2] F. Vaezi, S. J. Sadjadi, and A. Makui, "A portfolio selection model based on the knapsack problem under uncertainty," *PLoS ONE*, vol. 14, no. 5, May 2019, Art. no. e0213652.

[3] Z. X. Loke, S. L. Goh, G. Kendall, S. Abdullah, and N. R. Sabar, "Portfolio optimization problem: A taxonomic review of solution methodologies," *IEEE Access*, vol. 11, pp. 33100–33120, 2023.

[4] S. Martello and P. Toth, "Algorithms for knapsack problems," *North-Holland Math. Stud.*, vol. 132, pp. 213–257, Jan. 1987.

[5] R. Mansini, W. Ogryczak, and M. G. Speranza, *Linear and Mixed Integer Programming for Portfolio Optimization*. Cham, Switzerland: Springer, 2015, vol. 21.

[6] P. C. Chu and J. E. Beasley, "A genetic algorithm for the multidimensional knapsack problem," *J. Heuristics*, vol. 4, no. 1, pp. 63–86, Jun. 1998.

[7] W. Honggang, M. Liang, Z. Huizhen, and L. Gaoya, "Quantum-inspired ant algorithm for knapsack problems," *J. Syst. Eng. Electron.*, vol. 20, no. 5, pp. 1012–1016, Oct. 2009.

[8] A. Gunjan and S. Bhattacharyya, "Quantum-inspired meta-heuristic approaches for a constrained portfolio optimization problem," *Evol. Intell.*, vol. 17, no. 4, pp. 3061–3100, Aug. 2024.

[9] D. Herman, C. Googin, X. Liu, Y. Sun, A. Galda, I. Safro, M. Pistoia, and Y. Alexeev, "Quantum computing for finance," *Nat. Rev. Phys.*, vol. 5, no. 8, pp. 450–465, 2023.

[10] H. Cao, S. Garg, S. Mumtaz, M. Alrashoud, L. Yang, and G. Kaddoum, "Softwarized resource allocation in digital twins-empowered networks for future quantum-enabled consumer applications," *IEEE Trans. Consum. Electron.*, vol. 70, no. 1, pp. 800–810, Feb. 2024.

[11] A. Saxena, J. Mancilla, I. Montalban, and C. Pere, *Financial Modeling Using Quantum Computing: Design and Manage Quantum Machine Learning Solutions for Financial Analysis and Decision Making*. Birmingham, U.K.: Packt Publishing Ltd, 2023.

[12] E. Farhi, J. Goldstone, and S. Gutmann, "A quantum approximate optimization algorithm," 2014, *arXiv:1411.4028*.

[13] W. v. Dam, K. Eldefrawy, N. Genise, and N. Parham, "Quantum optimization heuristics with an application to knapsack problems," in *Proc. IEEE Int. Conf. Quantum Comput. Eng. (QCE)*, Oct. 2021, pp. 160–170.

[14] K. Kea, C. Huot, and Y. Han, "Leveraging knapsack QAOA approach for optimal electric vehicle charging," *IEEE Access*, vol. 11, pp. 109964–109973, 2023.

[15] H. M. Markowitz, "Markowitz revisited," *Financial Analysts J.*, vol. 32, no. 5, pp. 47–52, Sep. 1976.

[16] K. P. Anagnostopoulos and G. Mamanis, "A portfolio optimization model with three objectives and discrete variables," *Comput. Oper. Res.*, vol. 37, no. 7, pp. 1285–1297, Jul. 2010.

[17] A. Gunjan and S. Bhattacharyya, "A brief review of portfolio optimization techniques," *Artif. Intell. Rev.*, vol. 56, no. 5, pp. 3847–3886, May 2023.

[18] A. A. Kamil, C. Y. Fei, and L. K. Kok, "Portfolio analysis based on Markowitz model," *J. Statist. Manage. Syst.*, vol. 9, no. 3, pp. 519–536, 2006.

[19] A. J. Kleywegt and J. D. Papastavrou, "The dynamic and stochastic knapsack problem," *Oper. Res.*, vol. 46, no. 1, pp. 17–35, Feb. 1998.

[20] D. Du and P. M. Pardalos, *Handbook of Combinatorial Optimization*, vol. 4. Cham, Switzerland: Springer, 1998.

[21] T. Pradhan, A. Israni, and M. Sharma, "Solving the 0–1 knapsack problem using genetic algorithm and rough set theory," in *Proc. IEEE Int. Conf. Adv. Commun., Control Comput. Technol.*, May 2014, pp. 1120–1125.

[22] L. E. Schäfer, T. Dietz, M. Barbati, J. R. Figueira, S. Greco, and S. Ruzika, "The binary knapsack problem with qualitative levels," *Eur. J. Oper. Res.*, vol. 289, no. 2, pp. 508–514, Mar. 2021.

[23] D. Delahaye, S. Chaimatanan, and M. Mongeau, "Simulated annealing: From basics to applications," in *Handbook Metaheuristics*. Boston, MA, USA: Springer, 2019, pp. 1–35.

[24] X. Lai, J.-K. Hao, and D. Yue, "Two-stage solution-based Tabu search for the multidemand multidimensional knapsack problem," *Eur. J. Oper. Res.*, vol. 274, no. 1, pp. 35–48, Apr. 2019.

[25] S. Abidin, "Greedy approach for optimizing 0-1 knapsack problem," *Commun. Appl. Electron.*, vol. 7, no. 6, pp. 1–3, Sep. 2017.

[26] P. D. de la Grand'rive and J.-F. Hullo, "Knapsack problem variants of QAOA for battery revenue optimisation," 2019, *arXiv:1908.02210*.

[27] C. Patvardhan, S. Bansal, and A. Srivastav, "Quantum-inspired evolutionary algorithm for difficult knapsack problems," *Memetic Comput.*, vol. 7, no. 2, pp. 135–155, Jun. 2015.

[28] B. Chen, H. Wu, H. Yuan, L. Wu, and X. Li, "Quasi-binary encoding based quantum alternating operator ansatz," 2023, *arXiv:2304.06915*.

[29] G. E. Crooks, "Performance of the quantum approximate optimization algorithm on the maximum cut problem," 2018, *arXiv:1811.08419*.

[30] J. Zawalska and K. Rycerz, "Solving the traveling salesman problem with a hybrid quantum-classical feedforward neural network," in *Parallel Processing and Applied Mathematics*, R. Wyrzykowski, J. Dongarra, E. Deelman, and K. Karczewski, Eds. Cham, Switzerland: Springer, 2023, pp. 199–208.

[31] A. Borle, V. Elfving, and S. J. Lomonaco, "Quantum approximate optimization for hard problems in linear algebra," *SciPost Phys. Core*, vol. 4, no. 4, p. 31, Nov. 2021.

[32] G. Ausiello, P. Crescenzi, G. Gambosi, V. Kann, A. Marchetti-Spaccamela, and M. Protasi, "Combinatorial optimization problems and their approximability properties," in *Complexity and Approximation*. Cham, Switzerland: Springer, 1999.

[33] L. Zhou, S.-T. Wang, S. Choi, H. Pichler, and M. D. Lukin, "Quantum approximate optimization algorithm: Performance, mechanism, and implementation on near-term devices," *Phys. Rev. X*, vol. 10, no. 2, Jun. 2020, Art. no. 021067.

[34] A. Awasthi, F. Bär, J. Doetsch, H. Ehm, M. Erdmann, M. Hess, J. Klepsch, P. A. Limacher, A. Luckow, C. Niedermeier, L. Palackal, R. Pfeiffer, P. Ross, H. Safi, J. Schönmeier-Kromer, O. von Sicard, Y. Wenger, K. Wintersperger, and S. Yarkoni, "Quantum computing techniques for multi-knapsack problems," 2023, *arXiv:2301.05750*.

[35] S. Brandhofer, D. Braun, V. Dehn, G. Hellstern, M. Hüls, Y. Ji, I. Polian, A. S. Bhatia, and T. Wellens, "Benchmarking the performance of portfolio optimization with QAOA," *Quantum Inf. Process.*, vol. 22, no. 1, p. 25, Dec. 2022.

[36] S. Marsh and J. B. Wang, "A quantum walk-assisted approximate algorithm for bounded NP optimisation problems," *Quantum Inf. Process.*, vol. 18, no. 3, pp. 1–18, Mar. 2019.

[37] J. Shunza, M. Akinyemi, and C. Yinka-Banjo, "Application of quantum computing in discrete portfolio optimization," *J. Manage. Sci. Eng.*, vol. 8, no. 4, pp. 453–464, Dec. 2023.

[38] Y. S. Weinstein, M. A. Pravia, E. M. Fortunato, S. Lloyd, and D. G. Cory, "Implementation of the quantum Fourier transform," *Phys. Rev. Lett.*, vol. 86, pp. 1889–1891, Feb. 2001.

[39] P. Chawla, S. Singh, A. Agarwal, S. Srinivasan, and C. M. Chandrashekar, "Multi-qubit quantum computing using discrete-time quantum walks on closed graphs," *Sci. Rep.*, vol. 13, no. 1, p. 12078, Jul. 2023.

[40] D. S. Steiger, T. Häner, and M. Troyer, "ProjectQ: An open source software framework for quantum computing," *Quantum*, vol. 2, p. 49, Jan. 2018.

[41] S. C. Endres, C. Sandrock, and W. W. Focke, "A simplicial homology algorithm for Lipschitz optimisation," *J. Global Optim.*, vol. 72, no. 2, pp. 181–217, Oct. 2018.

[42] R. Martin, "PyPortfolioOpt: Portfolio optimization in Python," *J. Open Source Softw.*, vol. 6, no. 61, p. 3066, May 2021.

[43] S. Martello, D. Pisinger, and P. Toth, "Dynamic programming and strong bounds for the 0-1 knapsack problem," *Manage. Sci.*, vol. 45, no. 3, pp. 414–424, Mar. 1999.

[44] S. Johnstun and J.-F. Van Huele, "Understanding and compensating for noise on IBM quantum computers," *Amer. J. Phys.*, vol. 89, no. 10, pp. 935–942, Oct. 2021.

[45] M. Alam, A. Ash-Saki, and S. Ghosh, "Analysis of quantum approximate optimization algorithm under realistic noise in superconducting qubits," 2019, *arXiv:1907.09631*.

[46] B. Gaur, T. Humble, and H. Thapliyal, "Noise-resilient and reduced depth approximate adders for NISQ quantum computing," in *Proc. Great Lakes Symp. (VLSI)*, Jun. 2023, pp. 427–431.

[47] S. K. Satpathy, V. Vibhu, B. K. Behera, S. Al-Kuwari, S. Mumtaz, and A. Farouk, "Analysis of quantum machine learning algorithms in noisy channels for classification tasks in the IoT extreme environment," *IEEE Internet Things J.*, vol. 11, no. 3, pp. 3840–3852, Feb. 2024.

CHANSREYNICH HUOT received the bachelor's degree in information economics from the Royal University of Law and Economics, Phnom Penh, Cambodia, in 2020. She is currently pursuing the M.S. degree with the Department of AI Convergence, Pukyong National University, Busan, South Korea. Her research interests include deep learning, embedded systems, and quantum computing.

KIMLEANG KEA received the bachelor's degree in computer science and engineering from the Royal University of Phnom Penh, Phnom Penh, Cambodia, in 2020, and the M.S. degree from the Department of AI Convergence, Pukyong National University, Busan, South Korea, in 2024. His research interests include deep learning, embedded systems, and quantum computing.

TAE-KYUNG KIM received the M.E. and Ph.D. degrees from the Department of Information Industry Engineering, Chungbuk National University, Cheongju, Republic of Korea, in 2005 and 2010, respectively. From 2013 to 2019, he was the Director of Korea Software HRD Center, Phnom Penh, Cambodia. He was an Assistant Professor with the AI Department of Computer and Information Technology, Incheon Jaeneung University, Republic of Korea, from 2020 to 2023. Since 2024, he has been an Assistant Professor with the Department of Management Information Systems, Cheongju, Republic of Korea. His research interests include big data, artificial intelligence, and software education.

YOUNGSUN HAN (Member, IEEE) received the B.S. and Ph.D. degrees in electrical engineering from Korea University, Seoul, South Korea, in 2003 and 2009, respectively. He was a Senior Engineer with the System LSI, Samsung Electronics, Suwon, South Korea, from 2009 to 2011. He was an Assistant/Associate Professor with the Department of Electronic Hahahah Engineering, Kyungil University, Gyeongsan-si, South Korea, from 2011 to 2019. He is currently a Professor with the Department of Computer Engineering, Pukyong National University, Busan, South Korea. His research interests include quantum computing, compiler construction, microarchitecture, and high-performance computing.

● ● ●