Multi-Objective Optimization Using QAOA

Team 8

Bo-Yu Lien Jeongbin-Jo Eliška Krátká Hsiao-Nan Lu Shang-Ling Hung

Table of Contents

Multi-objective optimization using QAOA – IBM(Atsushi Matsuo, Vishal Bajpe)

Introduction:

Multi-objective optimization problems involve optimizing several conflicting objectives at once. Even when each individual objective is easy to solve on its own, tackling them simultaneously can be classically challenging. Thus, multi-objective optimization represents a compelling problem class to analyze with quantum computers. In this challenge, we solve a multi-objective vehicle routing problem (VRP) inspired by MAXCUT, using QAOA. Specifically, we consider a realistic scenario where a logistics provider must dispatch deliveries to city locations using two vehicles, balancing efficiency and service quality. Each customer's assignment to a vehicle can be modelled as a node partition, and edge weights can encode real-world costs such as travel distance and delivery satisfaction penalties.

Goal:

- **Fundamental:** Single objective optimization: Find the optimal solution for each objective function individually (e.g., minimize total distance or minimize delivery delay) using QAOA. You can select your own objective function criteria's for efficiency and service quality.
- **Advanced:** Find the Pareto front - the set of all Pareto-optimal vehicle assignments, where improving one objective necessarily worsens another and analyze the tradeoffs.
- **Bonus challenge:** Hardware implementation and Transpilation: Attempt to optimize your circuits and run your QAOA solution on real quantum hardware. Optimize your circuits for IBM Quantum Heron V2's connectivity, characteristics and layout with minimal circuit depth. Extrapolate your strategy and report feasibility of scaling up to bigger problem sizes (more customers, more objectives) to 100+ qubit regime and feasibly running using available hardware and error mitigation techniques.

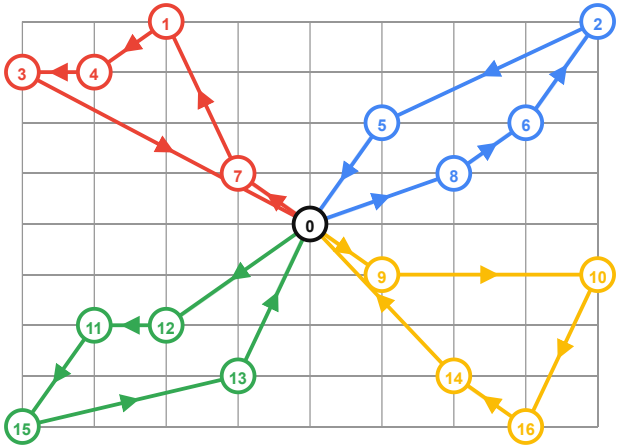
Reference:

1. Quantum approximate optimization algorithm, IBM Quantum Platform tutorial
2. Quantum Approximate Multi-Objective Optimization , arxiv 2503.22797
3. Variational Quantum Multi-Objective Optimization , arxiv 2312.14151
4. Qiskit Transpiler Module, IBM Quantum platform documentation
5. Qiskit Transpiler Service with AI transpiler passes, IBM Quantum Platform documentation

Background & Approach

Vehicle Routing Problem (VRP) finds the most efficient routes for vehicles delivering to multiple locations.

Single-objective VRPs are easy to solve.



Background & Approach

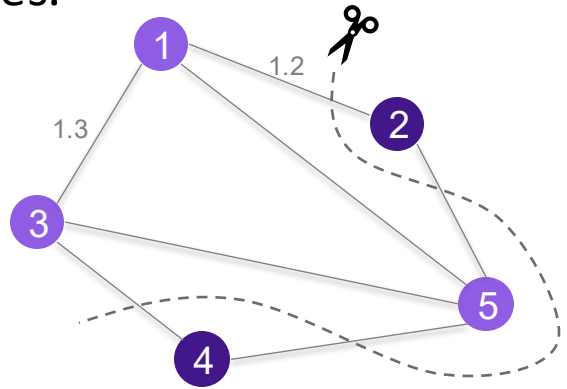
Multi-objective VRPs are **challenging** for classical computers due to increased complexity.

QAOA offers a scalable, flexible way to optimize **multiple objectives** in a **single optimization model**.

Background & Approach

We model a logistics provider dispatching **two vehicles** to city locations, **balancing efficiency and service quality**.

This constraint maps the VRP to a **Max-Cut** problem, where customer assignments form node partitions and edge weights encode real-world costs like distance and service penalties.



Method

arXiv:2503.22797v1 [quant-ph] 28 Mar 2025

Quantum Approximate Multi-Objective Optimization

Ayse Kotil,[1,2] Elijah Pelofske,[3] Stephanie Riedmüller,[2] Daniel J. Egger,[1]
Stephan Eidenbenz,[3] Thorsten Koch,[2,4] and Stefan Woerner[1,*]
[1] *IBM Quantum, IBM Research Europe - Zurich*
[2] *Zuse Institute Berlin*
[3] *Los Alamos National Laboratory*
[4] *Technische Universität Berlin*
(Dated: April 1, 2025)

The goal of multi-objective optimization is to understand optimal trade-offs between competing objective functions by finding the Pareto front, i.e., the set of all Pareto optimal solutions, where no objective can be improved without degrading another one. Multi-objective optimization can be challenging classically, even if the corresponding single-objective optimization problems are efficiently solvable. Thus, multi-objective optimization represents a compelling problem class to analyze with quantum computers. In this work, we use low-depth Quantum Approximate Optimization Algorithm to approximate the optimal Pareto front of certain multi-objective weighted maximum cut problems. We demonstrate its performance on an IBM Quantum computer, as well as with Matrix Product State numerical simulation, and show its potential to outperform classical approaches.

I. INTRODUCTION

Quantum computing is a new computational paradigm that has the potential to disrupt certain disciplines, with (combinatorial) optimization frequently being mentioned as one of them [1]. However, in many cases, classical approaches can find good solutions quickly, leaving little room for further improvements. Thus, it is important to identify the right problem classes and instances that are truly difficult classically with a potential for improvement through quantum computers.

The goal of multi-objective optimization (MOO) is to find optimal trade-offs between multiple objective functions by identifying all Pareto optimal solutions, i.e., those solutions where no single objective value can be improved without degrading another one [2]. MOO can be difficult even if corresponding single-objective problems can be solved efficiently [2, 3], particularly with an increasing number of objective functions [4]. In this context, sampling-based approximate quantum optimization algorithms can be beneficial as they can quickly produce a large variety of good solutions to approximate the Pareto front.

We demonstrate how the quantum approximate optimization algorithm (QAOA) can be efficiently applied to multi-objective combinatorial optimization by leveraging transfer of QAOA parameters across problems of increasing sizes. This eliminates the need to train QAOA parameters on a quantum computer and removes a computational bottleneck for the considered problems. Using an IBM Quantum computer, we present promising results demonstrating that our algorithm has the potential to outperform classical approaches for multi-objective weighted maximum cut (MO-MAXCUT). Additionally, we show how today's quantum computers can help forecast the performance of heuristics on future devices, serving as tools for algorithmic discovery.

The remainder of the paper is structured as follows. Sec. II introduces MOO. Sec. III discusses classical approaches to approximate the Pareto front. Then, Sec. IV defines the QAOA [5]–a core building block of our quantum algorithm–and the required theory. Afterwards, Sec. V discusses the related work on quantum algorithms for MOO and introduces our approach. Demonstrations of how the algorithm performs and how it compares to a classical benchmark are presented in Sec. VI. Sec. VII summarizes our work and discusses possible future research.

II. MULTI-OBJECTIVE OPTIMIZATION

The goal of MOO is to maximize a set of $m \in \mathbb{N}$ objective functions $(f_1, \ldots, f_m)$, with $f_i : X \to \mathbb{R}$, $i \in \{1, \ldots, m\}$, over a feasible domain $X \subset \mathbb{R}^n$, $n \in \mathbb{N}$ [2]:

$$\max_{x \in X} \quad (f_1(x), \ldots, f_m(x)). \tag{1}$$

Since it is usually not possible to optimize all f_i simultaneously, MOO refers to identifying the optimal trade-offs between possibly contradicting objectives.

Optimal trade-offs between objectives are defined via the concept of Pareto optimality. A solution $x \in X$ is said to dominate another solution $y \in X$ if $f_i(x) \geq f_i(y)$ for all $i \in \{1, \ldots, m\}$ and if there exists at least one $j \in \{1, \ldots, m\}$ with $f_j(x) > f_j(y)$. A solution x is called Pareto optimal if there does not exist any y that dominates x. In other words, x is Pareto optimal if we cannot increase any f_i without decreasing at least one f_j, $j \neq i$. The set of all Pareto optimal solutions is called the Pareto front or efficient frontier. Solving MOO exactly denotes computing the full Pareto front.

Suppose X is a convex set and all f_i are concave functions. Then, the Pareto front usually consists of infinitely

* wor@zurich.ibm.com

arXiv:2312.14151v3 [quant-ph] 12 Nov 2024

Variational Quantum Multi-Objective Optimization

Linus Ekstrøm,[1] Hao Wang,[2] and Sebastian Schmitt[1]
[1] *Honda Research Institute Europe GmbH, Carl-Legien-Str. 30, 63073 Offenbach, Germany*
[2] *LIACS/aQa, Leiden University, Niels Bohrweg 1, 2333 CA Leiden, Netherlands*
(Dated: February 5, 2024)

Solving combinatorial optimization problems on near-term quantum devices has gained a lot of attraction in recent years. Currently, most works have focused on single-objective problems, whereas many real-world applications need to consider multiple, mostly conflicting objectives, such as cost and quality. We present a variational quantum optimization algorithm to solve discrete multi-objective optimization problems on quantum computers. The proposed quantum multi-objective optimization (QMOO) algorithm incorporates all cost Hamiltonians representing the classical objective functions in the quantum circuit and produces a quantum state consisting of Pareto-optimal solutions in superposition. From this state we retrieve a set of solutions and utilize the widely applied hypervolume indicator to determine its quality as an approximation to the Pareto-front. The variational parameters of the QMOO circuit are tuned by maximizing the hypervolume indicator in a quantum-classical hybrid fashion. We show the effectiveness of the proposed algorithm on several benchmark problems with up to five objectives. We investigate the influence of the classical optimizer, the circuit depth and compare to results from classical optimization algorithms. We find that the algorithm is robust to shot noise and produces good results with as low as 128 measurement shots in each iteration. These promising result open the perspective to run the algorithm on near-term quantum hardware.

I. INTRODUCTION

Over the past few years, quantum computing experiments have indicated that the threshold of practically useful quantum computation is nearing [1–4]. Realistic application problems have increasingly moved into the focus of quantum algorithm research, in particular for the currently available noisy-intermediate scale quantum (NISQ) devices [5–10]. A promising class of approaches are the variational algorithms such as the quantum approximate optimization algorithm (QAOA) [11, 12]. However, the exact nature of the potential quantum advantage for real-world combinatorial optimization problems remains an open problem [12–14].

Current versions of variational quantum optimization approaches only deal with single-objective optimization problems. The final target for the optimized quantum circuit is to produce a quantum state where the probability of sampling the optimal solution of the original problem is as large as possible. However, many realistic application problems need to consider more than one objective, such as cost and quality, simultaneously [15, 16]. Since these objectives are mostly in a trade-off relationship where increasing the performance in one objective leads to a degradation in the other objectives, finding one single optimal solution is no longer a proper target setting. Instead, the goal is to find a set of Pareto-optimal solutions that represent the best possible trade-off solutions. Solving such multi-objective optimization problems directly with a variational quantum algorithm has not been possible until now.

While most investigations focus on qubit quantum computation, qubits can lead to significant encoding overhead for integer optimization problems [7, 17, 18]. In the NISQ-era of quantum computation, such overheads may be prohibitive to obtaining useful results. It has been suggested some of these limitations can be handled effectively by utilizing multi-level systems called qudits. There are indications that qudit systems may provide algorithmic benefits for circuit simplification [19–21], error correction [22, 23] or algorithm speed-up [24]. Qudit-based approaches have been used for several applications like max-cut, graph-coloring, constraints handling or machine learning problems [6, 25–32]. Generally, qudits represent a very promising route to efficient quantum information processing in enlarged Hilbert spaces. [21, 33–36]. Therefore, we formulate and evaluate the proposed quantum multi-objective optimization algorithm with qudit variables.

The structure of the paper is as follows. After a brief introduction to multi-objective optimization, we repeat the basics of qudit-based formulation of quantum algorithms. Then, we describe the proposed quantum multi-objective (QMOO) algorithm and introduce several benchmark functions for evaluating its performance. We present empirical results from algorithm simulations. Finally, we discuss the results, present our conclusions from this work, and give an outlook for future directions of investigations.

II. RELATED WORK

A recent proposal for solving multi-objective optimization problems with NISQ devices [37] utilizes a scalarizing approach. The objectives are combined into several single objectives by weighted sums. Each scalarized objective is then solved using a regular QAOA to obtain one Pareto-optimal solution. Combining the solutions of several different scalarizing functions allows the recon-

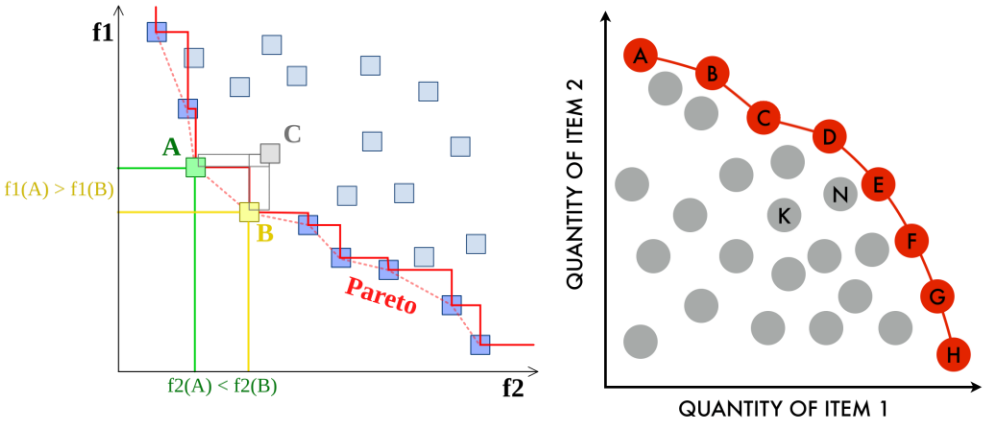
Weighted Sum Method (WSM)

Converts multiple objectives into a single objective by combining them with predefined weights.

$$f_c(x) = \sum_{i=1}^{m} c_i f_i(x) \longrightarrow H_c(x) = \sum_{i=1}^{m} c_i H_i(x)$$

Pareto Front

In multi-objective optimization, the Pareto front (also called Pareto frontier or Pareto curve) is the set of all Pareto efficient solutions.

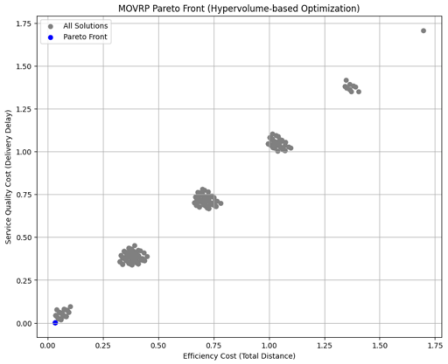


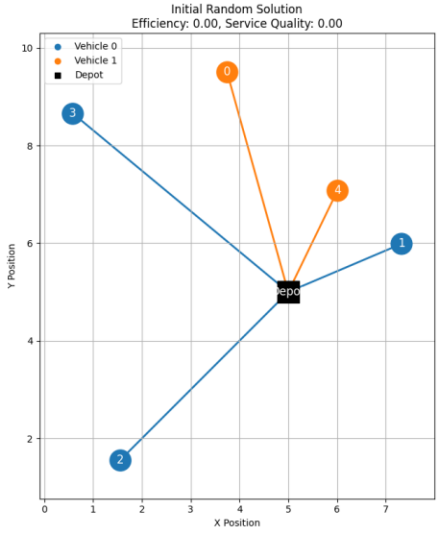
Implementation

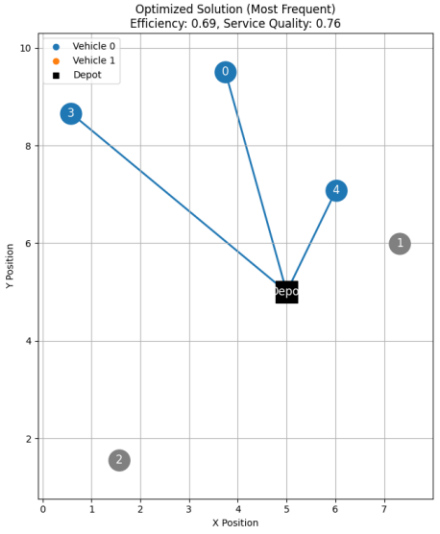
First Trial - Using QUBO for mulit-objective optimization.

Cost1 : Service quality / Cost2 : Efficiency (trade-off)

```python
def create_movrp_qubo_matrices(num_customers, num_vehicles, costs_eff, costs_ser):
    num_vars = num_customers * num_vehicles
    Q_eff = np.zeros((num_vars, num_vars))
    Q_ser = np.zeros((num_vars, num_vars))

    penalty_factor = 50

    for i in range(num_customers):
        for j in range(num_vehicles):
            var_idx = i * num_vehicles + j
            Q_eff[var_idx, var_idx] = costs_eff[i]
            Q_ser[var_idx, var_idx] = costs_ser[i]

    for i in range(num_customers):
        for j1 in range(num_vehicles):
            idx1 = i * num_vehicles + j1
            Q_eff[idx1, idx1] += -penalty_factor
            Q_ser[idx1, idx1] += -penalty_factor

            for j2 in range(j1 + 1, num_vehicles):
                idx2 = i * num_vehicles + j2
                Q_eff[idx1, idx2] += 2 * penalty_factor
                Q_ser[idx1, idx2] += 2 * penalty_factor

    return Q_eff, Q_ser
```







Flow: Simulator → Hardware

Optimization (Simulator): Estimate cost function on Aer simulator, but transpile for real backend.

Execution (Hardware): Run QAOA circuit with optimized parameters on real quantum device.

Hardware Experiments Summary

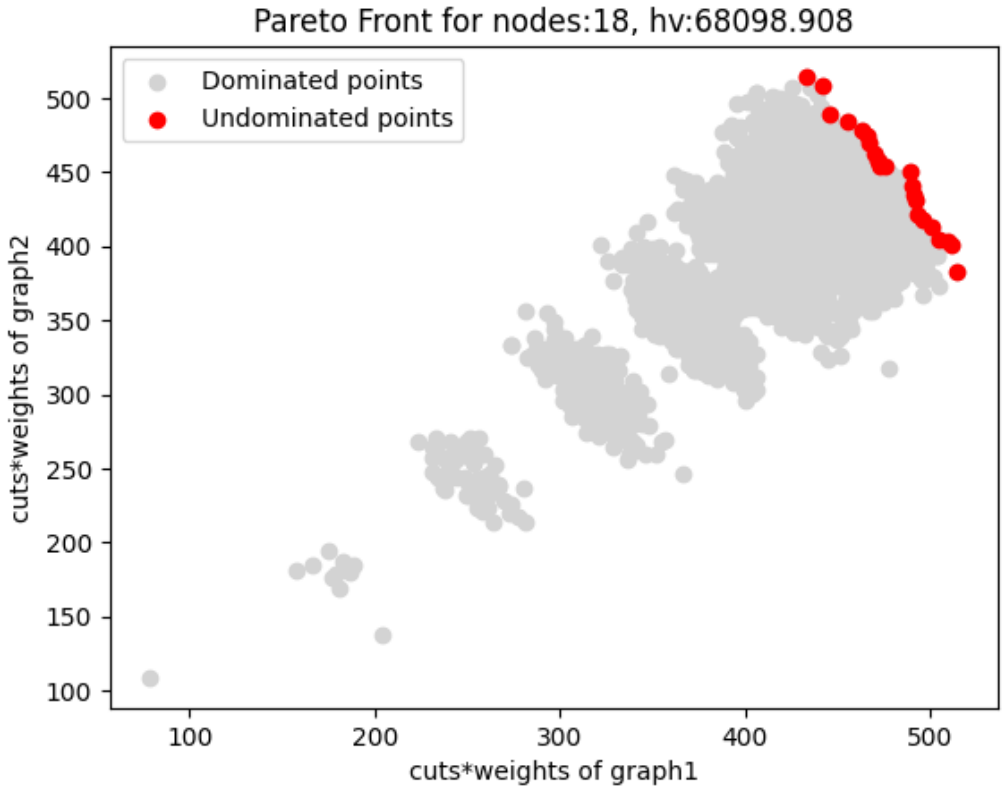
Optimization: Estimator on Aer simulator (transpilation for real backend).
Execution: Sampler on IBM hardware.

Experiment	Backend	n	Computational Time	Notes
1	ibm_strasbourg	6	2m 08s	Smallest instance
2	ibm_strasbourg	12	3m 49s	Medium instance
3	ibm_strasbourg	18	9m 09s	Largest on Strasbourg
4	ibm_brussels	18	9m 07s	Different backend

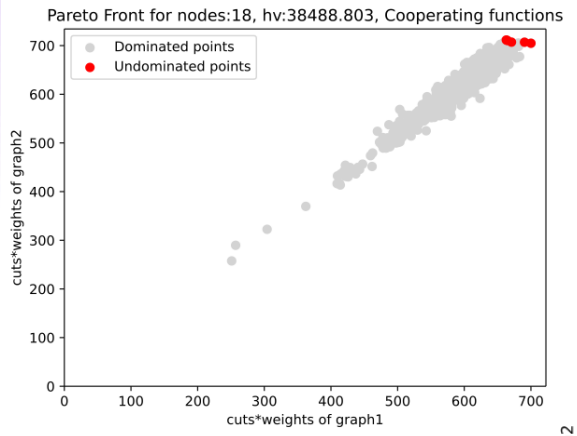
Experiment 3: Largest VRP Instance on IBM Strasbourg

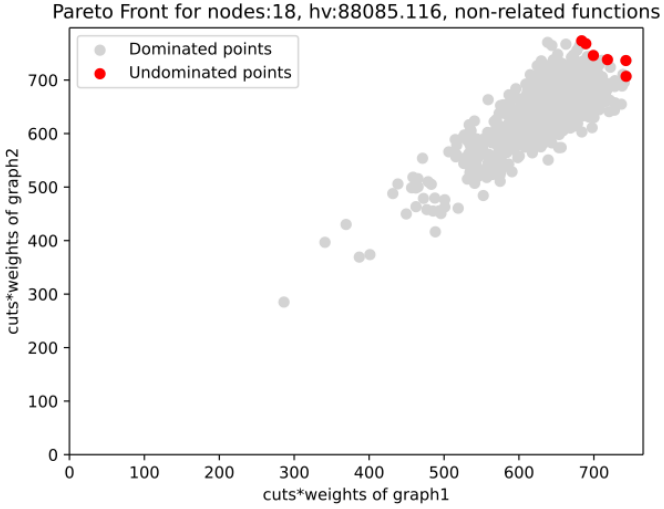
18 nodes

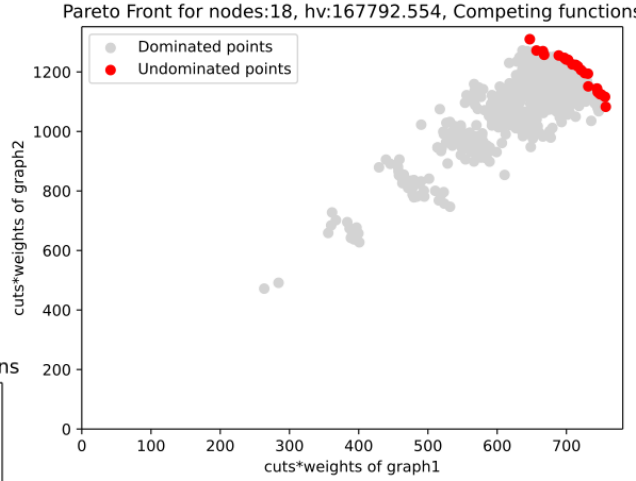
9m 09s computational time



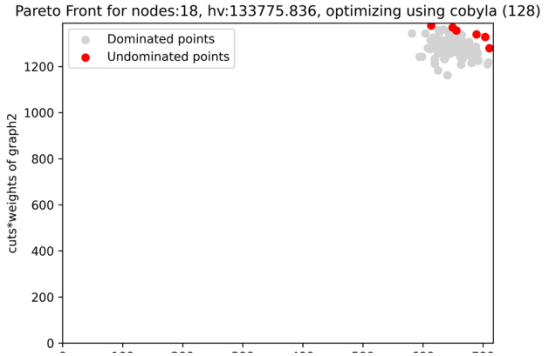
Cost Function (r>0, r=0, r<0)

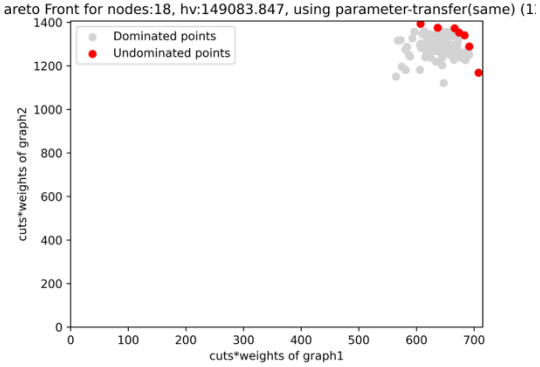


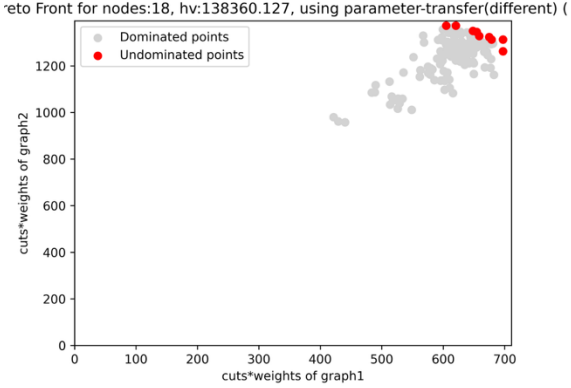


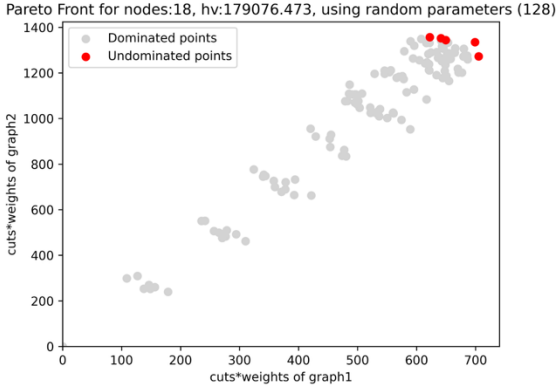


Power of Parameter Transfer

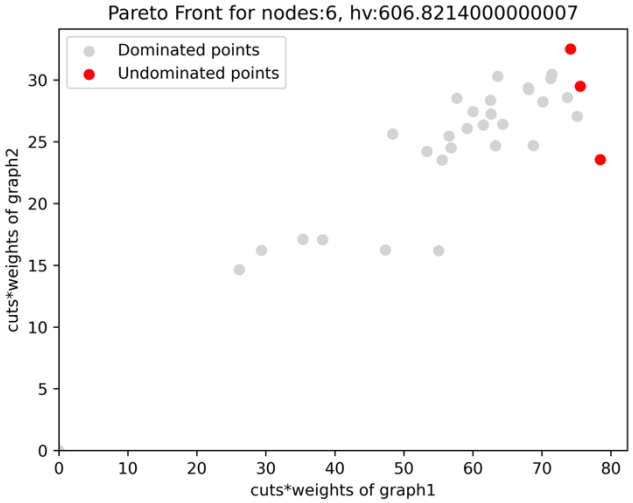


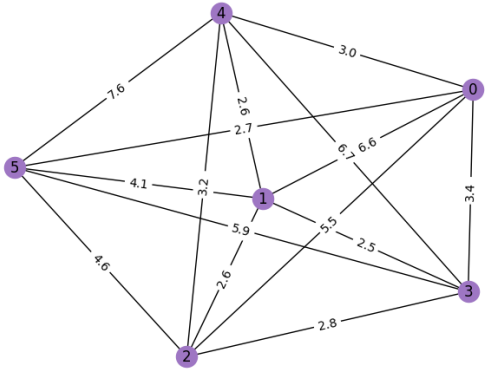


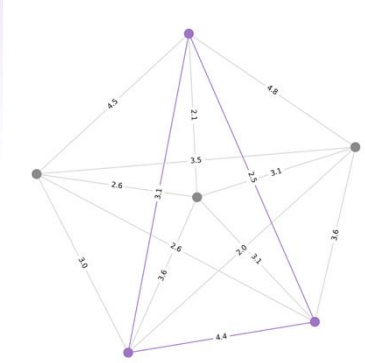


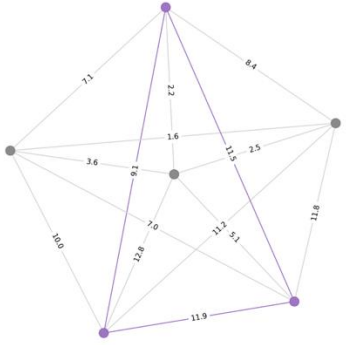


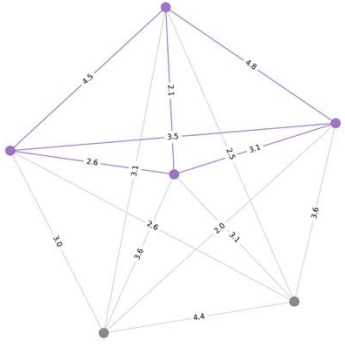
Result

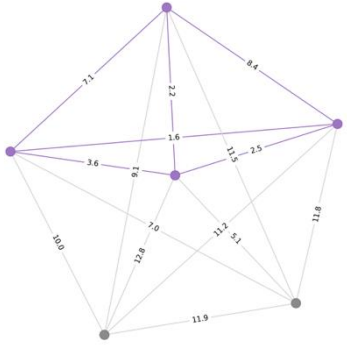

Pareto Front for nodes:6, hv:606.8214000000007
Dominated points
Undominated points
cuts*weights of graph2
cuts*weights of graph1
0
5
10
15
20
25
30
0
10
20
30
40
50
60
70
80


0
1
2
3
4
5









Summary

We **implememented** Quantum Approximate Multi-Objective Optimization.

We run multiple experiments on real IBM hardware.

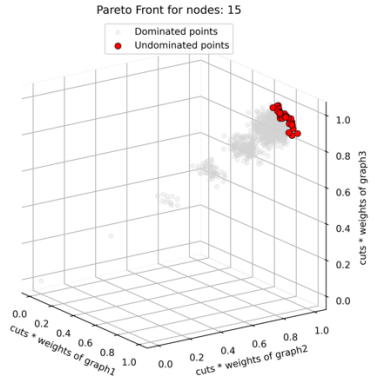
We run bunch of other **extra experiments on simulator**.

Future work

For example, optimize more cost functions simultaneously!

Commented code and all the results are on GitHub.

You can clone the repo immediately and check it yourself!



References

- [1] A. Kotil et al., “Quantum Approximate Multi-Objective Optimization,” arXiv:2503.22797v1 (2025).
- [2] A. Kotil et al., “Variational Quantum Multi-Objective Optimization,” arXiv:2312.14151 (2023).
- [3] R. Shaydulin et al., “Parameter Transfer for Quantum Approximate Optimization of Weighted MaxCut,” arXiv:2201.11785v2 (2023)
- Quantum approximate optimization algorithm, IBM Quantum Platform tutorial.