Quantum Algorithms for Portfolio Optimization

Beom Joon Baek*
Department of Physics, Columbia University, New York 10027, USA
(Dated: June 19, 2021)

Quadratic Unconstrained Binary Optimization (QUBO) is a NP-hard combinatorial optimization problem used in diverse fields. It is especially relevant in finance for portfolio optimization. With a couple of mathematical transformations, we can formulate a wide range of quadratic programs into a QUBO problem and to a corresponding Ising model. Using Variational Quantum Eigensolvers (VQE) and Quantum Approximate Optimization Algorithm (QAOA), we compare quantum algorithm's performance against classical optimization methods for different numbers of assets, budgets, and pricing models.

Keywords: Quantum Finance, Optimization, QUBO, VQE, QAOA

QUBO – Quadratic Unconstrained Binary Optimization problem, commonly called QUBO, is a combinatorial optimization problem that has a wide range of uses, especially in finances. It is also known as unconstrained binary quadratic programming (UBQP). Various NP hard problems, like Max Cut and Graph coloring, can be reformatted as a QUBO problem, and as such QUBO itself is an NP hard problem[1].

To parse through its name, quadratic means that terms involved in the equation to optimize are up to second order. Unconstrained means that there are no conditions on the variable to be optimized, although there can be workarounds by adding a penalty term to account for various forms of constraints. Binary means that the variable is a binary vector, consisting of either 1 or 0 (or -1 depending on the format).

Mathematically, QUBO is defined as:

$$\begin{gathered}\operatorname{Min} y = x^T Q\, x \\ \text{s.t. } x \in S\end{gathered}$$

where S is a binary discrete set $\{0,1\}^n$ and Q is an n-by-n square, symmetric matrix. y is the objective function that we wish to minimize. For cases when we are looking for maximum value, we can multiply the objective function with -1.

Classical Constraint	Equivalent Penalty
$x + y \leq 1$	$P(xy)$
$+y \geq 1$	$P(1 - x - y + xy)$
$x + y = 1$	$P(1 - x - y + 2xy)$
$x \leq y$	$P(x - xy)$
$x_1 + x_2 + x_3 \leq 1$	$P(x_1x_2 + x_1x_3 + x_2x_3)$
$x = y$	$P(x + y - 2xy)$

TABLE I. Table of Known Constraints and Corresponding Penalties in QUBO

The power of QUBO lies in its generalizability. It is easy to create additional variables to account for various constraints and make them unconstrained. Through the following sets of transformations, a large number of quadratic optimization problems could be formulated as QUBO.

First, while the problem is unconstrained, we can actually introduce effective constraints by applying a penalty term. By applying the penalty term that is roughly on the same order of magnitude as the term to be optimized, we can induce specific behaviors. For some constraints, there are commonly used penalties, shown in Table I.

P is a parameter we use to scale the penalty on the same magnitude as the term to optimize. For example, the classical constraint of $x + y \leq 1$ means that either or neither variable to be selected. The truth value of xy is the same as the classical constraint if both x and y are binary variables. Thus, the term could be modified to include the constraint and become an unconstrained problem:

$$\operatorname{Min} y = f(x) + P\, xy$$

The selection of the scalar penalty term P is more of art than exact science. Having too much of P overwhelms the original objective function while having too little makes the constraint irrelevant. Existing literature recommends using 75% to 150% of the original objective function value.

We can generalize the above constraints to any case of linear constraints. If given a linear constraint $Ax = b$, then we can add a quadratic penalty $P(Ax-b)^T(Ax-b)$ to the objective function.

$$\begin{aligned} y &= x^T Cx + P\,(Ax-b)^T(Ax-b) \\ &= x^T Cx + x^T Dx + c \\ &= x^T Qx + c \end{aligned}$$

where D and c are the corresponding matrix and vector through the matrix multiplication of the quadratic penalty term. In purposes of minimization, we can ignore the constant term c.

Secondly, we can expand QUBO to account for linear inequalities constraints by introducing slack variables. Suppose that our given objective function and its corresponding constraint are as follows:

$$\begin{gathered}\operatorname{Min} y = 2x_1 + x_2 + x_3 \\ \text{s.t. } x_1 + x_2 + x_3 \leq 5\end{gathered}$$

We introduce a slack variable s in the above constraint such that:

$$\begin{gathered}\text{Min}\, y = 2x_1 + x_2 + x_3 \\ \text{s.t. } x_1 + x_2 + x_3 + s = 5\end{gathered}$$

We can calculate the upper bounds of the slack variable by figuring out the left hand side's minimum. When x_1, x_2, and x_3 are 0, then the slack variable must be at most 5 to satisfy the constraint. Thus, our slack variable would be in the bounds $0 \leq s \leq 5$. We can further deconstruct this integer variable s into a binary variable through binary expansion where:

$$\begin{gathered}\min y = 2x_1 + x_2 + x_3 \\ \text{s.t. } x_1 + x_2 + x_3 + 1s_0 + 2s_1 + 2s_2 = 5\end{gathered}$$

With these sets of transformations, we can formulate any quadratic objective function with linear constraint into a QUBO form, thus demonstrating the problem's adaptability.

Portfolio Optimization – One application of QUBO in finance is the mean-variance portfolio optimization problem. Mean-variance portfolio optimization problem is a type of optimization problem where given the n universe of possible investments, you choose a number of investments such that you will maximize the following function with corresponding risk appetite, returns, and budget:

$$\begin{gathered}\min_{x \in \{0,1\}^n} qx^T \Sigma x - \mu^T x \\ \text{subject to: } P^T x \leq B\end{gathered}$$

where we use the following notation:

- $x \in \{0,1\}^n$ denotes the vector of binary decision variables, which indicate which assets to pick ($x[i] = 1$) and which not to pick ($x[i] = 0$)
- $\mu \in \mathbb{R}^n$ defines the expected returns for the assets
- $\Sigma \in \mathbb{R}^{n \times n}$ specifies the covariances between the assets
- $q > 0$ controls the risk appetite of the decision maker
- B denotes the budget, i.e. the available money to spend on assets
- $P \in \mathbb{R}^n$ represents the price associated with each assets

This formulation of the mean-variance portfolio optimization problem is the an extended version of the recipe shown in the paper [2]. In that paper, all assets had same price and we had to buy exactly B number of assets. Instead, this formulation has different prices for each asset, and there is no set amount of assets needed to buy.

Ising Model and Quantum Algorithms – QUBO has a set of classical algorithms, such as genetic algorithms methods, simulated annealing, and neural network solutions [3]. Yet in order to formulate QUBO as a quantum computing problem, the objective function must be recast as an Ising model with a corresponding Hamiltonian. Having a Hamiltonian makes it possible to use quantum computing methods like Variational Quantum Eigensolver or Quantum Approximate Optimization Algorithm.

Fortunately, the way to convert QUBO to an Ising model is fairly straightforward. If given a QUBO problem:

$$f(x_1, x_2, \ldots, x_n) = \sum_i^N h_i x_i + \sum_{i<j} J_{ij} x_i x_j$$

we can express this as a corresponding Hamiltonian:

$$H(x_1, x_2, \ldots, x_n) = -\sum_i^N h_i \sigma_i^z - \sum_{i<j} J_{ij} \sigma_i^z \sigma_j^z,$$

where σ_i is $\left(\frac{I - Z_i}{2}\right)$. All linear coefficients are expressed in h vector and quadratic coefficients are expressed in J matrix. After formulating QUBO problem as an Ising model with a corresponding Hamiltonian, then we could use either VQE or QAOA to optimize the objective function.

Methodology – Qiskit has a native Optimization module. The module has all the functions needed to define a quadratic equation, add linear constraints, convert the quadratic problem into QUBO, and formulate QUBO as an Ising model. Additionally, it has Qiskit Aqua with various quantum algorithms such as VQE and QAOA and classical optimizers implemented for us.

In order to simulate possible stock prices, we randomly picked various stocks from Nasdaq Exchange and got their price from March 2020 to April 2020. We then calculated the corresponding mean and the co-variance matrix of the stock prices.

With added flexibility in formulating the Portfolio Optimization problem, we wanted to explore the problem space. We picked two input parameters to calibrate: number of stocks(N_s) and the budget (B). We iterated four, eight, ten, and twelve stocks while we explored the budget range from plus two to minus two of the average price of stocks.

Due to the binomial expansion step when formulating a quadratic problem into QUBO, a budget that is extensively large would have created more slack variables, wasting precious qubits. Thus, we had to randomly assign prices of each stocks from 1 to 6 in order to keep the budget down. We calculated the budget as starting from the mean of all prices.

We tested three quantum algorithms for these different input parameters: VQE, QAOA, and CVaR (an extension of VQE) elaborated in here [4]. We expect to see different behaviors for three algorithms.

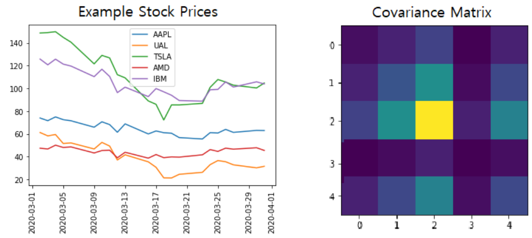

FIG. 1. Stock Prices Used in the Data Analysis and Covariance Matrix

Results – For the first test, we changed the budget for the problem. The mean price was 15, so we looked at 5 budget prices around the mean to see what different behaviors would three quantum algorithms show. **??** shows that Exact solution is identical with QAOA solution, perhaps indicating that QAOA is most resilient to changes in the problem statement. VQE-based methods seem to experience difficult in finding the global minimum.

Likewise, when we adjust the Risk Factor, we see the similar results in that QAOA and Exact solutions are closely aligned. Considering the fact that Risk Factor is just a linear change, it makes sense to conclude that VQE-based methods are especially hard to converge to global minimum.

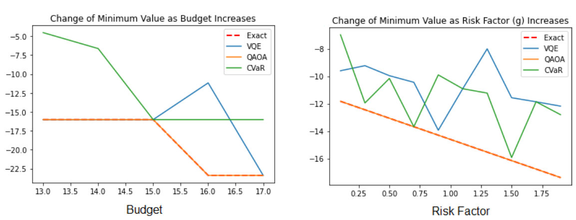

FIG. 2. Minimum Value Across Budget and Risk Factor

For the final case, we looked at what might happen if we were to increase the asset number. Keep in mind that increasing the asset number, in turn, increases the number of variables as both slack variables and original variables increase. For example, if there are eight assets, there are extra three variables because of slack variables. Thus, our quantum algorithms have to use eleven qubits, which took long time to simulate, and almost impossible to actually perform without IBM credits.

Conclusion – There are more ways to experiment and build on top of this research. First, I have only used two types of quantum algorithms - VQE and QAOA (CVaR is a type of VQE). Quantum optimization is a field of increasing research, with new quantum algorithms being developed. In order to continue this project, there needs to be a more diverse set of algorithms being tested. Additionally, it would be interesting to explore different kinds of financial data, not only stocks, but also options and other financial assets. Thirdly, due to the limitation of computing power, I had to scale down the number of assets that I wished to explore. Having slack variables made having higher number of assets a more computationally expensive operation. We wish to build upon this work to experiment with higher number of qubits. Fourthly, I could not run experiments on an actual quantum computing. It would have been better to try some experiments on a real quantum computer. Finally, there has been some doubt whether quantum computing could provide quantum advantage over classical computers in problems of optimization, mainly as there has not been any mathematical proof that quantum computers are faster than classical computers in theory [2]. Finding any theoretical proof of quantum advantage in optimization problems, especially in QUBO, would be a great change in the field.

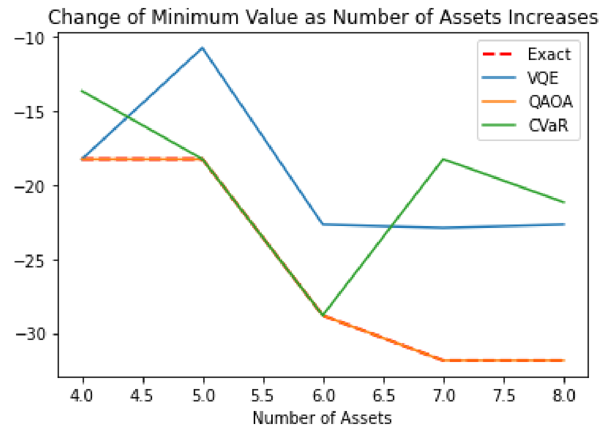

FIG. 3. Minimum Value With Increase In Assets

* bb2763@columbia.edu

[1] F. Glover, G. Kochenberger, and Y. Du, , 53.
[2] D. J. Egger, C. Gambella, J. Marecek, S. McFaddin, M. Mevissen, R. Raymond, A. Simonetto, S. Woerner, and E. Yndurain, IEEE Transactions on Quantum Engineering **1**, 1 (2020), conference Name: IEEE Transactions on Quantum Engineering.
[3] G. Kochenberger, J.-K. Hao, F. Glover, M. Lewis, Z. Lü, H. Wang, and Y. Wang, Journal of Combinatorial Optimization **28**, 58 (2014).
[4] P. K. Barkoutsos, G. Nannicini, A. Robert, I. Tavernelli, and S. Woerner, Quantum **4**, 256 (2020), arXiv: 1907.04769.
[5] F. Glover, G. Kochenberger, and Y. Du, 4OR **17**, 335 (2019).
[6] Qiskit, en"A Walkthrough of Qiskit's New Optimization

Module," (2021).
[7] S. Karimi and P. Ronagh, Quantum Information Processing **18**, 94 (2019), arXiv: 1706.01945.
[8] G. Tavares, , 460.