HamilToniQ: Comprehensive Optimization and Benchmarking for Mixer Hamiltonian in QAOA with Error Mitigation

Elif Cetiner, Jessica Link
TU Munchen
`{elif.cetiner, jessica.link}@tum.de`

Kuan-Cheng (Louis) Chen, Xiaotian (Felix) Xu
Imperial College London
`{kc2816, xx719}@ic.ac.uk`

Tomasz Andrzejewski, Pascal Windhager
TU Wien Atominstitut
`{tomasz.andrzejewski@tuwien.ac.at, pascal.windhager99@gmail.com}`

Abstract

This paper presents a comprehensive study on the Quantum Approximate Optimization Algorithm (QAOA), a hybrid quantum-classical algorithm proposed for solving combinatorial optimization problems on near-term, noisy intermediate-scale quantum (NISQ) devices. We investigate the performance of QAOA on three distinct problem types: the Hardware Grid problem, the Three Regular problems, and the Sherrington-Kirkpatrick (SK) model problem. Each problem type presents unique challenges and characteristics, providing a diverse range of conditions for evaluating QAOA's performance.

We focus on the impact of the choice of mixer Hamiltonian on QAOA's performance, specifically the XY mixer and its three versions: the ring mixer, the parity mixer, and the full mixer. Our results indicate that the ring mixer exhibits superior performance, achieving the best results and converging to a minimum faster than the X-model. Furthermore, we delve into the critical aspect of error mitigation in quantum computing. We employ the quasiprobability method for mitigating measurement errors and apply it in the context of the QAOA algorithm. Our findings suggest that error mitigation techniques can effectively reduce the impact of errors during runtime sessions, enhancing the performance and reliability of quantum computations.

In terms of practical implementation, we develop a tailored strategy for each specific coupling map to optimize the circuit depth in quantum circuit design. We approach the depth optimization problem as a variation of the graph colouring problem and employ a greedy colouring algorithm to provide a valid solution that satisfies the constraints of the problem.

In conclusion, our study provides valuable insights into the performance of QAOA and its implementation on near-term quantum devices. It contributes to the understanding of QAOA and guides future research in this area, facilitating the practical use of quantum computing for optimization problems.

***K*eywords** QAOA, Error Mitigation, Benchmarking, Mixer Hamiltonian, Graph Theory, Open-source Software

1 Introduction

Quantum computing, with its potential to solve certain problems exponentially faster than classical computers, has been the subject of intense research and development. One promising application of quantum computing lies in optimization problems, which are ubiquitous in various fields such as logistics, finance, and machine learning[1], [2]. However, solving these problems on a quantum computer poses significant challenges due to the nascent stage of quantum hardware and the complexity of quantum algorithms.

The Quantum Approximate Optimization Algorithm (QAOA) is a hybrid quantum-classical algorithm that has been proposed as a practical method to solve combinatorial optimization problems on near-term, noisy intermediate-scale quantum (NISQ) devices [3]. QAOA operates by approximating the ground state of a problem Hamiltonian H_P, which encodes the optimization problem to be solved [4]. The QAOA ansatz state is given by:

$$|\psi(\beta,\gamma)\rangle = U(\beta,\gamma)|s\rangle, \tag{1}$$

where $|\psi\rangle$ is the initial state, and $U(\beta,\gamma)$ is the QAOA unitary. Despite its potential, the performance of QAOA and the mechanisms underlying its success are not yet fully understood.

In this paper, we present a comprehensive study of QAOA, focusing on its performance, the mechanism of its operation, and its implementation on near-term quantum devices. We provide a detailed analysis of the algorithm's performance on different problem instances and investigate the role of quantum interference in the success of QAOA. Furthermore, we discuss practical considerations for implementing QAOA on current quantum hardware, including the impact of noise and the choice of initial states.

Our findings contribute to the understanding of QAOA and provide valuable insights for its application on near-term quantum devices. We hope that our work will guide future research in this area and facilitate the practical use of quantum computing for optimization problems.

1.1 Description of QAOA

The QAOA algorithm consists of two components; the quantum evolution of a state, and the classical optimizer. The evolution of the state consists of the repeated application of two types of operators to the ground state: the mixer Hamiltonian and the phase separation operator. The final, parametrized state is then given by: [5, p.2]

$$|\psi_{\gamma,\beta}\rangle = \hat{U}_M(\beta_p)e^{-i\gamma_p\hat{F}}...\hat{U}_M(\beta_2)e^{-i\gamma_p\hat{F}}\hat{U}_M(\beta_1)e^{-i\gamma_1\hat{F}_A}\left|\psi_)\right\rangle_M, \tag{2}$$

where the unitary operators $\hat{U}_M(\beta_p)$ are the mixing matrices and the $e^{-i\gamma\hat{F}}$ are the phase shifting factors, $\hat{F}$ represents the cost function encoded as a Hamiltonian. The circuit is therefore a Trotterized time evolution, shifting the ground state from an eigenstate of $\hat{M}$ to $\hat{F}$ [5]. The standard implementation of the mixing matrices consists of a series of $\hat{X}$ gates applied to each qubit given by $e^{i\beta\sum_{i=1}^{n}\hat{X}_i}$. The improved versions makes use of the system's underlying symmetries in order to simplify the computation of the expectation value. These can take on various forms, as described in section 2.2. The expectation value of the cost function based on this final state then serves as an input to a classical optimizer, which chooses a new parameter for another time evolution until the expectation value is minimized.

In the MaxCut problem, the cost function takes on the form [6]

$$C(x) = \sum_{i,j} w_{ij} \sum_{p} x_{i,p}x_{j,p+1} \tag{3}$$

When implemented as a Hamiltonian, this maps to minimizing the energy in the Ising model with the Hamiltonian [7]:

$$\hat{H} = \sum_i w_i Z_i + \sum_{i<j} w_{ij} Z_i Z_j. \tag{4}$$

2 Implementation

2.1 Problem Formulation

In our comprehensive study of the Quantum Approximate Optimization Algorithm (QAOA), we focus on three distinct problem types, each presenting unique challenges and characteristics. These problem types are the Hardware Grid problem, the Three Regular problem, and the Sherrington-Kirkpatrick (SK) model problem.

2.1.1 Hardware Grid Problem

This problem type is inspired by the physical layout of qubits in quantum hardware. The qubits are arranged in a two-dimensional grid, and interactions are allowed between neighboring qubits. This problem type is particularly relevant for near-term quantum devices, as it mirrors the connectivity constraints of actual quantum hardware.

```
import networkx as nx
import matplotlib.pyplot as plt

# Create a 3x3 grid graph
grid_graph = nx.grid_2d_graph(3, 3)
plt.figure(figsize=(4,4))
plt.title("Hardware Grid Problem")
nx.draw(grid_graph, with_labels=True)
plt.show()
```

2.1.2 Three Regular Problem

In this problem type, each node in the graph is connected to exactly three others, forming a '3-regular' graph. This problem type is interesting because it represents a balance between the highly constrained hardware grid problem and the fully connected SK model problem. It allows us to explore how the connectivity of the problem graph impacts the performance of QAOA.

```
# Create a 3-regular graph with 6 nodes
three_regular_graph = nx.random_regular_graph(d=3, n=6)
plt.figure(figsize=(4,4))
plt.title("Three Regular Problem")
nx.draw(three_regular_graph, with_labels=True)
plt.show()
```

2.1.3 Sherrington-Kirkpatrick (SK) Model Problem

The SK model is a well-known model in statistical mechanics, representing a system of spins with random interactions. In the context of QAOA, the SK model problem is a fully connected graph, where each node (or spin) can interact with every other. This problem type is computationally challenging due to its high connectivity and randomness, providing a stringent test for the performance of QAOA.

```
# Create a fully connected graph (SK model) with 5 nodes
sk_graph = nx.complete_graph(5)
plt.figure(figsize=(4,4))
plt.title("SK Model Problem")
nx.draw(sk_graph, with_labels=True)
plt.show()
```

By studying these diverse problem types, we aim to gain a deeper understanding of the performance of QAOA under different conditions and constraints. Our findings will shed light on the practical considerations for implementing QAOA on near-term quantum devices to guide future research in this area.

2.2 XY-Mixer Formulation

In the Quantum Approximate Optimization Algorithm (QAOA), the choice of mixer Hamiltonian can significantly impact the algorithm's performance. In this study, we focus on the XY mixer, which has shown promise in various applications. We consider three different versions of the XY mixer: the ring mixer, the parity mixer, and the full mixer.

2.2.1 Ring Mixer

The ring mixer applies the XY operation only to neighbouring pairs of qubits, effectively forming a ring. This mixer is particularly relevant for problems with a circular or cyclic structure.

```
# XY-Mixer
def xy_mixer_ring(qc, n, b):
    for i in range(n-1):
        qc.append(XXPlusYYGate(2*b, 0), [i, i+1])
    qc.append(XXPlusYYGate(2*b, 0), [0, n-1])
    return qc
```

2.2.2 Parity Mixer

The parity mixer applies the XY operation to pairs of qubits that have the same parity, i.e., their indices sum to an even number. This mixer can be useful for problems with a parity or symmetry structure.

```
# Parity-Mixer
def xy_mixer_parity(qc, n, b):
    for i in range(n-1)[::2]:
        qc.append(XXPlusYYGate(2 * b), [i, i + 1])
    for i in range(n-1)[1::2]:
        qc.append(XXPlusYYGate(2 * b), [i, i + 1])
    return qc
```

2.2.3 Full Mixer

The full mixer applies the XY operation to all pairs of qubits, resulting in a fully connected mixer. This mixer is the most general and can be used for any problem, but it may not exploit specific problem structures.

```
# Full-Mixer
def xy_mixer_full(qc, n, b):
    for i in range(n):
        for j in range(i+1, n):
            qc.append(XXPlusYYGate(2 * b), [i, j])
    return qc
```

2.2.4 Quantum Alternate Mixer-Phase Ansatz (QAMPA)

Proposed by [8], this ansatz merges the phase separation gate with the mixer phase gate, reducing the number of CNOT gates by a factor of $\frac{3}{4}$. One step in the evolution of the wave function is then

$$\hat{U}(\beta,\gamma) = \prod_{(i,j)\in S_{full}} \exp\Big\{ i\beta(\hat{X}_i\hat{X}_j + \hat{Y}_i\hat{Y}_j) - i\gamma W_{ij}\hat{Z}_i\hat{Z}_j \Big\} \prod_{i=1}^{n} \exp\Big\{ -i\gamma w_i \hat{Z}_i \Big\} \tag{5}$$

2.3 Depth Reduction of Mixer Formulation

The optimization of circuit depth in quantum circuit design is a critical consideration in the field of quantum computing. Brutally implementing XY mixers without careful consideration can result in a higher circuit depth and an overwhelming number of SWAP gates. To overcome this challenge, it is crucial to develop a tailored strategy for each specific coupling map.

A coupling map in quantum computing represents the allowed interactions or connections between qubits in a quantum computer. It defines which pairs of qubits can directly interact with each other. It is important to consider the coupling map when designing quantum algorithms or circuits to ensure the operations align with the physical constraints of the hardware. Optimization techniques are used to make the most efficient use of the available coupling map and minimize additional operations.

Our problem revolves around determining the most efficient sequence of edge deletions in a graph representation of the circuit, aiming to minimize the overall number of steps required. However, a crucial constraint must be observed: it is not possible to delete two edges that share a common node simultaneously.

This depth optimization problem can be seen as a variation of the graph coloring problem, where the objective is to assign colors to the vertices of a graph in a way that no adjacent vertices share the same color. In the context of coupling optimization, the "colors" represent the steps in which edges are deleted, and the "vertices" correspond to the edges in the original graph. Two edges are considered "adjacent" if they share a common node.

One approach to solving this problem is by employing a greedy coloring algorithm. This algorithm assigns the smallest available color to each vertex in the given order. While the result may not yield an optimal coloring, it does provide a valid solution that satisfies the constraints of the problem.

```
def greedy_coloring(graph):
    colour = {v: 0 for v in graph.nodes()}
    for v in graph.nodes():
        used_colors = {color[u] for u in graph.neighbors(v)}
        color[v] = min(set(range(len(graph))) - used_colors)
    return colour
```

Let’s take a relatively complex coupling map as an example.

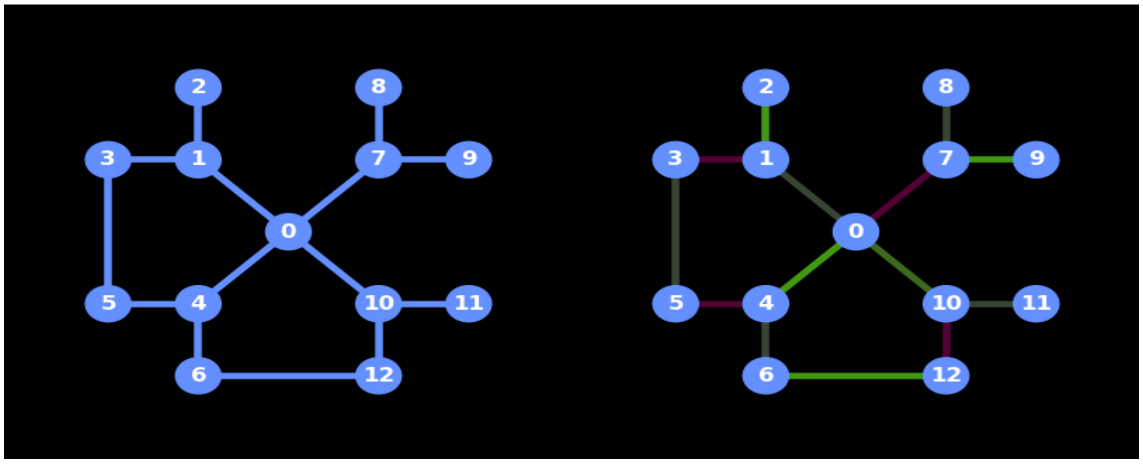


Figure 1: An illustration of the depth reduction function for mixer Hamiltonian. The left one is the original coupling map, the right one shown the implementation strategy of XY mixers.

In Figure 5, the original coupling map is depicted on the left-hand side. It is challenging for users to determine the optimal sequence for implementing XY mixers based on intuition alone. Implementing XY mixers randomly typically results in a mixer depth of 6 [9]. However, our depth reduction algorithm (`find_sub_group` in the Jupyter Notebook) reveals that the lowest achievable depth is 4. The corresponding optimized sequence is displayed on the right-hand side, where different colors represent operations at different depths.

Using this strategy, we can effectively reduce the circuit depth and achieve a depth equal to the number of nodes in the quantum circuit. This algorithm is applicable to any coupling map, regardless of its complexity or size. To further illustrate its capabilities, an example of the same algorithm applied to 10,000 qubits is shown below.

and the depth is only 4.

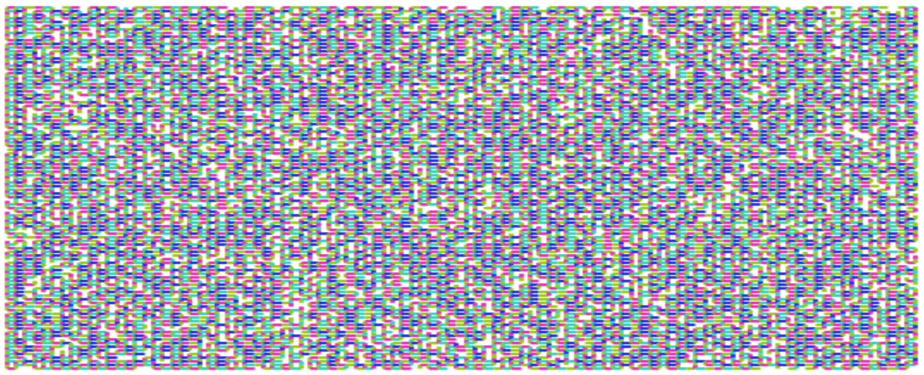


Figure 2: Another illustration of the depth reduction function on a random coupling map with 10000 qubits.

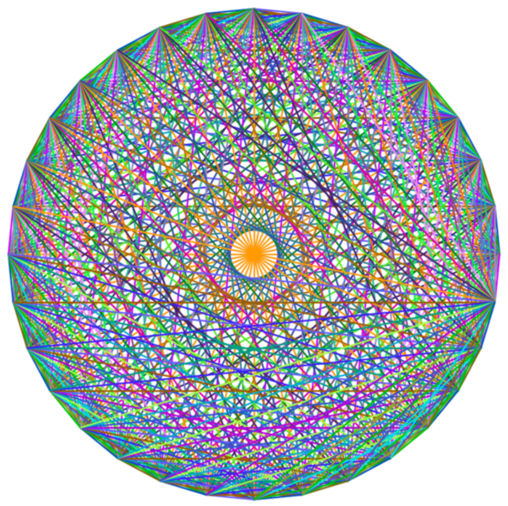


Figure 3: Another illustration of the depth reduction function on a random coupling map with SK fully-connected model.

2.4 Error Mitigation

Quantum computing, a promising technology that leverages the principles of quantum mechanics to perform computations, is currently in the noisy intermediate-scale quantum (NISQ) era. In this stage, quantum devices are subject to various types of errors that can significantly affect the accuracy of quantum computations. Therefore, error mitigation, the process of reducing the impact of these errors, is a critical aspect of quantum computing research and development.

One of the primary sources of errors in quantum computations is measurement errors. These errors occur when the state of a quantum bit (qubit) is measured, causing the qubit to collapse to a definite state. The collapse is probabilistic, and the probabilities are determined by the quantum state of the qubit. However, due to various factors such as environmental noise and imperfections in the quantum device, the measured state may not accurately reflect the true quantum state, leading to measurement errors.

In the paper "Error mitigation extends the computational reach of a noisy quantum processor" by McArdle et al., a novel technique for mitigating measurement errors called the quasiprobability method is proposed. This method involves expressing the quantum state as a linear combination of simpler states, each associated with a quasiprobability. By measuring these simpler states and recombining the results weighted by their quasiprobabilities, one can effectively mitigate the impact of measurement errors. The quasiprobability method has been shown to significantly improve the accuracy of quantum computations, particularly in the context of the Variational Quantum Eigensolver (VQE) algorithm, a hybrid quantum-classical algorithm used for solving eigenvalue problems. However, it is important to note that while the quasiprobability method can significantly reduce the impact of measurement errors, it does not eliminate all types of errors and comes with its own set of challenges, such as the need for a large number of measurements and the potential amplification of other errors.

In this work, we will discuss the theory and implementation of the quasiprobability method for error mitigation in quantum computations. We will also explore its application in the context of the QAOA algorithm and discuss the practical considerations for implementing this method on current quantum hardware.

2.4.1 Error mitigation with sampler

Qiskit Runtime includes two predefined programs: the Sampler primitive and the Estimator primitive. The Options class can be utilized to specify several options, with one commonly used option being "resilience_level." The Sampler function estimates the entire quasiprobability distribution by sampling from input circuits. Problem circuits are created using various types of mixer Hamiltonians and set the resilience level to both 0 and 1 to observe the differences in error reduction introduced by mitigation. The resilience level corresponds to the type of error mitigation applied. In the scope of this project, we focus on two levels: 0 for no error mitigation and 1 for twirled readout error extinction (T-REx). T-REx is an ansatz for noise reduction that makes no assumption about the specific noise present, and is therefore a general, effective technique [1]. To simulate a noisy model, we utilized FakeManila.

```
fake_backend = FakeManila()
noise_model = NoiseModel.from_backend(fake_backend)

options_with_em = Options(
simulator={
    "noise_model": noise_model,
     "seed_simulator": 42,
},
resilience_level=1   # Error mitigation
)

sampler = Sampler(options=options_with_em, backend=backend)
```

```python
options = Options(simulator={"seed_simulator": 42}, resilience_level=1)
with Session(service=service, backend=backend):
    sampler = Sampler(options=options)
    job = sampler.run(circuits=[create_qaoa_circ(graph, theta, type="x_mixer"),
    create_qaoa_circ(graph, theta, type="xy_mixer_full"),
    create_qaoa_circ(graph, theta, type="xy_mixer_parity"),
    create_qaoa_circ(graph, theta, type="xy_mixer_ring")])
result_res1 = job.result()
```

Our objective was to compare the cost efficiency of mitigated and non-mitigated versions for Pauli X and XY mixers. Additionally, we plotted quasi-distribution functions for each model with mitigation. This analysis aims to provide insights into the potential benefits of implementing mitigation techniques.

3 Results and Discussion

For a given sequence of vertices to map, the correlation between these nodes can impact the difficulty in optimizing these instances. The correlation is given by: [5]

$$s^2 = var(\mu_i). \tag{6}$$

The correlation between the expectation values of a particular sequence can affect the respective variances of the bits to simulate [5], [10]. At high correlations, the cost function can be dominated by other underlying features of the problem. Heuristically, at low variances it is clearer to choose the best parameters if they are distinct[5].

3.0.1 Stability Test

Figure 5 compares four different mixers applied to three different problems executed on a noise-free simulator, a noisy simulator and on hardware. The colour map is defined in a way that a dark spot corresponds to low energy. Therefore the solution set of possible ground states is restricted by the colour maps. The X-mixer restricts the solution set only slightly but the ring, parity and full mixer improve the result significantly. In the rightmost picture, one can identify the noise by comparing the X-mixer results with the noise-free simulation.

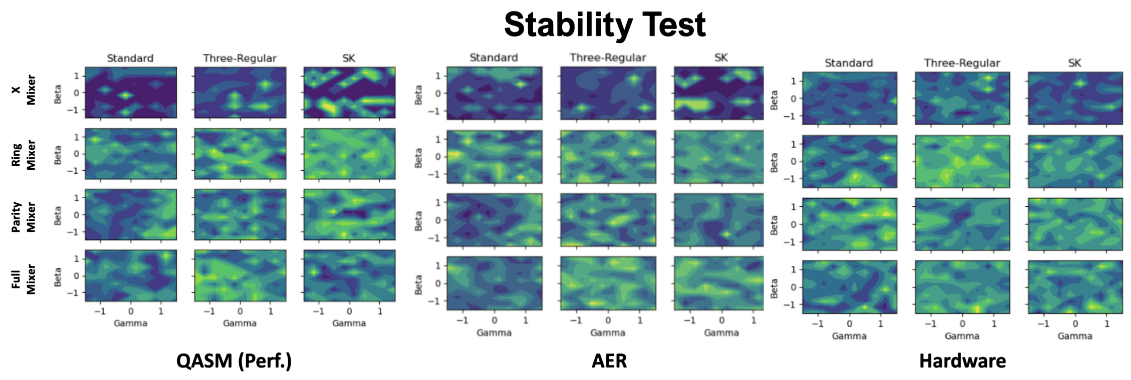


Figure 4: An illustration of the depth reduction function for mixer Hamiltonian. The left one is the original coupling map, the right one shown the implementation strategy of XY mixers.

3.0.2 Mixed Hamiltonian Benchmark

In the results above, the quantum circuit was measured directly, which introduces sampling errors. In order to prevent these errors, one can directly compute the state vector of the quantum circuit. This leads to a significant improvement in the plot because the periodic structure of the different mixers can be seen. Furthermore, it is now possible to identify the possible ground states by the dark spots on the colour map. So in the standard problem, for example, we could reduce the possible ground states from eight to two with two side peaks. In the case of the three-regular problem, the possible ground states could be reduced by half. As the SK model is completely symmetric and the amount of excitations is preserved in the XY-mixer the colour map needs to be completely flat.

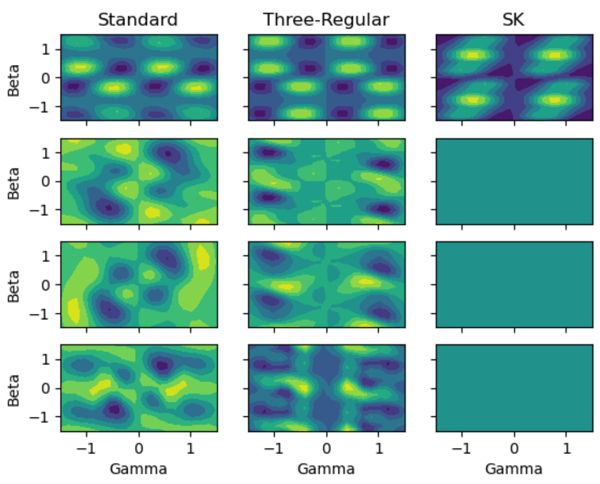
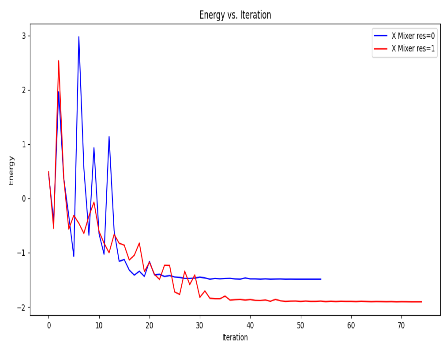


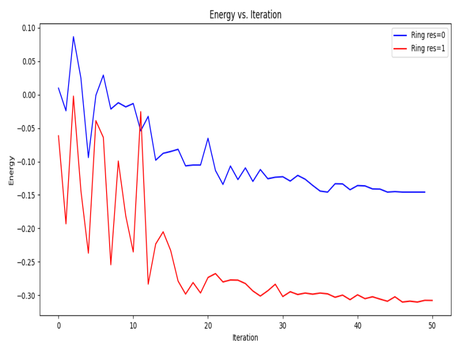
Figure 5: An illustration of the error reduction given by a mixed Hamiltonian. Using a mixed Hamiltonian, we can significantly reduce the number of possible ground states to compute, visible in the dark spots in the graphic.

3.1 Error Mitigation

In the non-mitigated version, resilience level is set to zero, so the errors caused by noise during these runs are more prevalent and significant. As a result, the accuracy and reliability of the quantum computation at residual level 0 are significantly affected. In the error-mitigated version, the resilience level is set to 1, which means the Twirled readout error extinction (T-REx) technique was used. This technique is also known as Pauli twirling, which is used to reduce the noise introduced during the process of quantum measurement. Unlike other techniques, this assumes no specific form of noise, which makes it very general and effective. As seen by the figures, we observed a lower error rate or error probability for residual level 1. This suggests that the error mitigation techniques employed in this version have effectively reduced the impact of errors during the runtime sessions at level 1. Consequently, the quantum computation outcomes at this level are more accurate and reliable. By reducing the impact of errors, these techniques enhance the performance and reliability of quantum computations, making them more suitable for various practical applications such as optimization problems [11], [12].



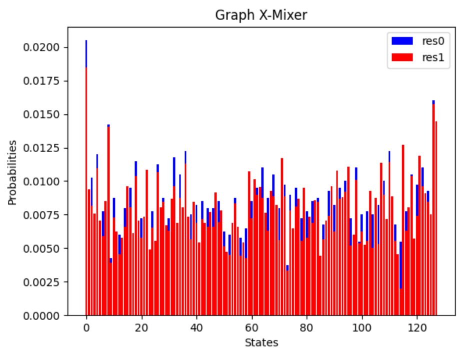
(a)



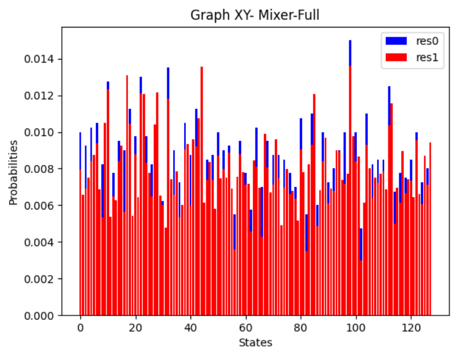
(b)

Figure 7: Comparison of energy changes over iterations for mitigated and non-mitigated X- mixer (a) and Ring- mixer (b)

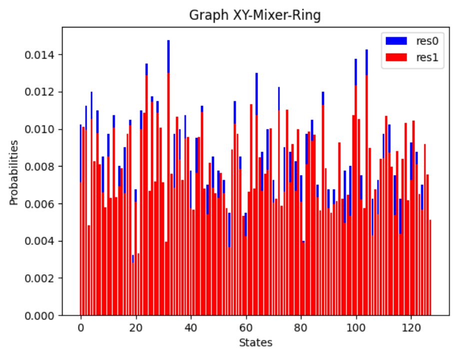
By comparing the energy changes over iterations of the mitigated and non-mitigated versions of both the X-mixer and the Ring-mixer, the faster convergence rate of the error-mitigated versions of the mixers is easily seen 7. This means that the error mitigation techniques implemented in these mixers have proven to be effective in reducing the negative effects of noise and errors.



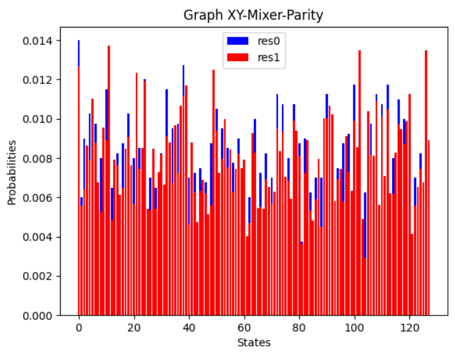
(a)



(b)



(c)



(d)

Figure 6: Setting resilience level 0 (blue) and 1 (red) effects of the error mitigation has been observed.

In the case of the X-Mixer, the unmitigated version may exhibit fluctuations and slower convergence due to noise and errors. However, with the implementation of error mitigation techniques, the mitigated X-mixer shows smoother and faster convergence.

Similarly, with the ring mixer, the uncorrected version is likely to show slower convergence and irregular energy changes due to the influence of noise and errors. On the other hand, the error-corrected ring mixer shows a faster convergence rate with more consistent and predictable energy changes over iterations.

By mitigating their effects, these techniques allow the mixers to converge more efficiently to the desired outcome. These comparisons overall highlight the benefits of error mitigation in improving the performance and convergence characteristics of both the X-mixer and the ring mixer.

4 Conclusion

This comprehensive study focuses on the Quantum Approximate Optimization Algorithm (QAOA), a hybrid quantum-classical algorithm that has been proposed as a practical method to solve combinatorial optimization problems on near-term, noisy intermediate-scale quantum (NISQ) devices. The research investigates the performance of QAOA on different problem instances, the mechanism of its operation, and its implementation on near-term quantum devices.

Three distinct problem types were studied: the Hardware Grid problem, the Three Regular problem, and the Sherrington-Kirkpatrick (SK) model problem. Each of these problem types presents unique challenges and characteristics, providing a diverse range of conditions under which the performance of QAOA can be evaluated.

The study also explores the impact of the choice of mixer Hamiltonian on the performance of QAOA. In particular, it focuses on the XY mixer and its three different versions: the ring mixer, the parity mixer, and the full mixer. The research reveals that the ring mixer exhibits superior performance in terms of the energy landscape.5 It not only achieves the best performance among the mixers studied, but also converges to a minimum faster than the X-model. This rapid convergence is a desirable property as it can significantly reduce the computational resources required for solving optimization problems using QAOA.7.

The research also delves into the critical aspect of error mitigation in quantum computing. It employs the quasiprobability method for mitigating measurement errors and applies it in the context of the QAOA algorithm. The study finds that error mitigation techniques can effectively reduce the impact of errors during runtime sessions, enhancing the performance and reliability of quantum computations. In terms of practical implementation, the study develops a tailored strategy for each specific coupling map to optimize the circuit depth in quantum circuit design. The depth optimization problem is approached as a variation of the graph coloring problem, and a greedy coloring algorithm is employed to provide a valid solution that satisfies the constraints of the problem.

In conclusion, this comprehensive study provides valuable insights into the performance of QAOA and its implementation on near-term quantum devices. It contributes to the understanding of QAOA and guides future research in this area, facilitating the practical use of quantum computing for optimization problems.

Acknowledgments

This work was carried out as part of the IBM Quantum Hackathon at the World of Quantum. We would like to express our deepest gratitude to IBM Quantum for providing this invaluable opportunity to delve into the fascinating world of quantum computing. We are particularly thankful for the support and guidance we received from the dedicated members of the Qiskit community.

References

[1] Stefan Woerner Daniel J. Egger, Jakub Marecek. Warm-starting quantum optimization. 2020.

[2] Burak Mete. A novel approach for solving constrained optimization problems with qaoa using encoders. 2022.

[3] Bryan O'Gorman Eleanor G. Rieffel Davide Venturelli Rupak Biswas Stuart Hadfield, Zhihui Wang. From the quantum approximate optimization algorithm to a quantum alternating operator ansatz. 2017.

[4] Jason M. Dominy Eleanor G. Rieffel Zihui Wang, Nicholas C. Rubin. Xy-mixers: analytical and numerical results for qaoa. 2020.

[5] Vanessa Dehn Gerhard Hellstern Matthias Huels Yanjun Ji Ilia Polian Amandeep Singh Bhatia Sebastian Brandhofer, Daniel Braun and Thomas Wellens. Benchmarking the performance of portfolio optimization with qaoa. 2021.

[6] Jeffrey Goldstone Edward Farhi. A quantum approximate optimization algorithm. 2014.

[7] IBM Qiskit documentation. Max-cut and traveling salesman problem.

[8] Davide Venturelli Ryan LaRose, Eleanor Rieffel. Mixer-phaser ansatze for quantum optimization with hard constraints. 2022.

[9] Stephan Eidenbenz Andreas Bartschi. Grover mixers for qaoa: Shifting complexity from mixer design to state preparation.

[10] Johanna Barzen Frank Leymann Vladimir Yussupov Felix Truger, Martin Beisel. Selection and optimization of hyperparameters in warm-started quantum optimization for the maxcut problem. 2022.

[11] Kristan Temme Ewout van den Berg, Zlatko K. Minev. Model-free readout-error mitigation for quantum expectation values. 2020.

[12] Antonio D Córcoles Antonio Mezzacapo Jerry M Chow Jay M Gambetta Abhinav Kandala, Kristan Temme. Error mitigation extends the computational reach of a noisy quantum processor. 2019.