A Review on Quantum Approximate Optimization Algorithm and its Variants

Kostas Blekos,
Dean Brand,
Andrea Ceschini,
Chiao-Hui Chou,
Rui-Hao Li,
Komal Pandya,
Alessandro Summer

CERN Quantum Journal Club

20/07/2023

QOSF PROJECT: QAOA REVIEW

- Literature still contains many conflicting opinions on whether the QAOA can actually outperform classical algorithms with current quantum devices.

- The lack of commonly acknowledged guidelines on QAOA may limit its impact in the quantum tech ecosystem.

- We aim to offer a guide for suggested uses of the QAOA:

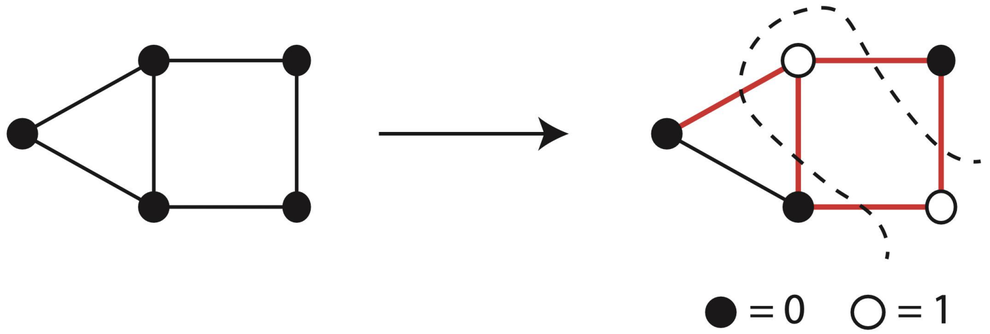
  - *"Can the QAOA outperform classical algorithms?"*
  - *"When and how should QAOA be used?"*

STRUCTURE OF THE PAPER

- Background on MaxCut and QUBO
- QAOA
- Analysis of different aspects of QAOA
- Our experimental results
- Discussion
- Practical guidelines

(WEIGHTED) MAXCUT

- Graph: Set of **vertices** (*V*) connected by (weighted) **edges** (*E*)
- MaxCut: Partition of vertices into two disjoint subsets (labeled by 0 and 1), such that the total **weight** of the edges between the two subsets is maximized
- For equal-weight edges, the goal is simply to maximize the number of edges connecting the two subsets.



Input:
(Weighted) graph $G = (V, E)$

Output:
Maximum cut $x \in (0,1)^n$
$x = [0, 1, 0, 1, 0]$

QUBO

- Quadratic Programs: Optimize (maximize or minimize) a quadratic objective function subject to linear constraints on the variables

  e.g., minimize $\mathbf{x}^T Q \mathbf{x} + \mathbf{c}^T \mathbf{x}$
  subject to $A\mathbf{x} \leq \mathbf{b}$

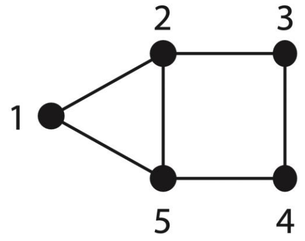
- Special case: Quadratic Unconstrained Binary Optimization (QUBO)
  - Quadratic objective function
  - No constraints on variables
  - Binary variables

- QUBO examples
  - MaxCut
  - Number partitioning
  - Graph coloring
  - . . .

QUBO

MaxCut

Weight matrix:



$$W = \begin{pmatrix} 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 1 & 0 \end{pmatrix}$$

Cost function:

$$C(\mathbf{x}) = \sum_{i,j=1}^{n} W_{ij} x_i (1 - x_j)$$

QUBO

QUBO matrix and vectors:

$$c_i = \sum_{j=1}^{n} W_{ij}, \qquad Q_{ij} = -W_{ij}$$

Cost function:

$$\begin{aligned} C(\mathbf{x}) &= \mathbf{x}^T Q \mathbf{x} + \mathbf{c}^T \mathbf{x} \\ &= \sum_{i,j=1}^{n} Q_{ij} x_i x_j + \sum_{i=1}^{n} c_i x_i \end{aligned}$$

QUANTUM APPROXIMATE OPTIMIZATION ALGORITHM (QAOA)

- The QAOA was first introduced by Farhi et al. (2014) as a VQA able to find approximate solutions to the MaxCut problem (QUBO instances), suitable to be run on NISQ devices
- It is considered one of the most promising quantum algorithms for quantum advantage on NISQ devices
- Can be regarded as a special case of Variational Quantum Eigensolvers (VQE)
- Layerized variational form based on Trotterized adiabatic process (related to adiabatic quantum computing)
- Key idea: encode the cost function of the optimization problem in the cost Hamiltonian, then encode it into a variational quantum circuit and leverage time evolution and layering to find a good approximated solution.

QUBO TO HAMILTONIAN

Goal: Find the cost Hamiltonian operator H_C that encodes the cost function $C(x)$, i.e.,

$$H_C \left|x\right\rangle = C(x) \left|x\right\rangle , \qquad x = \{0, 1\}.$$

Using the fact that

$$Z_i \left|x\right\rangle = (-1)^{x_i} \left|x\right\rangle = (1 - 2x_i) \left|x\right\rangle \implies \frac{\mathbb{1} - Z_i}{2} \left|x\right\rangle = x_i \left|x\right\rangle ,$$

we have

$$C(x) = \sum_{i,j} Q_{ij} x_i x_j + \sum_i c_i x_i$$

$$\implies \quad H_C = \sum_{i,j} Q_{ij} \left(\frac{\mathbb{1} - Z_i}{2}\right)\left(\frac{\mathbb{1} - Z_j}{2}\right) + \sum_i c_i \left(\frac{\mathbb{1} - Z_i}{2}\right)$$

$$= \sum_{i,j} \frac{1}{4} Q_{ij} Z_i Z_j - \sum_i \frac{1}{2} \left(c_i + \sum_j Q_{ij}\right) Z_i + \left(\sum_{i,j} \frac{Q_{ij}}{4} + \sum_i \frac{c_i}{2}\right).$$

INSPIRATION FROM TROTTERIZED AQC

Typical steps of Trotterized adiabatic quantum computing (AQC):

- Prepare initial state as the highest energy state of some "mixer" Hamiltonian (that does not commute with H_C), $H_M = \sum_i X_i$, i.e., $|\psi_0\rangle = \otimes_i |+\rangle$.
- Set the total Hamiltonian $H(t) = f(t)H_C + g(t)H_M$ with slowly varying control functions $f(t) = t/T$ $g(t) = 1 - t/T$.
- Through **adiabatic evolution**, the system will end up in the highest energy state of the cost Hamiltonian H_C, which then solves the QUBO problem.
- In practice, to implement the adiabatic evolution, one decomposes the time-evolution operator (for a time-dependent Hamiltonian) into a sequence of small steps through the **Trotter-Suzuki formula** :

$$U(t) := \mathcal{T} \exp\left[-i \int_0^T H(t)dt\right] \approx \prod_{a=0}^{k-1} \exp\left[-iH(a\tau)\tau\right]$$
$$= \prod_{a=0}^{k-1} \exp\left[-if(a\tau)H_C\tau\right] \exp\left[-ig(a\tau)H_M\tau\right].$$

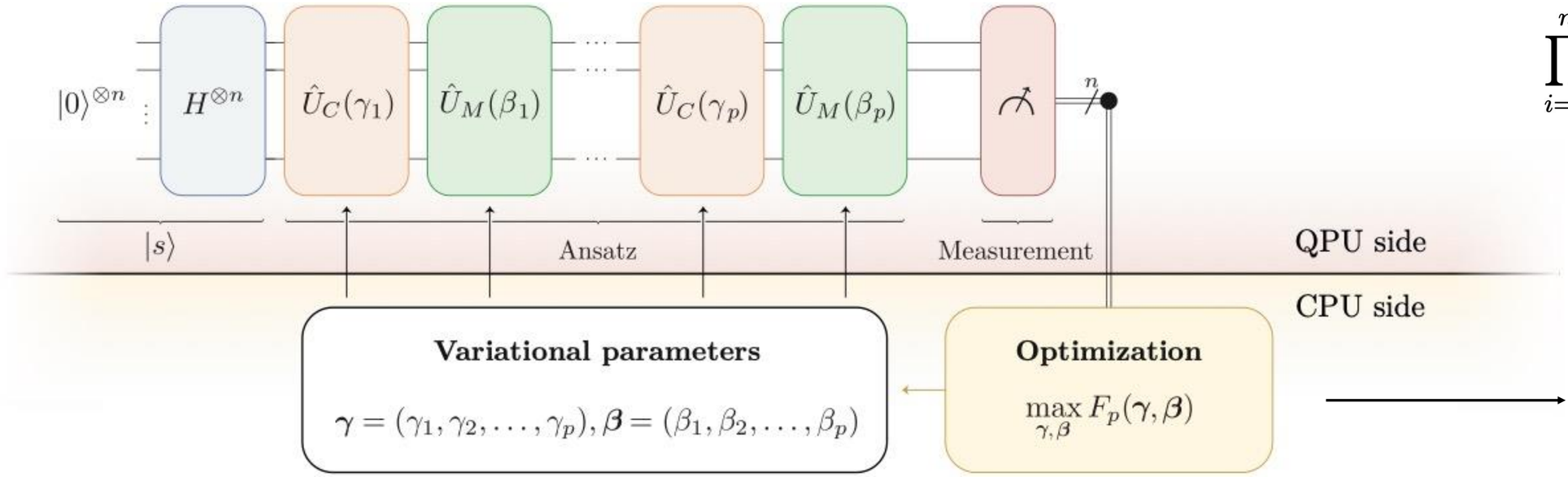
QUANTUM APPROXIMATE OPTIMIZATION ALGORITHM (QAOA)

- Inspired by Trotterized AQC, QAOA was designed to be a variational algorithm with repeated cost and mixer layers.
- The quantum circuit has *p* repetitions of those layers.
- For $p \to \infty$ QAOA can at least exactly approximate adiabatic quantum evolution and can therefore find the exact optimal solution.

$$U_M(\beta) = e^{-i\beta H_M} = \prod_{i=1}^{n} R_{X_i}(2\beta)$$

$$U_C(\gamma) = e^{-i\gamma H_C} = \prod_{i,j=1}^{n} R_{Z_i Z_j}\left(\frac{1}{2} Q_{ij}\gamma\right)$$

$$\prod_{i=1}^{n} R_{Z_i}\left(-\left(c_i + \sum_{j=1}^{n} Q_{ij}\right)\gamma\right).$$

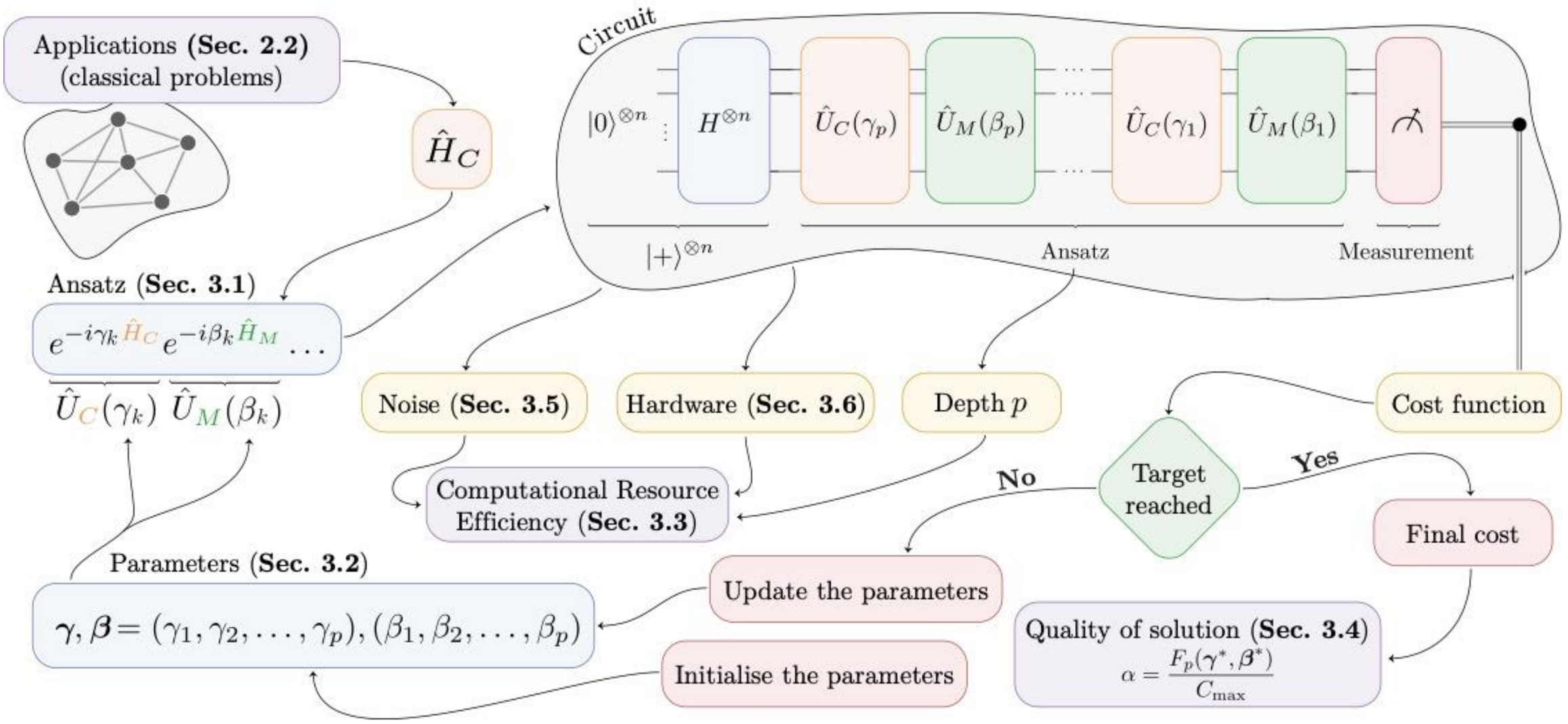


$$\alpha = \frac{F_p(\boldsymbol{\gamma}^*, \boldsymbol{\beta}^*)}{C_{\max}}$$

QAOA STEPS

1. Initialize $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$ with suitable real values.
2. Prepare the state $|\psi(\boldsymbol{\beta}, \boldsymbol{\gamma})\rangle$ using the QAOA circuit and measure it in the computational basis.
3. Compute the expectation value $\langle\psi(\boldsymbol{\beta}, \boldsymbol{\gamma})|H_C|\psi(\boldsymbol{\beta}, \boldsymbol{\gamma})\rangle$.
4. Find a new set of parameters $(\boldsymbol{\beta}_{\text{new}}, \boldsymbol{\gamma}_{\text{new}})$ with a classical optimization algorithm (e.g., gradient descent) and use this new set of parameters in the QAOA circuit.
5. Repeat steps 2 - 4 until some suitable convergence criterion is met.
6. The solution is then approximated as $|\psi(\boldsymbol{\beta}_{\text{opt}}, \boldsymbol{\gamma}_{\text{opt}})\rangle$ which maximizes $\langle\psi(\boldsymbol{\beta}_{\text{opt}}, \boldsymbol{\gamma}_{\text{opt}})|H_C|\psi(\boldsymbol{\beta}_{\text{opt}}, \boldsymbol{\gamma}_{\text{opt}})\rangle$

GENERAL QAOA FRAMEWORK



QAOA ANSATZ IMPROVEMENTS

- Multiple approaches have been proposed, including **ma-QAOA**, **QAOA+**, **ab-QAOA**, and **PG-QAOA**, which introduce new parameters, layers, and optimization techniques to enhance the traditional QAOA ansatz.
- **ADAPT-QAOA** provides an iterative method that selects QAOA mixers from a pool of operators, leading to faster convergence and reduced resource requirements.
- **RQAOA** is a non-local variant that iteratively reduces problem size.
- **WS-QAOA** attempts to initialize QAOA based on the solution to relaxed QUBO problems and is capable of retaining the performance guarantees of classical algorithms, like the GW bound when solving MaxCut.
- **Quantum Alternating Operator Ansatzes** offer a more flexible framework by alternating between general sets of operators, making them suitable for a broader range of problems with hard and soft constraints.
- **FALQON** and **FQAOA** tackle quantum optimization from different angles: FALQON focuses on feedback-based optimization without classical intervention, while FQAOA uses fermion particle number preservation to impose constraints intrinsically. Both methods show potential performance advantages over existing techniques.
- Lastly, **Quantum Dropout** offers an innovative approach to handle hard cases in combinatorial optimization problems, selectively modifying the quantum circuit to improve QAOA performance.

PARAMETER OPTIMIZATION

- Various techniques have been proposed to address the challenge of finding **good initial parameters**, such as heuristic strategies, parameter fixing, graph neural networks, and parameter transferability across graphs.

- Classical optimization algorithms such as **gradient-based** and **gradient-free** methods have been used to explore the QAOA parameter space. Despite gradient-based approaches are widely used to optimize the parameters of the QAOA, they can have high computational times when compared to gradient-free methods.

- **Machine learning** techniques may automatically learn the ideal parameter patterns by training models to find optimal parameters for QAOA. For instance, reinforcement learning and meta-learning have been applied to improve parameter optimization in QAOA , often resulting in faster convergence and better performance.

- Strategies like layerwise learning and warm-start techniques have been proposed to mitigate **barren plateaus**. Furthermore, the ability to transfer and reuse optimal parameters across different problem instances can potentially enhance the optimization process.

- Using **parameter symmetries** can significantly enhance the optimization process in QAOA by reducing degeneracies and enabling more efficient performance.

EFFICIENCY AND PERFORMANCE

- Several works have already highlighted QAOA's speedup potential compared to classical algorithms

- Despite this, quantum speedup with QAOA on near-term devices is unlikely:
  - Difficulties in optimizing the variational parameters in large QAOA circuits;
  - Substantial overheads of error correction, which slows down the algorithm significantly;
  - Problem structure should be exploited to achieve high-order speedups

- Many approaches to improve QAOA's runtime efficiency have been proposed, primarily targeting the parameter optimization process, modifying the ansatz, optimizing gate operations, reducing samplings, etc.

- There are some theoretical solution guarantees, that is, lower bounds on QAOA's performance in asymptotic limits when applied to various problems and compared to state-of-the-art classical algorithms.

- Empirical studies have also been conducted to benchmark QAOA's performance against classical solvers, using both simulators and real hardware. These factors have significant impact on QAOA's quality of solution:
  - Circuit depth and entanglement;
  - Parameter optimization;
  - Graph properties;
  - …

HARDWARE CONSIDERATIONS

- Both local and correlated errors pose significant challenges to QAOA's scalability and performance.
- In case of **local errors**: the QAOA performance suffers from an exponential degradation with an increasing noise strength.
  - Exponential time complexity to make QAOA effective
  - Noise-induced barren plateaus during optimization.

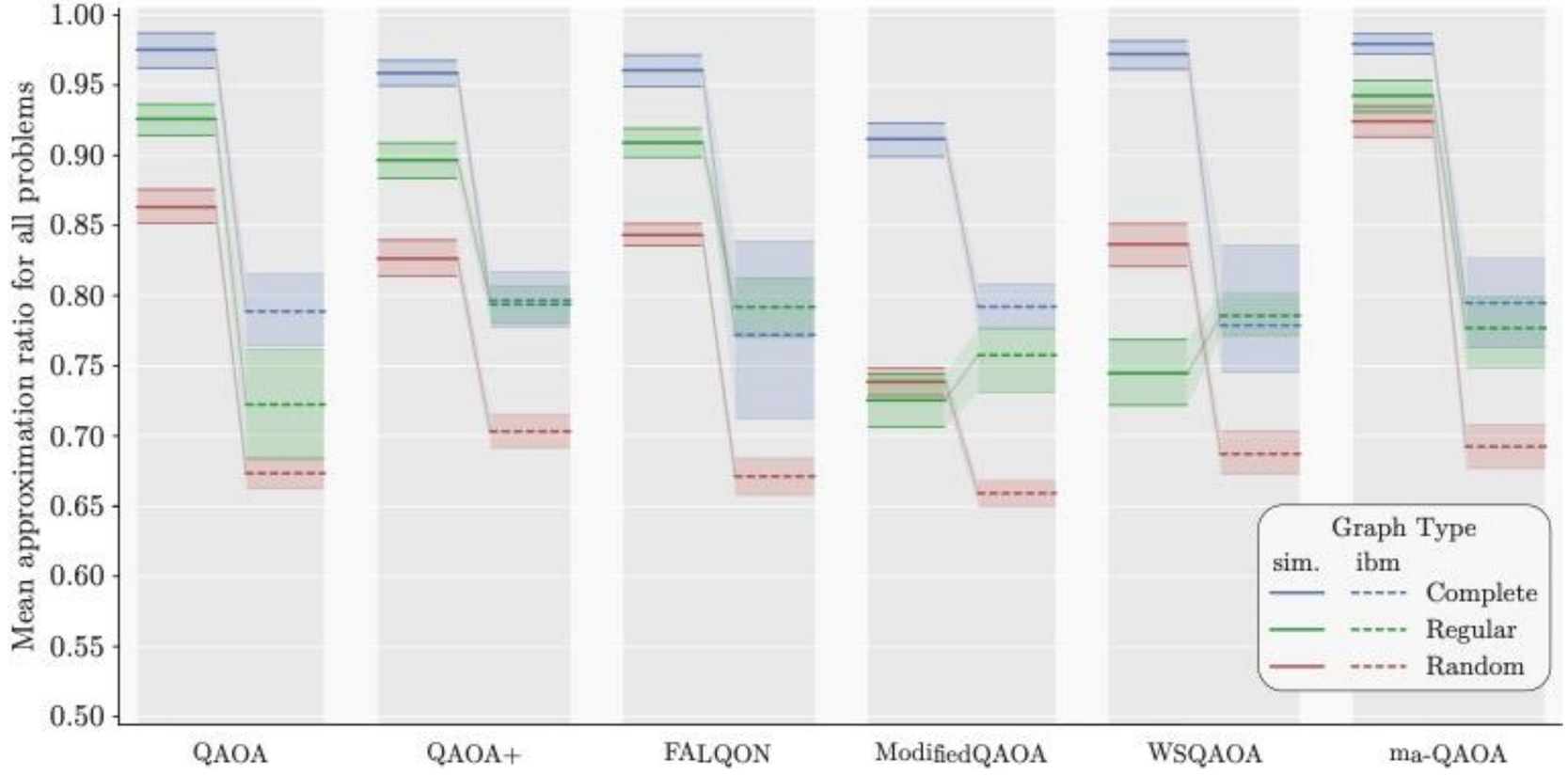
The adverse effects of **correlated errors** have also been investigated:

- crosstalk noise
- precision errors.
- coherent error induced by residual ZZ-couplings
- …

- Both theoretical and empirical studies suggest that **quantum advantage** is **unlikely** with current noise levels!
- Some **hardware-specific** approaches have been proposed to enhance QAOA's performance, focusing on leveraging platform capabilities like trapped ions, neutral atoms, superconducting qubits, and photonic quantum computers.

RESULTS ON SIMULATOR VS REAL HW

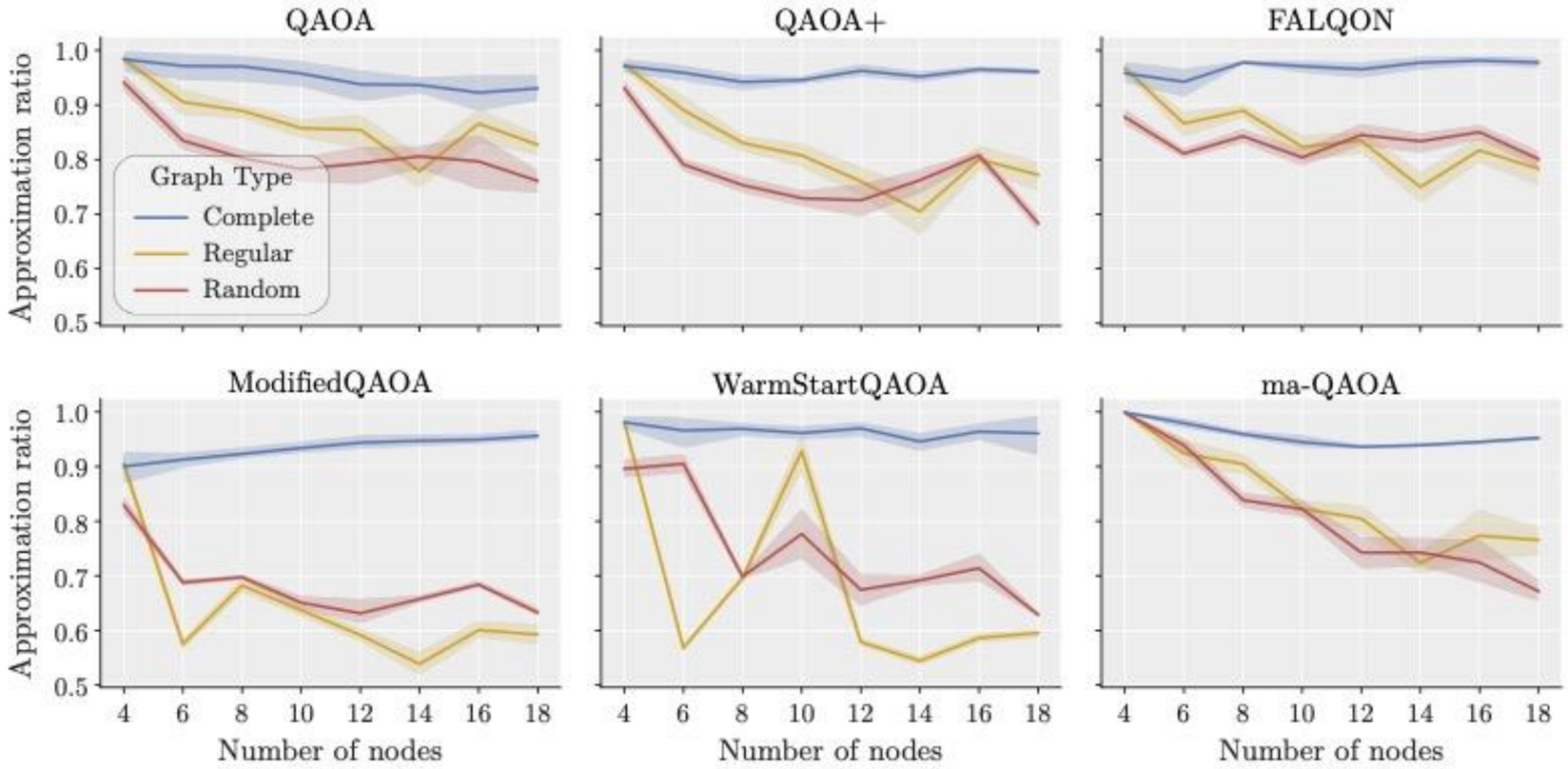
- We tested several QAOA variants on different MaxCut problems (complete, 3-regular and random graphs).
- We used Qiskit for the code implementation and simulations, and IBM Quantum systems including `ibm_oslo`, `ibm_lagos`, `ibm_nairobi` and `ibm_perth` for the real-hardware experiments.



APPROX. RATIO VS NUMBER OF NODES

- All variants demonstrate superior approximation ratios when applied to complete and regular graphs. Conversely, when applied to random graphs, the approximation ratio drops significantly.

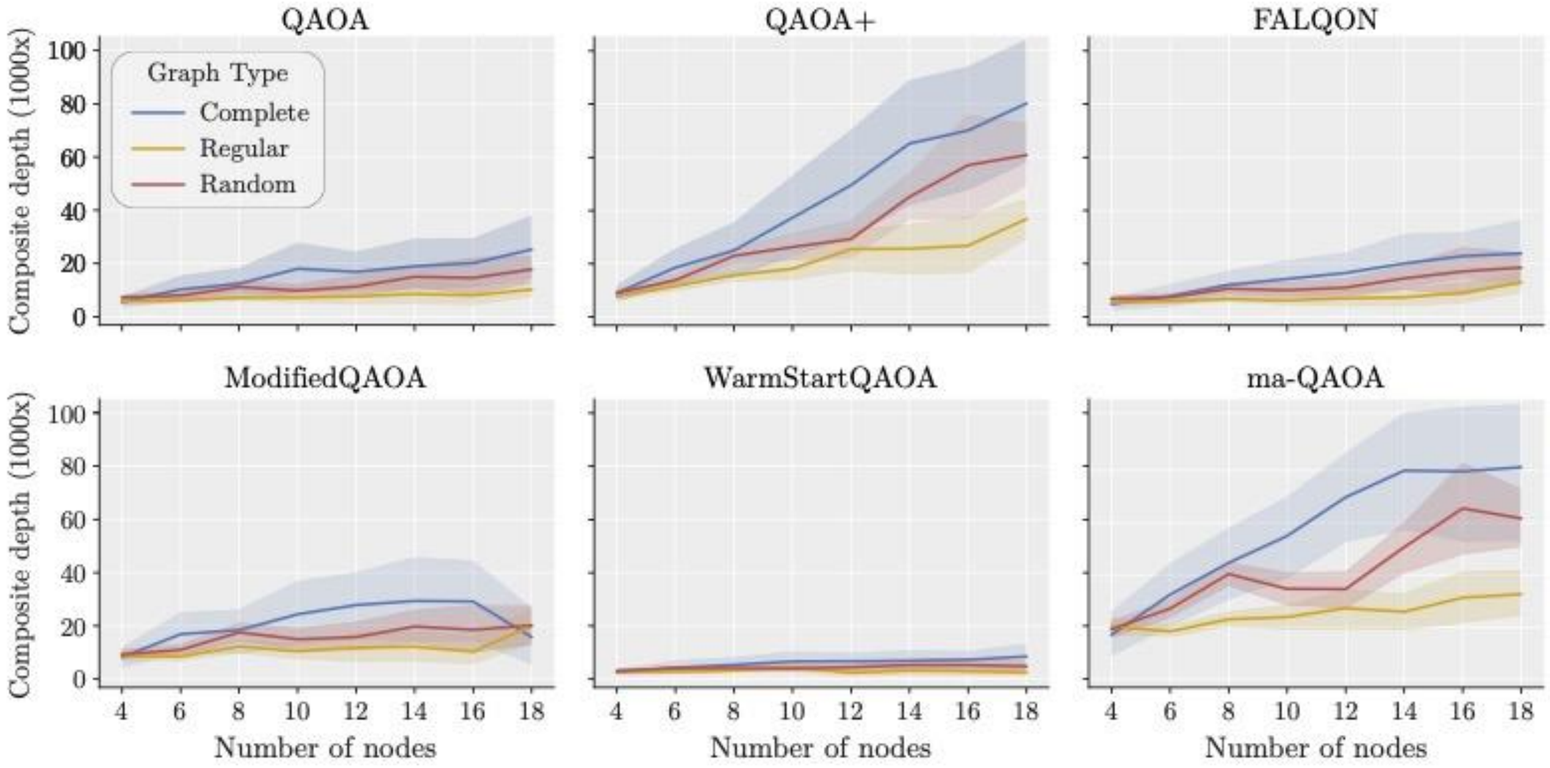
- This trend gets even more pronounced as the number of nodes increases.



EFFICIENCY VS NUMBER OF NODES

- Balancing approximation ratio and resource use: the trade-off between the mean approximation ratio and the computational resources required by each QAOA variant is a vital metric to consider.

- Some variants achieve higher approximation ratios but require more gates, have higher circuit depth, or need more circuit evaluations, resulting in increased computation time and resource usage.



OTHER METRICS AND RESULTS

- Effect of circuit layer depth:

depth (p)	1	2	3	4	5	6	7	8
QAOA	0.810	0.854	0.860	0.865	0.901	0.892	0.917	0.909
QAOA+	0.789	0.856	0.855	0.858	0.855	0.853	0.838	0.889
FALQON	0.807	0.849	0.865	0.890	0.890	0.885	0.899	-
ModifiedQAOA	0.684	0.746	0.726	0.734	0.733	0.724	0.725	0.735
WS-QAOA	0.785	0.756	0.770	0.806	0.804	0.796	0.790	0.817
ma-QAOA	0.846	0.990	0.999	0.994	0.998	0.948	0.960	0.992

- Noise-free simulations vs real quantum hardware:

	QAOA	**QAOA+**	**FALQON**	**ModifiedQAOA**	**WS-QAOA**	**ma-QAOA**
Simulation	0.899 ± 0.008	0.867 ± 0.009	0.881 ± 0.007	0.743 ± 0.011	0.802 ± 0.014	0.936 ± 0.008
Real hardware	0.702 ± 0.019	0.710 ± 0.014	0.716 ± 0.020	0.705 ± 0.016	0.727 ± 0.016	0.732 ± 0.016

- Proximity of optimal parameters to initial random guess:

	QAOA	**QAOA+**	**FALQON**	**ModifiedQAOA**	**WS-QAOA**	**ma-QAOA**
Mean cosine similarity	0.988	0.963	0.511	0.986	0.978	0.976
Standard Deviation	0.009	0.037	0.257	0.016	0.067	0.015

POTENTIAL APPLICATIONS AND USE CASES

- Despite the difficulty in implementing QAOA on modern quantum devices, there are already several practical applications which demonstrate the relevance of this algorithm in the industry.

- **General optimization problems**:
  - Constrained-optimization problems (QAOAnsatz, XY-QAOA);
  - Vertex-*k*-coloring for scheduling, mobile radio frequency assignment, register allocation, etc… (RQAOA);
  - Maximum *k*-Vertex Cover for network security, social network analysis, etc… (QAOAnsatz);
  - Tail Assignment Problem for aircraft assignment and Vehichle Routing Problem (QAOA).

- **Financial problems**:
  - Portfolio optimization (QAOA, QAOAnsatz, DC-QAOA).

- **Computer vision**:
  - Partially occluded object detection, QUBO (QAOA).

- **Machine Learning**:
  - Accelerate training of machine learning models (QAOA).

THE QUANTUM ADVANTAGE OF QAOA

- QAOA has **potential** for **quantum advantage** over classical algorithms under certain conditions.

- **Computational runtime efficiency**:
  - QAOA has demonstrated superiority in solving the MaxCut problem on dense graphs, offers exponential acceleration for large Minimum Vertex Cover problems, and achieves a superlinear quantum speedup compared to Simulated Annealing for the Maximum Independent Set problem on specific graph instances.
  - QAOA has shown potential in other areas, such as unstructured search problems and Quantum Linear System Problems, where it outperforms various classical and quantum algorithms.

- **Quality of solution**:
  - Studies comparing QAOA to classical algorithms on various optimization problems (e.g., MaxCut, Max-kXOR, and CSPs) indicate that QAOA outperforms them in specific conditions or for certain problems.

- Despite these results, **classical algorithms can** still match or **surpass QAOA's performance** in many scenarios.

- QAOA's quantum advantage is not yet fully realized due to **various challenges**:
  - Noise and hardware limitations in NISQ devices;
  - Competitiveness of state-of-the-art classical solvers;
  - Parameter optimization.

WHEN AND HOW TO USE QAOA EFFECTIVELY

We try to answer key questions about the algorithm. In particular, we tackled practical questions like:

- Which QAOA ansatz variant should I use for my problem?
  - Some variants seem particularly suitable for problems with both hard and soft constraints
  - QAOA can benefit from leveraging problem symmetries, odd cycles, density, and other structural features.
  - QAOA works for hardware grid problems but not for problems with specific constraints or without a clear graph representation that cannot be easily represented by Hamiltonians. It also may not work well if the graph topology differs significantly from the quantum hardware's connectivity.
- How to optimize the QAOA parameters?
  - There is no universal solution for the selection of the parameter optimization method.
  - Gradient-free methods are more computationally efficient, but struggle with many layers and noise.
  - Gradient-based and Machine Learning methods may provide better results, but are computationally expensive

More details and a schematic guide can be found in the paper! 

QOSF PROJECT: FUTURE DEVELOPMENTS

1 QAOA review

Review

2 Comprehensive analysis of several QAOA variants, both simulations and real-hardware executions

Experimental analysis

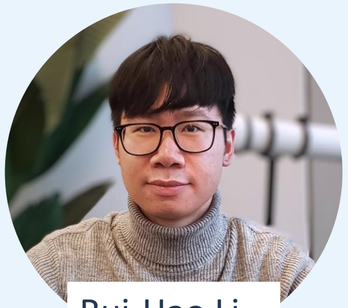
3 Release of our QAOA framework as open access
Python package – software library

Software package

Thank *you* for your attention…

…and thanks to *my colleagues* for making it possible!


Dean Brand


Rui-Hao Li


Komal Pandya


Chiao-Hui Chou


Kostas Blekos


Alessandro Summer


Andrea Ceschini