Development of a new variational quantum algorithm for MaxCut: QAOA and QEMC hybrid algorithm

Afonso Sequeira Azenha
afonso.azenha@tecnico.ulisboa.pt

Instituto Superior Técnico, Lisboa, Portugal

June 2024

Abstract

In this work, we introduce the Interpolated QAOA/QEMC (iQAQE) Framework, a novel approach inspired by the Quantum Approximate Optimization Algorithm (QAOA) and the Qubit-Efficient MaxCut Heuristic Algorithm (QEMC) for designing Variational Quantum Algorithms (VQAs) to solve the Maximum Cut (MaxCut) problem. This Framework builds on the core components of QEMC while integrating concepts from QAOA to harness the strengths of both algorithms, requiring fewer qubits and demonstrating greater resilience to statistical uncertainty associated with small shot numbers compared to QEMC. This framework provides adjustable parameters to create various VQAs, and we introduce heuristics for selecting these parameters, including the number of qubits, list cardinality, and mappings. Our evaluations show that iQAQE often matches or exceeds the performance of QEMC and can outperform classical algorithms like Goemans-Williamson in specific scenarios. We also propose two alternative approaches for solving the MaxCut problem derived from our iQAQE research, which offer additional insights and potential research directions. Additionally, we present a small machine learning model to determine optimal mappings for specific graphs based on statistical properties of the mappings themselves. Ultimately, the iQAQE Framework serves as a versatile testbed for developing new VQAs, potentially leading to significant advancements in quantum computing.

Keywords: Hybrid Quantum-Classical Computing (HQCC), Variational Quantum Algorithms (VQAs), Maximum Cut (MaxCut) Problem, Quantum Approximate Optimization Algorithm (QAOA), Qubit-Efficient MaxCut Heuristic Algorithm (QEMC).

1. Introduction

In recent decades, quantum computing has made significant strides [1], leveraging quantum mechanics to potentially revolutionize fields like cryptography, optimization, and machine learning. Despite these advances, current Noisy Intermediate Scale Quantum (NISQ) devices have yet to achieve quantum advantage, where quantum computers outperform classical ones in solving specific problems efficiently.

Currently, the most promising approaches for achieving meaningful quantum advantage are found in ”Hybrid Quantum-Classical Computing” (HQCC), which merges the computational power of quantum devices with the reliability of classical systems. Variational Quantum Algorithms (VQAs) have emerged as a leading area of research within this framework, primarily due to their compatibility with near-future quantum systems. They demand fewer qubits and operate with shallower circuit depths, making them highly adaptable to current quantum technology. These algorithms are intentionally designed as hybrids, leveraging classical optimizers to fine-tune the parameters of Parameterized Quantum Circuits (PQCs).

Among the applications, combinatorial optimization has seen notable progress. The Quantum Approximate Optimization Algorithm (QAOA) [2] shows potential for solving the MaxCut problem, but requires more qubits than current devices can handle when scaled to larger problem instances. This limitation has prompted the development of alternative approaches such as the Qubit-Efficient MaxCut Heuristic Algorithm (QEMC) [3], which addresses MaxCut with fewer qubits. Continued research in hybrid quantum-classical methods is essential for achieving quantum advantage.

1.1. Motivation

The search for efficient algorithms to solve combinatorial optimization problems is critical in numerous fields, such as logistics, finance, and telecommunications. The MaxCut problem, for example, has broad applications in areas like machine learn-

ing [4], statistical physics [5], circuit design [5], and data clustering [6]. Creating more efficient algorithms to solve this problem can improve the performance of these applications, driving substantial progress in their respective fields.

Moreover, there is a strong interest from the computer science community in terms of computational complexity. The MaxCut problem is recognized as NP-hard, and the search for efficient algorithms to solve it may offer valuable insights into the boundaries between classical and quantum computing. It might even contribute to unraveling one of the most perplexing questions in theoretical computer science: the $P = NP$ problem. Imagining a world where the $P = NP$ conjecture is proven true, albeit improbable, is intriguing. It would mean that every problem in NP could be solved in polynomial time, including MaxCut. This would also extend to problems like the Traveling Salesman Problem (TSP) and the Knapsack Problem, among others. The implications would be profound, as this would dramatically increase our ability to solve previously difficult optimization problems, with direct applications in areas like vehicle routing, job scheduling, and broader logistics. Additionally, the impact on cryptography would be as significant – if not more so – since many cryptographic techniques rely on the complexity of NP problems. For example, integer factorization is a key component of RSA (Rivest-Shamir-Adleman) encryption, a popular asymmetric encryption algorithm used extensively in public-key cryptography for secure message transmission over the internet. If $P = NP$, RSA encryption could easily be broken, leading to major security risks. Hence, the importance of studying these problems to fully understand their complexity.

The aforementioned considerations drive our efforts to develop a new algorithm for solving the MaxCut problem with greater efficiency and accuracy. This algorithm, the Interpolated QAOA/QEMC (iQAQE) Hybrid Algorithm, will be the focus of this work. More specifically, we will develop what we call the iQAQE Framework, which will serve as the foundation for developing numerous distinct iQAQE algorithms.

1.2. Topic Overview

In this project, we propose the Interpolated QAOA/QEMC Framework. This will enable us to develop novel VQAs by leveraging the strengths of two existing VQAs, QAOA and QEMC, for improved performance in solving the MaxCut problem. QAOA requires a qubit for each graph vertex, making it difficult to scale. In contrast, QEMC uses exponentially fewer qubits by assigning one basis state to each graph node, requiring only $\log_2(n)$ qubits (for n graph vertices). However, this compression leads to limitations in QEMC's results. By interpolating both VQAs, we aim to create an algorithm that utilizes fewer qubits than QAOA and performs better than QAOA and QEMC. The new algorithm, constructed through the iQAQE Framework, assigns multiple basis states to each node, in contrast to QEMC's single basis state approach. This design tentatively allows for a more practical implementation on present-day NISQ devices, thanks to its reduced qubit requirements compared to QAOA, fewer measurement shots than QEMC, and potentially greater trainability than QAOA.

1.3. Objectives

The primary objective of this thesis is to develop and analyze the iQAQE Framework. Algorithms built within this framework will be implemented and tested using classical simulations of quantum machines. The deliverables for this project include the iQAQE Framework's code[1], the results obtained from the simulations, and the analysis of these results. The expected outcomes are improvements in the performance of the iQAQE algorithm compared to QAOA and QEMC, with a focus on accuracy, efficiency, and scalability.

2. Theoretical Background

Here, we'll introduce some key concepts essential for grasping the foundations of this work.

2.1. Hybrid Quantum-Classical Computing

Hybrid quantum-classical computing refers to a computational approach that combines elements of both classical and quantum computing paradigms to leverage the strengths of each. In this model, classical processors and quantum processors work in tandem to solve complex problems more efficiently than either could achieve alone. This collaborative strategy aims to harness quantum computing's unique capabilities while mitigating the challenges and limitations associated with quantum systems, such as error correction and decoherence. As of today, the synergy between classical and quantum elements holds promise for addressing complex real-world problems in areas like optimization, machine learning, and cryptography [7].

2.1.1. Variational Quantum Algorithms

The essence of HQCC lies in VQAs, which use parameterized quantum circuits (Ansätze). These algorithms use classical optimization to adjust the quantum circuit parameters, similar to how neural networks are trained to minimize a cost function. This approach leverages classical optimization tools and keeps quantum circuit depth shal-

[1] Reach out to the author to obtain access to the code.

low, reducing noise and making VQAs suitable for the current NISQ era. VQAs have diverse applications and are considered the most promising for achieving near-term quantum advantage, despite challenges in trainability, accuracy, and efficiency.

Starting the development of a VQA involves defining a cost function, C, representing the problem's solution. Then, an ansatz, a parameterized quantum circuit, is proposed, with parameters $\boldsymbol{\theta}$ to be optimized. The ansatz undergoes training in a hybrid quantum-classical loop to minimize $C(\boldsymbol{\theta})$ (Eq. 1). Quantum computation estimates $C(\boldsymbol{\theta})$ and its gradient, while classical routines optimize $\boldsymbol{\theta}$. Further details for each step of the VQA framework are provided.

$$\boldsymbol{\theta}^{\star} = \arg\min_{\boldsymbol{\theta}} C(\boldsymbol{\theta}) \tag{1}$$

Cost function: Defining the VQA involves constructing a cost function that assigns numerical values to trainable parameters $\boldsymbol{\theta}$. This function shapes a cost landscape for optimization. The aim is to identify its global minimum, representing the solution. Typically, the cost takes the form of Eq. 2, consisting of functions f_k, a parameterized unitary $U(\boldsymbol{\theta})$, input states ρ_k, and observables O_k. Oftentimes, it is convenient to design the cost function to be given by the expectation value of a Hamiltonian (e.g., the cost Hamiltonian in QAOA), which is why we mention that it is beneficial for it to have this specific form. Ensuring the faithfulness and efficient estimation of $C(\boldsymbol{\theta})$ are essential for successful VQA implementation, requiring experimental validation.

$$C(\boldsymbol{\theta}) = \sum_k f_k \left(Tr\left[O_k U(\boldsymbol{\theta}) \rho_k U^{\dagger}(\boldsymbol{\theta}) \right] \right) \tag{2}$$

Ansatz: The ansatz (parameterized quantum circuit) plays a crucial role in determining the parameters $\boldsymbol{\theta}$ and their optimization to minimize the cost. Ansatz design can be problem-inspired or agnostic, with each approach offering distinct advantages. Problem-agnostic Ansätze are versatile but may require more parameters, complicating optimization. In contrast, problem-oriented Ansätze incorporate problem-specific information, potentially reducing parameter space and improving interpretability. Choosing between these approaches involves balancing accuracy and generality, considering resource utilization. While problem-inspired Ansätze often yield superior performance, their design presents challenges. Common ansatz types include hardware-efficient, unitary coupled clustered, and quantum alternating operator Ansätze, among others.

Gradients and training: After defining the cost function and ansatz, the next step involves training the parameters $\boldsymbol{\theta}$ to optimize Eq. 1. Analytically computing the gradient of the cost function is a notable advantage of many VQAs, achieved using the parameter-shift rule (Eq. 3), where $\boldsymbol{\theta}_{\pm} = \boldsymbol{\theta} \pm \alpha \boldsymbol{e_l}$, holds for any real number α. Practically, $\alpha = \pi/4$ is commonly used for accuracy. The VQA workflow resembles traditional machine learning, with input states processed by an ansatz parameterized by $\boldsymbol{\theta}$, followed by measurement and cost function estimation. A classical optimizer updates $\boldsymbol{\theta}$ iteratively until convergence to find the optimal parameters $\boldsymbol{\theta}^{\star}$. This high-level overview illustrates how VQAs function within a hybrid loop. Therefore, specifying the cost function and ansatz will be sufficient to characterize them.

2.2. MaxCut Problem

The MaxCut problem, a fundamental problem in graph theory and combinatorial optimization, involves partitioning a graph $G = (V, E)$ into two disjoint subsets, S_1 and S_2, such that the number of edges connecting vertices from different subsets is maximized. Formally, the objective is to find a partition (S_1, S_2) that maximizes:

$$\text{Cut}(S_1, S_2) = \sum_{(u,v)\in E} \chi(u, v), \tag{3}$$

where $\chi(u, v) = 1$ if u and v belong to different subsets, and $\chi(u, v) = 0$ if they belong to the same subset. This quantity is referred to as the "cut" of the partition (S_1, S_2). Pictorially, this can be represented as cutting the edges of the graph (Figure 1), hence the name MaxCut. What we describe here is the un-directed, un-weighted MaxCut problem. A more general formulation would involve the specific graph's adjacency matrix, W_{ij}.

Since the MaxCut problem is NP-hard, finding an optimal solution efficiently is a significant computational challenge, especially as the graph size grows. However, researchers have developed various approximation algorithms and heuristics to approach near-optimal solutions in a reasonable time, including both classical and quantum approaches. These methods are designed to tackle the intrinsic

$$\frac{\partial C}{\partial \theta_l} = \sum_k \frac{1}{2\sin\alpha} \left(Tr\left[O_k U(\boldsymbol{\theta}_+) \rho_k U^{\dagger}(\boldsymbol{\theta}_+) \right] - Tr\left[O_k U(\boldsymbol{\theta}_-) \rho_k U^{\dagger}(\boldsymbol{\theta}_-) \right] \right) \tag{3}$$

complexity of the problem, providing practical solutions that have real-world applications. As mentioned earlier, the MaxCut problem finds use in a variety of fields, including machine learning [4], statistical physics [5], circuit design [5], and data clustering [6].

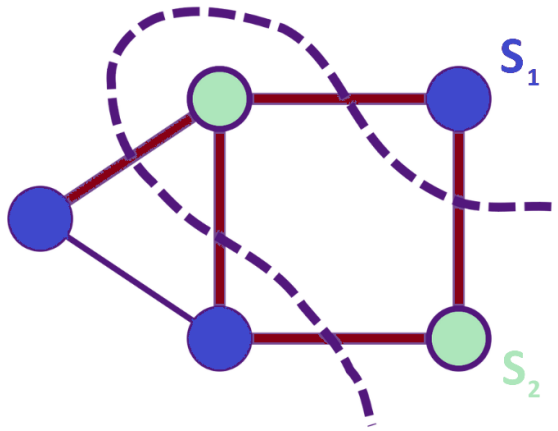

Figure 1: An example of a graph with a partition that maximizes the number of cut edges (red). Note that the MaxCut partition might not be unique. The number of edges that are "cut" is promptly labeled as the "cut".

2.3. State-of-the-Art Algorithms

Next, we introduce the state-of-the-art algorithms for tackling the MaxCut problem, encompassing both classical (Goemans-Williamson) and hybrid quantum-classical (QAOA and QEMC) approaches.

2.3.1. Goemans-Williamson Algorithm

In this section, we explore the Goemans-Williamson (GW) Algorithm, pivotal in combinatorial optimization, especially for MaxCut. Developed by Goemans and Williamson in 1995, it employs semidefinite programming to create an approximate solution, later refined for a near-optimal cut. Semidefinite programming optimizes a linear function over a symmetric matrix while ensuring positive semidefiniteness. It finds applications in control theory, nonlinear programming, geometry, and combinatorial optimization, as detailed in [8] and related references.

Formally, the Goemans-Williamson Algorithm is presented as follows, based on the original formulation [8]. Given a graph with vertex set $V = \{1, ..., n\}$ and non-negative weights $w_{ij} = w_{ji}$ for each pair of vertices i and j, the weight of the maximum cut $w(S_1, S_2)$ is obtained from the integer quadratic program:

$$\begin{aligned} &\text{Maximize} \quad \frac{1}{2} \sum_{\substack{i<j: \\ (i,j)\in E}} w_{ij}(1 - y_i y_j) \\ (Q) \qquad &\text{Subject to: } y_i \in \{-1, 1\} \qquad \forall i \in V. \end{aligned} \tag{4}$$

Where $S_1 = \{i \mid y_i = 1\}$ and $S_2 = \{i \mid y_i = -1\}$ correspond to a cut of weight

$$w(S_1, S_2) = \frac{1}{2} \sum_{\substack{i<j \\ (i,j)\in E}} w_{ij} \left(1 - y_i y_j\right). \tag{5}$$

Solving this NP-hard program with the GW algorithm involves relaxing constraints, resulting in a semidefinite programming relaxation (P):

$$\begin{aligned} &\text{Maximize} \quad \frac{1}{2} \sum_{\substack{i<j: \\ (i,j)\in E}} w_{ij}(1 - v_i \cdot v_j) \\ (P) \qquad &\text{Subject to: } v_i \in S_n \qquad \forall i \in V. \end{aligned} \tag{6}$$

The GW algorithm then proceeds as follows:

1. Solve (P) to obtain optimal vectors v_i;
2. Select a random vector r uniformly distributed on S_n;
3. Partition vertices into S_1 and S_2 based on whether $v_i \cdot r \geq 0$ or $v_i \cdot r < 0$.

This method essentially partitions vertices based on a randomly chosen hyperplane in n dimensions, dividing them into S_1 and S_2 accordingly. Furthermore, it can be shown that the GW algorithm has a performance guarantee of:

$$\alpha = \min_{0 \leq \theta \leq \pi} \frac{2}{\pi} \frac{\theta}{1 - \cos\theta} > 0.878. \tag{7}$$

A comprehensive proof outlining the derivation of this 0.878 value is available in section 3 of [8], along with a more in-depth exposition of the algorithm.

2.3.2. Quantum Approximate Optimization Algorithm (QAOA)

The QAOA, renowned for its prowess in quantum-enhanced optimization, was initially crafted to tackle a range of combinatorial optimization problems such as constraint-satisfaction and MaxCut. QAOA utilizes a cost Hamiltonian, denoted as H_C, which serves as the basis for extracting the objective function. This Hamiltonian is formed by associating each conventional classical variable in Eq. 4 with a Pauli spin $1/2$ operator, represented as $\boldsymbol{Z}_j$. Formally, the usual MaxCut objective $L(s) = \frac{1}{2} \sum_{\substack{i<j: \\ (i,j)\in E}} (1 - s_i s_j)$, denoting the cut of partition

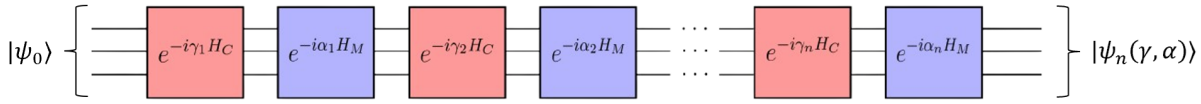

Figure 2: Quantum alternating operator ansatz. Adapted from [9].

$s = (s_1, ..., s_N)$, is transformed into the cost Hamiltonian

$$H_C = \frac{1}{2} \sum_{\substack{i<j: \\ (i,j)\in E}} (1 - \boldsymbol{Z}_i \boldsymbol{Z}_j). \tag{8}$$

The Trotter-inspired QAOA ansatz involves p cycles of alternating evolution between H_C and a mixer Hamiltonian, H_M. The role of this mixer Hamiltonian can be understood as introducing quantum fluctuations, or transitions, between different states, helping the algorithm explore the solution space more effectively, ideally preventing it from getting trapped in sub-optimal local minima. This ansatz, depicted in Figure 2, illustrates n QAOA layers, each comprising alternating applications of H_C and H_M.

The cost function $C(\boldsymbol{\gamma}, \boldsymbol{\alpha})$ is determined by the expectation value of H_C over the ansatz state $|\psi_p(\boldsymbol{\gamma}, \boldsymbol{\alpha})\rangle$, obtained at the end of the quantum circuit (Figure 2). In practice, defining $\boldsymbol{\theta} = \{\boldsymbol{\gamma}, \boldsymbol{\alpha}\}$, the cost function is $C(\boldsymbol{\gamma}, \boldsymbol{\alpha}) = \langle\psi_p(\boldsymbol{\gamma}, \boldsymbol{\alpha})| H_C |\psi_p(\boldsymbol{\gamma}, \boldsymbol{\alpha})\rangle$, with

$$|\psi_p(\boldsymbol{\gamma}, \boldsymbol{\alpha})\rangle = e^{-i\alpha_p H_M} e^{-i\gamma_p H_C} ... e^{-i\alpha_1 H_M} e^{-i\gamma_1 H_C} |\psi_0\rangle, \tag{9}$$

where $|\psi_0\rangle$ is the initial state entering the ansatz. In QAOA, it is customary to start with a uniform superposition over the n bit-string[2] basis states, i.e., $|\psi_0\rangle = |+\rangle^{\otimes n} = \frac{1}{\sqrt{2^n}} \sum_{z\in\{0,1\}^n} |z\rangle$. This is achieved by applying a Hadamard gate to each of the qubits, at the start of the quantum circuit.

In practice, the mixer Hamiltonian terms $e^{-i\alpha_l H_M}$ correspond to $R_x(2\alpha_l)$ and are easy to implement. The cost Hamiltonian terms, on the other hand, are a bit more tricky, requiring the use of 2 CNOT gates. Each of the terms $e^{-i\gamma_l(1-\boldsymbol{Z}_j\boldsymbol{Z}_k)/2}$ can be transpiled into a quantum circuit, as shown in Figure 3, below. For each edge in the graph, we'll have one of these terms, in each of the p layers of the QAOA ansatz.

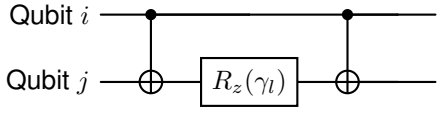

Figure 3: $e^{i\gamma_l \boldsymbol{Z}_i \boldsymbol{Z}_j/2}$ decomposition.

[2]Keep in mind, there's one qubit allocated for each graph node.

Subsequently, a classical routine is employed to optimize the values of the parameters by minimizing the negative of the cost (analogous to the "cut"), therefore enabling the extraction of the Max-Cut partition.

2.3.3. Qubit-Efficient MaxCut Heuristic Algorithm (QEMC)

QEMC [3] is somewhat similar to QAOA. However, it has a number of crucial differences. First, it only requires $n = \log_2(N)$ qubits, instead of N, where N is the number of nodes in the graph. Additionally, the QEMC algorithm is based on a novel probability threshold encoding scheme, a suitable cost function, and a parameterized unconstrained quantum circuit. Going in order:

Probability threshold encoding scheme: With n qubits, each of the $N = 2^n$ basis states will represent one of the graph's nodes. Following the sampling of the quantum circuit, a probability distribution is generated. Nodes with probabilities exceeding a certain threshold, $p_{th} = \frac{1}{2B}$, belong to "Set 1", while probabilities below this value indicate inclusion in "Set 0". This strongly diverges from how we encode set inclusion in QAOA.

Cost function: The objective function utilized in QEMC is the following:

$$L(\{p(i)\}) = \sum_{\substack{j<k: \\ (j,k)\in E}} \left[\left(d(j,k) - \frac{1}{B} \right)^2 + \left(s(j,k) - \frac{1}{B} \right)^2 \right], \tag{10}$$

where $d(j,k) = |p(j) - p(k)|$ and $s(j,k) = p(j) + p(k)$ are the absolute difference and sum of the corresponding states' probabilities. The idea is that as both $d(j,k)$ and $s(j,k)$ tend towards $1/B$, the probability of one node approaches zero (distinctive "Set 0"), while the probability of the other node approaches $1/B$ (distinctive "Set 1"), without specifying which is which. Ultimately, just like for QAOA, connections between nodes of different sets are favoured. Note, however, that this probability threshold encoding scheme assumes, *a priori*, that one of the sets ("Set 1") has B nodes. Nevertheless, this is not an issue, as we can efficiently iterate through all potential values of $B = 1, ..., \lfloor \frac{N}{2} \rfloor$. Frequently, it is reasonable to set $B = N/2$, and we shall use this as our starting point.

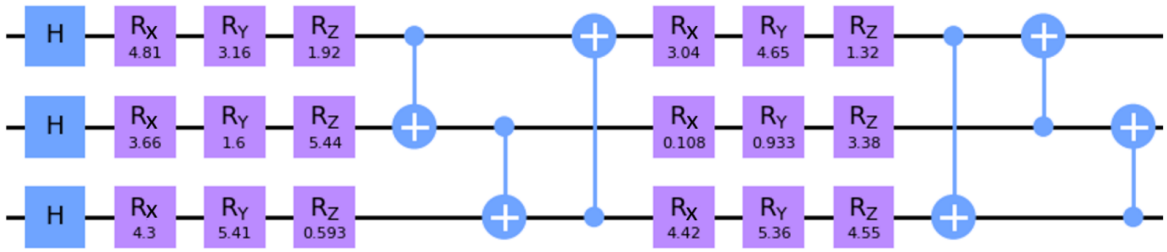

Figure 4: "Strongly Entangling Layers" circuit ansatz: case of $n = 3$ qubits and $p = 2$ layers. After a single layer of Hadamard gates, each subsequent layer consists of $3n$ single-qubit parameterized rotation gates and n CNOT gates (entangling gates). Reproduced from [3].

Problem-agnostic ansatz: The QEMC circuit ansatz is agnostic to specific graph instances, a departure from QAOA where the graph structure is explicitly encoded in the quantum circuit. Instead, the graph is implicitly encoded through the cost function. As a result, the QEMC quantum circuit is not bound to any particular form and only needs to be expressive enough to approximate the optimal states in the Hilbert space. Such problem-independent ansatz approach provides considerable flexibility in ansatz selection. Frequently, the circuit ansatz known as "Strongly Entangling Layers" is employed, as depicted in Figure 4. As can be seen, this ansatz applies a series of parameterized single-qubit rotations interspersed with controlled entangling gates to generate a highly entangled quantum state. We use Pennylane's `qml.StronglyEntanglingLayers` implementation for this purpose.

One notable, and perhaps unfortunate, property of QEMC is that it is efficiently simulable classically. Due to the exponential compression of the number of qubits, the algorithm can be feasibly run on a classical computer, even for large graphs, hence defeating its purpose as a quantum algorithm. This is something the authors of the algorithm [3] realized in hindsight. For this reason, it is now termed a quantum-inspired classical algorithm.

3. iQAQE Framework

We introduce a new framework, inspired by QAOA and QEMC, for seamlessly designing multiple distinct VQAs. This framework builds on QEMC's core components while integrating concepts from QAOA to leverage the strengths of both algorithms for improved performance. Tentatively named the iQAQE Framework, it opens the door to numerous unexplored and unknown VQAs, offering a variety of parameters to experiment with, such as the number of qubits, list cardinality, and mappings.

In iQAQE, we depart from the QEMC approach by associating each graph node with a list of basis states, in contrast to QEMC's consideration of a single basis state for each node. Each of these lists comprises $c \in \left[1, 2^{\texttt{n\_qubits}-1}\right]$ basis states, where `n_qubits` represents the number of qubits. c denotes the cardinality of the lists. What we formerly referred to as "mapping" involves distributing basis states among these lists. The term "mapping" can also signify a specific allocation of basis states among the lists. This design allows for potential overlap among states from different lists/nodes. Additionally, it is important to note that the encoding of these states will utilize a qubit range expected to fall between the QEMC and QAOA requirements, specifically in $[\log_2 N, N]$, for an N-node graph.

As a default approach, we compute node probabilities straightforwardly by summing the probabilities of associated basis states and then normalizing the result. This is necessary because we utilize QEMC's probability threshold encoding scheme. Additionally, we consider iQAQE to use the same ansatz and cost function as QEMC, adjusted for the appropriate number of qubits. However, we also explore the potential for slight deviations from this formula, such as alternative methods for computing node probabilities and the adoption of problem-inspired ansatz variations.

We can postulate that such a hybrid approach might have some potential advantages. Namely, it should allow for less shots than in QEMC and fewer qubit requirements than in QAOA, in addition to, arguably, being better trainable. Another notable feature of this algorithm is its tunable "quantumness." By interpolating between QAOA (a hybrid quantum-classical method) and QEMC (a classical, quantum-inspired method), we can selectively adjust the degree of quantum and classical components, which could prove to be advantageous.

We designate this as a framework due to the vast array of possibilities in qubit number, list cardinality, and mappings, which facilitate the creation of multiple unique VQAs, each with its own merits and drawbacks. In this study, we will introduce heuristics for parameter selection and detail the results corresponding to each.

4. Implementation

In our numerical implementation, we leveraged PennyLane, a Python library tailored for differentiable programming of quantum computers. PennyLane facilitates the execution of variational quantum circuits and their simultaneous training, akin to training a classical neural network, within the same Python environment. We utilized two PennyLane devices: `default.qubit` for smaller circuits and exact expectation values, and `lightning.qubit` for larger circuits. Furthermore, NetworkX was instrumental in generating and manipulating graphs for addressing the MaxCut problem, while CVXPY, tailored for convex optimization, assisted in solving the semidefinite programming relaxation of MaxCut within the Goemans–Williamson algorithm framework.

Despite the versatility of our setup, testing on actual quantum hardware was unavailable, limiting our evaluation to simulations on a personal computer. This constraint also impacted scalability and comprehensive testing, particularly for larger graphs. Simulations were conducted using the Adam optimizer, with learning rates fine-tuned as hyperparameters.

Moreover, it's important to elucidate the methodologies employed for benchmarking and testing the algorithms' efficacy. Central to our evaluation are the cost and approximation ratio[3] plots, essential for comparing convergence rates and final cut values across the algorithms. Additionally, recognizing the influence of initial parameters on outcomes, we adopt statistically robust metrics such as average cut values and the best-so-far (BSF) cut value, obtained through a monotonic transformation[4]. Furthermore, to mitigate the impact of outliers, we incorporate the median BSF cut value in our analysis, computed from multiple training curves to ensure statistical robustness. Frequently, we use the average cut or approximation ratio curves generated from 10 different random initializations. This approach enables us to evaluate the algorithm's performance on average post-training. Essentially, we assess the likelihood of obtaining favorable results following a single run.

In our pursuit of optimization, grid searches are conducted on hyperparameters like Adam's learning rate, the number of layers (`n_layers`), and qubits (`n_qubits`), albeit constrained by the unavailability of an HPC cluster. We also refine our evaluations by benchmarking against the Goemans-Williamson algorithm, offering insights into the relative performance of our algorithms compared to state-of-the-art solutions. Additionally, we extend our comparative analysis to include commonly used quantum optimization algorithms such as QAOA and QEMC. Through these multifaceted evaluations, we aim to provide nuanced insights into the strengths and limitations of each algorithm under consideration.

[3] The approximation ratio denotes the proportion of the achieved "cut" value to the MaxCut value.

[4] If a dip occurs in the graph, it is adjusted to match the previous graph value, ensuring that the final result is monotonically increasing. That's what the BSF transformation does.

5. iQAQE Schemes and Results

Something about the iQAQE Schemes and Results goes here. I'll have to choose which results to include here, as I definitely have way too many graphs for just 10 pages.

6. Alternative Schemes

Something about the alternative schemes goes here. What I put here will depend on what I put in the results section, before, due to space restrictions.

7. Conclusions

Something about the conclusions goes here. This can be probably just be a summary of my thesis's conclusion chapter. Also mention future work, of course.

References

[1] John Preskill. Quantum computing 40 years later, 2023.

[2] Edward Farhi, Jeffrey Goldstone, and Sam Gutmann. A quantum approximate optimization algorithm, 2014.

[3] Yovav Tene-Cohen, Tomer Kelman, Ohad Lev, and Adi Makmal. A variational qubit-efficient maxcut heuristic algorithm, 2023.

[4] Y.Y. Boykov and M.-P. Jolly. Interactive graph cuts for optimal boundary & region segmentation of objects in n-d images. In *Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001*, volume 1, pages 105–112 vol.1, 2001.

[5] Francisco Barahona, Martin Grötschel, Michael Jünger, and Gerhard Reinelt. An application of combinatorial optimization to statistical physics and circuit layout design. *Operations Research*, 36(3):493–513, 1988.

[6] Jan Poland and Thomas Zeugmann. Clustering pairwise distances with missing data: Maximum cuts versus normalized cuts. In Ljupčo Todorovski, Nada Lavrač, and Klaus P. Jantke, editors, *Discovery Science*, pages 197–208, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.

[7] M. Cerezo, Andrew Arrasmith, Ryan Babbush, Simon C. Benjamin, Suguru Endo, Keisuke Fujii, Jarrod R. McClean, Kosuke Mitarai, Xiao Yuan, Lukasz Cincio, and Patrick J. Coles. Variational quantum algorithms. *Nature Reviews Physics*, 3(9):625–644, August 2021.

[8] Michel X. Goemans and David P. Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *J. ACM*, 42(6):1115–1145, nov 1995.

[9] Intro to QAOA. `https://pennylane.ai/qml/demos/tutorial_qaoa_intro/`, 2024. Accessed: 2023-12-18.