Grad Project - José Ossorio

Portfolio Optimization Using Quantum Computing

I. Introduction

Currently available quantum computers are small and not capable of solving many industry problems. Nevertheless, quantum computing (QC) is a promising technology that is expected to bring exponential speedups to certain areas of science and engineering. As a software engineer, I am interested in the application of quantum computers to current and future industry problems, and how to integrate this technology with classical computing in order to produce effective software applications. Despite the limited processing power of current quantum processors, there are still areas of industry that can benefit from QC right now.

In the past few years, QC applications have transitioned from being a hope for the future to a reality in some promising fields, such as quantum chemistry, simulations, and operations research. Among the applications of QC, portfolio optimization promises to deliver great value to the industry. A portfolio is a collection of assets that interact with each other and have potential returns. According to Orús et al. in their paper *Quantum computing for finance: Overview and prospects*, an optimal portfolio will pick the best subset of assets to maximize the return according to a desired risk. This also indicates that for a given return there is a portfolio that minimizes the risk.

In order to model this problem, we need to take into consideration each possible asset, as well as the interactions that occur among them. With the addition of assets to the set of possible options, the difficulty of finding an optimal solution quickly increases. As mentioned in chapter four of the book *Portfolio optimization: Applications in quantum computing*, "The optimization techniques traditionally used to solve this problem are so-called classical optimization techniques that rely on mathematically well-defined gradient based or descent/directional indications that must be well defined and constrained". Such heuristics provide a way to solve combinatorial optimization problems using classical computers, but may lead to suboptimal results and poor runtime.

The portfolio optimization problem can be modeled as a quadratic unconstrained binary optimization (QUBO) problem. According to Orús et al., demonstrated speedups in solving optimization problems using QC can be described as follows: "when the number of classical bits needed to specify the input data is increased, the number of operations needed to run the quantum algorithm increases slower than the best known classical alternative".

Moreover, QC startup D-Wave provides powerful quantum computers designed specifically to solve optimization problems. According to their website, "to solve a problem on quantum samplers, you formulate the problem as an objective function, usually in Ising or QUBO format. Low energy states of the objective function represent good solutions to the problem." This means that if we can formulate our problem accordingly (using a QUBO or an Ising model formulation) we can find optimal solutions using a quantum computer. The D-Wave computers make use of a QC paradigm called quantum annealing, but this is not the only algorithm used to solve optimization problems.

Hybrid quantum-classical approaches to solving optimization problems include the Variational Quantum Eigensolver (VQE) and the Quantum Approximate Optimization Algorithm (QAOA). These two algorithms are designed for universal quantum processors and are built using parameterized quantum circuits. These circuits have a set of variational parameters which need to be optimized to minimize the problem's cost function. The optimization process consists of multiple iterations of the algorithm. The circuit runs on quantum processors and according to the results after each run, classical techniques are used to optimize the parameters for the next iteration. Hybrid approaches such as this one have proven to be very effective at taking advantage of the limited capabilities of quantum computers and utilizing them to solve the part of the problem that benefits from quantum processing while solving other parts using classical techniques, such as gradient descent, that work well on classical hardware.

I find this application area to be very exciting, since it allows us to make use of current quantum computers to generate cost-effective solutions to hard problems. It is a great opportunity to show that quantum computers are useful in real world problems and that if we pair them together with classical computers, we can build efficient and effective software applications that could benefit society. In this project, I will document my findings after researching different approaches to solving the portfolio optimization problem. For this first part, I will briefly describe some of the different approaches I came across while doing a literature search.

A. Quantum annealing

In the review paper by Orús et al., it is stated that the most prominent approach to solving optimization problems involving quantum computers is using an adiabatic algorithm. In adiabatic QC, one must first map the optimization problem to a function known as a Hamiltonian. The Hamiltonian encodes the cost function to be minimized, which is equivalent to finding the ground state of the Hamiltonian function. The state of the energy function is slowly evolved (in the case of D-Wave, this is done by applying an external magnetic field to the superconducting qubits) until it reaches the ground state of our desired Hamiltonian. At this point, a measurement of the system has a high probability of returning the correct answer to our problem. This process is known as quantum annealing.

B. Reverse quantum annealing

In the paper titled *Reverse quantum annealing approach to portfolio optimization problems*, Venturelli et al. attempted a hybrid quantum-classical method that implements a reverse annealing protocol. In this study, the researchers used the D-Wave Quantum Annealer 2000Q to sample various portfolio optimization problems and compare the reverse annealing protocol to the traditional forward quantum annealing algorithm. They found that their approach is 100 times faster on average.

According to the authors of the same study, the theory of reverse annealing is still being investigated. For this particular experiment described in the paper, a classical computer will use a 'greedy search' algorithm to seed the quantum annealer with a possible (but not necessarily optimal) solution, i.e., a local minimum of the objective function. Then, the reverse annealing protocol starts, pausing at one point to then continue with forward annealing and measurement. It is highlighted in the study that "the quality of the initial state is likely to influence dramatically the reverse annealing process".

C. Using quantum walks to optimize portfolios

In a recent study by Slate et al., the researchers evaluated the performance of a newly developed Quantum Walk Optimization Algorithm (QWOA) and compared it to approches that use the QAOA algorithm. Simillar to other hybrid approaches, the problem is solved iteratively, adjusting the parameters each time according to a cost function through the use of classical algorithms. The authors found that using the QWOA the search space was reduced by a significant factor, allowing the algorithm to arrive at an optimal solution using fewer iterations.

D. A real-world application

Quantum computers have yet to deliver much of what the theory has promised us. Implementing a quantum computer capable of solving hard problems is no trivial task, but as it has been shown in the different studies mentioned in this document, obtaining useful results with current quantum technology is possible. In fact, the Spanish company Multiverse has been working with the bank BBVA to create optimized portfolios using quantum computers. The company deviced an algorithmic approach using the D-Wave hybrid solver service to generate portfolios based on a set of constraints and real financial data. Ultimately, this shows that current quantum computers can already be used (together with classical computers) to achieve valuable results.

II. Solving the Portfolio Optimization Problem with a Hybrid Approach

As we learned, a hybrid quantum-classical approach is well suited for this specific problem. In this case, I will describe how to solve the portfolio optimization problem using the quantum annealing algorithm in combination with classical techniques. I will describe the problem specification for the D-Wave hybrid solver. The hybrid solver is a framework that allows the user to create custom hybrid workflows to solve optimization problems. It's called hybrid since it uses classical and quantum resources to find a solution in the most efficient manner.

A. Problem Specification

In order to solve the problem, we must first build a model to represent our portfolio's variables, constraints and variable interactions. According to the D-Wave documentation, the hybrid solver takes as input a quadratic model. In fact, D-Wave has three different quadratic model hybrid solvers readily available: the Binary Quadratic Model solver (BQM), the Discrete Quadratic Model solver (DQM) and the Constrained Quadratic Model solver (CQM).

The BQM was the first solver made available by D-Wave and it can be used to solve optimization problems that deal with binary variables. For example, if we want to find the best combination of variables to maximize (or minimize) a specific value, this solver will tell us which subset of variables we should choose to achieve the best result (i.e., which variables should be present and which should not). However, it does not handle cases in which the variables can take discrete values (i.e., 1, 2, 3, ...), nor does it allow for constraints (e.g., a limiting budget for a portfolio).

Then came the DQM, a more capable model that expands the functionality of the hybrid solver to include discrete variables. This solver could tell you how many shares to buy from a given stock if you want to maximize the return. However, it also does not allow for constraints.

Finally, the CQM encompases both previous functionalities and adds the possibility to have constraints. Thus, for the specific problem of portfolio optimization I will describe the specification of a CQM. In a CQM, as in the other two models, we have a representation of the problem that uses linear and quadratic variables. The linear terms, in this case, refer to the number of shares for each of the stocks, their expected monthly return per dollar spent based on previous data, and the purchase value. This expression represents the total expected return. Mathematically, it looks like this:

$$Obj = \sum_{i=1}^{n} r_i p_i x_i$$

Where we have n different stocks to choose from, and r_i refers to the expected monthly return per dollar spent, p_i to the price, and x_i is the discrete variable that will tell us how many shares we should buy per each stock. This variable is what we ultimately want to find to solve our problem.

The previous expression is our objective function **Obj**, as it is called in the CQM. In the quantum computer, these linear terms will be encoded to each of the qubits.

Moreover, the quadratic terms refer to the interactions between the different stocks. In this case, we care the directional relationship of the returns of two stocks. This is also known as the covariance, and a way to think about it is to consider the scenario of having shares of two different stocks in a portfolio. If there are two different stocks that belong to competing companies they will move inversely, i.e., if one's return increases the other will decrease. This means that they have negative covariance. In the opposite case when the two assets move together we say that they have positive covariance. Ultimately, the covariance metric will be an indicator of the risk of our portfolio, which is one of the constraints that we will set. A lower covariance means that most of the stocks in our portfolio move inversely, leading to less risk. These quadratic terms will be encoded in the quantum computer's couplers, which control the interactions between qubits. Another way to think about this problem formulation is to imagine a weighted graph with vertices (linear terms) and edges connecting them (quadratic terms).

Mathematically, the quadratic terms are given by:

$$\sum_{i=1}^{n}\sum_{j=1}^{n}\sigma_{ij}p_i x_i p_j x_j$$

Where σ_{ij} refers to the covariance between stocks i and j, and the other terms are the same as in the total expected return expression.

Lastly, we must consider the other constraint that we have for our problem. In this case, I will limit the available budget for the portfolio, meaning that our total cost C must be less than or equal to our budget B

$$C \leq B = \sum_{i=1}^{n} p_i x_i \leq B$$

Our goal is to maximize profit while not exceeding a certain risk and budget. Thus, by changing the values for the x_i we can maximize our objective function. One important point about this algorithm is that it will actually minimize the function, meaning that if we want to maximize we will have to minimize the negative profit.

Moreover, the risk or budget constraints will be added in the form of equalities or inequalities to the expression.

All that is left to do now is to pick a dataset and implement the mathematical expressions in Python using the D-Wave hybrid solver module and send them to the D-Wave computer to get the results back.

B. Algorithms Used

As mentioned in this section, the D-Wave hybrid solver will internally perform quantum and classical techniques to find a solution to the problem. The hybrid solver system receives the problem model as input and uses something they call the heuristic module, which are threads running in parallel on state of the art AWS CPUs and GPUs. These threads explore the solution landscape and communicate with a QPU through a module called the quantum module. The QPU answers queries from the quantum module and tells the heuristic module where to move in the energy landscape.

It's interesting that they do it in this way, since it's the opposite to what we usually would do in other optimization algorithms such as VQE and QAOA, where the classical part helps the quantum.

III. Implementation

For the implementation part I decided to use QPLEX, which is a library that I'm currently working on with my colleague Juan from the Rigi Research Lab. QPLEX is an extension to an existing library called DOcplex, which allows for the specification of a model for an optimization problem. The goal of QPLEX is to extend the execution capabilities to allow for the use of quantum resources.

The following code snippets illustrate the implementation of the portfolio optimization problem using QPLEX and running on D-Wave hardware. For this problem, I used a dataset from the S&P 500. The example uses around 200 different stock variables and sets 2 constrains explained in previous sections. The budget is set to $5000 and the risk to $2500

In [ ]:
```
from qplex.library.qmodel import QModel

portfolio_model = QModel('portfolio')

stocks = ['AAL', 'AAPL', 'AAP', 'ABBV', 'ABC', 'ABT', ...]
x = portfolio_model.integer_var_list(n, name=lambda index: stocks[index])

cost = sum(price[s] * x[index] for index, s in enumerate(stocks))

risk = 0
for i, s1 in enumerate(stocks):
    for j, s2 in enumerate(stocks):
        coefficient = covariance_matrix[s1][s2] * price[s1] * price[s2]
        risk = risk + coefficient * x[i] * x[j]

returns = 0
for index, stock in enumerate(stocks):
    returns = returns + avg_monthly_returns[stock] * x[index] * price[stock]

portfolio_model.add_constraint(cost <= budget)
portfolio_model.add_constraint(risk <= max_risk)
portfolio_model.set_objective('max', returns)

portfolio_model.solve('quantum', backend='d-wave')
```

After running the code, we get the following results back:

Creating QModel

Adding variables... done

Computing cost constraint... done

Computing risk constraint... done

Computing return... done

Adding constraints... done

Solving... done

Solution:

objective: 48.219

AEP=1

AET=1

AKAM=1

ATVI=2

BBY=2

BSX=1

CBOE=3

CI=1

CME=2

DLR=1

DPS=1

DRI=1

DTE=1

EXPE=1

EXR=2

FB=1

FTI=1

This would be the asset distribution to maximize the return and limit the risk.

IV. Findings and Challenges

- **Finding a good dataset**. The predictions achieved by this model are as good as the data that we use. Finding good and accurate data is crucial. In this project, the example was somewhat simplified, but in a real world scenario we would want to add more constraints or make multiple investments using different strategies to have a better prediction
- **The model doesn't allow us to buy part of a share**. In this case, we are modeling this problem as a quadratic integer constrained optimization problem, which means we can't buy half a share, also known as a fractional share. This limits the investment options, since we cannot invest a fixed amount of money, but we are instead dependent on the price per share and the number of shares to determine the money spent.
- **How can we determine if we should use a hybrid or a purely-classical solution?** The first thing that comes to mind is benchmarking. By benchmarking multiple quantum and classical algorithms we should be able to determine when it's best to use a hybrid

approach. In this case by "best" I mean it would be faster, cheaper, or have better scalability. I must say, however, that I tested the example described in previous paragraphs using the classical high performance solver CPLEX on my machine and it yielded a higher value for the objective function (in this case this is good since we want to maximize the return). I ran multiple tests with different numbers of variables and still got better results using the classical algorithm. I suspect this could be because my example is too simplistic, and therefore a classical algorithm is more than enough to solve the problem. Some future work would be to increase the complexity of the problem and to run more benchmarks to determine if there is indeed an advantage to using a hybrid algorithm (which I suspect there is).

- **The preprocessing can take a long time.** As I said, I got the stock data from a dataset containing information about the stocks in the S&P 500 stock index. One of the constraints that I put in place for my example was the risk constraint. The risk of the portfolio could not go above a certain maximum risk, which would be defined when running the program. Computing the risk, however, is no easy task. At least not for 503 different stocks, which is the size of the dataset. For this number of variables we are talking about a 503 x 503 matrix (although it is a diagonal matrix). Computing this constraint expression took from 40 to 60 seconds on my machine, making it very annoying everytim I had to run the model. I can imagine that for other constraints that involve quadratic interactions the case would be the same. To solve this, one suggestion would be to simply use a high performance computer (HPC) for any preprocessing needed. This way we would achieve much better performance results.

- **It's good to know we can actually implement cool things using quantum in the current era.** There is a lot of talk about quantum right now, but it's not very common to see problems we can actually solve right now using this new technology. That's what I find exciting about D-Wave and their hybrid solver service. They managed to put out a product that uses vanguard technology like quantum and can actually be used in the industry. I'm really excited to see what we will do once we get more powerful gate-based quantum computers from IBM, Google, and others. This is an exciting time to be in tech.

References

- Marzec, M. (2016). Portfolio optimization: Applications in quantum computing. Handbook of High-Frequency Trading and Modeling in Finance, 73-106.
- Orús, R., Mugel, S., & Lizaso, E. (2019). Quantum computing for finance: Overview and prospects. Reviews in Physics, 4, 100028.
- Venturelli, D., & Kondratyev, A. (2019). Reverse quantum annealing approach to portfolio optimization problems. Quantum Machine Intelligence, 1(1-2), 17-30.
- Hegade, N. N., Chandarana, P., Paul, K., Chen, X., Albarrán-Arriagada, F., & Solano, E. (2022). Portfolio optimization with digitized counterdiabatic quantum algorithms. Physical Review Research, 4(4), 043204.

- Slate, N., Matwiejew, E., Marsh, S., & Wang, J. B. (2021). Quantum walk-based portfolio optimisation. Quantum, 5, 513.
- Getting started with D-wave solvers. D. (n.d.). Retrieved February 12, 2023, from https://docs.dwavesys.com/docs/latest/doc_getting_started.html
- Multiverse Computing Releases New Version of Singularity SDK for Portfolio Optimization with Quantum Computing. Retrieved February 12, 2023, from https://multiversecomputing.com/resources/multiverse-computing-releases-new-version-of-singularity-sdk-for-portfolio-optimization-with
- Rieffel, E. G., Venturelli, D., O'Gorman, B., Do, M. B., Prystay, E. M., & Smelyanskiy, V. N. (2015). A case study in programming a quantum annealer for hard operational planning problems. Quantum Information Processing, 14, 1-36.
- Mugel, S., Kuchkovsky, C., Sanchez, E., Fernandez-Lorenzo, S., Luis-Hita, J., Lizaso, E., & Orus, R. (2022). Dynamic portfolio optimization with real datasets using quantum processors and quantum-inspired tensor networks. Physical Review Research, 4(1), 013006.
- Hybrid Solvers — Ocean Documentation 6.3.0 documentation. (n.d.).Retrieved March 12, 2023, from https://docs.ocean.dwavesys.com/en/stable/overview/hybrid.html