Quantum Machine Intelligence (2026) 8:32
https://doi.org/10.1007/s42484-026-00354-z

RESEARCH ARTICLE

QAOA-Driven PMU placement optimization with graph learning-based parameter initialization refinement

Yuqi Jiang[1] · Xiangyue Wang[2] · Zhiding Liang[3] · Yan Li[1] · Thomas Morstyn[2] · Pedro L. S. Lopes[4] · Matthew J. Brandsema[5]

Received: 31 July 2025 / Accepted: 7 January 2026

Abstract

With the significant expansion of renewable energy integration, the scale of the power grid also increases rapidly. To effectively monitor the operational state of large-scale power grids, optimizing the placement of Phasor Measurement Units (PMUs) is a critical research focus. Optimizing PMU Placement (OPP) is a typical combinatorial optimization problem with NP-hard complexity, which brings substantial challenges to classical computers. The sheer scale of modern grids forces classical solvers into prohibitive runtimes and sub-optimal local minima, degrading both execution speed and solution quality. Recent advances in quantum computing have opened new opportunities for tackling combinatorial optimization problems, particularly through the Quantum Approximate Optimization Algorithm (QAOA). However, QAOA operates as a hybrid quantum–classical framework, where determining the optimal parameters depends on a classical optimization process that remains computationally challenging and inherently NP-hard. On the other hand, in the Noisy Intermediate-Scale Quantum (NISQ) era, obtaining optimal optimization results typically requires a large number of quantum measurement shots. In this work, we propose a graph-learning-based strategy to provide QAOA with guided parameter initialization, enabling effective operation under limited quantum resources, particularly when the number of available measurement shots is restricted. Both the OPP problem and the channel-limitation task are investigated, where the proposed graph-learning-based parameter predictor enhances QAOA performance on both tasks, improving both the approximation ratio and computational efficiency. Furthermore, due to the complexity of the channel-limitation task and the scarcity of its pretraining data, a transfer learning strategy is employed to leverage knowledge from the original OPP task, where QAOA parameter datasets are more readily available to train the graph learning framework for the QAOA parameter predictor. The transfer learning approach also outperforms both random initialization and graph learning trained solely on the channel-limitation dataset in terms of the approximation ratio and the time efficiency. In general, this work is aimed at providing a new benchmark for solving complicated real-world power system optimization problems in the current NISQ era.

Keywords Quantum Approximate Optimization Algorithm (QAOA) · Power system optimization · Phasor Measurement Unit (PMU)

✉ Yan Li
yql5925@psu.edu

Yuqi Jiang
yzj5282@psu.edu

Xiangyue Wang
xiangyue.wang@eng.ox.ac.uk

Zhiding Liang
liangz9@rpi.edu

Thomas Morstyn
thomas.morstyn@eng.ox.ac.uk

Pedro L. S. Lopes
plopes@quera.com

Matthew J. Brandsema
mjb619@arl.psu.edu

[1] Department of Electrical Engineering, Pennsylvania State University, University Park, PA 16802, USA

[2] Department of Engineering Science, University of Oxford, Parks Road, Oxford OX1 3PJ, UK

[3] Department of Computer Science, Rensselaer Polytechnic Institute, Troy, NY 12180, USA

[4] QuEra Inc., Boston, MA 02135, USA

[5] Applied Research Laboratory, Pennsylvania State University, University Park, PA 16802, USA

Published online: 16 March 2026

Nomenclature

G	Power grid
V	Power grid bus set
N	Bus number
$\mathcal{L}(\cdot)$	Function to the adjacency node set
$\mathcal{N}(\cdot)$	Function to calculate the number of adjacent neighbors of a specific node
θ	Voltage phase angle
O	Number of fully observable buses
U	Bus voltage
I	Branch current
Z	Power grid impedance matrix
B	Number of PMUs placed
P	Partially observable set
$\mathcal{M}$	Observability modeling function

1 Introduction

The rapid penetration of renewable energy resources has reshaped the power-system landscape, enlarging power networks and complicating their supervision. The monitoring and control of the expansive power grids require an overall observation of the operating state of the power grid (Almasabi and Mitra 2019). Phasor Measurement Units (PMUs) can provide a real-time measurement of the current and voltage phasors of the buses, which provides a comprehensive understanding of the system. Because each PMU (device + installation + communication) can cost thousands of dollars and only a limited subset of buses are typically instrumented, their placement becomes a high-value, complex combinatorial optimization problem. In addition, the currents and the voltage phasor of the buses without installing PMU can still be calculated via Kirchhoff's Current Law (KCL) and Ohm's Law. Specifically, a bus placed with a PMU can measure its voltage phasor and the currents on its connected branches. Thus, it is essential to determine suitable PMU placements to minimize the cost and ensure grid observability (Maji and Acharjee 2017).

Determining the Optimal PMU Placement (OPP) is a prototypical combinatorial optimization problem, which is NP-hard (Yuill et al. 2011). Maji et al. propose a multistage Exponential Binary PSO Algorithm for OPP studies (Maji and Acharjee 2017). Huang et al. study the OPP with Integer Linear Programming (ILP) for OPP with measurement redundancy (Huang et al. 2013). However, in large-scale power networks, the sheer complexity of the search space drives classical optimization routines toward prohibitively long runtimes and frequent entrapment in local minima, pushing practitioners toward heuristics. Yet heuristic methods lack worst-case guarantees and are still prone to getting stuck in local optima. This motivates novel heuristic paradigms that explore the search space under different rules (Shaydulin et al. 2022; Boulebnane et al. 2025).

Recent quantum advancements offer one such paradigm: it heuristically targets high-quality solutions using quantum state superposition and interference, potentially reshaping the optimization landscape compared with classical procedures (Lubinski et al. 2024). Industry progress from IBM, Google, QuEra, and other quantum hardware providers shows rapid gains in scale and fidelity, expanding the horizon for heuristic quantum optimization, though performance remains instance-specific, which underscores the need for clear use cases and tailored benchmarks to guide the co-evolution of quantum hardware and algorithm design (Mandelbaum et al. 2025; Wurtz et al. 2023; Arute et al. 2019; Blekos et al. 2024; Brandhofer et al. 2023; Bucher et al. 2024).

Quantum optimization algorithms have demonstrated inspiring performance in solving real-world optimization problems. These algorithms leverage the unique capabilities of quantum computing to provide solutions that have better qualities than traditional methods, particularly on certain applications across various fields such as portfolio optimization in finance (Brandhofer et al. 2023), traffic flow optimization (Zhang et al. 2021), and drug discovery (Bonde et al. 2023). For instance, Bhasin et al. utilize quantum machine learning to optimize financial portfolio management (Bhasin et al. 2024). Salloum et al. propose a quantum annealing strategy for traffic congestion optimization (Salloum et al. 2024). Li et al. use quantum machine learning for new drug discovery (Li et al. 2021).

Quantum computing has also been pioneered in power system applications. Vlachogiannis et al. utilize a quantum genetic algorithm for reactive power and voltage control (Vlachogiannis and Ostergaard 2009). Jiang et al. develop quantum computing strategies for PMU optimization, and the developed quantum framework can outperform the classical baseline methods with better placement solutions and theoretical computational complexity (Jiang et al. 2025). Morstyn applies quantum annealing for power flow calculations (Morstyn 2022). However, the development of quantum hardware is still in the nascent Intermediate Noisy Scale Quantum (NISQ) stage, where early-stage quantum computers are plagued by problems of noise and scalability. The substantial implementation time and cost force researchers to restrict runtime and the number of execution shots as the problem size increases and the quantum device scales up in size.

The Quantum Approximate Optimization Algorithm (QAOA) has been applied to a variety of combinatorial optimization problems on NISQ devices (Farhi and Goldstone 2014). QAOA is a hybrid quantum-classical algorithm that solves such problems by variationally minimizing a cost

Hamiltonian using parameterized quantum circuits. Specifically, it maps classical optimization problems into quantum energy landscapes, where the ground state encodes the optimal solution (Zhou et al. 2020). The classical optimizer is responsible for tuning the QAOA parameters, which is a significant computational bottleneck since poor initialization may lead to slow convergence or suboptimal solutions (Sack and Serbyn 2021). Furthermore, finite measurement shots on NISQ hardware reduce expectation-value precision, potentially slowing or destabilizing QAOA's parameter updates. Therefore, techniques that warm-start the QAOA parameters are essential to improve efficiency and solution quality under hardware constraints.

Given the limited quantum computation resources and the hybrid nature of QAOA, Machine Learning (ML) has provided a way to optimize the QAOA parameter searching process (Khairy et al. 2020). Notably, ML has demonstrated great potential in efficiently searching the QAOA parameter space. Liang et al. apply Graph Neural Network (GNN) to QAOA, boosting the approximation ratio of the Max-Cut problem (Liang et al. 2024). Falla et al. propose a Transfer Learning (TL) strategy for QAOA, solving the Max-Cut problem (Falla et al. 2024). Montanez-Barrera et al. combine QAOA with TL to solve multiple graph optimization problems, i.e., Max-Cut, bin packing problem (BPP), etc (Montanez-Barrera et al. 2025). Bach et al. propose GNN to accelerate Hybrid Quantum-Classical Multilevel QAOA (Bach et al. 2024). Nevertheless, graph-learning-driven warm-start strategies have yet to be adopted for quantum optimization of power-system problems, primarily because suitable training datasets and standardized pipelines that map complex grid topologies to quantum-circuit parameters are still lacking, making it difficult to build and validate models that generalize across real-world networks.

In this work, we leverage the grid's graph-topological structure and characterize realistic instances via connectivity, graph density, and other related statistics to inform algorithm design and average-case benchmarking. We pioneer the ML strategy to optimize the QAOA parameters initialization to determine the optimal placement of PMUs in power systems. Specifically, GNN is utilized to learn the power system task representations to optimize the QAOA parameter space. In addition, several downstream PMU optimization problems are also studied, where Transfer Learning is utilized to transfer a decent set of learned parameters from one PMU optimization task to another to enhance the representation. Specifically, two specific PMU optimization scenarios are investigated: (1) Optimal PMU Placement (OPP), which aims to determine the minimum number of PMUs required to achieve full observability of the power grid; and (2) Channel Limitations, where each PMU is subject to a constraint on the number of measurement channels it can support (Korkali and Abur 2009).

Recognizing that quantum shots become increasingly costly as quantum devices scale up in size, we study strategies under a restricted-shot regime, i.e., a fixed, limited number of circuit executions. It is discovered that a significant amount of repetitions of the quantum circuit execution are required to obtain the optimal results (Jiang et al. 2025). In this work, GNN will be utilized to learn the representation of the optimal QAOA parameter sets and provide the parameter knowledge to the target graph, even with limited quantum resources. In addition, given that it is much more difficult to obtain the QAOA parameter pretraining dataset of the channel limitation task, transfer learning is then utilized in the channel-limited case to transfer the learned knowledge representation from the OPP dataset, and the channel limitation knowledge is used to further finetune the graph learning backbone. For the classical methods, both the analytical and the heuristic approaches generally suffer from a heavy computational burden on large-scale systems with complex constraints and provide no guarantee of escaping local optima to achieve global optimality. In contrast, the developed QAOA framework, equipped with graph-based parameter initialization, more effectively explores the solution landscape and thus has the potential to approach globally optimal solutions with improved performance. The main contribution of this paper can be summarized as follows:

- This work presents a graph learning strategy to enhance QAOA performance for the power system PMU placement optimization problem when the quantum computation resources are restricted. The specific QAOA parameter datasets for both the OPP and the channel limitation constraints are generated, pioneering feasible quantum computing for power system applications at the current NISQ era.
- The proposed graph learning strategy demonstrates superior empirical results for both the OPP and the channel limitation scenarios in terms of the approximation ratio and the time efficiency under restricted quantum computing power. This strategy enhances the adaptability and efficiency of QAOA in solving large-scale optimization problems.
- We introduce a transfer learning strategy that leverages knowledge from standard OPP tasks to accelerate and enhance QAOA parameter prediction in channel-limited scenarios where the specific QAOA parameter datasets are harder to obtain. The proposed transfer learning strategy can significantly improve the QAOA performance and time efficiency, providing a few-shot

vision for QAOA parameter prediction in real-world applications.

The remainder of this paper is organized as follows: Section 2 illustrates the formulation of the PMU optimization problem, Section 3 introduces the QAOA solution process for the tailored objective function, Section 4 demonstrates the graph learning framework for the QAOA parameter prediction, Section 5 introduces the transfer learning strategy for the QAOA parameter predictor, Section 6 shows the numerical exampls, and Section 7 summarizes this paper.

2 Problem formulation

Suppose $G = (V, E)$ is a power grid, where V denotes the bus set (node) and E denotes the branch set (edge). The number of buses is N and the number of branches is M. Assume the adjacency matrix of G is A. An adjacency list function $\mathcal{L}$ is also defined, where $\mathcal{L}(\cdot)$ will output the set of neighbors of the input node. Therefore, a function $\mathcal{N}(\cdot)$ can also be utilized to count the total number of adjacent neighbors of a specific node. We assume that the line impedance of the branches is known and denote the impedance matrix as Z.

2.1 OPP Objective function

For the demonstration, in this work, we only consider the zero-injection scenario. The main objective of the OPP problem is to apply the minimal number of PMUs to ensure full observability of the power grid G. If the voltages on all the buses V and the currents on all the branches M are known, the system is fully observable, and each bus $i \in V$ is also fully observable. A PMU placed at bus i can measure the voltage on bus i and all the currents on the branches connecting to i. Apart from this, there are other ways to determine observability:

(i) The branch current can be calculated by Ohm's Law if the voltage phasor at both ends of that branch is known.
(ii) If the voltage phasor at one end of the branch and the current on the branch are known, the voltage phasor at the other end can be calculated via Ohm's Law.
(iii) If a bus i is connected to $\mathcal{N}(i)$ branches and $(\mathcal{N}(i) - 1)$ branch currents are known, the last unknown current can be calculated via KCL.

Suppose $X \in \mathbb{R}^{N\times 1}$ is the binary decision sequence, $x_i \in X$ indicates the i-th binary variable for the installation of the PMU on bus i. If $x_i = 1$, then a PMU will be installed on bus i, and the opposite for $x_i = 0$. The objective function can be expressed as:

$$\begin{aligned} &\min \sum_{i=1}^{N} x_i \\ &\text{s.t. } O = N, \end{aligned} \tag{1}$$

where O denotes the number of fully observable buses. This objective function can be further expressed as:

$$\min \sum_{i=1}^{N} x_i + \lambda(N - O), \tag{2}$$

where λ is the penalty coefficient. If the system is not fully observable under a given placement configuration, a penalty term λ is applied to the corresponding placement sequence. To determine the observability O given a specific placement X, a Depth First Search (DFS) strategy is adopted and will be introduced in the next section.

2.2 OPP observability calculation algorithm

In this work, we propose a DFS-based recursion algorithm to determine the observability O under a specific placement X as shown in Fig. 1, and the detailed pseudo code is summarized in Appendix A, where KCL is leveraged to design the recursion strategy.

Beginning with the bus set I that has PMU placement, the recursive process will go through all the fully observable buses. Assume B PMUs are installed initially, and these buses j with PMUs placed are fully observable. A set D will be utilized to record all the fully observable buses, and it will be initialized with I. For $j \in D, j = 1, ..., B$, its neighbor $k \in \mathcal{L}(j)$ will be at least partially observable since

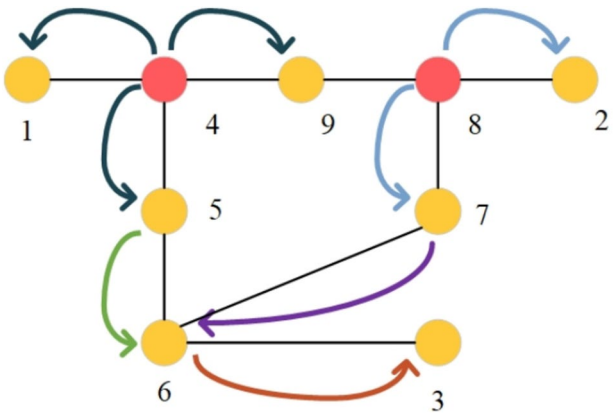

Fig. 1 The recursion process of the developed DFS-based observability algorithm on the IEEE 9-bus system. Starting from PMU-equipped bus 4, the search visits buses 1, 5, and 9 (partially observing bus 6), then proceeds from PMU-equipped bus 8 through buses 2 and 7 to finally render bus 6 and its neighbors fully observable

the voltage is $\mathrm{U}_k = \mathrm{U}_j - \mathrm{Z}_{jk}\mathrm{I}_{jk}$, and one connecting current I_{jk} is known. During the recursive process, any fully observable bus will have at least one neighbor that is at least partially observable.

Given this partially observable condition, the recursion rule can be designed. Starting with any observable bus placed with the PMU $j \in D$, all of its neighbor nodes will be partially observable. For $k \in \mathcal{L}(j)$, leveraging rule (iii), if the number of known currents of k is no less than $\mathcal{N}(k) - 1$, then k is fully observable, and the recursion will continue on k immediately. Otherwise, k is partially observable, and a partially observable set P will be utilized to store the number of known currents of k during the recursion. As the process continues, if $P(k) \geq \mathcal{N}(k) - 1$, k will be determined as fully observable and the recursion will be processed on k. As a result, beginning with the initial placement set I, the observable buses will be recursively computed, and the process will be terminated if all observable buses are processed. The detailed algorithm is summarized in the Appendix A. The DFS-based recursion is adopted here because the observability propagation from PMU-equipped buses naturally follows long KCL-based chains, and a recursive DFS traversal can exploit this structure with linear-time complexity on sparse grid graphs.

As illustrated in Fig. 1 using the IEEE 9-bus system as an example, suppose the depth-first recursion starts from bus 4. Bus 4 has three neighboring buses, 1, 5, and 9. Thus, each of them is visited in turn. Bus 1 has only one neighbor (bus 4), and thus becomes fully observable. Similarly, bus 9 has two neighbors and is fully observable by applying KCL. Its remaining neighbor, bus 8, is a PMU-equipped bus that has already been included in the recursion list, so the recursion at bus 9 terminates. The procedure then proceeds to bus 5, which is also fully observable via KCL. Next, the recursion continues from bus 5 to its other neighbor, bus 6. However, bus 6 has three neighbors and only I_{56} is known. Therefore, bus 6 is partially observable and $P(6) = 1$. As a result, the recursion that starts from PMU-placed bus 4 terminates, and the search proceeds to the other PMU-placed bus 8.

For bus 8, its neighboring buses 2 and 7 likewise become fully observable. Since there is no further recursion from bus 2, the recursion continues from bus 7. For bus 7, the current I_{67} can be calculated via KCL, and the partially observable set can be updated as $P(6) = 2 \geq \mathcal{N}(6) - 1$. Therefore, bus 6 can be determined as fully observable now, and it will enter the recursion. Finally, the remaining neighbor of bus 6 can also be determined as fully observable, and the overall recursion terminates.

As for the computational complexity, if the power grid is fully observable, every bus will only enter the recursion process once. In addition, every branch will be checked twice, yielding the $\mathcal{O}(N + 2M)$ complexity. Thus, the total complexity is $\mathcal{O}(N + M)$, and no extra auxiliary qubit is required for this recursion process.

2.3 OPP with channel limitations

The installation cost of a PMU is driven mainly by its technical features, most notably, the number of measurement channels it supports (Shafiullah et al. 2019). Specifically, each measurement channel is capable of measuring the current on the branch where it is installed. Due to the significant expense associated with PMUs that offer unrestricted measurement capabilities, the OPP task with channel limitations is studied under a more practical assumption.

To handle the channel limitation constraints, we leverage a Breadth-First Search (BFS)-based algorithm to calculate the observability of the system given a specific placement, where the details of the algorithm can be found in the Appendix B. In contrast to the DFS-based scheme used for the unconstrained OPP case, BFS is chosen here because the channel capacity constraint is naturally enforced in a level-wise manner: a queue-based traversal allows us to allocate a limited number of measurement channels from each PMU outward and update observability in a controlled, breadth-wise fashion.

Given B placed PMUs, a queue $\mathcal{Q}$ is utilized and initialized with each bus that has a PMU installed. Assume the channel limitation is set to F. Starting with the PMU placed bus j, if the PMU placed bus j has the number of neighbors that satisfies $\mathcal{N}(j) \leq F + 1$, then bus j becomes fully observable, and each of its neighboring buses $k \in \mathcal{N}(j)$ will be denoted as visited with their observability count updated as $P(k) = P(k) + 1$. If $P(k) \geq \mathcal{N}(k) - 1$, k is fully observable and will enter the queue $\mathcal{Q}$. After all $k \in \mathcal{N}(j)$ is handled, j will be out of the queue, the branch jk will not be visited in the following recursion. A symmetric matrix $\mathcal{Y}$ is utilized, where the entry $\mathcal{Y}[jk] = \mathcal{Y}[kj] = 1$ if jk is visited, otherwise $\mathcal{Y}[jk] = \mathcal{Y}[kj] = 0$.

If $\mathcal{N}(j) > F + 1$, then the observability of j is not determined. In this paper, a greedy strategy is used for placing the measurement channels on the connecting branches of j. Specifically, the measurement channel will be installed on the branches $jk, k \in \mathcal{N}(j)$ where: (1), jk has not been visited, (2), $k \in \mathcal{N}(j)$ has more neighbors. If the number of unvisited branches of j is less than F, there will be measurement channel redundancy at j, and these redundant channels will still be placed at k with more neighbors, where $k \in \mathcal{N}(j)$. Thus, after the updating of the channel placement for j, the total known branches of j can be counted as $P = \sum_j \mathcal{Y}[jk], k \in \mathcal{N}(j)$. If j is determined to be fully observable, j will enter $\mathcal{Q}$ again.

Otherwise, j is partially observable. If any PMU-installed bus j is partially observable during its first recursion, but is further determined to be fully observable in the following iterations, j will enter $\mathcal{Q}$ again. For the PMU-installed buses, though they may be determined to be fully observable, some of their neighbors may not be visited. Therefore, if the PMU-installed buses are determined to be fully observable during the recursion, they will always enter $\mathcal{Q}$ twice to ensure there are no unvisited neighbors. The details of the algorithm are shown in the Appendix B.

As for the computational complexity of the observability modeling algorithm under the channel limitation scenario, if the power grid is fully observable, all the branches will be visited only once, the PMU-installed buses will be visited twice, and the remaining buses will be visited once, resulting in $\mathcal{O}(M+N)$ complexity. In the next section, the quantum solution pipeline for OPP will be introduced.

3 Quantum solutions for OPP

In this section, the detailed QAOA solution process for the OPP problem is developed, which can be visualized in Fig. 2. QAOA is a hybrid algorithm utilizing parameterized quantum evolution and classical optimizers.

Initially, the input transformation is conducted to convert the $\{0,1\}$ classical binary string to Z with $\{-1,1\}$ as:

$$z_i = -2x_i + 1. \tag{3}$$

Thus, the classical objective function is converted to:

$$\min\ C(Z) = \sum_{i=1}^{N} \frac{1-z_i}{2} + \lambda(N - \mathcal{M}(\frac{1-Z}{2})), \tag{4}$$

where $\mathcal{M}$ denotes the observability modeling algorithm that will output the observability value O. Since there are 2^N candidate binary strings, for a specific binary string Z_p, it can be mapped to the quantum form as:

$$C(Z_p) = \sum_{i=1}^{N} \frac{1-z_{p,i}}{2} + \lambda(N - \mathcal{M}(\frac{1-Z_p}{2})) \tag{5}$$

$$= \sum_{i=1}^{N} \frac{1-\langle z_{p,i}|\sigma_i^z|z_{p,i}\rangle}{2} \tag{6}$$

$$+ \lambda(N - \mathcal{M}(\frac{1-Z_p}{2})) \tag{7}$$

$$= \sum_{i=1}^{N} \frac{1-\langle z_{p,i}|\sigma_i^z|z_{p,i}\rangle}{2} + \lambda(N - \mathcal{M}(cat(\frac{1-\langle z_{p,i}|\sigma_i^z|z_{p,i}\rangle}{2}))) \tag{8}$$

$$= \langle Z_p|H_C|Z_p\rangle \equiv C(|Z_p\rangle), \tag{9}$$

where $cat(\cdot)$ is the operation to concatenate the input into a string. H_C is the problem Hamiltonian, which can be expressed as:

$$H_C = \sum_{i=1}^{N} \frac{1-\sigma_i^z}{2} + \lambda(N - \mathcal{M}(cat(\frac{1-\sigma_i^z}{2}))). \tag{10}$$

After this, all the 2^N candidate strings will be mapped to the quantum form, and the 2^N $|Z_p\rangle$ strings form the Hilbert space for the quantum state. Thus, $|Z\rangle$ is the linear combination of all $|Z_p\rangle$ as:

$$|Z\rangle = \sum_{p=1}^{2^N} \alpha_p |Z\rangle_p, \tag{11}$$

where $\sum_{p=1}^{2^N} |\alpha_p|^2 = 1$. Thus, we can have:

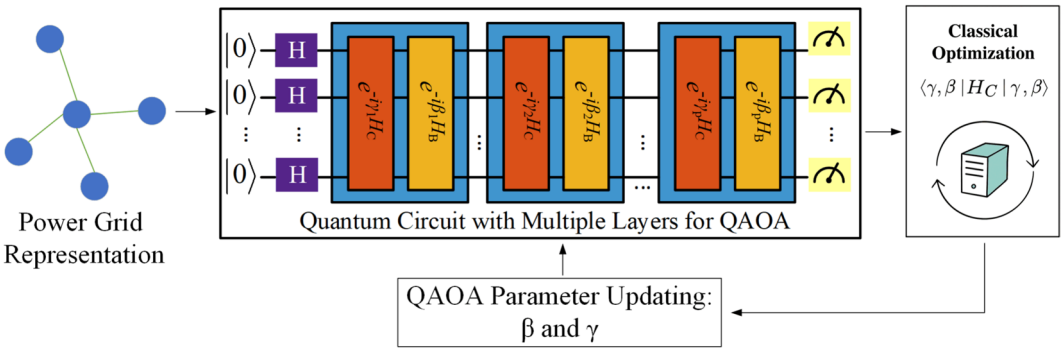

Fig. 2 The original solution process for OPP and channel limitation tasks using QAOA. The QAOA parameters $\vec{\gamma}$ and $\vec{\beta}$ are randomly initialized

$$C(Z) = \langle Z|H_C|Z\rangle = \sum_{k=1}^{2^N} \alpha_k^2 C(|Z_k\rangle). \tag{12}$$

Therefore, the original OPP is transformed into finding the minimum quantum energy state value.

To implement the optimization process on a quantum computer, a mixer Hamiltonian H_B is selected for circuit initialization as $H_B = \sum_{l=1}^{N} \sigma_l^x$, where σ_l^x is the Pauli-X operator. The mixer Hamiltonian H_B prepares all N qubits into a superposition, where all the qubits are initialized as $|+\rangle$ (Zhou et al. 2020). Thus, we can apply the mixing Hamiltonian H_B and the problem Hamiltonian H_C to represent the variational wavefunction of QAOA as:

$$\begin{aligned} |\psi_q(\vec{\gamma},\vec{\beta})\rangle \quad &= e^{-i\beta_q H_B} e^{-i\gamma_q H_C} \cdots e^{-i\beta_2 H_B} \\ &\quad e^{-i\gamma_2 H_C} e^{-i\beta_1 H_B} e^{-i\gamma_1 H_C} |+\rangle^{\otimes N}, \end{aligned} \tag{13}$$

where the variational wavefunction is parameterized by $2q$ trainable parameters $\vec{\gamma} = (\gamma_1, \gamma_2, ..., \gamma_q)$ and $\vec{\beta} = (\beta_1, \beta_2, ..., \beta_q)$. Classical optimization methods are utilized to optimize $\vec{\gamma}$ and $\vec{\beta}$ to minimize the expectation value of the quantum state $F(\vec{\gamma},\vec{\beta}) = \langle\psi(\vec{\gamma},\vec{\beta})|H_C|\psi(\vec{\gamma},\vec{\beta})\rangle$.

4 Graph learning for QAOA

As shown in the last section, the landscape of QAOA is parameterized by $2q$ parameters. However, the main bottleneck of optimizing the QAOA is to find the optimal parameter set with classical optimization methods, which is an NP-hard problem (Montanez-Barrera et al. 2025). In addition, the quantum energy landscape has a great number of local minima. Because the NP-hard, multi-minima QAOA landscape easily traps random starts, a well-informed parameter initialization is essential to guide the optimizer toward near-global optima with far fewer circuit evaluations.

Graph learning can extract the topology patterns, for instance, degree distribution, clustering coefficients, and spectral characteristics, as well as reveal their underlying relation with the quantum energy landscape encoded by the QAOA parameters. Thus, a promising approach is to apply graph learning strategies to learn from the QAOA landscapes that are trained by optimal parameters and predict the QAOA parameter initialization. The overall graph learning process is shown in Fig. 3 (a) and (b).

In this work, we further pretrain the QAOA on representative graphs to obtain near-optimal parameters that encode structural patterns of the quantum energy landscape to form the training dataset, as shown in Fig. 3 (a). Our key objective is to demonstrate that these pretrained parameters, informed by graph learning, can boost QAOA performance in resource-limited scenarios, where only a restricted number of measurement shots is available and classical optimization is costly and prone to local minima. Graph Transformer is selected as the graph learning backbone model due to its superior ability to capture long-range dependencies and global structural patterns that are crucial for understanding the topology-QAOA parameter relationship. In general, as shown in Fig. 3 (a) and (b), the formed QAOA parameter dataset and the corresponding graph instance are input into the Graph Transformer to predict the QAOA parameter initialization.

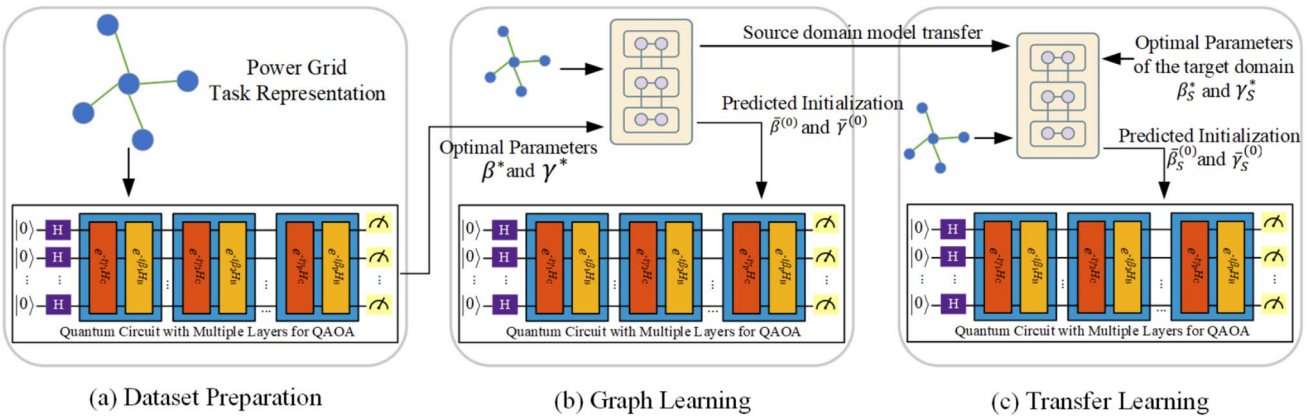

Fig. 3 The overall framework of the proposed graph-learning–based QAOA initialization and its transfer to new power-grid tasks. (**a**) For each power-grid optimization task, the corresponding graph $G_t = (V_t, E_t)$ is encoded and input into the QAOA circuit, and the task-specific optimal parameters $\theta_t^* = (\beta_i^*, \gamma_i^*)$ are collected as the training labels. (**b**) A graph learning model is then trained on these $\mathcal{D} = \{G_t, \theta_t^*\}$ pairs to predict QAOA initializations $\bar{\beta}^{(0)}$ and $\bar{\gamma}^{(0)}$ directly from a given grid graph. (**c**) For a new target domain under the channel-limitation scenario, the backbone weights of the source-domain graph model of the vanilla OPP task Θ_S are transferred and fine-tuned on the target-domain dataset so that the adapted model can forecast good QAOA initializations for the target tasks. The input of the target domain model is $\mathcal{D}_T$

Let $\mathcal{D} = \{G_t, \theta_t^*\}$ as the training dataset, where each sample consists of a graph $G_t = (V_t, E_t)$ as the input, and the corresponding pretrained optimal QAOA parameters $\theta_t^* = (\beta_i^*, \gamma_i^*) \in \mathbb{R}^{2q}$ as the learning target for q QAOA layers. It should be noted that the target parameters θ_t^* are obtained through extensive QAOA pretraining that converges to:

$$\theta_t^* = \text{argmin}_\theta \langle \psi_t(\theta) | H_C^{(t)} | \psi_t(\theta) \rangle. \tag{14}$$

This pretraining process is very computationally expensive, often requiring substantial quantum measurement shots per parameter set, which also motivates our graph learning approach to predict the optimal parameter set and transfer the learned representations to the much fewer shot QAOA learning.

Given a graph $G_t = (V_t, E_t)$ with the adjacency matrix $A_t \in \{0,1\}^{|V_t| \times |V_t|}$ The initial graph representation can be constructed as:

$$H^{(0)} = E_{\text{node}} + E_{\text{pos}} + E_{\text{degree}}, \tag{15}$$

where E_{node} is the node embedding, E_{pos} is the positional embedding (Vaswani et al. 2017), and E_{degree} is the node degree embedding (Wu et al. 2019).

For each attention layer l, the multi-head attention with graph structure bias is computed as:

$$\begin{aligned} &\text{Attention}(Q^{(l)}, K^{(l)}, V^{(l)}) \\ &= \text{Softmax}\Big(\frac{Q^{(l)}(K^{(l)})^T}{\sqrt{d_k}} + \mathcal{B}_{adj}\Big) V^{(l)}, \end{aligned} \tag{16}$$

where $Q^{(\ell)} = H^{(\ell)} W_Q^{(\ell)} \in \mathbb{R}^{|V| \times d_k}$ is the Transformer query, $K^{(\ell)} = H^{(\ell)} W_K^{(\ell)} \in \mathbb{R}^{|V| \times d_k}$ is the Transformer key, $V^{(\ell)} = H^{(\ell)} W_V^{(\ell)} \in \mathbb{R}^{|V| \times d_k}$ is the Transformer value, and $\mathcal{B}_{\text{adj}} = \alpha \cdot A_i \in \mathbb{R}^{|V| \times |V|}$ is the adjacency bias matrix with learnable parameter α. Then the multi-head attention can be constructed as:

$$\text{MultiHead}(H^{(l)}) = \text{Cat}(\text{head}_1, ..., \text{head}_h) W_O, \tag{17}$$

where $W_O \in \mathbb{R}^{d_{model} \times d_{model}}$ is the output projection matrix, and each attention head processes different representational subspaces.

In each Transformer block, it applies as:

$$\tilde{H}^{(l+1)} = \text{LayerNorm}\big(H^{(\ell)} + \text{MultiHead}(H^{(\ell)})\big), \tag{18}$$

$$H^{(\ell+1)} = \text{LayerNorm}\big(\tilde{H}^{(\ell+1)} + \text{FFN}(\tilde{H}^{(\ell+1)})\big), \tag{19}$$

where $\text{FFN}(x) = W_2 \cdot \text{ReLU}(W_1 x + b_1) + b_2$ is the Feed-Forward Network (FFN). The Feed-Forward Network (FFN) introduces non-linearity and feature transformation at each position independently, enabling the model to capture complex patterns beyond self-attention.

After L Transformer layers, the global graph representation is obtained through attention-weighted pooling:

$$\mathcal{M} = \text{softmax}\big(\text{GlobalAttention}(H^{(L)})\big), \tag{20}$$

$$h_G = \sum_{v=1}^{|V|} a_v \cdot H_v^{(L)}, \tag{21}$$

where

$$\begin{aligned} \text{GlobalAttention}(H^{(L)}) = W_{\text{attn}_2} \cdot \tanh\big(W_{\text{attn}_1} H^{(L)} \\ + b_{\text{attn}_1}\big) + b_{\text{attn}_2}. \end{aligned} \tag{22}$$

The final QAOA parameter forecasting is yielded by the decoder as:

$$\begin{aligned} \text{QAOADecoder}(h_G) = W_3 \cdot \text{ReLU}\big(W_2 \cdot \text{ReLU}\big(W_1 h_G \\ + b_1\big) + b_2\big) + b_3. \end{aligned} \tag{23}$$

The overall training and prediction process is visualized in Fig. 3 (b).

5 Transfer learning for channel limitation applications

Utilizing QAOA to predict the OPP under the channel limitation scenario will take much longer training time and more QAOA measurement shots to guarantee optimal results. Thus, the QAOA parameter pretraining dataset for the channel limitation scenario is much more complicated to obtain than the normal OPP problem.

Given this computational complexity and the prohibitive cost of generating extensive pretraining datasets for channel-limited scenarios, we propose a transfer learning approach to leverage the learned knowledge from a donor task: the standard OPP problem, to boost the learning performance for the target task of OPP under channel limitation scenarios with limited quantum resources. The transfer learning process can be visualized in Fig. 3 (c), where the parameters of the source domain model will be transferred to the target domain model, and the optimal parameter sets and the corresponding graph instances will be used as the input.

Let the OPP problem be the source domain task and the source domain dataset be $\mathcal{D}_S = \{G_{S,t}, \theta_{S,t}\}_{t=1}^{M_S}$. Suppose the parameters of the source model as Θ_S, where $\Theta_S = \{\Theta_{en}, \Theta_{de}\}$ (including the parameters from the

Encoder and Decoder). The OPP under channel limitation is the target domain task, and the target domain dataset is $\mathcal{D}_T = \{G_{T,t}, D_t, \theta_{T,t}\}_{t=1}^{M_T}$, where D_t is the channel limitation, and $M_T \ll M_S$. The backbone model of the target task is still Graph Transformer. Assume the target domain model as $\bar{\theta}_{T,t} = \mathcal{H}(G_{T,t}, D_t; \Theta_S)$.

For the channel limitation scenario, the channel limitation constraint is also embedded in the input representation process as:

$$H^{(0)} = E_{\text{node}} + E_{\text{pos}} + E_{\text{degree}} + g(\mathrm{e}_D), \tag{24}$$

where the last term is the channel limitation constraints embedding. e_D is the channel-specific embedding vector for channel limitation capacity D, $\mathrm{e}_D = \mathcal{E}(D)$, where $\mathcal{E} : D \to \mathbb{R}^{d_{model}/\epsilon}$, d_{model} is the dimension of the model, and ϵ is the scaling factor. g is an MLP to process the raw channel embedding. $H^{(0)}$ is the final initial node representations incorporating channel constraints. Thus, the main idea is to learn and transfer a predictive prior over QAOA parameters based on graph structure and constraints from the original OPP problem.

To implement the transfer learning, the parameters of the source domain model Θ_S are transferred to $\mathcal{H}$ since both the source and the target domain model have the same backbone. For $\mathcal{H}$, the extra parameters are the channel limitation constraints embedding Θ_{CL}. To train the target domain model $\mathcal{H}$, MSE loss is utilized as:

$$\text{MSE} = \frac{1}{M_T} \sum_{t=1}^{M_T} ||\theta_{T,t} - \bar{\theta}_{T,t}||_2^2, \tag{25}$$

where $\bar{\theta}_{T,t}$ is the predicted QAOA parameter.

To finetune the target domain model $\mathcal{H}$, different parameter groups require distinct learning rates to balance the preservation of valuable pre-trained knowledge with the acquisition of new domain-specific capabilities (Li et al. 2020). Specifically, different learning rate strategies are applied for different parameter groups, where the target domain model parameters can be denoted as $\Theta_T = \Theta_{en} \cup \Theta_{CL} \cup \Theta_{de}$. The updating of Θ_{en}, Θ_{CL}, and Θ_{de} can be expressed as:

$$\Theta_{en}^{(u+1)} = \Theta_{en}^{(u)} - \alpha_{en} \nabla_{\Theta_{en}} \text{MSE}, \tag{26}$$

$$\Theta_{CL}^{(t+1)} = \Theta_{CL}^{(t)} - \alpha_{CL} \nabla_{\Theta_{CL}} \text{MSE}, \tag{27}$$

$$\Theta_{de}^{(t+1)} = \Theta_{de}^{(t)} - \alpha_{de} \nabla_{\Theta_{de}} \text{MSE}, \tag{28}$$

where u denotes the iteration steps of the finetuning process, and the learning rate $\alpha_{en} < \alpha_{de} < \alpha_{CL}$ to preserve transferred knowledge while adapting to new constraints.

6 Numerical examples

In this section, the numerical examples are carried out, where QAOA is leveraged for the OPP scenario and the channel limitation scenario optimizer. The proposed graph learning strategy is used to predict the QAOA parameter initialization. Specifically, the proposed graph learning strategy is applied to the OPP problem with the IEEE 14- and 24-bus systems as the test cases, where the number of qubits required is 14 and 24, respectively. For the channel limitation scenario, the proposed graph learning strategy is used on the IEEE 24-bus system. In addition, the proposed transfer learning strategy is also used to transfer the learned OPP representations of the channel limitation task, where the pretrained datasets are more complicated to obtain. The QAOA is conducted on both the IBM quantum simulators and the classical servers (2 Intel Xeon 6542 CPUs, 2 Nvidia L40S, and 512 GB RAM). The graph learning framework is also trained on the classical servers. The sizes of the training datasets for the vanilla scenario M_T and the channel-limitation scenario M_S are 80,000 and 20,000, respectively. The number of QAOA layers p is set to 7. The number of quantum circuit implementation repetition times $\mathcal{R}$ is set from 1,000 to 6,000, and all the experiments are repeated 5 times.

6.1 Backbone model architecture evaluation

Initially, we compare the performances of different GNN architectures for our applications. As proof of concept, we compare the utilized graph Transformer with GraphSAGE, Graph Isomorphism Networks (GIN), Graph Convolutional Network (GCN), and Graph Attention Network (GAT) for the OPP scenario. The test system is the IEEE 24-bus system, and the number of QAOA measurement shots is 1,000. To quantitatively assess the performance of the graph learning predicted QAOA parameter initialization by different methods, the QAOA approximation ratio is used as the baseline:

$$\mathcal{A} = \frac{Y_{\text{G}}}{Y_{\text{P}}}, \tag{29}$$

where Y_{P} is the predicted number of PMUs to make the power grid fully observable given by QAOA and Y_{G} is the global optimal result. In the following experiment, the

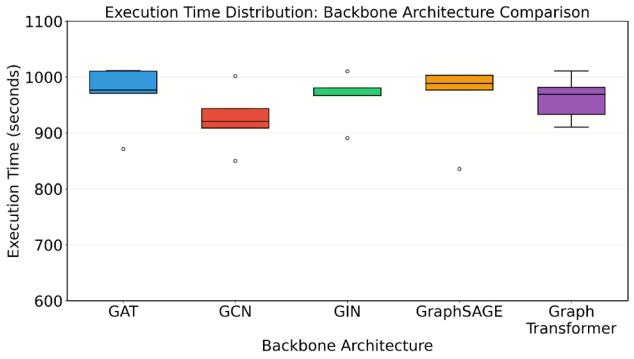

Fig. 4 The implementation time comparison between the GL and the RI strategies for the IEEE 24-bus system

Table 1 Approximation Ratio Comparison Across Different Backbone Architectures

Model	Approximation Ratio
Graph Transformer	0.83
GAT	0.71
GCN	0.71
GIN	0.63
GraphSAGE	0.71

Table 2 Minimum number of PMUs For Full Observability: Baseline comparison

Model	14-bus system	24-bus system
QAOA with GL	3	6
QAOA with RI	3	6
Ref. Rahman and Zobaa 2016	4	7
Ref. Kumar 2014	NA	7
Ref. Bhattacharjee and De 2023	4	NA
Ref. Chen et al. 2020	4	NA

QAOA approximation ratio $\mathcal{A}$ is also used as the comparison benchmark. In Fig. 4 and Table 1, we present the approximation ratios $\mathcal{A}$ of the predicted QAOA parameter initializations generated by both the benchmark methods and the proposed Graph Transformer, as well as the total prediction time of different graph learning models when utilizing the pretrained QAOA parameter datasets.

As shown in Fig. 4 and Table 1, the Graph Transformer framework attains the highest approximation ratio of 0.83 among all candidate backbone architectures, while its execution time ranks second overall, only slightly slower than that of GAT. Thus, in this work, Graph Transformer is selected as the QAOA parameter predictor empirically.

6.2 OPP Experiment results

As for the OPP scenario, the global optimal results Y_G for IEEE 14- and 24-bus systems are placing 3 and 5 PMUs to make the power grid fully observable. For the simplicity of illustration, the graph learning based QAOA parameter initialization strategy is denoted as GL and the random initialization strategy is denoted RI. In addition, the implementation time of both strategies is also recorded.

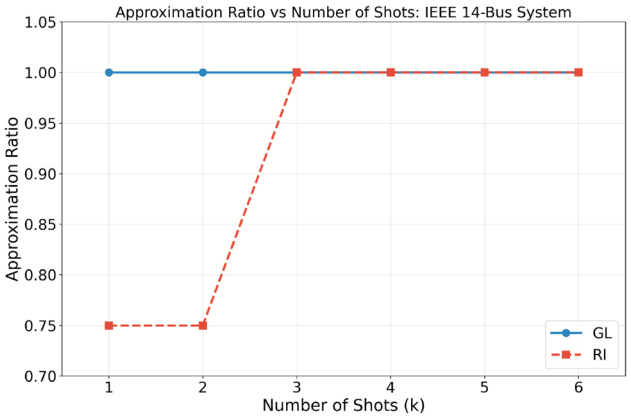

Fig. 5 The approximation ratio $\mathcal{A}$ comparison between the GL and the RI strategies for the IEEE 14-bus system

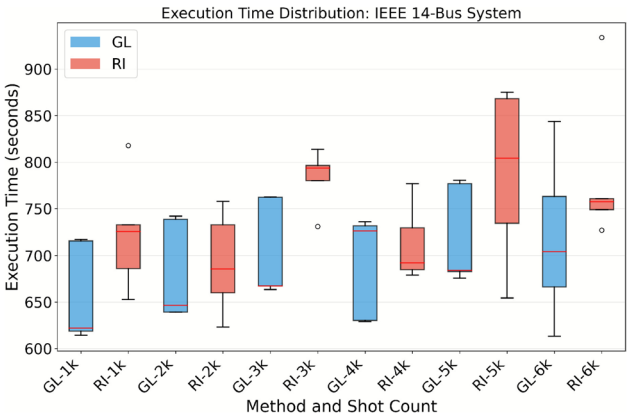

Fig. 6 The implementation time comparison between the GL and the RI strategies for the IEEE 14-bus system

Firstly, the best placement results of the classical baseline results (NA indicates that the method is not tested on the system in the reference), QAOA with RI strategy, and the developed QAOA with GL strategy are compared in Table 2. As for the outcome, the developed QAOA with GL and the vanilla QAOA achieve fewer PMUs than representative classical methods, demonstrating that applying QAOA can outperform the classical benchmarks in terms of the solution quality.

In addition, as shown in Figs. 5, 6, 7, and 8, for both the IEEE 14- and 24-bus systems, the proposed graph learning strategy demonstrates significant performance improvement when the quantum measurement shots are limited. It should be noted that due to the limited quantum measurement shots, QAOA may not yield the global optimal results in several scenarios. For the 14-bus system, the proposed GL strategy achieves the optimal results as the measurement shots vary from 1,000 to 6,000, while the RI strategy performs worse when the number of shots

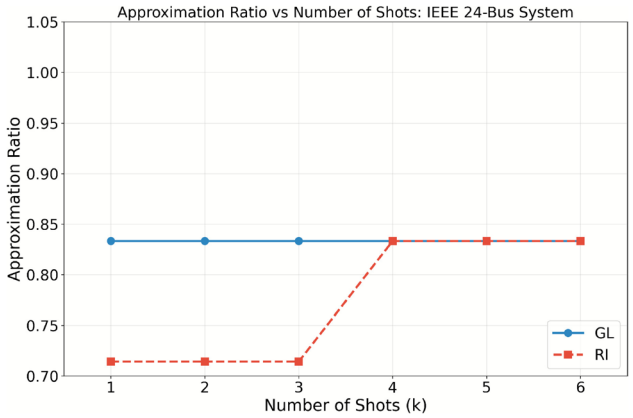

Fig. 7 The approximation ratio $\mathcal{A}$ comparison between the GL and the RI strategies for the IEEE 24-bus system

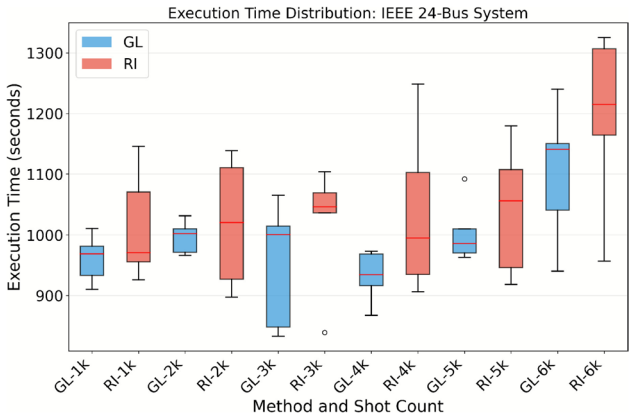

Fig. 8 The implementation time comparison between the GL and the RI strategies for the IEEE 24-bus system

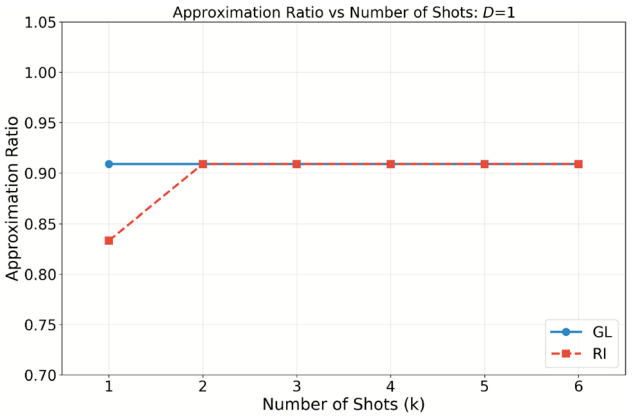

Fig. 9 The approximation ratio $\mathcal{A}$ comparison between the GL and the RI strategies for channel limitation $D = 1$

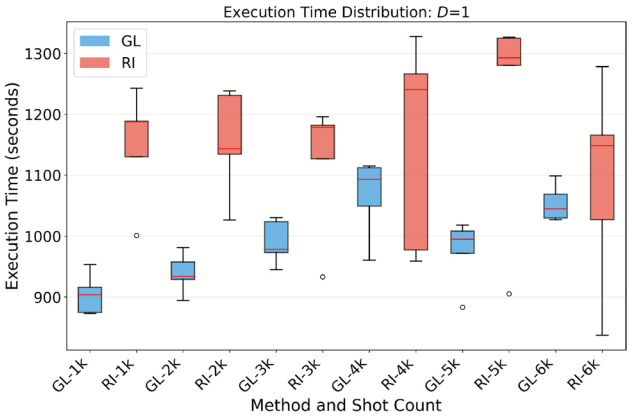

Fig. 10 The implementation time comparison between the GL and the RI strategies for the channel limitation $D = 1$

is 1,000 or 2,000. On the other hand, the implementation time with the GL strategy improves approximately 7% on average. For the 24-bus system, with limited quantum measurement shots, both strategies cannot guarantee the global optimal results since the system is much more complicated. The GL strategy can yield an approximation ratio of 83.3% (using 6 PMUs) for all 1,000 to 6,000 shots, while the RI strategy performs worse when the number of shots is 1,000 or 2,000. In addition, the implementation time improves by about 6% on average. Notably, the GL method shows execution times ranging from approximately 620-780 seconds across different shot counts, with relatively low variance indicated by the compact box plots. The RI strategy demonstrates slightly higher variance in execution times, particularly at higher shot counts, with median execution times ranging from 680-750 seconds. The RI method exhibits several outliers, suggesting less consistent computational performance.

In general, the GL initialization strategy achieves optimal or near-optimal solutions more reliably, especially at lower shot counts where quantum noise effects are more pronounced. the GL strategy not only improves the time efficiency, but also exhibits more consistent performance with lower variance, reducing the risk of exceptionally long optimization runs.

6.3 Channel limitation experiment results

In this section, the channel limitation constraints are incorporated. The IEEE 24-bus system is utilized as the case study, and the channel limitation D is set from 1 to 5, where the global optimal results are 10, 7, 6, 5, and 5, respectively.

As shown in Figs. 9, 10, 11, 12, 13, 14, 15, 16, 17, and 18, the proposed GL strategy performs overall better than the RI in terms of the approximation ratio and the time efficiency when the quantum computation resources are restricted.

Across all channel limitation scenarios, GL demonstrates superior quantum resource efficiency by achieving target approximation ratios with significantly fewer shots. Under moderate channel constraints (D=2, D=4), GL reaches optimal performance (A=1.0) at 2k shots while RI requires 4k shots, representing a 50% reduction in quantum circuit executions. This pattern is particularly evident in the D=2

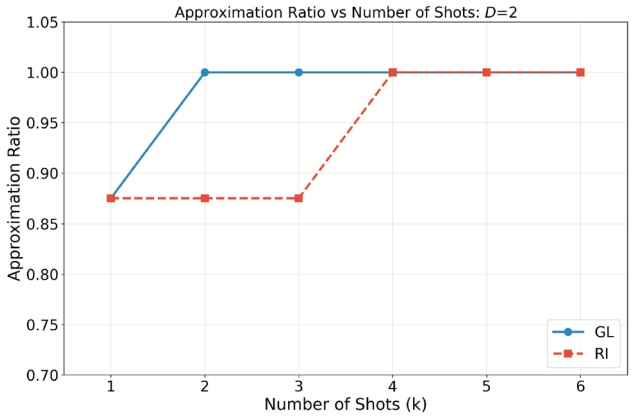

Fig. 11 The approximation ratio $\mathcal{A}$ comparison between the GL and the RI strategies for channel limitation $D = 2$

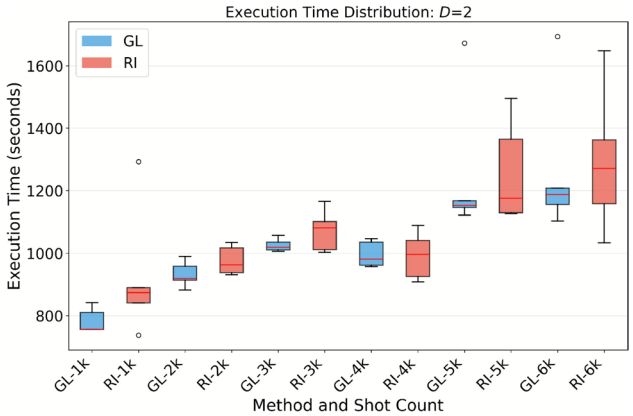

Fig. 12 The implementation time comparison between the GL and the RI strategies for the channel limitation $D = 2$

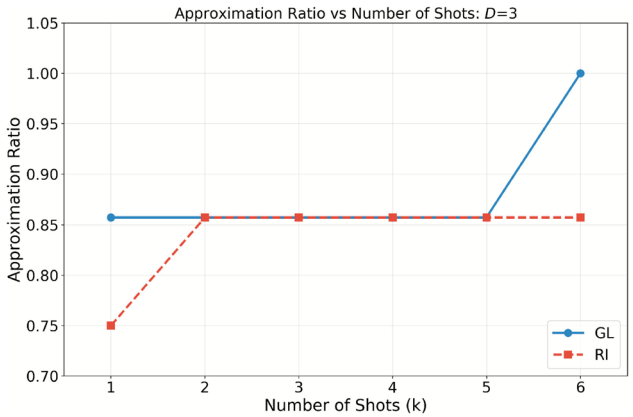

Fig. 13 The approximation ratio $\mathcal{A}$ comparison between the GL and the RI strategies for channel limitation $D = 3$

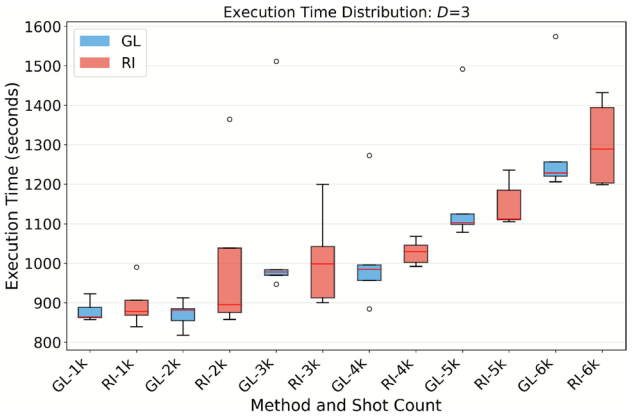

Fig. 14 The implementation time comparison between the GL and the RI strategies for the channel limitation $D = 3$

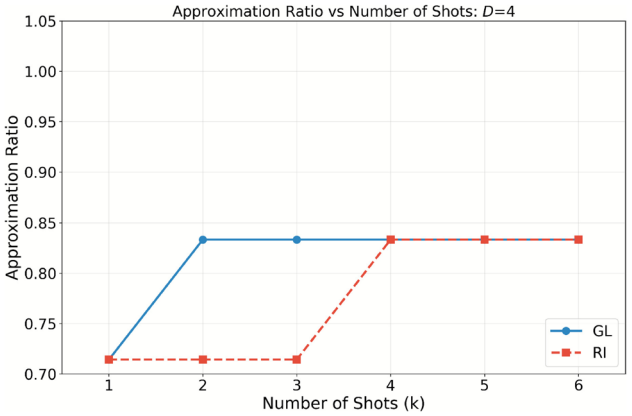

Fig. 15 The approximation ratio $\mathcal{A}$ comparison between the GL and the RI strategies for channel limitation $D = 4$

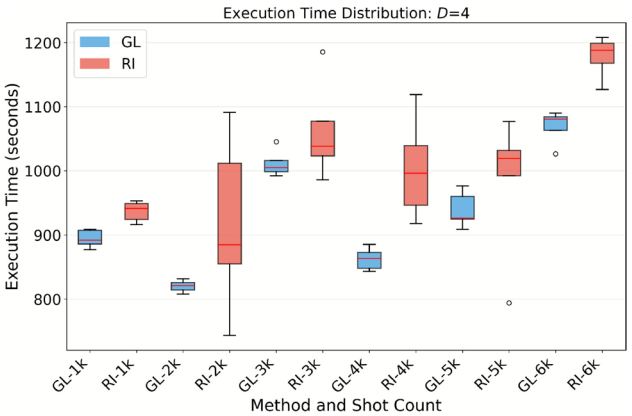

Fig. 16 The implementation time comparison between the GL and the RI strategies for the channel limitation $D = 4$

case, where GL maintains perfect approximation ratios from 2k-6k shots while RI shows degraded performance at lower shot counts. Even under the most restrictive single-channel constraint (D=1), both methods achieve similar final performance, but GL provides more consistent results across all shot counts.

The execution time analysis reveals significant computational efficiency advantages for GL initialization. Examining the average execution times across different channel constraints, GL demonstrates substantial time savings: approximately 5-20% faster execution times compared to RI across most configurations. For instance, under D=1

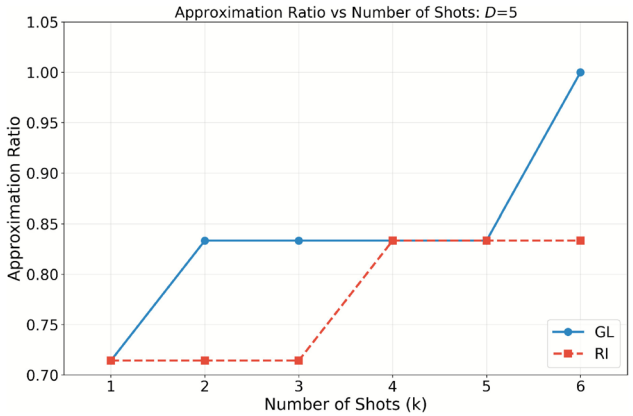

Fig. 17 The approximation ratio $\mathcal{A}$ comparison between the GL and the RI strategies for channel limitation $D = 5$

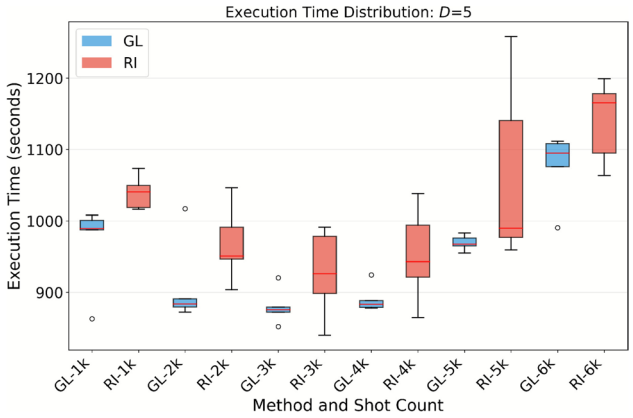

Fig. 18 The implementation time comparison between the GL and the RI strategies for the channel limitation $D = 5$

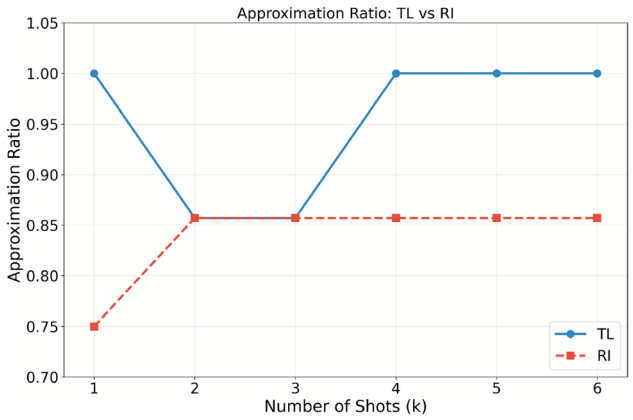

Fig. 19 The approximation ratio $\mathcal{A}$ comparison between the TL and the RI strategies for channel limitation $D = 3$

constraints, GL shows average time around 900-1,100 seconds while RI requires 1,150-1,300 seconds. Under D=2 constraints, GL maintains execution times of 950-1,200 seconds compared to RI's 950-1,370 seconds. This efficiency advantage becomes even more pronounced when considering that GL achieves superior approximation ratios with

Fig. 20 The implementation time comparison between the TL and the RI strategies for the channel limitation $D = 3$

reduced execution shots, effectively providing both better solution quality and faster convergence.

6.4 Transfer learning strategy for channel limitation

Given the high cost of training the QAOA parameter dataset under the channel limitation scenario and the limited amount of pretraining data, Transfer learning is applied to transfer the backbone graph Transformer of the OPP task to the backbone forecasting model for the channel limitation scenario (denoted as TL in the following). As proof of the concept, we test the TL strategy on the IEEE 24-bus system with the channel limitation $D = 3$.

In Figs. 19 and 20, the approximation ratio and the time efficiency of the proposed TL strategy on the 24-bus test case with $D = 3$ channel limitation constraints are demonstrated. Compared with the RI strategy, we observe exceptional performance with perfect approximation ratios achieved at both low-resource (1k shots) and high-resource (4k-6k shots) configurations, indicating robust optimization capabilities across varying quantum resource constraints. In contrast, random initialization exhibits gradual improvement from 0.75 to 0.86 but plateaus at significantly suboptimal performance levels. These results represent a substantial 14-33% improvement by the proposed TL strategy in solution quality when the quantum computation resources are limited. It should be noted that with 1k shots the proposed TL strategy can yield a global optimal result. The superior performance observed with reduced shot counts can be attributed to the stochastic sampling variance that introduces beneficial perturbations into the objective function landscape, thereby preventing premature convergence of the COBYLA optimizer to suboptimal local minima. In terms of time efficiency, transfer learning uses around 750 to 1,100 seconds of implementation time, achieves up to a 25% reduction in average computational time,

and exhibits significantly lower variation compared to random initialization.

In addition, compared with the GL strategy that is trained on a small amount of channel limitation dataset shown in Fig. 13, the utilized TL strategy can also demonstrate superior advantages when operating under data-limited conditions. While the graph learning method with channel limitation (D=3) can also work on a smaller pretraining dataset due to computational constraints, it achieves only gradual performance improvement, maintaining stable but suboptimal approximation ratios across most shot counts before reaching optimal performance only at the highest resource allocation (6k shots). In stark contrast, transfer learning demonstrates immediate superior performance, achieving perfect optimization at minimal resource expenditure and maintaining this optimal performance consistently across higher shot counts. As for the execution time, the TL strategy can improve slightly in time efficiency over the GL strategy by approximately 8%. As a result, TL is recommended for the QAOA parameter forecasting when the feasible pretrained datasets for some real-world applications are hard to obtain.

In interpreting these results, it is important to emphasize that all numerical experiments are restricted to the IEEE 14- and 24-bus benchmarks. These systems are intentionally chosen because they are standard test cases in power systems and are among the largest instances for which the current quantum hardware can handle. However, as quantum computing continues to advance rapidly, it is reasonable to expect more real-world quantum applications for large-scale optimization problems in the near future.

7 Conclusion

In this work, we propose a graph learning strategy to provide an informed initialization of the QAOA parameters for the OPP task and the channel limitation scenario. When the quantum computation resources are restricted, especially the number of quantum measurement shots, using the Graph Transformer as the backbone of the graph learning architecture, the proposed strategy for the QAOA parameter initialization forecasting illustrates a superior advantage over the random initialization strategy in terms of approximation ratio and time efficiency. In addition, to address the difficulty of obtaining a sufficient number of QAOA parameter pretraining datasets for the channel limitation task, transfer learning is utilized to transfer the learned representation from the normal OPP task. The utilized transfer learning method has also demonstrated the advantage in both the approximation ratio and the time efficiency over random initialization of QAOA parameters. Generally, this work provides a new benchmark for NISQ-era quantum optimization pipelines in real-world applications.

In future work, we will incorporate the QAOA circuit depth as a learning target into the graph learning framework, ensuring the model is adaptive to different problem sizes and can automatically optimize both parameter initialization and circuit architecture simultaneously. Additionally, we plan to extend the transfer learning approach to larger power system networks and investigate its applicability to other combinatorial optimization problems in quantum computing.

Appendix A: Observability modeling for the OPP scenario

In this section, we illustrate the detailed algorithm for the OPP scenario in Algorithm 1.

Algorithm 1 Observability modeling algorithm for OPP.

Input : Adjacency matrix A, placement sequence X, bus set V
Output: Total number of observed buses O

```
Observability(A, X, V):
// Initialize placement and observation sets
I ← Initial PMU installing based on X;
D ← I;
P ← {};
// Calculate neighbors counter for each node in V
Q ← 𝒩(V);  // Q ∈ ℝ^{N×1}, counter of neighbors for all the nodes in 𝒱
for j ∈ I do
    TryNode(j);
O ← length of D;
return O;
TryNode(j):
for p ∈ ℒ(j) do
    if p ∉ D then
        if p ∉ P then
            // Add p to the potential observation set P if not already included
            P.add(p);
        // Increase the counter for potential observation
        P[p] ← P[p] + 1;
        if P[p] ≥ Q[p] − 1 then
            D.add(p);
            // Remove p from potential observation list
            delete P[p];
            TryNode(p);
```

Appendix B: Observability modeling for the channel limitation

In this section, the algorithm details of the OPP scenario with channel limitations are illustrated in Algorithm 2.

Algorithm 2 Observability modeling algorithm: channel limitations.

Input : Adjacency matrix A, placement sequence X, bus set V, channel limitation F
Output: Total number of observed buses O

```
CalObservability(A, X, V, F):
I ← Initial placement based on X;
Enqueue(𝒬, I) ; // All the PMU-installed buses enqueue
Θ ← 𝒬.size;
Y ← 0_{N×N};
O ← {};
D ← {} ;                  // Fully observable set
P ← {} ;              // Partially observable set
Q ← 𝒩(V);  // 𝒬 ∈ ℝ^{N×1}, counter of neighbors
// Continue to Algorithm 2
```

Algorithm 3 Channel limitation algorithm: Main processing.

```
// Continued from Algorithm 2
for θ = 1 to Θ do
    j ← Dequeue(Q);
    χ = Q[j];
    if χ > F + 1 then
        ρ ← sorted.L(j) ;      // Sort neighbors of j descending
        ϑ ← [p ∈ ρ and Y[jp] = Y[pj] = 0] ;
        // Unvisited branches
        ϒ ← ϑ[1 : F] ;                  // Top F buses
        if len(ϒ) < F then
            Ξ ← [p ∈ ρ and Y[pj] = Y[jp] = 1];
            ϒ += Ξ[1 : (F − len(ϒ))] ;      // Ensure F channels
        Δ ← [p ∈ Y[jp] = Y[pj] = 1] ;       // Visited branches of j
        P[j] = len(Δ ∪ ϒ);
        if P[j] ≥ Q[j] then
            D.add(j) ;              // j is observable
            Enqueue(Q, j) ;
            delete P[j];
        for p ∈ ϒ do
            if Y[jp] = Y[pj] = 0 then
                Y[jp] = Y[pj] = 1;
                if p ∉ O then
                    P[p] ← P[p] + 1;
                    if P[p] ≥ Q[p] − 1 then
                        D.add(p);
                        Enqueue(Q, p) ;
                        delete P[p];
    else
        D.add(j);
        for p ∈ L(j) do
            if Y[jp] = Y[pj] = 0 then
                Y[jp] = Y[pj] = 1;
                if p ∉ O then
                    P[p] ← P[p] + 1;
                    if P[p] ≥ Q[p] − 1 then
                        D.add(p);
                        Enqueue(Q, p) ;
                        delete P[p];
// Continue to Algorithm 2
```

Algorithm 4 Channel limitation algorithm: Final processing.

```
// Continued from Algorithm 2
while Q is not empty do
    κ ← Dequeue(Q);
    for p ∈ L(κ) do
        if Y[κp] = Y[pκ] = 0 then
            Y[κp] = Y[pκ] = 1;
            if p ∉ O then
                P[p] ← P[p] + 1;
                if P[p] ≥ Q[p] − 1 then
                    D.add(p);
                    Enqueue(Q, p) ;
                    delete P[p];
O ← length of R;
return O;
```

Acknowledgements This work is supported by the Office of Naval Research under the award N00014-22-1-2504, the National Science Foundation under the award OAC-2417773.

Author Contributions Y.J. conceived the research idea, conducted the experiments, processed the data and results, and drafted the manuscript. X.W. contributed to data preparation and manuscript editing. Z.L. assisted with manuscript editing. Y.L. supervised the research and provided guidance throughout the study. T.M., P.L., and M.B. contributed to manuscript editing. All authors reviewed and approved the final manuscript.

Data Availability Data will be made available on request.

Declarations

Competing interests The authors declare that they have no conflicts of interest.

References

Almasabi S, Mitra J (2019) A fault-tolerance based approach to optimal PMU placement. IEEE Trans Smart Grid 10(6):6070–6079

Arute F et al (2019) Quantum supremacy using a programmable superconducting processor. Nature 574:505–510. https://doi.org/10.1038/s41586-019-1666-5

Bach B, Falla J, Safro I (2024) MLQAOA: Graph Learning Accelerated Hybrid Quantum-Classical Multilevel QAOA. In: 2024 IEEE International conference on quantum computing and engineering (QCE), vol 1, pp 1–12. https://doi.org/10.1109/QCE60285.2024.00072

Bhasin NK (2024) Enhancing quantum machine learning algorithms for optimized financial portfolio management. In: 2024 Third international conference on intelligent techniques in control, optimization and signal processing (INCOS). IEEE, pp 1–7

Bhattacharjee R, De A (2023) A novel bus-ranking-algorithmbased heuristic optimization scheme for PMU placement. IEEE Trans Industr Inf 19(9):9921–9932

Blekos K et al (2024) A review on Quantum Approximate Optimization Algorithm and its variants. In: Physics Reports 1068, pp 1–66. https://doi.org/10.1016/j.physrep.2024.03.002

Bonde B, Patil P, Choubey B (2023) The future of drug development with quantum computing. In: High performance computing for drug discovery and biomedicine. Springer, pp 153–179

Boulebnane S et al (2025) Evidence that the Quantum Approximate Optimization Algorithm Optimizes the Sherrington-Kirkpatrick Model Efficiently in the Average Case. In: arXiv preprint arXiv:2505.07929. https://doi.org/10.48550/arXiv.2505.07929

Brandhofer S et al (2023) Benchmarking the performance of portfolio optimization with QAOA. Quantum Inf Process 22(1):25. https://doi.org/10.1007/s11128-022-03766-5

Bucher D et al (2024) Towards Robust Benchmarking of Quantum Optimization Algorithms. In: 2024 IEEE International Conference on Quantum Computing and Engineering (QCE). IEEE, pp 159–170. https://doi.org/10.1109/QCE60285.2024.11030870

Chen X et al (2020) PMU placement for measurement redundancy distribution considering zero injection bus and contingencies. IEEE Syst J 14(4):5396–5406

Falla J et al (2024) Graph representation learning for parameter transferability in quantum approximate optimization algorithm. Quantum Mach Intell 6(2):46

Farhi E, Goldstone J, Gutmann S (2014) A quantum approximate optimization algorithm. In: arXiv preprint arXiv:1411.4028

Huang L et al (2013) Optimal PMU placement considering controlled islanding of power system. IEEE Trans Power Syst 29(2):742–755

Jiang Y et al (2025) Optimal PMU Placement via Quantum Optimization. IEEE Trans Smart Grid 16(4):3125–3141. https://doi.org/10.1109/TSG.2025.3564889

Khairy S et al (2020) Learning to optimize variational quantum circuits to solve combinatorial problems. In: Proceedings of the AAAI conference on artifficial intelligence 34(03):2367–2375

Korkali M, Abur A (2009) Placement of PMUs with channel limits. In: 2009 IEEE power & energy society general meeting. IEEE, pp 1–4

Kumar S (2014) Optimal placement of PMU using probabilistic approach. In: 2014 Recent Advances in Engineering and Computational Sciences (RAECS). IEEE, pp 1–6

Liang Z et al (2024) Graph learning for parameter prediction of quantum approximate optimization algorithm. In: Proceedings of the 61st ACM/IEEE Design automation conference, pp 1–4

Li H et al (2020) Rethinking the hyperparameters for fine-tuning. In: International Conference on Learning Representations (ICLR). OpenReview preprint. https://openreview.net/forum?id=B1g8VkHFPH

Li J et al (2021) Drug discovery approaches using quantum machine learning. In: 2021 58th ACM/IEEE Design automation conference (DAC). IEEE, pp 1356–1359

Lubinski T et al (2024) Optimization Applications as Quantum Performance Benchmarks. In: ACM Transactions on quantum computing 5(3)1–8:44. https://doi.org/10.1145/3678184

Maji TK, Acharjee P (2017) Multiple solutions of optimal PMU placement using exponential binary PSO algorithm for smart grid applications. In: IEEE Transactions on industry applications 53(3):2550–2559

Mandelbaum R et al (2025) How IBM will build the world's First large-scale, fault-tolerant quantum computer. https://www.ibm.com/quantum/blog/large- cale-ftqc. IBM Quantum Blog, 10 June 2025. Accessed 24 July 2025

Morstyn T (2022) Annealing-based quantum computing for combinatorial optimal power flow. In: IEEE Transactions on smart Grid 14(2):1093–1102

Montanez-Barrera JA, Willsch D, Michielsen K (2025) Transfer learning of optimal QAOA parameters in combinatorial optimization. In: Quantum information processing 24(5):1–18

Rahman NH, Zobaa AF (2016) Optimal PMU placement using topology transformation method in power systems. In: Journal of advanced research 7(5):625–634

Salloum H et al (2024) Quantum congestionfocused traffic optimization (Q-CFTO): enhancing traffic congestion solutions with quantum annealing. In: Authorea Preprints

Sack SH, Serbyn M (2021) Quantum annealing initialization of the quantum approximate optimization algorithm". In: quantum, 5:491

Shafiullah Md et al (2019) A modified optimal PMU placement problem formulation considering channel limits under various contingencies. Measurement 135:875–885

Shaydulin R et al (2022) Parameter Transfer for Quantum Approximate Optimization of Weighted MaxCut. In: arXiv preprint. https://doi.org/10.48550/arXiv.2201.11785

Vaswani A et al (2017) Attention is all you need. In: Proceedings of the 31st International conference on neural information processing systems. NIPS'17. Long Beach, California, USA: Curran Associates Inc., 2017, pp 6000–6010. ISBN: 9781510860964

Vlachogiannis JG, Ostergaard J (2009) Reactive power and voltage control based on general quantum genetic algorithms. In: Expert systems with applications 36(3):6118–6126

William Y et al (2011) Optimal PMU placement: A comprehensive literature review. In: (2011) IEEE Power and energy society general meeting. IEEE 2011:1–8

Wu J, He J, Xu J (2019) Net: Degree-specific graph neural networks for node and graph classification. In: Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining, pp 406–415

Wurtz J et al (2023) Aquila: QuEra's 256-qubit neutral-atom quantum computer. In: arXiv preprint (2023). arXiv: 2306.11727 [quant-ph]

Zhang Y, Zhang R, Potter AC (2021) QED driven QAOA for network-ow optimization. In: Quantum 5, p 510

Zhou L et al (2020) Quantum approximate optimization algorithm: Performance, mechanism, and implementation on near-term devices. In: Physical review X 10(2):021067