Solving MaxCut Problem using

Quantum Approximate Optimisation Algorithm (QAOA) under a proposed Hybrid Quantum-Classical Parameter Optimisation Strategy

Chee Chong Hian

SGInnovate Summation Programme
Apprentice@Entropica Labs

24th June 2020

About Company and Me

Entropica Labs

- Create Quantum Models, Algorithms and Software tools to make QC more useful.

Me

- Graduating from NTU with a Bachelor in Physics.
- Pursing a PhD at CQT at NUS (Angelakis Research Group).

Contents

MaxCut Problem

Challenge:

What is the maximum number of edges can I possibly cut on a graph in a single cut?

Rules:

A cut *must*...

1. Start and end outside of graph.
2. Not intersect itself.
3. Not cut any edge more than once.

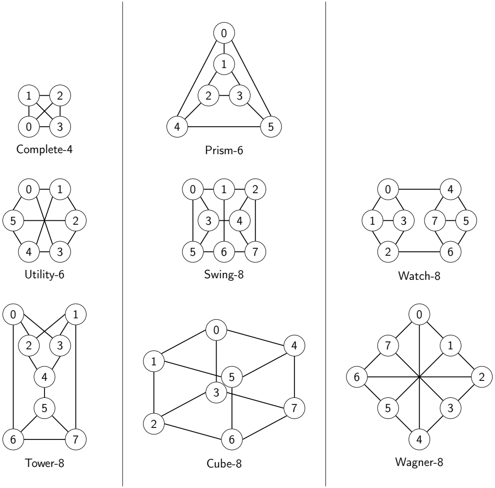
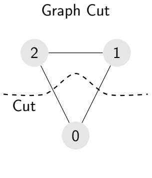


Figure 1: Want to solve all unique 3 Regular unweighted graphs up to 8 vertices.

Computational MaxCut

Binary Reformulation

- A cut splits the graph into two parts, which are identified using bits.
- Use the vertex label to create a binary string s that represents the cut.
- Define a binary function $f(s)$ to count the number of edge cuts.
  - $f(s) = \sum_{(i,j)\in \text{Edges}} \frac{1}{2}(1 - s_i s_j)$ (MaxCut Function)



Vertices	0 1 2
Binary Strings s	0 1 1
	1 0 0

Figure 2: Converting a cut into binary.

Quantum Computing

Want to use a *quantum* algorithm that finds s^* that maximises $f(s)$.

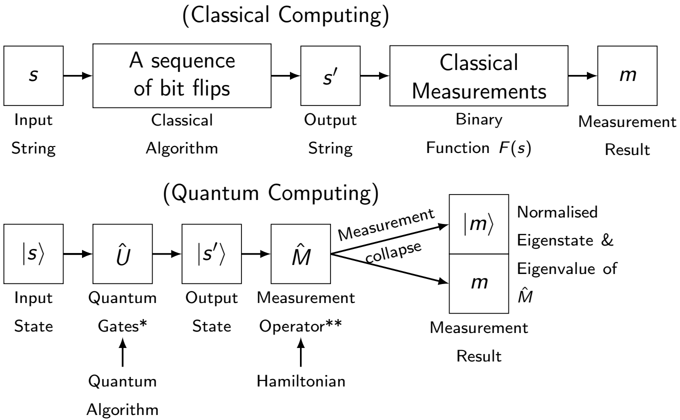


Figure 3: A basic comparison between classical and quantum computing. In mathematical physics, we call * a unitary and ** a hermitian.

Quantum Approximate Optimisation Algorithm

QAOA

- A parametrised quantum algorithm invented by Farhi et al [FGG14].
- Designed to solve MaxCut by optimising QAOA parameters $\vec{\beta}_P = (\beta_1, \ldots, \beta_P)$, $\vec{\gamma}_P = (\gamma_1, \ldots, \gamma_P)$.
- Get Output State $|\phi_P\rangle = \left[\prod_{p=1}^{P} \hat{U}_B(\beta_p)\, \hat{U}_C(\gamma_p)\right] |\psi\rangle$.

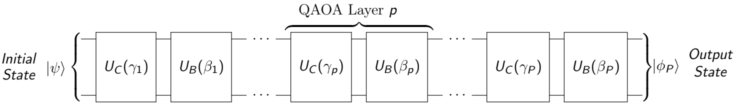
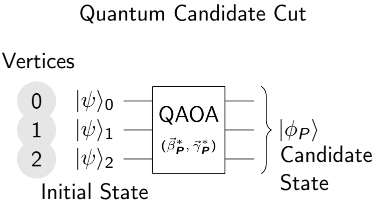


Figure 4: Quantum circuit decomposition of QAOA of depth P.

Quantum MaxCut

Quantum Reformulation

- Treat each vertex as a qubit.
- The output $|\phi_P\rangle$, generated by QAOA, is a superposition of the solution state $|s^*\rangle$ and others.
- Treat $|\phi_P\rangle$ as the candidate state for MaxCut.



$$|\phi_p\rangle = \alpha_1|s^*\rangle + \alpha_2|Others\rangle$$
$$\text{where} \quad |\alpha_1|^2 \gg |\alpha_2|^2$$

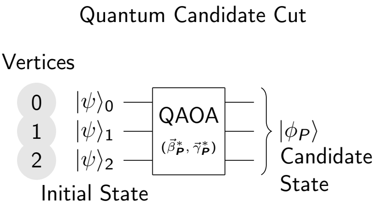
Figure 5: Quantum mechanical attempt to solve MaxCut.

Want to optimise $\vec{\beta}^*_P, \vec{\gamma}^*_P$ to maximise probability of $|s^*\rangle$

Quantum Approximate Optimisation Algorithm (QAOA)

Quantum Reformulation (Con't)

- Define a Measurement Ops $\hat{C}$ to count the edge cuts of the candidate $|\phi_P\rangle$.
    - $\hat{C} = \sum_{(i,j)\in \mathrm{Edges}} -\frac{1}{2}\left(\mathbb{I} - \hat{Z}_i \hat{Z}_j\right)$ (MaxCut Hamiltonian)
- But, a quantum measurement of $\hat{C}$ will collapse $|\phi_P\rangle$.
    - Estimating an average is better.
        - Find $\left\langle \hat{C} \right\rangle_p = \langle \phi_P | \, \hat{C} \, | \phi_P \rangle$ (Expectation of $\hat{C}$)



$$|\phi_p\rangle = \alpha_1 |s^*\rangle + \alpha_2 |Others\rangle$$
$$\text{where} \quad |\alpha_1|^2 \gg |\alpha_2|^2$$
$$\text{and} \quad \langle \phi_p | \, \hat{C} \, | \phi_p \rangle = \text{optimal}$$

Maximising probability of $|s^*\rangle \Rightarrow$ Optimising $\langle \phi_P | \, \hat{C} \, | \phi_P \rangle$

Hybrid Q-C Optimisation Strategy

Goal: Find the optimal $\left(\vec{\beta}^*, \vec{\gamma}^*\right)$ that optimise $\left\langle \hat{C} \right\rangle_p$ to get the best $|\phi_P\rangle$ that solves MaxCut.

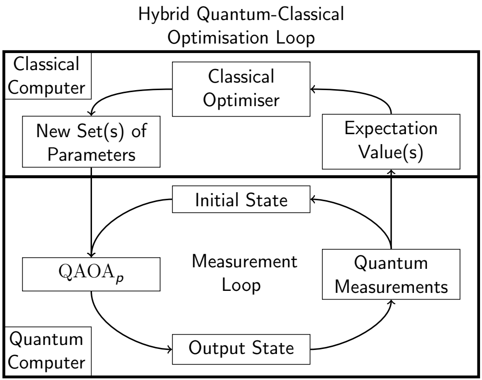


Figure 6: Hybrid Quantum-Classical Optimisation Strategy Framework

Greedy-Based Strategies

Greedy Heuristic

- Optimise QAOA parameters (β_p, γ_p) layer by layer.

Project Focus: Greedy Strategies

1. Standard Greedy Search
2. Greedy Subsearch
3. Greedy Newton (Proposed New Strategy)

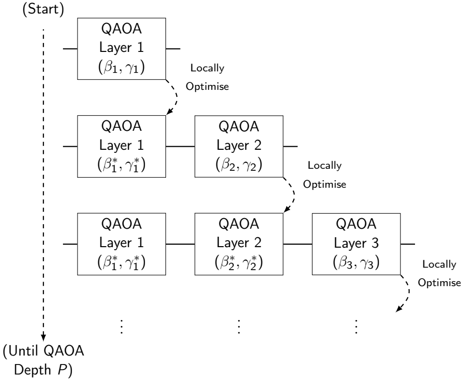


Figure 7: A general greedy-based strategy.

Standard Greedy Search

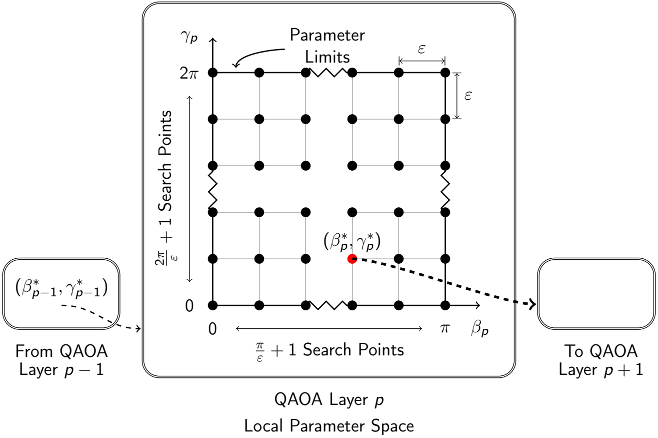


Figure 8: Standard Greedy Search hybrid strategy optimizer for a quality QAOA parameter optimisation. Note: ε is the division size.

Greedy Subsearch

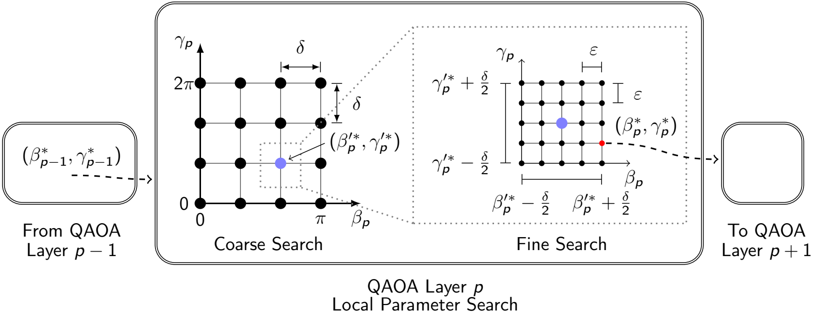


Figure 9: Greedy Subsearch hybrid strategy optimizer for a faster QAOA parameter optimisation. Note: $\delta = \sqrt{\pi\varepsilon} > \varepsilon$ is coarse division size.

Proposed Strategy

Greedy Newton

Adapts Greedy Subsearch by incorporating Newton's Descent.

Newton's Descent

A second-order 'Gradient Descent': Finds the local optimal $\vec{y}^* = (\beta_p^*, \gamma_p^*)$ of $\left\langle \hat{C} \right\rangle_p$.

1. Start with any initial $\vec{y}_0$.
2. If Newton's conditions are met, get the next $\vec{y}_{n+1} = \vec{y}_n - \alpha\,[\text{gradient}]\,[\text{Hessian}]^{-1}$.
3. Stops when $\underbrace{|\vec{y}_{n+1} - \vec{y}_n|}_{\text{Parameter Difference}} < \vec{\epsilon}$.

Note: Step Size: α, Gradient $(\partial\beta_p, \partial\gamma_p)$

Hessian $\begin{pmatrix} \partial\beta_p^2 & \partial\gamma\partial\beta \\ \partial\beta\partial\gamma & \partial\gamma_p^2 \end{pmatrix}$

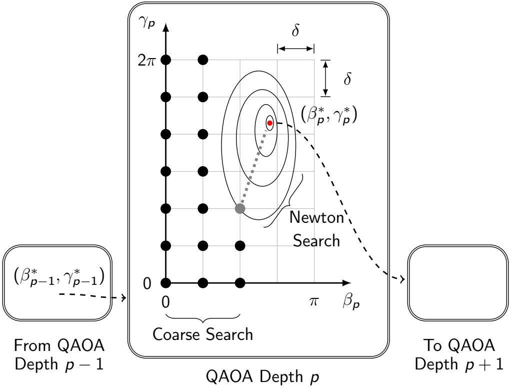


Figure 10: Greedy Newton Optimiser.

Greedy Newton

Difficulties

- How to get the gradient and Hessian, $\nabla^{1,2}\left\langle \hat{C} \right\rangle_p$ of QAOA layer p using Quantum Computers?

Solution by Others

- Use Quantum Gradient / Hessian Algorithms. [Jor04; Bul05; Reb+16]
- Consider Quasi-Newton Descent by approximating $\nabla^{1,2}\left\langle \hat{C} \right\rangle_p$ using Finite Difference Method. [GS17; Zho+18]
- Switch to ML gradient descent instead. [Per+13; Cro18; Pag+19]

My Solution

- Differentiate the Pauli Decomposition of $\left\langle \hat{C} \right\rangle_p$ w.r.t β_p, γ_p.

What & Why Pauli Decomposition?

Pauli Decomposition

- Re-expressing any measurement ops $\hat{M}$ in terms of Pauli operators.

$$\hat{M} = \sum_{i} \alpha_{1i}\sigma_i + \sum_{i,j} \alpha_{2ij}\sigma_i\sigma_j + \sum_{i,j,k} \alpha_{3ijk} \underbrace{\sigma_i\sigma_j\sigma_k}_{\substack{\text{Pauli} \\ \text{Compositions}}} + \dots$$

- Pauli Operators σ_i (Eg. $\hat{X}$, $\hat{Y}$, $\hat{Z}$) are special measurements ops that typically used in today's quantum computers to get measurement results.

My Solution Steps (Con't)

1. $\left\langle \hat{C} \right\rangle_p \underset{\text{Expand}}{=} \langle \phi_{p-1} | \underbrace{\hat{U}_C^\dagger(\gamma_p)\,\hat{U}_B^\dagger(\beta_p)\,\hat{C}\,\hat{U}_B(\beta_p)\,\hat{U}_C(\gamma_p)}_{\text{Pauli Decompose this...}} | \phi_{p-1} \rangle$

2. Differentiate $\left\langle \hat{C} \right\rangle_p$ w.r.t local parameters β_p, γ_p.

Full Pauli Decomposition

Full Pauli Decomposition Equations of $\left\langle \hat{C} \right\rangle_p$ of QAOA layer p

$$\left\langle \hat{C} \right\rangle_p = \sum_{(i,j)\in \mathrm{E}} \frac{1}{2} \left(A_1 \cos^2 2\beta_p + (A_2 + A_3) \frac{1}{2} \sin 4\beta_p + A_4 \sin^2 2\beta_p - 1 \right)$$

where

$$A_1 = \langle \phi_{p-1} | \hat{Z}_i \hat{Z}_j | \phi_{p-1} \rangle$$

$$A_2 = \sum_{k=0}^{\ell_{ij}} \sum_{\lambda \in \Lambda_k^{\ell_{ij}}} \left[(\cos \gamma_p)^{\ell_{ij}-k} (-i \sin \gamma_p)^k \left(\cos \gamma_p \langle y_i z_j z_i^k \rangle_\lambda + \sin \gamma_p \langle x_i z_i^k \rangle_\lambda \right) \right]$$

$$A_3 = \sum_{k=0}^{\xi_{ij}} \sum_{\lambda \in \Lambda_k^{\xi_{ij}}} \left[(\cos \gamma_p)^{\xi_{ij}-k} (-i \sin \gamma_p)^k \left(\cos \gamma_p \left\langle z_i y_j z_j^k \right\rangle_\lambda + \sin \gamma_p \left\langle x_j z_j^k \right\rangle_\lambda \right) \right]$$

$$A_4 = \sum_{n=0}^{\xi_{ij}} \sum_{m=0}^{\ell_{ij}} \sum_{\lambda_1 \in \Lambda_m^{\ell_{ij}}} \sum_{\lambda_2 \in \Lambda_n^{\xi_{ij}}} \left[(\cos \gamma_p)^{\ell_{ij}+\xi_{ij}-m-n} (-i \sin \gamma_p)^{m+n} \left\langle y_i y_j z_i^m z_j^n \right\rangle_{\lambda_1, \lambda_2} \right]$$

- Notice that β_p and γ_p are outside of the expectation $\langle \_ \rangle$.
- Thus, the differentiation to get $\nabla^{1,2} \left\langle \hat{C} \right\rangle_p$ is straightforward, though tedious.

Calculation Flowchart

My Solution Steps (Con't)

3. Have equations $\nabla^{0,1,2}\left\langle \hat{C} \right\rangle_p$ in terms of β_p, γ_p and $\langle \phi_{p-1} | \, \sigma \, | \phi_{p-1} \rangle$.
4. Estimate $\langle \phi_{p-1} | \, \sigma \, | \phi_{p-1} \rangle$ using Quantum Computer with QAOA_{p-1}.
5. Load $\langle \phi_{p-1} | \, \sigma \, | \phi_{p-1} \rangle$ into Classical Computer.
6. Substitute (β_p, γ_p) to calculate $\nabla^{0,1,2}\left\langle \hat{C} \right\rangle_p$.

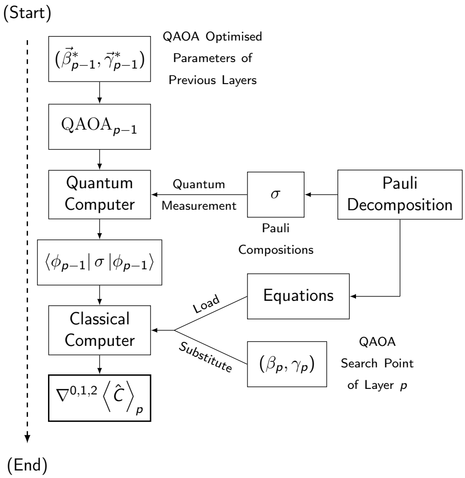


Figure 11: Proposed Calculation of $\nabla^{0,1,2}\left\langle \hat{C} \right\rangle_p$ Flowchart.

Comparing the Performance of Strategies

KPIs	Actual Measurement
Total Computation Time	Overall time to solve the MaxCut problem.
Total Quantum Computer Calls	The number of independent executions of QAOA.
Approximation Ratio	The ratio of the approximated to the actual MaxCut value.

Table 1: Key Performance Indicators (KPIs) for the Strategies.

Numerical Experiment Setup

Total Depth: 10
Initial State: $|+\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$
Parameter Domain: $\beta_p \in [0, \pi]\,,\ \gamma_p \in [0, 2\pi]$
Quantum Computer: Statevector Simulator (Matrix Multiplication)
Unweighted Graphs: 1 Regular, Cycle, Path, Complete & 3 Regular ($n \leq 8$ vertices)

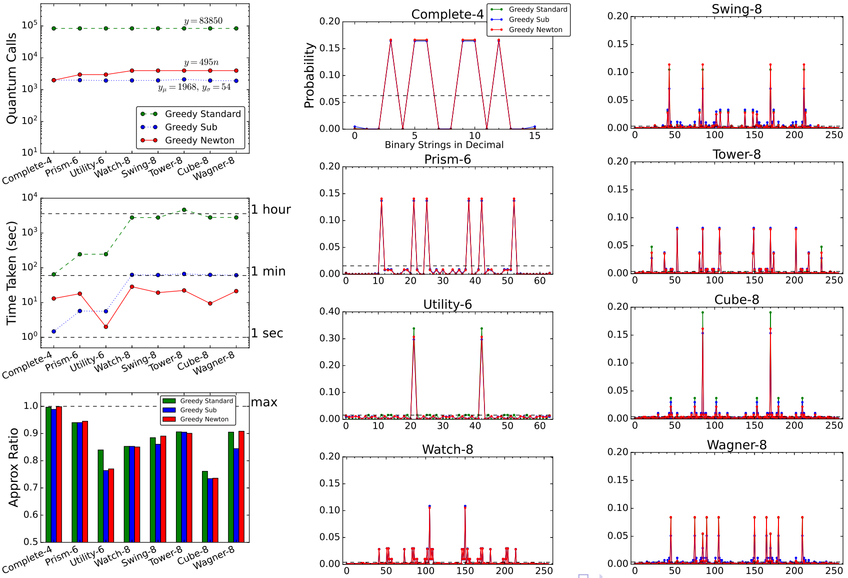
Standard Greedy Search
Division size:
$\varepsilon = \frac{\pi}{64}$

Greedy Subsearch
Division sizes:
$\delta = \sqrt{\pi\varepsilon} = \frac{\pi}{8}$ (Coarse)
$\varepsilon = \frac{\pi}{64}$ (Fine)

Greedy Newton
Division size: $\delta = \frac{\pi}{8}$ (Coarse)
Max Iteration: 100
Step Size: 0.35
Start Condition:
$\mathrm{Det}\left(\nabla^2 \left\langle \hat{C} \right\rangle_p\right) > 10^{-9}$
Stop Condition:
$\Delta\beta_p$ and $\Delta\gamma_p < 0.001$ rad

Numerical Results

3 Regular



3 Regular MaxCut Solutions

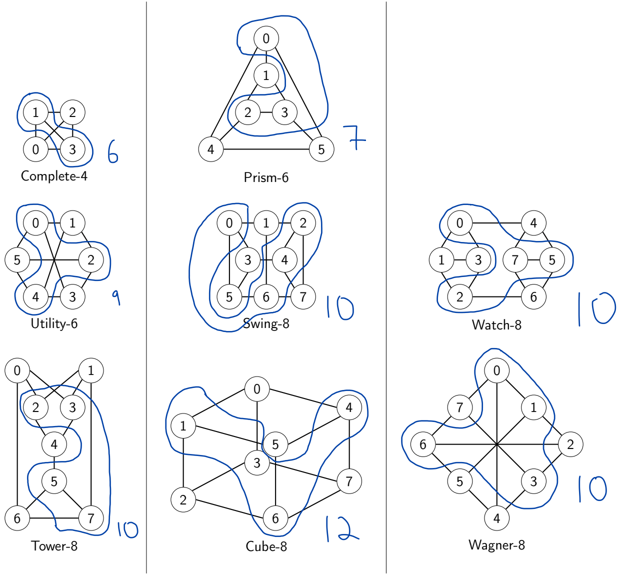


Figure 12: MaxCut Solution of 3 Regular Graph.

Performance Summary

Strategies	Quantum Calls	Simulated Computational Time (Exponential in n)	Solution Quality
Standard Greedy Search	Quadratic in $\frac{1}{\varepsilon}$	Longest	Best
Greedy Subsearch	Linear in $\frac{1}{\varepsilon}$	Short	Good
Greedy Newton	Linear in n (Fixed g regularity) Exponential in n (For $g = n - 1$)	Shortest* (For small $n \leq 8,\ g \leq 3$) Long** (For $g = n - 1$)	Better

Table 2: Performance Comparison. * only if Newton's Method is successful, or ** if otherwise.

Future Research

Project Extensions

- Use *real* quantum computers.
- Apply the Greedy Newton to solve weighted MaxCut (I have the equations ready).
- Compare performance with other non-greedy strategies.

Project Enhancements

- Use more efficient Tensor Contraction simulation [Fri+17] to replace Statevector.

Closing Thoughts

Greedy Newton is designed to be a *better* greedy strategy.

Given the right conditions, it should be competitive with other strategies that are not greedy.

However, Greedy Newton is unable to overcome the inherent limitations of its greedy nature.

Nonetheless, I believe its practicality and ease of implementation will confidently trumps over such inconvenient limitations.

Thank You

Thank you for listening to my presentation.

My project thesis will be up online soon.

Acknowledgements:

Mentor: Ewan Munro

Supervisor: Tommaso Demaire

Funded by: SGInnvoate & EntropicaLabs (under Summation Programme)

Supported by: Professor Mile Gu (SPMS, NTU) & Dr Jayne Thompson (CQT, NUS)

Citations I

David Bulger. "Quantum computational gradient estimation". In: (2005). arXiv: 0507109 [quant-ph]. URL: http://arxiv.org/abs/quant-ph/0507109.

Gavin E Crooks. "Performance of the Quantum Approximate Optimization Algorithm on the Maximum Cut Problem". In: (2018), pp. 15–17. arXiv: 1811.08419. URL: http://arxiv.org/abs/1811.08419.

Edward Farhi, Jeffrey Goldstone and Sam Gutmann. "A Quantum Approximate Optimization Algorithm". In: (2014). arXiv: 1411.4028. URL: http://arxiv.org/abs/1411.4028.

Citations II

E Schuyler Fried et al. "qTorch: The Quantum Tensor Contraction Handler". In: (2017). DOI: 10.1371/journal.pone.0208510. arXiv: 1709.03636. URL: http://arxiv.org/abs/1709.03636.

Gian Giacomo Guerreschi and Mikhail Smelyanskiy. "Practical optimization for hybrid quantum-classical algorithms". In: (2017), pp. 1–25. arXiv: 1701.01450. URL: http://arxiv.org/abs/1701.01450.

Stephen P Jordan. "Fast quantum algorithm for numerical gradient estimation". In: 02139 (2004), pp. 1–4. DOI: 10.1103/PhysRevLett.95.050501. arXiv: 0405146 [quant-ph]. URL: http://arxiv.org/abs/quant-ph/0405146.

Citations III

G. Pagano et al. "Quantum Approximate Optimization of the Long-Range Ising Model with a Trapped-Ion Quantum Simulator". In: 1.1 (2019), pp. 1–17. arXiv: 1906.02700. URL: http://arxiv.org/abs/1906.02700.

Alberto Peruzzo et al. "A variational eigenvalue solver on a quantum processor". In: 1.2 (2013), pp. 1–10. DOI: 10.1038/ncomms5213. arXiv: 1304.3061. URL: http://arxiv.org/abs/1304.3061.

Patrick Rebentrost et al. "Quantum gradient descent and Newton's method for constrained polynomial optimization". In: (2016). arXiv: 1612.01789. URL: http://arxiv.org/abs/1612.01789.

Citations IV

Leo Zhou et al. "Quantum Approximate Optimization Algorithm: Performance, Mechanism, and Implementation on Near-Term Devices". In: (2018). arXiv: 1812.01041. URL: http://arxiv.org/abs/1812.01041.