


Algorithm-oriented qubit mapping for variational quantum algorithms

YanJun Ji^{1,*}, Xi Chen^{2,†}, Ilia Polian^{1,‡} and Yue Ban^{2,§}

¹*Institute of Computer Architecture and Computer Engineering, University of Stuttgart, Pfaffenwaldring 47, 70569 Stuttgart, Germany*

²*Instituto de Ciencia de Materiales de Madrid (CSIC), Cantoblanco, E-28049 Madrid, Spain*

 (Received 23 July 2024; revised 19 November 2024; accepted 11 February 2025; published 11 March 2025)

Quantum algorithms implemented on near-term devices require qubit mapping due to noise and limited qubit connectivity. In this paper we propose a strategy called algorithm-oriented qubit mapping (AOQMAP) that aims to bridge the gap between exact and scalable mapping methods by utilizing the inherent structure of algorithms. While exact methods provide optimal solutions, they become intractable for large circuits. Scalable methods, like SWAP networks, offer fast solutions but lack optimality. AOQMAP bridges this gap by leveraging algorithmic features and their association with specific device substructures to achieve depth-optimal and scalable solutions. The proposed strategy follows a two-stage approach. First, it maps circuits to subtopologies to meet connectivity constraints. Second, it identifies the optimal qubits for execution using a cost function and performs postselection among execution results across subtopologies. Notably, AOQMAP provides both scalable and optimal solutions for variational quantum algorithms with fully connected two-qubit interactions on common subtopologies including linear, T-, and H-shaped, minimizing circuit depth. Benchmarking experiments conducted on IBM quantum devices demonstrate significant reductions in gate count and circuit depth compared to Qiskit, Tket, and SWAP network. Specifically, AOQMAP achieves up to an 82% reduction in circuit depth and an average 138% increase in success probability. This scalable and algorithm-specific approach holds the potential to optimize a wider range of quantum algorithms.

DOI: [10.1103/PhysRevApplied.23.034022](https://doi.org/10.1103/PhysRevApplied.23.034022)

I. INTRODUCTION

Recent strides in variational quantum algorithms (VQAs) [1,2] have demonstrated considerable potential in solving complex problems, such as combinatorial optimization [3] and quantum simulation of materials [4], surpassing the efficiency of classical algorithms. However, existing constraints of quantum processing units (QPUs) pose a significant obstacle to realizing the full capabilities of VQAs. A major challenge is the presence of noise, limited number of qubits, and restricted connectivity between qubits. These constraints impede the scalability and applicability of VQAs in tackling larger and more intricate problems. Efforts to overcome these challenges are critical for unlocking the complete potential of VQAs in practical applications. Specifically, the execution of quantum algorithms requires compilation before being deployed on a quantum device, which involves addressing connectivity constraints by introducing SWAP gates and decomposing algorithms into native hardware basis gates. Moreover,

optimizing circuits to minimize the influence of noise is crucial to ensure accurate performance of algorithms [5].

A crucial step in compilation is mapping logical qubits to physical qubits available on quantum devices, a complex problem commonly referred to as qubit mapping [6–10]. The main objective of qubit mapping is to minimize the number of inserted SWAP gates or circuit depth and to maximize circuit fidelity. This problem is identified as NP-hard [11], underscoring the necessity for efficient and effective methods to tackle it. Qubit mapping can be expressed as a mathematical optimization problem and solved using constraint satisfaction techniques. These approaches are called exact methods and have been investigated in various studies [12–16]. While exact methods provide high-quality and stable solutions, their compilation time increases exponentially with problem size. In contrast, heuristic approaches [9,14,15,17–19] prioritize efficiency by providing fast solutions without guaranteeing optimality. Another approach involves constructing swap layers [20–24] aimed at providing scalable solutions. However, similar to heuristic methods, optimality is not guaranteed.

This paper proposes an efficient, two-stage approach to qubit mapping, prioritizing both optimality and scalability. In the first stage, quantum algorithm is mapped to the

*Contact author: quantumji@hotmail.com, y.ji@fz-juelich.de

†Contact author: xi.chen@csic.es

‡Contact author: ilia.polian@informatik.uni-stuttgart.de

§Contact author: yue.ban@csic.es

target QPU's subtopologies to address connectivity constraints. Here, we provide optimal and scalable solutions with minimal circuit depth for VQAs with all-to-all connected two qubit interactions on common subtopologies including linear, T-, and H-shaped. For VQAs with partially connected two-qubit interactions, solutions can be obtained by optimizing the initial qubit order to minimize CX gate count. These solutions are applicable to various noisy intermediate-scale quantum (NISQ) devices, such as Google's Sycamore, IBM's QPUs, and Rigetti's processors. After addressing connectivity constraints, the second stage focuses on determining optimal qubits for execution and postselecting the execution outcomes associated with different subtopologies. Each subtopology-adapted circuit is mapped onto the QPU taking into account real-time noise characteristics. After executing all subtopology-adapted circuits on the quantum device, the results corresponding to different subtopology types are postselected to ensure high performance of algorithms.

Unlike conventional methods that usually map predefined circuits, our approach begins directly from the algorithm's Hamiltonian, allowing for optimization during the conversion process from Hamiltonian to circuit representation. This codesign strategy generates mappings tailored to the algorithm's structure and hardware constraints. This methodology promises not only optimality but also enhanced adaptability and scalability in tackling the qubit mapping challenge. Benchmarks on six IBM QPUs with 7, 27, and 127 qubits demonstrate significant performance gains. In particular, we achieve an average reduction of 31% (up to 82%) in circuit depth and an average increase of 138% in success probability compared to Qiskit [25], Tket [26], and SWAP network [20].

The paper is structured as follows. Section II provides background on NISQ devices and the qubit mapping problem. Section III details our proposed approach, including the identification of subtopologies, analysis of routing solutions, introduction of mapping strategies, and discussion on optimality and scalability. Section IV demonstrates practical applications of our method to VQAs, in particular, the quantum approximate optimization

algorithm (QAOA) and variational quantum eigensolver (VQE), as well as the broader implications for executing other algorithms on various quantum devices. Section V presents benchmarking results comparing our technique to existing methods using both simulators and real quantum hardware. Finally, Sec. VI concludes.

II. BACKGROUND

A. Near-term quantum devices

Prominent near-term quantum platforms exhibit distinctive characteristics. Superconducting qubits demonstrate increased connectivity that extends beyond nearest-neighboring qubits [27–29], while trapped ions showcase relatively long coherence times [30], and photons exhibit low noise [31]. In the era of NISQ computing [32], the development of practical algorithms encounters substantial constraints due to the inherent limitations of quantum hardware. For instance, the 7-, 27-, and 127-qubit IBM QPUs illustrated in Fig. 1, exhibit restricted connectivity. Specifically, each qubit can only directly interact with up to three neighboring qubits. Additionally, the noise in quantum systems varies over time, leading to temporal fluctuations in errors. These hardware limitations pose challenges for implementing quantum algorithms that rely on long-range qubit interactions or high precision. Current endeavors concentrate on enhancing the control of qubits by improving coherence times, elevating gate fidelities, and expanding qubit connectivity. Moreover, compiling quantum algorithms to reduce circuit depth and mitigate errors is essential to fully realize the potential of these devices.

To address the limitations of NISQ devices, researchers have developed VQAs [1], which are hybrid algorithms combining classical optimization techniques with quantum resources. However, implementing VQAs on NISQ devices presents challenges. One significant hurdle is selecting an appropriate classical optimization algorithm, which can profoundly impact the success of VQAs. Moreover, mapping algorithms onto physical qubits of quantum devices, especially for larger circuits, is difficult.

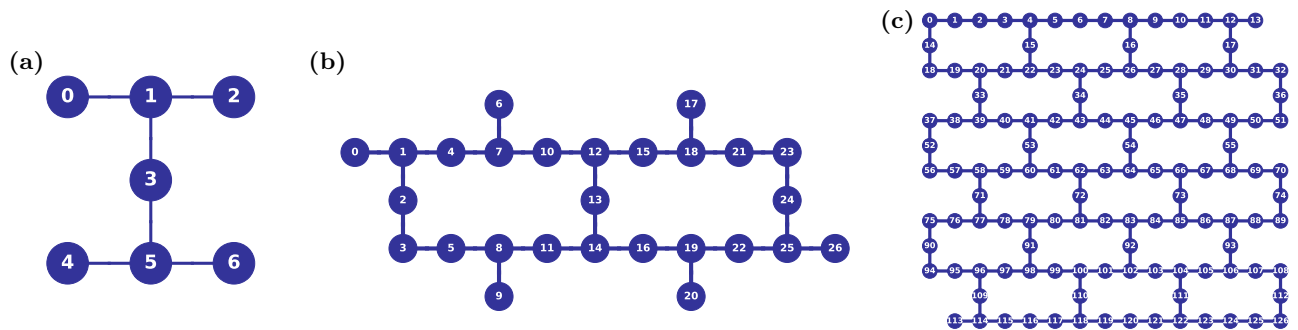


FIG. 1. Topologies of IBM QPUs with (a) 7, (b) 27, and (c) 127 qubits. Circles denote individual qubits. Lines connecting the circles represent available two-qubit gates between corresponding qubit pairs.

Strategies to tackle these challenges have been developed, such as employing machine-learning techniques to optimize parameters [33], developing alternative ansatz to capitalize on qubit connectivity [34], and implementing efficient compilation processes [35–40]. In this study, our focus is on efficient qubit mapping, which is a crucial step in implementing VQAs. We examine the challenges and opportunities associated with qubit mapping and propose an approach to improve scalability and optimality. We assess the effectiveness of our approach and demonstrate its application on real quantum hardware.

B. Qubit mapping problem

The primary objective of qubit mapping is to minimize errors inherent in the implementation of algorithms on QPUs. This task is critical in mitigating errors arising from the noisy nature of qubits, particularly in the context of high error rates associated with two-qubit gates such as CNOT or CX that can significantly impact the overall performance of algorithms. Optimizing the qubit mapping process requires a careful balance between two crucial factors. First, the insertion of SWAP gates introduces errors when connecting two qubits that are not directly linked, as the implementation of a SWAP gate necessitates three CX gates. Second, the quality of qubits and qubit pairs can vary, necessitating the identification of the most suitable qubits for circuit execution. While searching for solutions of qubit mapping on a specific subtopology can markedly reduce computational complexity compared to exploring solutions across the entire topology, particularly for larger topologies with hundreds of qubits, it is crucial to strike the right balance between the number of SWAP gates inserted on specific topologies and the quality of qubits on those topologies.

III. METHODOLOGY AND PRELIMINARIES

This section details our methodology. Figure 2 illustrates the algorithm-oriented qubit mapping (AOQMAP) flow. The process begins with decomposing the input VQA into a two-qubit Hamiltonian. After identifying N subtopologies within the target QPU, the Hamiltonian is routed onto subtopologies by introducing SWAP gates, ensuring compliance with hardware connectivity constraints. Each of these routed circuits is then decomposed into native basis gates of the target QPU, followed by an optimization step to reduce redundant gates in circuit and improve fidelity. In this work, the decomposition and optimization process is performed using Qiskit transpiler [25] with default settings (optimization level 1). Finally, a cost function is utilized to select the optimal set of qubits for execution. After executing N circuits on the target QPU, a postselection process can be applied to determine results associated with different subtopologies by minimizing the expectation value of the problem Hamiltonian, as will be detailed later. Verification is a technique used to protect against implementation errors that can occur during processes such as routing, decomposition, and optimization. While optional, the verification process, represented by dashed lines, is particularly recommended for small circuits. For verification, a reference circuit is constructed directly from the two-qubit Hamiltonian, matching parameters and gate sequences of the routed circuit. The Hellinger distance between output distributions of the reference and routed circuits can be employed to assess correctness. Additional details on the verification process are provided in Appendix A. This comprehensive flow guarantees effective adaptation and optimization of quantum circuits within constraints imposed by target hardware topology.

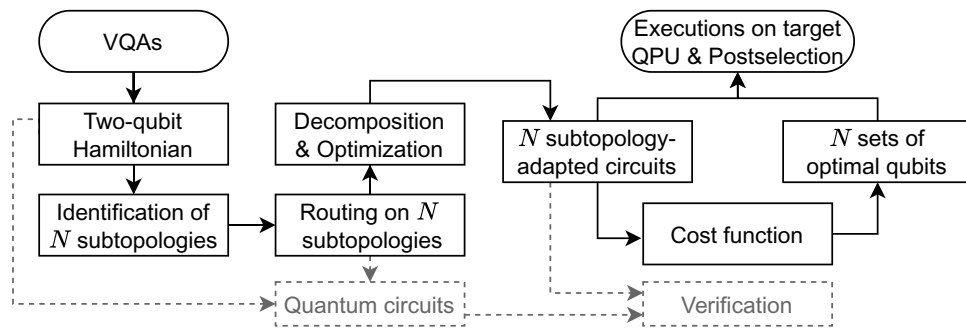


FIG. 2. Algorithm-oriented qubit mapping (AOQMAP) flow is outlined for mapping VQAs onto a target QPU. The AOQMAP approach differs from traditional qubit mapping methods in that it starts from a two-qubit Hamiltonian, rather than a predefined circuit. The process of AOQMAP involves two main steps: (1) adapting the VQA to N subtopologies of target QPU; and (2) selecting N optimal mapping schemes to implement such subtopology-adapted circuits, followed by executions on the QPU and postselection. The adaptation process ensures algorithm excitability, while the mapping process guarantees the use of high-quality qubits for execution. By first adapting circuits and then choosing an optimal mapping scheme, AOQMAP minimizes errors and optimizes algorithm performance. Finally, the N results produced by the QPU, each associated with a different subtopology, are postselected by minimizing the expectation value of the problem Hamiltonian. Dashed lines represent the circuit verification process, which is optional but recommended for small circuits. A more detailed discussion of circuit verification can be found in Appendix A.

Below, we detail each step. Beginning with the identification of three common subtopologies, we present optimal and scalable solutions for routing VQAs on these topologies, focusing on fully connected two-qubit interactions. Additionally, we employ a qubit selection strategy to map subtopology-adapted circuits onto high-quality qubits of the QPU and introduce the postselection process to determine the results obtained from different types of subtopologies. Finally, we conclude by analyzing the optimality and scalability of our proposed methods.

A. Identification of subtopologies

We identify $N = 3$ prevalent subtopologies within IBM QPUs: linear, T-, and H-shaped configurations. Given the NP-hard nature of qubit mapping, we focus on symmetric subtopologies to facilitate the development of optimal and scalable solutions. Linear configurations serve as a fundamental topology, while T- and H-shaped topologies (formally defined in Sec. III B) represent the simplest symmetric extensions. Moreover, to achieve optimality and scalability, we will employ exact methods for small-scale problems and then develop scalable approaches (see Sec. III B). The symmetric T- and H-shaped subtopologies enable efficient analysis of solutions and offer potential for extending these solutions to other topologies. Future research will explore these extensions. Additionally, our analysis of 27-qubit IBM QPUs reveals that these subtopologies are dominant for the problem sizes considered. Specifically, we exhaustively identify all possible subtopologies of up to seven qubits and calculate their corresponding layouts within the QPU using mapomatic [41]. Figure 3 illustrates these subtopologies, and Table I summarizes the numbers of their corresponding layouts. As shown in Table I, linear topologies exhibit the

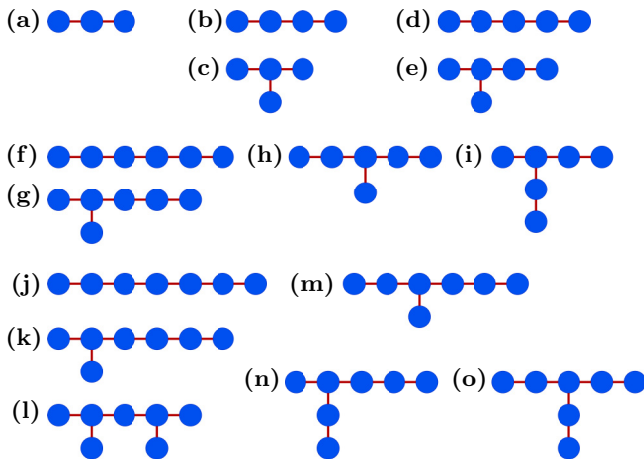


FIG. 3. All possible subtopologies within the heavy-hex topology for varying numbers of qubits, ranging from 3 to 7. The most common subtopology on a 27-qubit topology [Fig. 1(b)] is linear, followed by T-shaped and then H-shaped.

TABLE I. Number of layouts within a 27-qubit QPU that enable the execution of circuits satisfying the connectivity constraints of subtopologies shown in Figs. 3(a)–3(o).

Topology	3	4	5	6	7
Linear	74 (a)	80 (b)	100 (d)	104 (f)	132 (j)
T-shaped		48 (c)	36 (e)	64 (g)	48 (k)
H-shaped					56 (l)
T-variant				24 (h)–(i)	44 (m)–(n), 12 (o)

highest number of layouts, followed by T- and H-shaped configurations.

To identify the optimal subtopology among the three candidates, a postselection strategy is employed. This approach is essential when the optimal subtopology is not immediately evident. The cost function, which will be detailed in Sec. III C, incorporates gate fidelity to select the optimal set of qubits for executing subtopology-adapted circuits. In addition to enabling the selection of qubits for a given topology, this cost function allows for the distinction between various types of subtopologies. However, each subtopology exhibits unique crosstalk effects that are not accounted for in the calibration data of IBM QPUs. Furthermore, as will be demonstrated subsequently, increased qubit connectivity reduces the number of two-qubit gates required but increases circuit depth, creating a trade-off between circuit fidelity and execution time. The cost function based solely on gate fidelity is insufficient for accurately assessing the quality of a circuit. To overcome these limitations, a postselection procedure is performed after executing circuits corresponding to each subtopology on QPUs. The criterion for postselection is based on minimizing the expectation value of the problem Hamiltonian, which allows us to identify the most effective subtopology. This will be investigated in Sec. V C. The method is also demonstrated by applying AOQMAP to a digitized counterdiabatic quantum optimization algorithm in Ref. [42].

B. Subtopology-aware circuit adaptation

This section introduces the methodology for subtopology aware circuit adaptation on NISQ devices. We focus on linear, T-, and H-shaped configurations, and develop strategies for adapting VQAs to each subtopology.

In this study, we concentrate on the QAOA applied to dense portfolio optimization problems, where each qubit necessitates interaction with all other qubits. As detailed in Ref. [43], the problem Hamiltonian for n asset portfolio optimization is expressed as

$$H_c = \sum_{i=1}^{n-1} \sum_{j=i+1}^n c_{ij} Z_i Z_j + \sum_{i=1}^n c_i Z_i + c_0. \quad (1)$$

Here, $Z_i Z_j$ represents ZZ interaction between qubits i and j , defined by $ZZ(\theta) = e^{-i\frac{\theta}{2}Z \otimes Z}$ with rotation angle θ . Z_i denotes Pauli-Z operator acting on qubit i . The coefficients c_{ij} , c_i , and c_0 are real numbers determined by parameters in the portfolio optimization problem, where c_0 is a constant term. The coefficients c_{ij} and c_i are defined as follows:

$$c_{ij} = \frac{\lambda}{2}(q\sigma_{ij} + A), \quad (2)$$

$$c_i = \frac{\lambda}{2} \left[A(2B - n) + (1 - q)\mu_i - q \sum_{j=1}^n \sigma_{ij} \right], \quad (3)$$

where λ is the global scaling factor, q represents risk preference, σ_{ij} is the covariance between assets i and j , A is the penalty factor, B is the number of assets to be selected, and μ_i is the expected return of asset i .

We consider the mixing operator in QAOA with the form

$$H_m = \sum_{i=1}^n X_i, \quad (4)$$

where X_i is Pauli- X operator acting on qubit i . The QAOA commences by selecting an initial state, denoted by $|\psi_0\rangle$, which is the eigenstate of mixer Hamiltonian H_m . Subsequently, problem and mixer Hamiltonians are applied alternately to $|\psi_0\rangle$ in a parameterized quantum circuit of depth p . This yields a quantum state

$$|\psi_{\gamma, \beta}\rangle = \prod_{k=1}^p e^{-i\beta_k H_m} e^{-i\gamma_k H_c} |\psi_0\rangle, \quad (5)$$

where β and γ are p -dimensional vectors of rotation angles that control the evaluation of H_c and H_m in each layer k . The $2p$ parameters (γ, β) can be optimized using a classical solver to minimize the expectation value $\langle \psi_{\gamma, \beta} | H_c | \psi_{\gamma, \beta} \rangle$. This tuning process adjusts $|\psi_{\gamma, \beta}\rangle$ to approximate the ground state of H_c , offering a well-approximated solution to the optimization problem encoded in H_c .

To achieve optimal qubit-mapping solutions, a comprehensive analysis of problem and mixer Hamiltonians described in Eqs. (1) and (4) is essential. Notable observations guide our approach: (i) the two-qubit gates in Eq. (1) exclusively involve ZZ interactions. Their summation form implies that the order of application does not affect the outcome, allowing flexibility in mapping H_c ; (ii) the single rotation gates R_Z and R_X in H_c and H_m can be independently applied to each qubit without affecting others. Hence, qubit mapping is unnecessary for these gates; (iii) for high depths, as depicted in Eq. (5), a fixed gate arrangement for two-qubit gates is required at each depth

to maintain an equivalent H_c . However, since all ZZ gates commute with each other, they can be assigned in each depth with an arbitrary gate arrangement. This implies that the ZZ gate arrangement needed for implementing QAOA at each depth can be arbitrary, providing additional flexibility in qubit mapping. By leveraging these observations, we can formulate a tailored mapping method for quantum algorithms to optimize their performance, minimizing the number of CX gates, reducing circuit depth, and thereby maximizing the algorithm's overall performance.

In the upcoming sections, we explore qubit-mapping solutions on three subtopologies, emphasizing QAOA with all-to-all connected two-qubit gate interactions. Although our analysis focuses on this specific instance of VQAs, the implications extend to algorithms with analogous features. Specifically, we underscore the relevance of our findings to Hamiltonian with partially connected interactions, VQE, and other NISQ devices in Sec. IV.

1. Linear subtopology

In quantum computing, linear subtopology arranges qubits sequentially in a one-dimensional configuration, forming a linear chain or arrangement. To determine optimal mapping on this subtopology, we initially employ an exact method aimed at minimizing the circuit depth [16]. However, instead of mapping the entire algorithm, we focus solely on mapping the ZZ gates in Eq. (1), treating them as single entities and avoiding the decomposition into two CX gates to streamline the mapping process. Additionally, we narrow our attention to the first QAOA depth, significantly reducing the effort required to identify a solution.

The routing solution for ZZ gates in a five-qubit QAOA on linear topology is illustrated in Fig. 4(a). Assuming an arbitrary initial qubit order $[a, b, c, d, e]$, after each swap layer, qubit orders evolve as follows: $[a, c, b, e, d]$, $[c, a, e, b, d]$, and $[c, e, a, d, b]$. Since each qubit needs to interact with all other qubits, ZZ gate acts on following qubit pairs: for qubit a , (a, b) , (a, c) , (a, d) , and (a, e) ; for qubit b , (b, c) , (b, d) , and (b, e) ; for qubit c , (c, d) , and (c, e) ; and for qubit d , (d, e) . Our analysis reveals that all ten ZZ gates required in a five-qubit QAOA can be executed on a linear subtopology, regardless of the initial qubit order. This observation is highly beneficial for achieving scalability, as it facilitates straightforward extension to high QAOA depths by repeating these swap layers. Similarly, as depicted in Fig. 4(b), the ZZ gates required for six-qubit QAOA can be implemented using four swap layers. We observe that in an n -qubit QAOA, ZZ gates are organized into n layers to minimize circuit depth. The swap layers, excluding the first and last layers, are positioned after each ZZ layer. As shown in Fig. 4(c), due to the cancellation of CX gates between ZZ and SWAP gates, each SWAP gate following a ZZ gate introduces only one

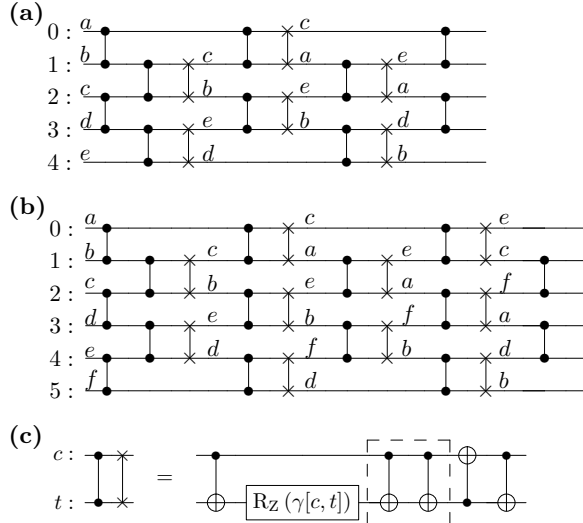


FIG. 4. Routing solutions of ZZ gates in QAOA at depth $p = 1$ with (a) five and (b) six qubits. (c) ZZ gate on qubit pair (c, t) with rotation angle $\gamma[c, t]$, followed by a SWAP gate, with the box showing CX gate cancellation between ZZ and SWAP gates.

additional CX gate. The optimal routing solutions of ZZ gates in QAOA on linear subtopology exhibit a structure similar to the SWAP network presented in Ref. [20], with the absence of the first and last swap layers. This similarity enables the extension of solutions to an arbitrary number of qubits. The dispensability of the first and last swap layers stems from the flexibility to adjust initial qubit order and measurement order to eliminate SWAP gates in the first and last ZZ layers, leveraging commutativity of ZZ and SWAP gates. Once two-qubit gates are mapped, we can construct the entire QAOA circuit.

Algorithm 1 presents pseudocode for mapping QAOA on linear subtopology. The reconstruction process starts with an initialized qubit order $O = \{0, 1, \dots, i, \dots, j, n - 1\}$ for n qubits. We first prepare an initial state $|\psi_0\rangle$ by applying a Hadamard gate to each qubit. Then, we apply ZZ or ZZ-SWAP gate layer continuously according to the solutions presented in Fig. 4. The order of qubits i and j is exchanged only when a SWAP gate is applied to qubit pair (i, j) . To improve algorithm efficiency, it is crucial to optimize ZZ and ZZ-SWAP gates using strategies such as gate cancellation or pulse-level optimization techniques [44,45]. Finally, R_Z gates in problem Hamiltonian and R_X gates in mixer Hamiltonian are implemented on qubits with appropriate parameters. Figure 5 shows the resulting circuit of a five-qubit QAOA at depth $p = 1$. All ZZ, R_Z , and R_X gates are executed according to the current qubit order, followed by measurement of qubits.

As demonstrated, an arbitrary qubit order can implement all required ZZ gates in QAOA with n qubits using $n - 2$ swap layers. A solution of depth p can be obtained

ALGORITHM 1. AOQMAP for QAOA on linear subtopology.

Input: Number of qubits n , QAOA depth p ,
Parameters $\gamma[c, t]$, $\alpha[i]$, and $\beta[j]$ of gates ZZ
on qubit pair (c, t) , R_Z on qubit i , and R_X on
qubit j , respectively, where
 $c, t, i, j \in \{0, \dots, n - 1\}$ and $c < t$

Output: Circuit satisfying connectivity constraints

```

1 Function ApplyZZGate( $O, i, j$ )
2   | Apply ZZ( $\gamma[O[i], O[j]]$ ) on  $(i, j)$ 
3 end
4 Function ApplyZZSWAPGate( $O, i, j$ )
5   | Apply ZZ( $\gamma[O[i], O[j]]$ ) on  $(i, j)$ 
6   | Apply SWAP on  $(i, j)$ 
7   |  $O[i] \leftrightarrow O[j]$  // Exchange qubit order
8 end
9  $O \leftarrow \{0, 1, \dots, n - 1\}$  // Initialize the qubit order
10 Prepare the initial state  $|0\rangle^{\otimes n}$ 
11 Apply  $H^{\otimes n}$ 
12 while  $p > 0$  do
13    $s \leftarrow 0$ 
14   while  $s < n$  do
15     for  $q := 0$  to  $n - 1$  step 2 do
16       if  $s == 0$  or  $s == n - 1$  then
17         | ApplyZZGate( $O, q, q + 1$ )
18       else
19         | ApplyZZSWAPGate( $O, q, q + 1$ )
20       end
21     end
22      $s \leftarrow s + 1$ 
23     if  $s < n$  then
24       for  $q := 1$  to  $n - 1$  step 2 do
25         if  $s == 0$  or  $s == n - 1$  then
26           | ApplyZZGate( $O, q, q + 1$ )
27         else
28           | ApplyZZSWAPGate( $O, q, q + 1$ )
29         end
30       end
31     end
32      $s \leftarrow s + 1$ 
33   end
34   for  $k := 0$  to  $n$  do
35     | Apply  $R_Z(\alpha[O[k]])$  on  $k$ 
36     | Apply  $R_X(\beta[O[k]])$  on  $k$ 
37   end
38    $p \leftarrow p - 1$ 
39 end
40 Measure the qubits ( $[0, \dots, n - 1] \rightarrow [O[0], \dots, O[n - 1]]$ )

```

by repeating p times swap layers in depth $p = 1$ circuit. Inserting these swap layers consecutively, the initial order returns after $2n$ swap layers. Since each QAOA depth introduces $n - 2$ swap layers, circuit repeats every $\text{LCM}(2n, n - 2)/(n - 2)$ depths, where $\text{LCM}(2n, n - 2)$ is the least common multiple of $2n$ and $n - 2$. For odd numbers of qubits, one repetition of swap layers in depth $p = 1$ circuit demonstrates symmetry, and at depth two, the final qubit order returns directly to the initial qubit order. For even numbers of qubits, one repetition of such swap layers

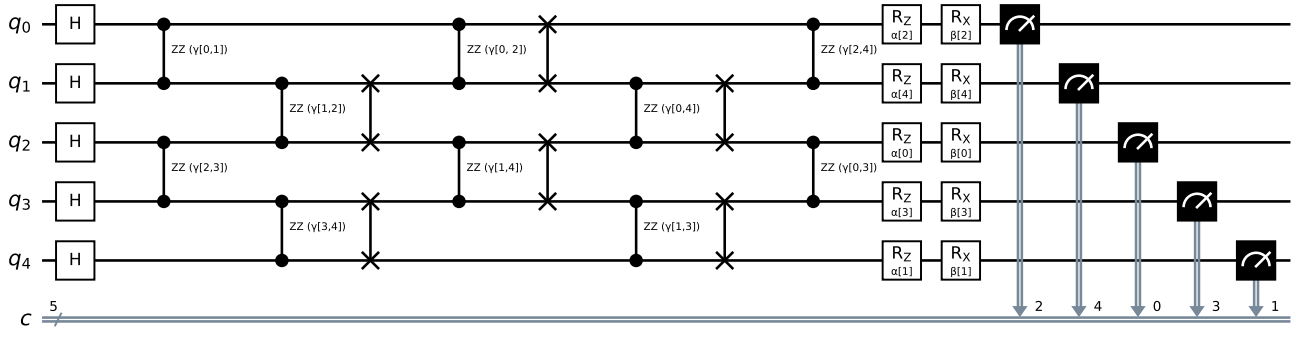


FIG. 5. Resulting circuit of five-qubit QAOA for portfolio optimization on a linear subtopology using the AOQMAP approach. This circuit is further decomposed and optimized for the target QPU. The optimal mapping scheme is then selected for execution based on the QPU’s noise information.

demonstrates alternating odd- and even-numbered swap layers, resulting in circuits that repeat a subcircuit with the same gate arrangement every $\text{LCM}(2n, n-2)/(n-2)$ layers. Since all ZZ gates commute with each other, these two constructions of high depth solutions are equivalent. However, we can utilize mirror symmetry of swap layers to construct high depth circuits for even numbers of qubits and obtain circuits repeating every two depths. Additionally, mirror symmetry can also be utilized to construct algorithms with partially connected two qubit gates, which we discuss in Sec. IV.

2. T-shaped subtopology

We now analyze solutions obtained by exact method [16] for two-qubit gates in QAOA with five and six qubits on T-shaped subtopology. A T-shaped subtopology

features a central qubit at the apex of the “T” shape, serving as the primary qubit. The arms of T-shaped topology consist of one or more qubits that are linearly connected to the central qubit. The qubits in arms typically do not have direct interactions with each other. We note that the minimum number of qubits on a T-shaped subtopology is four. Figure 6(a) shows the definition of T-shaped topology for n qubits with qubit 2 as the center qubit. Consider an arbitrary initial qubit order $[a, b, c, d, e]$ for the solution of five qubit, as shown in Fig. 6(b). Following each swap layer, the qubit order evolves sequentially: $[a, b, d, c, e]$, $[d, b, a, e, c]$, and $[d, b, e, a, c]$. This sequence allows us to execute all required ten ZZ gates for five-qubit QAOA. Similarly, the four swap layers in six-qubit QAOA are capable of performing the necessary 15 ZZ gates, as shown in Fig. 6(c).

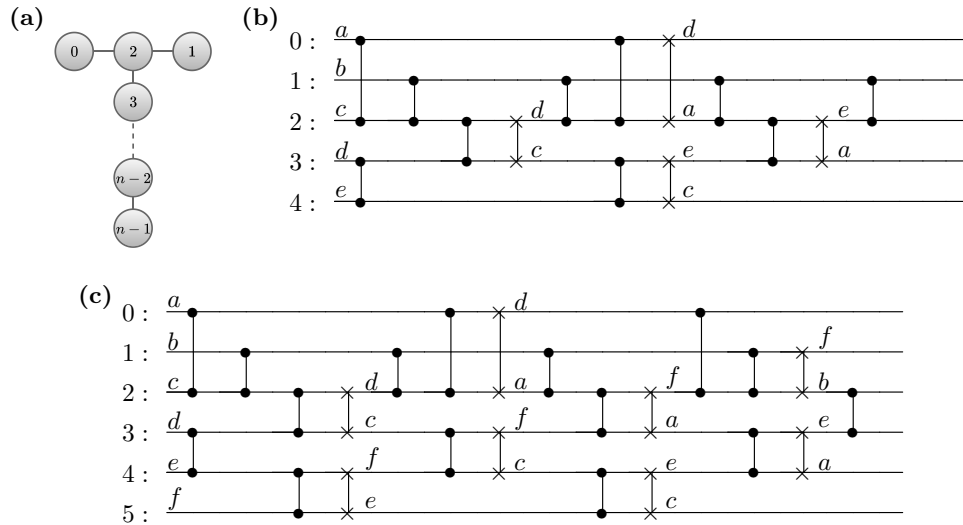


FIG. 6. Definition of T-shaped topology and corresponding routing solutions of ZZ gates in QAOA at depth $p = 1$. (a) T-shaped topology defined for n qubits with qubit 2 as the center qubit. The connectivity of this subtopology for n qubits is given by $\{(0, 2), (1, 2), (2, 3), (3, 4), \dots, (n-2, n-1)\}$. (b) Five-qubit routing solution. (c) Six-qubit routing solution.

Algorithm 2 outlines the procedure for generating n swap layers for n -qubit QAOA on T-shaped subtopology. As depicted in Fig. 6(c), the initial swap layer starts at center qubit 2 and consists of SWAP gates on qubit pairs $\{(2, 3), (4, 5), \dots, (n_{\text{odd}} - 1, n_{\text{odd}})\}$ with $n_{\text{odd}} = (n - 1) - 1 + (n - 1) \bmod 2$. This layer alternates with another layer that starts from two different qubits in short arms of a T-shaped structure, forming two distinct layers $\{(0, 2), (3, 4), \dots, (n_{\text{even}} - 1, n_{\text{even}})\}$ and $\{(1, 2), (3, 4), \dots, (n_{\text{even}} - 1, n_{\text{even}})\}$ with $n_{\text{even}} = (n - 1) - (n - 1) \bmod 2$. After constructing n swap layers, all required ZZ gates can be implemented on connected qubit pairs. Algorithm 3 presents pseudocode for AOQMAP applied in n -qubit QAOA at depth $p = 1$ on T-shaped subtopology. We begin the process by initializing a qubit order $O = \{0, 1, \dots, i, \dots, j, n - 1\}$ for n qubits, with qubit 2 being center qubit. This initial qubit order enables executing one layer of ZZ gates. One possible option is gates on qubit pairs of the center qubit and two qubits in short arm, and gates on qubit pairs starting with the first qubit in long arm. For instance, the first ZZ layer in six-qubit QAOA consists of $\{(0, 2), (1, 2), (3, 4)\}$. The second layer starts with center qubits and proceeds to qubits on long arm to form ZZ gates on qubit pairs $\{(2, 3), (4, 5)\}$, followed by the first swap layer that introduces new qubit order and provides opportunity for additional gate implementation. We first execute remaining ZZ gates that are not on qubit pairs of the second swap layer. Then, we implement remaining ZZ gates on qubit pairs of the second swap layer, followed by SWAP gates. This process ensures that ZZ gate is positioned immediately before SWAP gate on the corresponding qubit pair, thereby enabling CX gate cancellation. We maintain this iterative process until all ZZ gates are implemented.

ALGORITHM 2. Swap layers on T-shaped subtopology.

Input: Number of qubits n , Connectivity of T-shaped topology defined in Fig. 6(a)
Output: List of swap layers S

```

1 begin
2    $n_{\text{odd}} \leftarrow (n - 1) - 1 + (n - 1) \bmod 2$ ;
3    $n_{\text{even}} \leftarrow (n - 1) - (n - 1) \bmod 2$ ;
4    $S \leftarrow$  empty List;
5    $j \leftarrow 0$ ;
6   while  $j < n$  do
7      $S[j] \leftarrow S[j] \cup \{(2, 3), (4, 5), \dots, (n_{\text{odd}} - 1, n_{\text{odd}})\}$ ;
8     if  $++j \geq n$  then break;
9      $S[j] \leftarrow S[j] \cup \{(0, 2), (3, 4), \dots, (n_{\text{even}} - 1, n_{\text{even}})\}$ ;
10    if  $++j \geq n$  then break;
11     $S[j] \leftarrow S[j] \cup \{(2, 3), (4, 5), \dots, (n_{\text{odd}} - 1, n_{\text{odd}})\}$ ;
12    if  $++j \geq n$  then break;
13     $S[j] \leftarrow S[j] \cup \{(1, 2), (3, 4), \dots, (n_{\text{even}} - 1, n_{\text{even}})\}$ ;
14    if  $++j \geq n$  then break;
15  end
16 end

```

ALGORITHM 3. AOQMAP for QAOA at depth $p = 1$ on T-shaped subtopology.

Input: Number of qubits n , Parameters $\gamma[c, t]$, $\alpha[i]$, and $\beta[j]$ of gates ZZ on qubit pair (c, t) , R_z on qubit i , and R_x on qubit j , respectively, where $c, t, i, j \in \{0, \dots, n - 1\}$ and $c < t$
Output: Circuit satisfying connectivity constraints

```

1  $O \leftarrow \{0, 1, \dots, n - 1\}$  // Initialize the qubit order
2  $L \leftarrow$  List of all ZZ gates
3  $E \leftarrow$  List of connected edges on T-shaped subtopology
4  $E_O \leftarrow \{(O[r], O[s]) \text{ for } (r, s) \in E\}$ 
5  $S \leftarrow$  List of  $n$  swap layers // Algorithm 2
6  $S_O \leftarrow \{(O[r], O[s]) \text{ for } (r, s) \in S[k]\}$ 
7 Function ApplyZZGate( $O, c, t$ )
8   | Apply ZZ( $\gamma[c, t]$ ) on  $(O.\text{index}(c), O.\text{index}(t))$ 
9 end
10 Function ApplySWAPGate( $O, i, j$ )
11   | Apply SWAP on  $(i, j)$ 
12   |  $O[i] \leftrightarrow O[j]$  // Exchange qubit order
13 end
14 Prepare the initial state  $|0\rangle^{\otimes n}$ 
15 Apply  $H^{\otimes n}$ 
16 for  $k := 0$  to  $n$  do
17   if  $L$  is empty then continue
18   foreach ZZ( $\gamma[c, t]$ )  $\in L$  do
19     if  $(c, t) \in E_O$  and  $(c, t) \notin S_O$  then
20       | ApplyZZGate( $O, c, t$ )
21       |  $L.\text{remove}(\text{ZZ}(\gamma[c, t]))$ 
22     end
23   end
24   if  $L$  is empty then continue
25   foreach  $(i, j) \in S[k]$  do
26      $c, t = O[i], O[j]$ 
27     if  $c > t$  then
28       |  $c \leftrightarrow t$  // Exchange
29     end
30     if ZZ( $\gamma[c, t]$ )  $\in L$  then
31       | ApplyZZGate( $O, c, t$ )
32       |  $L.\text{remove}(\text{ZZ}(\gamma[c, t]))$ 
33       if  $L$  is empty then continue
34       | ApplySWAPGate( $O, i, j$ )
35     end
36   end
37 end
38 foreach SWAP on  $(i, j)$  located at the circuit end do
39   | Remove SWAP
40   |  $O[i] \leftrightarrow O[j]$ 
41 end
42 for  $k \leftarrow 0$  to  $n$  do
43   | Apply Rz( $\alpha[O[k]]$ ) on  $k$ 
44   | Apply Rx( $\beta[O[k]]$ ) on  $k$ 
45 end
46 Measure the qubits  $([0, \dots, n - 1] \rightarrow [O[0], \dots, O[n - 1]])$ 

```

Similar to linear subtopology, there are two solutions for higher depths: repeating swap layers in depth $p = 1$ circuit and leveraging mirror symmetry of swap layers. Alternating swap layers and their mirrors causes gates in QAOA

at depth p to invert gates at the previous depth. For commuting gates such as the ZZ gates used in QAOA, this reversal has no impact on algorithm performance. However, for noncommuting gates, this inversion may suppress Trotter errors [46,47], making AOQMAP promising for optimizing other algorithms. As with the linear topology, T-shaped subtopology also requires $n - 2$ swap layers for n -qubit QAOA, which we will discuss in more detail in Sec. III D. It is worth noting that SWAP gates at the end of a circuit can be eliminated, as no remaining two-qubit gates require the introduction of other qubit orders.

The T-shaped subtopology affords enhanced qubit connectivity for the central qubit, reducing required SWAP gates but increasing circuit depth. This implies that T-shaped subtopology is advantageous when the algorithm's fidelity is a primary factor affecting performance. Conversely, linear subtopology is more effective for larger circuit sizes where schedule duration is paramount.

3. H-shaped subtopology

The H-shaped subtopology shares similarities with T-shaped, but has two central qubits at each end instead of one. The horizontal segments of “H” serve as a bridge connecting central qubits. To implement an H-shaped subtopology, a minimum of six qubits is required. We define the connectivity of an H-shaped subtopology with center qubits 2 and $n - 3$ as $\{(0, 2), (1, 2), (2, 3), \dots, (n - 4, n - 3), (n - 3, n - 2), (n - 3, n - 1)\}$ for n qubits.

Algorithm 4 presents a pseudocode for generating n swap layers that enable the implementation of Hamiltonian with fully connected two-qubit gates on H-shaped topology. For odd numbers of qubits, the first and third swap layers differ in initial connections of $(0, 2)$ and $(1, 2)$, which alternate. Similarly, the second and fourth layers differ in final connections of $(n_{\text{even}} - 2, n_{\text{even}} - 1)$ and $(n_{\text{even}} - 2, n_{\text{even}})$. For even numbers of qubits, the first and third layers are identical, which include connections between two center qubits. In contrast, the second and fourth layers differ in the first and last connections, where $(1, 2)$ and $(n_{\text{odd}} - 2, n_{\text{odd}} - 1)$ are for the second layer and $(0, 2)$ and $(n_{\text{odd}} - 2, n_{\text{odd}})$ are for the fourth layer. This alternating pattern of connections between neighboring swap layers efficiently constructs the set of minimized SWAP gates for VQAs with arbitrary numbers of qubits mapped to H-shaped topology.

The procedure to construct a depth-one circuit with arbitrary qubit number n on H-shaped subtopology is similar to Algorithm 3 for T-shaped subtopology. The list of connected edges E is updated to reflect the H-shaped connectivity, and the list of n swap layers S is generated according to Algorithm 4 for H-shaped subtopology. Similarly, repeating the same swap layers at depth $p = 1$ circuit or leveraging mirror symmetry extends solutions to high depths. Compared to linear and T-shaped

ALGORITHM 4. Swap layers on H-shaped subtopology.

Input: Number of qubits n
Output: List of swap layers S

```

1 begin
2    $n_{\text{odd}} \leftarrow (n - 1) - 1 + (n - 1) \bmod 2;$ 
3    $n_{\text{even}} \leftarrow (n - 1) - (n - 1) \bmod 2;$ 
4    $S \leftarrow$  empty List;
5    $j \leftarrow 0;$ 
6   if  $n \bmod 2 \neq 0$  then
7     while  $j < n$  do
8        $S[j] \leftarrow$ 
9          $S[j] \cup [(0, 2), (3, 4), \dots, (n_{\text{odd}} - 2, n_{\text{odd}} - 1)];$ 
10      if  $++j \geq n$  then break;
11       $S[j] \leftarrow$ 
12         $S[j] \cup [(2, 3), (4, 5), \dots, (n_{\text{even}} - 2, n_{\text{even}} - 1)];$ 
13      if  $++j \geq n$  then break;
14       $S[j] \leftarrow$ 
15         $S[j] \cup [(1, 2), (3, 4), \dots, (n_{\text{odd}} - 2, n_{\text{odd}} - 1)];$ 
16      if  $++j \geq n$  then break;
17       $S[j] \leftarrow S[j] \cup [(2, 3), \dots, (n_{\text{even}} - 4, n_{\text{even}} - 3), (n_{\text{even}} - 2, n_{\text{even}})];$ 
18      if  $++j \geq n$  then break;
19    end
20  else
21    while  $j < n$  do
22       $S[j] \leftarrow$ 
23         $S[j] \cup [(2, 3), (4, 5), \dots, (n_{\text{even}} - 2, n_{\text{even}} - 1)];$ 
24      if  $++j \geq n$  then break;
25       $S[j] \leftarrow$ 
26         $S[j] \cup [(1, 2), (3, 4), \dots, (n_{\text{odd}} - 2, n_{\text{odd}} - 1)];$ 
27      if  $++j \geq n$  then break;
28       $S[j] \leftarrow$ 
29         $S[j] \cup [(0, 2), \dots, (n_{\text{odd}} - 4, n_{\text{odd}} - 3), \dots, (n_{\text{odd}} - 2, n_{\text{odd}})];$ 
30      if  $++j \geq n$  then break;
31    end
32  end
33 end

```

topologies, H-shaped topology enables additional connections for two center qubits that reduce required SWAP gates but increase circuit depth. Additionally, the H-shaped subtopology requires $n - 1$ swap layers to achieve full connectivity, compared to $n - 2$ for linear and T-shaped subtopologies. Further details can be found in Sec. III D.

C. Mapping of subtopology-adapted circuits

The next crucial step is to map these subtopology-aware circuits onto target quantum device by selecting an optimal qubit mapping scheme. To identify a high-quality qubit set, we utilize a cost function denoted as C , which takes into account the error rate of each gate and measurement in the

circuit and is given by

$$C = 1 - \prod_{i=1}^{N_g} (1 - p_{gi}) \prod_{j=1}^{N_m} (1 - p_{mj}), \quad (6)$$

where N_g is number of gates in circuit, p_{gi} is error rate of the i th gate, N_m is number of measurements, and p_{mj} is error rate of the j th measurement. A lower value of C indicates a higher estimated fidelity. The error rates of gates and measurements can be obtained from device calibration data. To select optimal qubits, we first utilize mapomatic [41] to identify all layouts on QPU matching connectivity of adapted circuit. We evaluate circuit on each layout, choosing the one with the highest fidelity for execution. Since IBM QPUs like Fig. 1 contain more linear and T-shaped subtopologies than H-shaped, we focus on mapping to linear and T-shaped in our demonstrations. Additionally, H-shaped subtopologies on IBM QPUs are limited to specific qubit numbers such as 7, 9, 11, 13, and 15, whereas linear and T-shaped provide more flexibility.

We adopt a postselection process to determine between linear and T-shaped mappings instead of relying solely on cost function evaluation. Specifically, we execute circuits on both subtopologies and use postselection to select the circuit corresponding to the minimum expectation value of the problem Hamiltonian. While fidelity estimates from the cost function are valuable, factors such as gate scheduling can also impact algorithm performance [45]. Moreover, different gate arrangements may be equivalent in the absence of noise but behave differently under actual hardware noise. Therefore, we further introduce an additional variant called AOQMAP-LS, which performs AOQMAP on linear subtopology with mirror symmetric swap layers in depth $p = 1$.

D. Optimality and scalability

The optimal swap strategy on line topology with n qubits has been proven to necessitate $n - 2$ total swap layers to achieve fully connected two qubit gates [23]. This reduction of two swap layers has also been reported in previous work (e.g., Ref. [48]). However, this paper reaches the same conclusion by analyzing solutions obtained from an exact solver for small-scale instances, providing independent verification and a potentially more generalizable perspective on minimizing swap layer overhead. These $n - 2$ swap layers ensure scalability with respect to the number of qubits, and the approach can be extended to arbitrary depth p by repeating the swap layers or alternating them with their symmetric counterparts. For Hamiltonians with partially connected two-qubit interactions, scalability in terms of depth is preserved through mirror symmetry. However, with respect to the number of qubits, this study emphasizes the critical role of the initial qubit order in reducing the number of swap gates required. In

particular, we can minimize the number of CX gates by optimizing the initial qubit mapping, which we will discuss in Sec. IV A, providing alternative insights into efficient routing strategies.

Similarly, we derive depth optimal and scalable solutions for VQAs with arbitrary depth p and number of qubits on T- and H-shaped topologies. The depth optimality generated by AOQMAP-T and AOQMAP-H can be established by analyzing the interactions between their unique structures and shared linear chains. For the T-shaped subtopology, two overlapping linear chains are present:

$$0 \leftrightarrow 2 \leftrightarrow \dots \leftrightarrow n - 1, \quad \text{and} \quad 1 \leftrightarrow 2 \leftrightarrow \dots \leftrightarrow n - 1, \quad (7)$$

which share the qubits $\{2, 3, \dots, n - 1\}$, referred to as the shared chain. Each chain has $n - 1$ qubits, requiring $n - 3$ swap layers to achieve full connectivity. However, branching qubits 0 and 1, both connected to qubit 2, cannot simultaneously interact with the rest of the qubits. Sequential swaps involving $(0, 2)$ and $(1, 2)$ introduce an additional layer. After incorporating qubit 0 into the shared chain (which takes one timestep), the longest remaining distance is between logical qubits located on physical qubits 1 and $n - 1$, corresponding to a chain of $n - 1$ qubits, requiring $n - 3$ additional swap layers. Thus, the T-shaped subtopology requires at least $n - 2$ total swap layers to achieve depth optimality, validating the depth optimality of AOQMAP-T.

For the H-shaped subtopology, there are four overlapping linear chains that share the qubits $\{2, 3, \dots, n - 3\}$:

$$\begin{aligned} 0 &\leftrightarrow 2 \leftrightarrow \dots \leftrightarrow (n - 3) \leftrightarrow (n - 2), \\ 0 &\leftrightarrow 2 \leftrightarrow \dots \leftrightarrow (n - 3) \leftrightarrow (n - 1), \\ 1 &\leftrightarrow 2 \leftrightarrow \dots \leftrightarrow (n - 3) \leftrightarrow (n - 2), \\ 1 &\leftrightarrow 2 \leftrightarrow \dots \leftrightarrow (n - 3) \leftrightarrow (n - 1). \end{aligned}$$

Each linear chain consists of $n - 2$ qubits, which require $n - 4$ swap layers to achieve full connectivity. However, the end-localized qubits $\{0, 1\}$, connected to qubit 2, and $\{n - 2, n - 1\}$, connected to qubit $n - 3$, cannot interact with the rest of the network until swaps propagate along the shared chain $\{2, 3, \dots, n - 3\}$. As demonstrated in Algorithm 4, three additional steps are required before the last end-localized qubit(s) are incorporated into the network for both even and odd numbers of qubits. These three steps account for the sequential propagation of swaps to fully integrate the branching qubits at both ends of the topology. Combining these three steps with the $n - 4$ swap layers required for the $n - 2$ qubit chain results in a total of $n - 1$ swap layers, thereby establishing the depth optimality of the H-shaped subtopology. Furthermore, our investigation reveals that the final swap layer requires only

a single SWAP gate to implement the last remaining ZZ gate.

The scalability of AOQMAP to large quantum systems, such as those comprising 100 qubits, is essential for its practical applicability. To compare scalability, we measure compilation time to obtain optimal mappings that minimize circuit depth using an exact approach proposed in Ref. [16]. The original work on the compiler in Ref. [16] focused on an entire topology rather than a specific substructure. Therefore, we map QAOA circuits onto the entire topology of a 27-qubit IBM QPU, as depicted in Fig. 1. For three-qubit QAOA circuits, the compilation time of an exact algorithm increases exponentially with higher depth p , expanding from seconds at depth 1 to over 5 days at depth 7. Similarly, four-qubit compilation requires 13 s for depth 1 but grows substantially to over 15 h for depth 2 and more than a week for depth 3. Furthermore, increasing qubit number from 3 to 9 lengthens compilation from 3 s to over 41 h. In contrast, our approach intrinsically generalizes solutions for arbitrary depth and number of qubits without requiring any computational effort.

AOQMAP achieves both optimality and scalability by analyzing the optimal solutions of small QAOA instances obtained using the exact method [16] in conjunction with the scalable solutions provided by SWAPNK [20]. Depth optimality is achieved by setting the objective to minimize circuit depth in the exact method [16], while scalability is ensured by preserving the structural properties of small-circuit solutions as the system size increases, leveraging symmetric subtopologies (linear, T-, and H-shaped). By integrating insights from both optimal and scalable approaches, AOQMAP provides depth-optimal and scalable solutions. This methodology not only bridges the gap between exact and scalable approaches but also offers significant insights for addressing other routing problems, advancing both theoretical understanding and practical applications.

While the mapping of Hamiltonians to circuits is inherently scalable, other stages, such as verification, qubit selection, and subtopology identification, require careful attention to ensure feasibility at larger scales. As detailed in Appendix A, the verification process currently relies on classical simulation to validate the correctness of routing, decomposition, and optimization processes, which is effective for smaller circuits but is not scalable to larger systems due to the exponential growth in computational resources. To overcome this limitation, the verification step can be validated on small circuits and subsequently omitted for larger ones. Alternatively, techniques such as ZX-calculus-based circuit verification [49] provide a practical solution, having demonstrated the capability to handle systems with hundreds of qubits. Qubit selection in AOQMAP is performed using the mapomatic framework, which has been demonstrated to scale to systems with hundreds of qubits

[41], ensuring its suitability for large problem instances. Postselection, the final step in the workflow, evaluates and selects the optimal outcome among the three identified subtopologies by minimizing the expectation value of the problem Hamiltonian. This process imposes minimal computational overhead, supporting the scalability of AOQMAP.

For subtopology selection, AOQMAP currently supports linear, T-, and H-shaped configurations. To meet the demands of larger or more complex systems, future work will extend these subtopologies to include variants with additional qubit connections, providing enhanced flexibility while maintaining computational efficiency. If no predefined subtopologies are sufficient, routing can still be performed on the available subtopologies using compilers such as Qiskit or Tket, which reduce computational complexity compared to routing on the entire topology while maintaining high-quality results. The advantages of mapping onto subtopologies followed by postselection will be discussed in Sec. VB4. By integrating these strategies into the workflow, AOQMAP ensures scalability for large quantum systems. This efficient and adaptive process balances classical and quantum resources, making it well suited for real-world problems on near-term quantum devices.

IV. APPLICATIONS

A. QAOA for MaxCut on noncomplete graphs

QAOA is a VQA designed specifically to address combinatorial optimization problems, such as the MaxCut problem for graphs [3]. This problem involves partitioning nodes of a graph into two distinct groups to maximize the number of edges that connect these groups [50]. The problem Hamiltonian in QAOA for MaxCut problem is represented as

$$H_p = \frac{1}{2} \sum_{i,j} (1 - Z_i Z_j), \quad (8)$$

where i and j are two nodes of an edge. Compared to QAOA for portfolio optimization [Eq. (1)], QAOA for the MaxCut problem does not include Pauli-Z items, implying the absence of R_Z gates in quantum circuit. QAOA for MaxCut starts with state preparation and then alternately applies the problem and mixer Hamiltonian to evolve the state toward an optimal solution. QAOA for MaxCut problem on complete graphs follows a similar approach to QAOA for portfolio optimization. However, in noncomplete graphs, the lack of connectivity necessitates the exclusion of corresponding ZZ gates, introducing additional challenges for qubit mapping.

For VQAs with partially connected two-qubit gates, swap layers obtained from fully connected interactions

can still ensure that the circuit satisfies connectivity constraints. In such cases, ZZ-SWAP gate corresponding to the missing edge is replaced by a SWAP gate, allowing for correct execution of quantum circuits on subtopologies. However, the presence of remaining SWAP gates due to the lack of two-qubit interactions significantly amplifies errors caused by noise. One solution is to optimize initial mapping or initial qubit order such that all SWAP gates are placed at the end of circuit. These end-located SWAP gates can then be removed by adjusting measurement order accordingly. Furthermore, if a SWAP gate is placed behind a ZZ gate in the first ZZ layer, it can be removed by adjusting the initial qubit order since ZZ and SWAP gates commute.

Algorithm 5 presents pseudocode for mapping QAOA for MaxCut on noncomplete graphs by optimizing initial qubit order. For n qubits, there are $n!/2$ distinguished permutations due to symmetry. Different initial qubit orders introduce different gate arrangements, resulting in different numbers of CX gates. By minimizing additional CX gates, we can obtain an optimized qubit mapping. Practically, we can calculate the optimal or minimum number of CX gates in the resulting circuit. This optimal solution arranges all existing ZZ gates in consecutive layers until all gates are implemented. Then, we remove every SWAP gate behind the first ZZ layer and the one located at the end of circuit. This yields solutions with the minimum number of CX gates. To accelerate the search process, we can set the calculated optimal CX gate count as a target and terminate optimization once an initial qubit order achieves this value. It is important to note that the optimal initial order is not unique. Alternatively, we can employ a heuristic approach, where a set of initial qubit order permutations is searched, and the one with the fewest number of CX gates is selected [42]. For higher depth, we can obtain the solution by utilizing mirror symmetry, which involves alternating between swap layers at depth $p = 1$ and their corresponding mirrors, resulting in a circuit repeating every two depths.

B. Variational quantum eigensolver

In the previous section, we presented optimal routing solutions for QAOA on different types of subtopologies, including linear, T-, and H-shaped. One notable advantage of our approach is its applicability to other VQAs, such as VQE, without requiring additional computational resources. This is because both VQAs involve sequences of parameterized single-qubit rotations and fixed two qubit operations. Therefore, optimal qubit routing solutions that we derived for QAOA based on subtopology connectivities can be easily adapted.

VQE [51–53] is designed to determine the ground-state energy or eigenvalue of a Hamiltonian. It has broad applications in various fields such as quantum chemistry [54],

ALGORITHM 5. AOQMAP for QAOA at depth $p = 1$ with partially connected ZZ interactions.

Input: Number of qubits n , Parameters $\gamma[c, t]$ and $\beta[j]$ of gates ZZ on qubit pair (c, t) and R_X on qubit j , respectively, where $c, t, j \in \{0, \dots, n-1\}$ and $c < t$

Output: Circuit satisfying connectivity constraints

```

1  $O \leftarrow \{0, 1, \dots, n-1\}$  // Initialize the qubit order
2  $L \leftarrow$  List of all existing ZZ gates
3 Function MapTwoQubitGates( $O, L, \gamma$ )
4    $O_0 \leftarrow O$  // Initial qubit order
5    $qc_{zz} \leftarrow$  Quantum circuit initialized to  $|0\rangle^{\otimes n}$ 
6   foreach ZZ in  $L$  do
7     Assign ZZ( $\gamma[c, t]$ ) to  $qc_{zz}$  according to  $O_0$  and
8     swap layers obtained for fully connected two
9     qubit interactions and update current qubit
10    order  $O$  if SWAP is applied
11  end
12   $O_f \leftarrow O$  // Final qubit order
13  foreach SWAP on  $(i, j)$  located at the end of  $qc_{zz}$ 
14  do
15    Remove SWAP
16     $O_f[i] \leftrightarrow O_f[j]$ 
17  end
18  foreach SWAP on  $(i, j)$  located at the end of the
19  first ZZ layer of  $qc_{zz}$  do
20    Remove SWAP
21     $O_0[i] \leftrightarrow O_0[j]$ 
22  end
23  return ( $qc_{zz}, O_f$ )
24 ( $qc_{zz}^{opt}, O_f$ ) = MapTwoQubitGates ( $O, L$ )
25  $n_{cx}^{opt} \leftarrow$  CX gate count of  $qc_{zz}^{opt}$ 
26  $\mathcal{O}_q \leftarrow$  List of defined qubit orders
27 for  $O_i$  in  $\mathcal{O}_q$  do
28   ( $qc_{zz}, O_{f_i}$ ) = MapTwoQubitGates ( $O_i, L$ )
29    $n_{cx} \leftarrow$  CX gate count of  $qc_{zz}$ 
30   if  $n_{cx} < n_{cx}^{opt}$  then
31      $n_{cx}^{opt} = n_{cx}$ 
32      $qc_{zz}^{opt} = qc_{zz}$ 
33      $O_f = O_{f_i}$ 
34   end
35 end
36 Prepare the initial state  $|0\rangle^{\otimes n}$ 
37 Apply  $H^{\otimes n}$ 
38 Apply  $qc_{zz}^{opt}$ 
39 for  $k \leftarrow 0$  to  $n$  do
40   Apply  $R_X(\beta[O_f[k]])$  on  $k$ 
41 end
42 Measure the qubits ( $[0, \dots, n-1] \rightarrow [O_f[0], \dots, O_f[n-1]]$ )

```

condensed-matter physics [55], and combinatorial optimization [56]. Let H be the Hamiltonian of a quantum system, and $|\psi\rangle$ a trial wave function. The Rayleigh-Ritz quotient is bounded below by the ground-state energy E_0

$$E_0 \leq \frac{\langle \psi | H | \psi \rangle}{\langle \psi | \psi \rangle}. \quad (9)$$

The objective is to determine a quantum state by examining a parameterized ansatz state, denoted as $|\psi(\theta)\rangle = U(\theta)|0\rangle$, to minimize expectation value of Hamiltonian. Here, $|0\rangle$ represents initial state, and $U(\theta)$ is the vector of parameters θ , also known as variational form or ansatz, which represents a parameterized unitary transformation achievable through quantum circuit. The selection of ansatz circuits plays a critical role in determining the efficacy of VQE. Three prominent categories of ansatz circuits include chemically inspired [57], hardware-efficient ansatz (HEA) [34], and Hamiltonian variational [58].

In this study, we examine VQE with full entanglement, as demonstrated in previous research (e.g., Ref. [59]). The ansatz circuit begins with a layer of parameterized R_Y gates, followed by controlled-Z (CZ) gates that serve as entangling gates. Unlike CX gate, which distinguishes between control and target qubits, CZ gate is undirected. After that, another set of parameterized R_Y gates is performed, succeeded by measurement. For higher p , the sub-circuit between the first layer and measurement is repeated p times. The circuit with n qubits and depth p contains $(p+1)n$ parameters that require optimization by a classical optimizer. The full entanglement is achieved through $n(n-1)/2$ CZ gates, each comprising one CX and two Hadamard gates.

Algorithm 6 describes the procedure of AOQMAP for VQE with full entanglement on linear subtopology. It differs from Algorithm 1 in the gates used. We initiate the implementation of CZ gate with one CX and two Hadamard gates that act on the physical qubit representing the target qubit of the CX gate. Subsequently, we perform CZ-SWAP gate by inserting a SWAP gate after CX gate and the second Hadamard gate on the physical qubit representing control qubit of CX instead of the target qubit, since the introduced SWAP gate alters qubit order. Analogously, we construct $n-2$ swap layers for n qubits on linear subtopology, meaning that CZ gates are performed on the first and last layers, while the constructed CZ-SWAP gates are implemented on the remaining layers. Finally, R_Y gates are assigned accordingly, followed by measurement. The qubit-mapping solution for VQE circuits on T- and H-shaped subtopologies can be attained similarly. Additionally, Algorithm 5 can be employed to obtain solutions for nonfully entangled VQE. The mapped circuit with depth $p=1$ on five qubits is depicted in Fig. 7. Each SWAP gate introduces another qubit order and adds one additional CX gate, resulting in a total of $p(n-1)^2$ CX gates. The incorporated SWAP gates guarantee that all requisite CZ gates can be executed. R_Y gates and measurement operators are then assigned according to the current qubit order.

The AOQMAP approach offers several advantages. First, solutions for different subtopologies can be easily adapted between VQAs. This means that once solutions are

ALGORITHM 6. AOQMAP for VQE on linear subtopology.

Input: Number of qubits n , VQE depth p , Vector of parameters θ with dimension $(p+1) \times n$
Output: Circuit satisfying connectivity constraints

```

1 Function ApplyCZGate( $i, j$ )
2   | Apply H on  $j$ 
3   | Apply CX on ( $i, j$ )
4   | Apply H on  $j$ 
5 end
6 Function ApplyCZSWAPGate( $O, i, j$ )
7   | Apply H on  $j$ 
8   | Apply CX on ( $i, j$ )
9   | Apply SWAP on ( $i, j$ )
10  |  $O[i] \leftrightarrow O[j]$  // Exchange qubit order
11  | Apply H on  $i$ 
12 end
13  $O \leftarrow \{0, 1, \dots, n-1\}$  // Initialize the qubit order
14 Prepare the initial state  $|0\rangle^{\otimes n}$ 
15 for  $i := 0$  to  $n$  do
16   | Apply  $R_Y(\theta[i])$  on  $i$ 
17 end
18  $p_{\max} \leftarrow p$ 
19 while  $p > 0$  do
20    $s \leftarrow 0$ 
21   while  $s < n$  do
22     for  $q := 0$  to  $n-1$  step 2 do
23       if  $s == 0$  or  $s == n-1$  then
24         | ApplyCZGate( $q, q+1$ )
25       else
26         | ApplyCZSWAPGate( $O, q, q+1$ )
27       end
28     end
29      $s \leftarrow s+1$ 
30     if  $s < n$  then
31       for  $q := 1$  to  $n-1$  step 2 do
32         if  $s == 0$  or  $s == n-1$  then
33           | ApplyCZGate( $q, q+1$ )
34         else
35           | ApplyCZSWAPGate( $O, q, q+1$ )
36         end
37       end
38        $s \leftarrow s+1$ 
39     end
40     for  $i := 0$  to  $n$  do
41       | Apply  $R_Y(\theta[i + n(p_{\max} - p + 1)])$  on  $O[i]$ 
42     end
43      $p \leftarrow p-1$ 
44 end
45 Measure the qubits ( $[0, \dots, n-1] \rightarrow [O[0], \dots, O[n-1]]$ )

```

found for a specific algorithm on target subtopology, they can be applied to other VQAs. Second, AOQMAP facilitates individual block optimization. For instance, the optimization of CZ-SWAP gate can be achieved by rearranging the SWAP gate before the second Hadamard gate, while altering the qubit that Hadamard gate acts upon. Moreover, this structure enables efficient pulse optimization [45] and application of error-mitigation strategies [42,60].

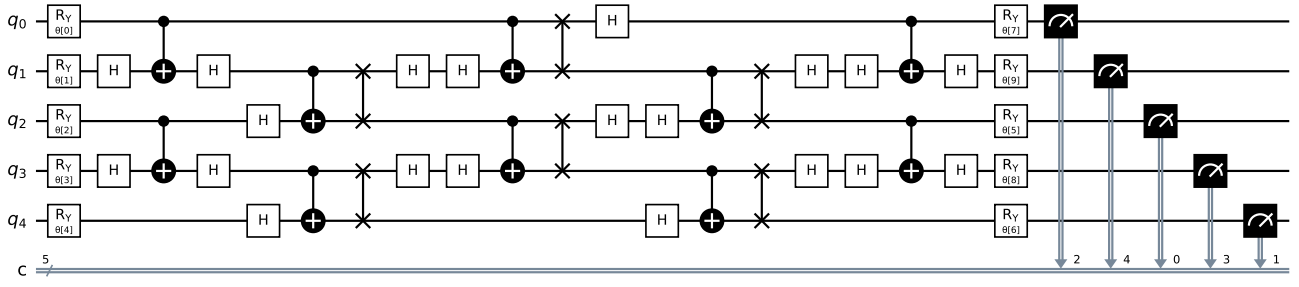


FIG. 7. Resulting circuit of a five-qubit VQE at depth $p = 1$ with full entanglement on a linear subtopology using AOQMAP. The circuit can be further optimized by canceling two Hadamard gates and performing CX gate cancellation for CX-SWAP gates.

C. NISQ devices and algorithms beyond VQAs

AOQMAP facilitates the transfer of solutions among diverse quantum devices, accounting for their specific architectures and noise characteristics. The routing solutions for VQAs on linear, T-, and H-shaped configurations can be adapted to other superconducting quantum devices. For instance, Google’s Sycamore processor [61] employs square lattice qubit connectivity providing more opportunities for the three subtopologies. An H-shaped subtopology can be readily implemented in the processor by identifying two neighboring square faces in the grid and their linear connection. Similarly, Rigetti’s processor [62] also exhibits such basic subtopologies. The adaptability is then achieved through a decomposition-optimization-remapping workflow, where the routed circuit on a subtopology is decomposed into basis gates of target device, optimized, and mapped onto hardware, taking into account the latest device calibration data. As alternative architectures proliferate, AOQMAP’s customized adaptability enables algorithms to be easily adjusted, making it a crucial advantage in the NISQ era.

The proposed method is also adaptable to other gate-based quantum architectures, such as trapped ions [63–65], where qubit connectivity and gate fidelities may differ significantly. Trapped-ion architectures typically support full connectivity on a one-dimensional chain. While this eliminates the need for SWAP layers, first decomposing the circuit into native gates and then selecting high-quality qubits remains essential for improving algorithm performance. Moreover, gate sequences introduced by SWAP layers provide guidance for implementing algorithms. Additionally, research has shown that gate fidelities decrease as the chain size increases [64,66], limiting scalability to a large number of qubits. An open question is whether focusing on neighboring qubit interactions can maintain high gate fidelities with increasing system size, thereby achieving scalability. Future research can compare the performance of fully connected two-qubit interactions versus neighboring qubit interactions using our routing solutions.

For photonic architectures [67–70], the universal quantum computing models, such as one-way or measurement-based [71–74], differ fundamentally from the standard quantum circuit model, requiring tailored compilation strategies for mapping circuits to photonic hardware [75–77]. While the proposed mapping strategy is not directly applicable to such architectures, this study provides valuable insights into optimizing algorithm performance. For instance, recent work has demonstrated VQA with hardware-efficient ansatzes (HEA) on a quantum photonic platform [78]. Our routing solutions with neighboring interactions can be adapted to build HEA for VQAs, including integrating symmetric structures to construct high VQA layers, as employed in AOQMAP-L.

The AOQMAP flow presented in Fig. 2 can be generalized to optimize various quantum algorithms. Specifically, the division of the process into subtopology adaptation and mapping of subtopology-adapted circuits is particularly promising for improving the performance of algorithms on real quantum devices, as it facilitates the efficient utilization of classical and quantum resources while enabling high-quality implementations. One of the most significant advantages of AOQMAP is its ability to find optimal and scalable solutions for VQAs with fully connected two-qubit interactions on linear, T-, and H-shaped subtopologies. These solutions can be directly adapted to variational-based algorithms, including machine-learning models leveraging variational circuits [79–82]. While these solutions cannot be straightforwardly transformed to algorithms beyond VQAs, such as Grover’s algorithm [83], our approach to finding solutions by bridging exact and scalable methods offers valuable guidance: (1) carefully analyze the specific structure of target algorithm from Hamiltonian to circuit level to identify properties such as symmetry and the most advantageous subtopologies. In the case where the optimal subtopology is not immediately apparent, start with linear, T-, and H-shaped topologies, as they frequently appear in most NISQ devices; (2) focus on two-qubit interactions in the algorithm and insert SWAP gates to satisfy connectivity constraints. This simplifies the

TABLE II. Characterization of IBM QPUs.

Property	ibm_perth	ibm_nairobi	ibmq_ehningen	ibmq_kaolkata	ibm_cusco	ibm_nazca	ibm_brisbane
Qubit number	7	7	27	27	127	127	127
Single qubit gate error (%)	0.049	0.030	0.033	3.732	0.270	0.086	0.024
Two qubit gate error (%)	1.14	0.87	1.03	18.55	8.86	5.49	0.75
Readout error (%)	2.44	3.01	1.23	2.60	5.51	5.18	14.08
Single qubit gate length (ns)	35.56	35.56	32.00	35.56	44.00	60.00	60.00
Two qubit gate length (ns)	485.93	306.96	346.98	450.29	487.22	658.17	660.0
Readout length (ns)	721.78	5560.89	846.22	640.0	4000.0	4000.0	4000.0
T_1 (μ s)	162.06	101.06	143.82	103.18	131.51	197.66	229.71
T_2 (μ s)	123.82	71.04	166.11	82.98	109.17	128.56	146.32

routing problem, as any single qubit gate can be assigned at the algorithmic level, which we have demonstrated with QAOA by first finding routing solutions on subtopologies and then assigning single qubit gates; (3) prioritize placing SWAP gates behind or before two-qubit gates. This allows for optimization of the two-qubit gate following a SWAP gate by leveraging CX gate cancellation (see Fig. 7 for CZ-SWAP and Ref. [42] for ZY-SWAP), thereby reducing the impact of noise due to the insertion of SWAP gates; (4) remove any possible initial and final SWAP gates by adjusting the initial and measurement orders. Integrating these guidelines into the design process of heuristic and exact methods is promising to improve the quality and efficiency of solving qubit-mapping problems.

V. BENCHMARKING EXPERIMENTS

This section presents benchmarking results of QAOA for portfolio optimization on several IBM QPUs. We evaluate the efficiency of our method against three other approaches: Qiskit [25], Tket [26], and SWAP network (SWAPNK) [20].

Table II summarizes the characteristics of IBM QPUs employed in our benchmarking experiments. It is worth noting that these characteristics fluctuate over time. The processors used in our study range from 7 to 127 qubits. The native gate set on 7- and 27-qubit QPUs is {CX, ID, R_Z , SX, X }, comprising CX, identity gate, single-qubit Z rotation, $\pi/2$ X rotation, and Pauli X . The 127-qubit QPUs implement a basis set of {ECR, ID, R_Z , SX, X }, where ECR is echoed cross-resonance two-qubit gate. The average error rate for single-qubit gates varies from 10^{-4} to 10^{-2} , while for two-qubit gates, it is typically an order of magnitude higher, around 10^{-2} . Additionally, average readout error rates are approximately 10^{-2} . The average gate lengths for single-qubit gates range from 32 to 60 ns, whereas for two-qubit gates, they range from 306.96 to 660.00 ns. The readout lengths vary between 640.0 and 5560.89 ns. Furthermore, the mean energy relaxation time T_1 ranges from 101.06 μ s to 229.71 μ s across different processors, and the average dephasing time T_2 ranges from 71.04 μ s to 166.11 μ s. These comprehensive devices

allow us to validate our optimization techniques across various qubit numbers, connectivity options, and noise properties.

The portfolio optimization instances utilized in this study are obtained from the Supplemental Material of Ref. [43]. Six different problem sizes are examined, involving the selection of the first 3, 4, 5, 6, 7, and 10 assets from the portfolio optimization dataset. In QAOA, the key parameters (q, B, A, λ), as defined in Sec. III B, are set based on the problem size as follows: (0.33, 2, 0, 20.97) for three-qubit, (0.33, 2, 0.13, 17.99) for four-qubit, (0.33, 3, 0.07, 17.51) for five-qubit, (0.33, 3, 0.12, 17.73) for six-qubit, (0.33, 3, 0.13, 14.70) for seven-qubit, and (0.33, 5, 0.05, 6) for ten-qubit scenarios. To determine the parameters (β, γ), we employ the constrained optimization by linear approximation (COBYLA) [84] optimizer along with Qiskit's QASM simulator.

A. Evaluation metrics

We utilize circuit metrics including two-qubit gate count and circuit depth to evaluate various qubit-mapping methods. Additionally, we employ approximation ratio (AR) and success probability (SP) to assess the performance of QAOA.

Portfolio optimization refers to the process of maximizing returns while minimizing risk by selecting a subset of n assets (n qubits). Generally, a defined number of assets needs to be selected, which is called the budget constraint. If the solution violates the budget constraint, the value of AR is defined as 0, otherwise it is defined as

$$r = \frac{F - F_{\max}}{F_{\text{opt}} - F_{\max}}, \quad (10)$$

where F , F_{opt} , and F_{\max} are the average value found by QAOA, the optimal and minimum value, and the worst-case and maximum value, respectively. The SP is defined as the probability of obtaining an optimal solution. A higher value of AR or SP implies a more effective algorithm performance, while a lower value of two-qubit

gate count or circuit depth indicates a high quality of the qubit-mapping solution.

B. Comparison of qubit-mapping strategies

To ensure fair benchmarking, we employ various qubit-mapping techniques to map circuits onto target quantum hardware, followed by decomposing and optimizing these circuits using Qiskit [25] with default settings (optimization level 1). For Tket [26], we employ NoiseAwarePlacement to select the lowest-noise qubits based on target QPUs' noise properties. The default Tket routing method *RoutingPass* is employed to insert SWAP gates. For Qiskit [25], we transpile with default settings. Finally, to map SWAPNK [20], we apply the same mapping strategy as AOQMAP, as detailed in Sec. III C.

1. Circuit properties

We first compare our approach with SWAPNK [20]. Figure 8 presents the reduction in SWAP gates for QAOA at depth $p = 1$ using AOQMAP on linear (AOQMAP-L), T- (AOQMAP-T), and H-shaped (AOQMAP-H) subtopologies compared to SWAPNK. Specifically, AOQMAP-L leads to a decrease of 67% in SWAP gates for three-qubit and 20% for ten-qubit. Moreover, AOQMAP-T reduces SWAP gates by 67% for four-qubit and 29% for ten-qubit, whereas AOQMAP-H reduces SWAP gates by 53% for six-qubit and 36% for ten-qubit. For QAOA with depth p , the reduction in SWAP gate count achieved is p multiplied by the reduction attained at depth $p = 1$.

We then compare our approach with Tket, Qiskit, and SWAPNK on IBM QPUs. The examined QAOA has depths ranging from 1 to 7 and qubit numbers 3, 5, 6, and 10. Tables III and IV present the reduction in two-qubit gate counts (CX or ECR) and circuit depth, respectively, of mapped circuits generated by AOQMAP compared to other approaches across various QPUs. We note that the

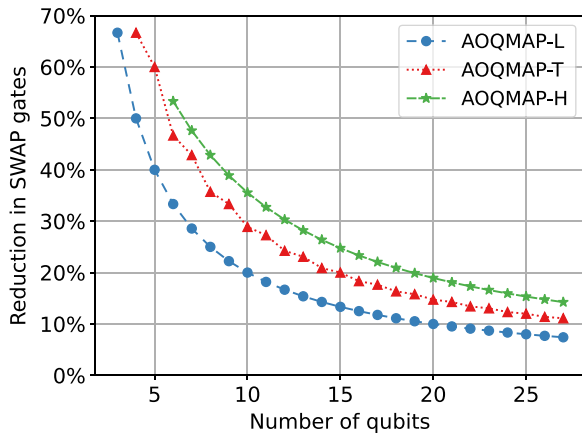


FIG. 8. Reduction in the number of SWAP gates for QAOA with depth $p = 1$ using AOQMAP-L, AOQMAP-T, and AOQMAP-H compared to SWAPNK.

127-qubit QPUs natively implement ECR gates instead of CX gates. The results demonstrate that AOQMAP leads to reductions in CX or ECR gates and/or circuit depth compared to other approaches. For three-qubit QAOA, the number of CX gates remains the same reduction for AOQMAP compared to Tket and SWAPNK since three qubits can only form a linear subtopology. The difference in reduction in Qiskit is due to the randomness of Qiskit's transpiler. However, for five-qubit QAOA with depths ranging from 2 to 7, AOQMAP exhibits a higher CX gate reduction but lower circuit depth reduction on *ibmq_perth* (5Q-Perth) than *ibmq_kolkata* (5Q-Kolkata). The reason is that a circuit adapted to T-shaped subtopology yields a lower expectation value of problem Hamiltonian and is selected. This selection of T-shaped topology further reduces the number of CX gates, but also increases circuit depth. For six-qubit QAOA on *ibmq_ehningen*, AOQMAP uses only linear subtopology, which has the least number of CX gates and the shortest depths compared to others. A T-shaped subtopology is also selected in AOQMAP for ten-qubit QAOA on *ibmq_kolkata* at depths 1 and 7. The solutions provided by AOQMAP achieve the shortest circuit depth and/or the fewest number of CX or ECR gates compared to other approaches. Specifically, AOQMAP reduces two-qubit gate count by 29% (up to 56%) and circuit depth by 31% (up to 82%) on average compared to Qiskit, Tket, and SWAPNK.

It is worthwhile to note that the solution quality in Qiskit is sensitive to the optimization level settings. As shown in Ref. [42], where AOQMAP is applied to improve digitized counterdiabatic QAOA, for QAOA circuits with depth $p = 1$, Qiskit's optimization levels 1 and 2 produce comparable results, while level 3 shows improved performance. However, AOQMAP consistently outperforms all Qiskit optimization levels tested, highlighting its superior efficiency and effectiveness.

2. Computational resources

This section examines the computational resources needed for AOQMAP and compares them with the requirements of Qiskit and Tket. We show that the performance enhancements achieved by AOQMAP do not come at the expense of increased computational complexity, making it a practical approach. Specifically, we first divide the runtime of AOQMAP into three parts: adaptation, verification, and qubit selection. As illustrated in Fig. 2, VQAs are routed, decomposed, and optimized on subtopologies in adaptation process, ensuring that the circuit can be executed on QPU's subtopologies. Then, the circuit correctness is verified by computing the Hellinger distance of adapted and original circuits using QASM simulator. Finally, mapomatic [41] is employed to select optimal qubits for execution.

TABLE III. Reduction in CX or ECR gate counts of AOQMAP compared to other qubit mapping methods using QAOA with n qubits (nQ) and depths from 1 to 7 on various IBM QPUs. Higher values indicate better performance of AOQMAP.

Benchmark	Tket	Qiskit	SWAPNK
3Q-Perth	(22%, 22%, 30%, 28%, 31%, 30%, 32%)	(22%, 22%, 30%, 28%, 31%, 30%, 32%)	(22%, 22%, 22%, 22%, 22%, 22%, 22%)
3Q-Kolkata	(22%, 22%, 30%, 28%, 31%, 30%, 32%)	(22%, 22%, 30%, 28%, 31%, 30%, 32%)	(22%, 22%, 22%, 22%, 22%, 22%, 22%)
3Q-Ehningen	(22%, 22%, 30%, 28%, 31%, 30%, 32%)	(30%, 48%, 40%, 45%, 35%, 38%, 42%)	(22%, 22%, 22%, 22%, 22%, 22%, 22%)
3Q-Nairobi	(22%, 22%, 30%, 28%, 31%, 30%, 32%)	(22%, 22%, 30%, 28%, 31%, 30%, 32%)	(22%, 22%, 22%, 22%, 22%, 22%, 22%)
5Q-Perth	(10%, 28%, 28%, 30%, 30%, 31%, 31%)	(19%, 34%, 35%, 37%, 35%, 39%, 36%)	(13%, 20%, 20%, 20%, 20%, 20%, 20%)
5Q-Kolkata	(10%, 22%, 22%, 25%, 26%, 25%, 26%)	(21%, 29%, 30%, 32%, 32%, 34%, 34%)	(13%, 13%, 13%, 13%, 13%, 13%, 13%)
5Q-Cusco	(28%, 25%, 27%, 26%, 27%, 26%, 27%)	(32%, 34%, 38%, 38%, 38%, 39%, 39%)	(13%, 13%, 13%, 13%, 13%, 13%, 13%)
5Q-Ehningen	(7%, 19%, 22%, 24%, 24%, 25%, 25%)	(48%, 50%, 54%, 52%, 53%, 56%, 53%)	(11%, 11%, 11%, 11%, 11%, 11%, 11%)
6Q-Ehningen	(41%, 40%, 41%, 42%, 42%, 42%, 42%)	(39%, 43%, 45%, 43%, 44%, 44%, 45%)	(10%, 7%, 7%, 7%, 7%, 7%, 10%)
10Q-Kolkata	(52%, 50%, 48%, 52%, 53%, 48%, 54%)	(44%, 43%, 45%, 47%, 48%, 47%, 44%)	(7%, 7%, 7%, 7%, 7%, 7%, 7%)
10Q-Nazca	(48%, 53%, 55%, 55%, 56%, 56%, 56%)	(46%, 46%, 48%, 49%, 48%, 49%, 49%)	(7%, 7%, 7%, 7%, 7%, 7%, 7%)
10Q-Cusco	(51%, 53%, 55%, 55%, 56%, 56%, 56%)	(46%, 46%, 48%, 49%, 48%, 49%, 49%)	(7%, 7%, 7%, 7%, 7%, 7%, 7%)

TABLE IV. Reduction in circuit depth. Same benchmarks as in Table III.

Benchmark	Tket	Qiskit	SWAPNK
3Q-Perth	(30%, 21%, 29%, 25%, 29%, 26%, 29%)	(27%, 21%, 22%, 25%, 25%, 26%, 26%)	(10%, 11%, 11%, 11%, 11%, 11%, 11%)
3Q-Kolkata	(30%, 21%, 29%, 25%, 29%, 26%, 29%)	(27%, 21%, 22%, 25%, 25%, 26%, 26%)	(10%, 11%, 11%, 11%, 11%, 11%, 11%)
3Q-Ehningen	(27%, 21%, 28%, 25%, 28%, 26%, 28%)	(14%, 28%, 22%, 26%, 19%, 22%, 25%)	(10%, 11%, 11%, 11%, 11%, 11%, 11%)
3Q-Nairobi	(30%, 21%, 29%, 25%, 29%, 26%, 29%)	(27%, 21%, 22%, 25%, 25%, 26%, 26%)	(10%, 11%, 11%, 11%, 11%, 11%, 11%)
5Q-Perth	(29%, 13%, 18%, 19%, 20%, 20%, 21%)	(43%, 23%, 27%, 34%, 28%, 36%, 28%)	(7%, -26%, -27%, -27%, -27%, -27%, -27%)
5Q-Kolkata	(29%, 36%, 40%, 46%, 47%, 46%, 47%)	(47%, 43%, 51%, 50%, 48%, 54%, 53%)	(7%, 7%, 8%, 8%, 8%, 8%, 8%)
5Q-Cusco	(53%, 50%, 50%, 49%, 50%, 49%, 49%)	(48%, 47%, 48%, 52%, 50%, 52%, 52%)	(13%, 13%, 12%, 12%, 11%, 12%, 11%)
5Q-Ehningen	(21%, 26%, 28%, 29%, 30%, 30%, 31%)	(51%, 55%, 57%, 57%, 57%, 59%, 57%)	(7%, 7%, 8%, 8%, 8%, 8%, 8%)
6Q-Ehningen	(48%, 43%, 44%, 48%, 45%, 46%, 48%)	(44%, 45%, 50%, 50%, 47%, 49%, 49%)	(6%, 6%, 7%, 7%, 7%, 7%, 7%)
10Q-Kolkata	(57%, 69%, 65%, 71%, 71%, 66%, 59%)	(54%, 66%, 67%, 69%, 72%, 71%, 27%)	(-33%, 4%, 4%, 4%, 4%, 4%, -35%)
10Q-Nazca	(74%, 75%, 79%, 80%, 81%, 82%, 82%)	(71%, 71%, 52%, 72%, 71%, 76%, 74%)	(6%, 6%, 6%, 5%, 5%, 5%, 5%)
10Q-Cusco	(75%, 75%, 79%, 80%, 81%, 81%, 81%)	(71%, 71%, 52%, 71%, 71%, 76%, 74%)	(7%, 7%, 6%, 6%, 6%, 6%, 6%)

TABLE V. Runtime [s] of three components in AOQMAP on T-shaped subtopology using QAOA with the number of qubits n ranging from 4 to 10 and depth p of 1, 3, 5, and 7. The runtime is divided into adaptation, verification, and qubit selection. Avg denotes the average runtime for different qubit numbers.

p	n	4	5	6	7	8	9	10	Avg
1	Adaptation	0.04	0.05	0.08	0.06	0.12	0.09	0.08	0.07
	Verification	0.13	0.12	0.16	0.16	0.19	0.25	0.21	0.17
	Qubit selection	0.03	0.05	0.11	0.07	0.09	0.09	0.14	0.08
3	Adaptation	0.02	0.02	0.03	0.03	0.05	0.06	0.08	0.04
	Verification	0.12	0.14	0.14	0.24	0.19	0.2	0.23	0.18
	Qubit selection	0.11	0.06	0.09	0.09	0.16	0.14	0.23	0.13
5	Adaptation	0.03	0.03	0.05	0.05	0.06	0.08	0.1	0.06
	Verification	0.12	0.2	0.16	0.31	0.2	0.23	0.28	0.21
	Qubit selection	0.05	0.06	0.11	0.13	0.2	0.2	0.33	0.15
7	Adaptation	0.03	0.03	0.06	0.06	0.08	0.11	0.2	0.08
	Verification	0.13	0.14	0.16	0.19	0.22	0.25	0.28	0.2
	Qubit selection	0.08	0.08	0.12	0.16	0.27	0.25	0.44	0.2

AOQMAP, Qiskit, and Tket employ the same experimental setup described at the beginning of Sec. VB. We benchmark the QAOA for MaxCut on complete graphs with varying qubit numbers and depths, mapping these circuits onto a 27-qubit QPU. We take AOQMAP-T as an example, and its analysis can be extended to AOQMAP-L and AOQMAP-H. Tables V and VI provide a comprehensive overview of runtime for three parts of AOQMAP and an overall comparison, respectively. Our investigation reveals that adaptation in AOQMAP consumes the least time, with its duration increasing minimally as the number of qubits grows. This process is also minimally affected by variations in QAOA depths. In contrast, verification is the most time intensive, requiring the execution of circuits with QASM simulator. Moreover, verification runtime increases with the number of qubits but is relatively unaffected by QAOA depths. The runtime for qubit selection increases with both the number of qubits and QAOA depths. We also observe that the runtime for deeper circuits might be shorter than that for shallower circuits.

This is potentially due to the stochastic nature of Qiskit transpiler employed for decomposition and optimization. As indicated in Table VI, Qiskit demonstrates the shortest runtime. Additionally, AOQMAP outperforms the heuristic strategy Tket. Tket's runtime increases with QAOA depths and qubit numbers, and exhibits significant fluctuations, potentially due to the interface transforming circuits implemented with Qiskit and Tket.

3. Simulation under noise

To examine the impact of noise on the performance of QAOA with different qubit-mapping strategies, we simulate algorithms mapped onto a 27-qubit QPU with the topology shown in Fig. 1. We use a depolarizing noise model [85,86], which is a common simple model used to approximate the effects of mixed noise processes in quantum systems. This model captures noise effects by applying random single-qubit bit-flip, phase-flip, and combined bit- and phase-flip errors to each gate, providing

TABLE VI. Runtime comparison of different qubit mapping methods using QAOA, with the number of qubits n ranging from 4 to 10 and depth p of 1, 3, 5, and 7. Avg denotes the average runtime for different qubit numbers.

p	n	4	5	6	7	8	9	10	Avg
1	AOQMAP-T	0.2	0.22	0.34	0.29	0.4	0.44	0.44	0.33
	Qiskit	0.02	0.03	0.03	0.03	0.03	0.04	0.05	0.03
	Tket	0.05	0.36	0.19	0.13	0.64	0.77	2.99	0.73
3	AOQMAP-T	0.25	0.22	0.27	0.36	0.39	0.41	0.55	0.35
	Qiskit	0.02	0.04	0.03	0.13	0.06	0.06	0.08	0.06
	Tket	0.07	0.42	0.27	0.23	1.42	0.92	3.16	0.93
5	AOQMAP-T	0.2	0.23	0.31	0.48	0.47	0.52	0.7	0.42
	Qiskit	0.05	0.05	0.05	0.09	0.08	0.12	0.14	0.08
	Tket	0.13	0.48	0.35	0.34	0.97	1.09	3.19	0.94
7	AOQMAP-T	0.24	0.25	0.34	0.41	0.56	0.62	0.92	0.48
	Qiskit	0.04	0.06	0.06	0.09	0.19	0.13	0.17	0.11
	Tket	0.16	0.53	0.49	0.44	1.13	1.34	3.67	1.11

a good approximation of the overall noise behavior and impact on algorithm performance. The depolarizing error channel, acting on qubits described by density matrix ρ , is defined as

$$E_D(\rho) = \frac{\epsilon}{4} (X\rho X + Y\rho Y + Z\rho Z) + \left(1 - \frac{3\epsilon}{4}\right) \rho, \quad (11)$$

where X , Y , and Z are Pauli matrices, and ϵ is noise strength.

We compare AOQMAP-L and AOQMAP-T with Tket, Qiskit, and SWAPNK. The mapped circuits are simulated under depolarizing noise with strength 0.005 for two-qubit gates and $\epsilon/10$ for single-qubit gates. Figure 9 presents the approximation ratio and success probability of QAOA on three, four, five, and six qubits at depths from 1 to 7. Compared to other quantum mapping strategies, AOQMAP-L and AOQMAP-T demonstrate better performance at higher depths and comparable performance at depth $p = 1$. This suggests that increased depth of QAOA leads to a more significant noise effect, emphasizing advantages of AOQMAP. AOQMAP-T requires fewer CX gates during the mapping stage compared to others, which mitigates the detrimental effects of noise accumulation, leading to improved performance. For three-qubit QAOA, AR and SP decrease with increased depth, indicating that noise effects outweigh performance improvement at higher depths. In comparison, for QAOA with four, five, and six qubits, AR and SP increase and then stabilize or slightly decrease, suggesting a balance between performance improvements and noise effects.

4. Advantages of postselection on subtopologies

In this section, we investigate the performance benefits of restricting circuit execution to identified subtopologies and applying postselection, compared to directly mapping circuits onto the full device topology. Specifically, we first compare subtopology-based circuits with those mapped to the entire topology of a 127-qubit QPU `ibm_brisbane` in terms of circuit depth and the number of two-qubit gates. We then show that postselection among results obtained on subtopologies, executed using a noise simulator configured to mimic the `ibm_brisbane`, yields higher performance than full-topology mapping. Moreover, we evaluate the performance of Qiskit and Tket in comparison to our AOQMAP approach and show that AOQMAP consistently outperforms both, underscoring its potential for large-scale quantum computing applications.

Here we employ the highest optimization level available in Qiskit, level 3, which is consistently applied for Qiskit and the decomposition process employed in AOQMAP and Tket to ensure a fair comparison. To accurately emulate the noise characteristics of a real quantum device, we configure the Qiskit AerSimulator using `AerSimulator.from_backend(device_backend)`. This method integrates the simulator with the noise model and characteristics of an actual backend, specifically `ibm_brisbane`. By replicating the practical noise environment of a quantum processor, this simulation framework enables a robust evaluation of algorithmic performance under realistic conditions.

Figure 10 illustrates the impact of topology selection on ECR gate count and circuit depth for n -qubit QAOA

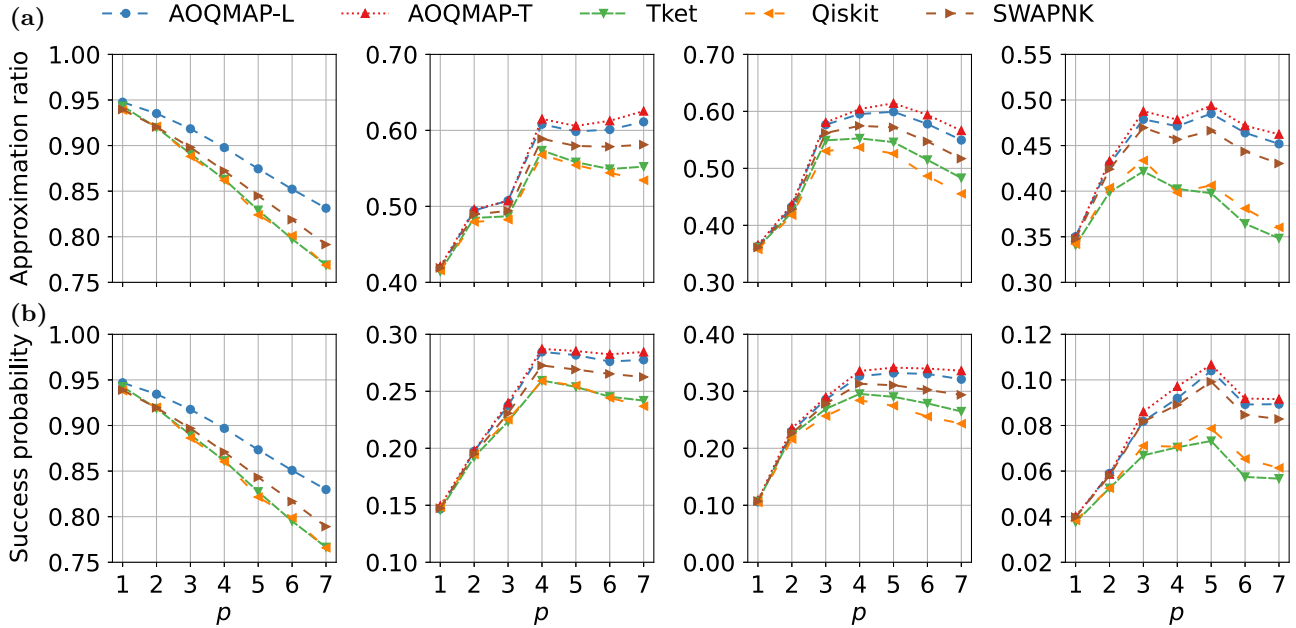


FIG. 9. Simulation under depolarizing noise of QAOA for portfolio optimization mapped with AOQMAP-L, AOQMAP-T, Tket, Qiskit, and SWAPNK. (a) Approximation ratio and (b) success probability for QAOA with three, four, five, and six qubits. Three-qubit QAOA has an absence of AOQMAP-T, as the minimum number of qubits required for the T-shaped subtopology is four.

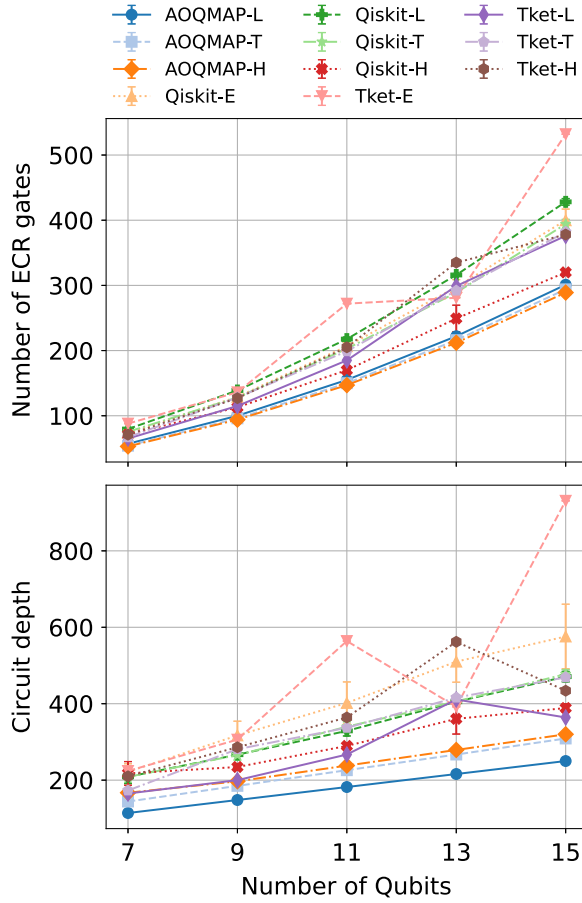


FIG. 10. Comparison of ECR gate count and circuit depth in n -qubit QAOA with $p = 1$, transpiled on the entire topology (denoted by “-E”) and subtopologies (denoted by “-L,” “-T,” and “-H”) of a 127-qubit QPU ibm_brisbane. Each data point represents the mean of 10 repetitions, with error bars indicating the standard error of the mean.

circuits with $p = 1$ on ibm_brisbane. Restricting the layout to subtopologies reduces the number of two-qubit gates and/or circuit depth compared to the full topology. Among Qiskit implementations, the H-shaped subtopology achieves the best performance, while Tket performs optimally on linear subtopologies. AOQMAP consistently outperforms Qiskit and Tket across all subtopologies, with AOQMAP-L yielding the lowest circuit depth and AOQMAP-H achieving the minimal ECR gate count. These results highlight the advantages of subtopology-based transpilation approaches. Table VII provides a quantitative summary of the average reductions in ECR count and circuit depth achieved by Qiskit and Tket on subtopologies relative to the full topology. For Qiskit, the H-shaped subtopology achieves the greatest reductions, with a 13.92% decrease in ECR gate count and a 23.56% decrease in circuit depth. In contrast, Tket performs best on the linear subtopology, achieving notable reductions of 19.30% in ECR gate count and 33.94% in

TABLE VII. Average reductions in ECR gate count and circuit depth with Qiskit and Tket on subtopologies compared to the full topology, derived from the dataset in Fig. 10.

Transpiler	Subtopology	ECR reduction	Depth reduction
Qiskit	Linear	-6.87%	15.75%
	T-shaped	0.90%	15.04%
	H-shaped	13.92%	23.56%
Tket	Linear	19.30%	33.94%
	T-shaped	16.17%	22.94%
	H-shaped	12.06%	11.65%

circuit depth. While Qiskit exhibits less efficient performance on the linear subtopology due to an increase in ECR gates, it still achieves a 15.75% reduction in circuit depth. Notably, Qiskit is not limited to linear configurations, as it demonstrates substantial decreases in both ECR gates and circuit depth on the H-shaped subtopology. This underscores Qiskit’s adaptability and improved performance when employing our proposed approach, which is designed to leverage multiple subtopology types effectively.

Figure 11 compares the performance of mapping strategies for a seven-qubit QAOA circuit with $p = 1$ and $p = 3$ on a noisy simulator modeled after the 127-qubit ibm_brisbane. The results indicate that subtopology-based postselection consistently outperforms direct mapping on the full topology. Notably, Tket demonstrates more significant performance improvements from postselection compared to Qiskit, particularly for higher-depth circuits ($p = 3$). Among the strategies evaluated, AOQMAP achieves the highest performance. These findings underscore the significant performance benefits of postselection on subtopologies, particularly in enhancing circuit efficiency and reliability across different transpilation frameworks. They also provide valuable insights into optimizing other classes of algorithms for large-scale quantum devices, demonstrating the broader applicability of this approach.

C. Demonstration on IBM quantum devices

This section evaluates the performance of QAOA for portfolio optimization across six IBM QPUs, comparing different mapping approaches on seven-qubit ibm_perth and ibm_nairobi, 27-qubit ibmq_kolkata and ibmq_ehningen, and 127-qubit ibm_nazca and ibm_cusco. The characterization of the IBM QPUs at the time of the demonstration is detailed in Appendix B. The problem sizes QAOA include three, four, five, six, and ten qubits, and depths ranging from 1 to 7.

As discussed in Sec. IIIB, two methods exist for constructing higher QAOA depth, including repeating swap layers in depth $p = 1$ circuit and leveraging their symmetry. To examine the influence of symmetry, we

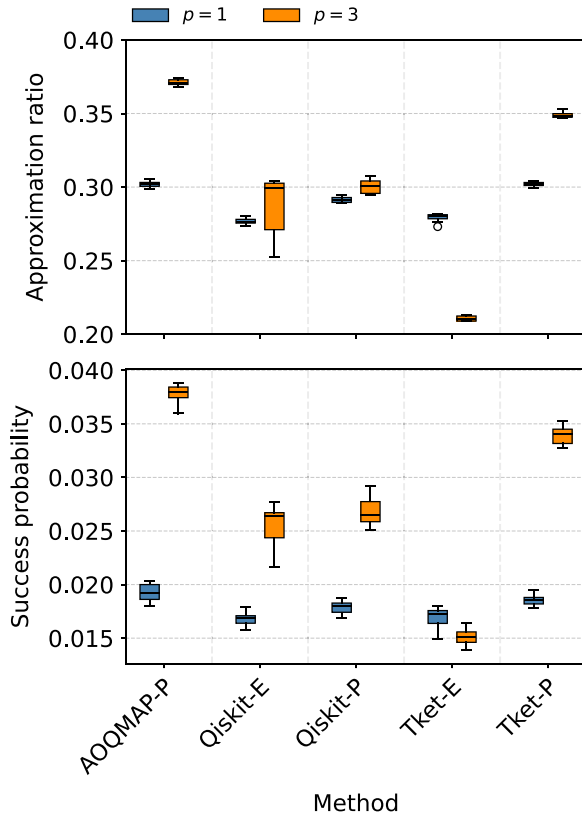


FIG. 11. Comparison of mapping strategies for a seven-qubit QAOA with $p = 1$ (blue, left) and $p = 3$ (orange, right) on a noisy simulator mimicking the 127-qubit QPU *ibm_brisbane*. Each box plot represents ten repetitions of the corresponding mapping strategy. The results contrast entire topology mapping (denoted by “-E”) with subtopology mapping followed by postselection (denoted by “-P”).

conduct a comparative analysis of AOQMAP on linear subtopology with repetitive routing at depth $p = 1$ (AOQMAP-L) and with mirror-symmetric swap layers (AOQMAP-LS), as well as AOQMAP on T-shaped subtopology (AOQMAP-T). We consider QAOA instances with four qubits and depths from 1 to 7. Figure 12 compares these variants to Tket, Qiskit, and SWAPNK on the seven-qubit *ibm_nairobi*. AOQMAP on linear subtopology utilizing symmetry significantly enhances the performance, increasing success probability by an average of 82% (up to $1.83\times$) compared to AOQMAP-L, 76% (up to $2.76\times$) compared to AOQMAP-T, $1.72\times$ (up to $3.80\times$) compared to Tket, $1.77\times$ (up to $3.21\times$) compared to Qiskit, and 73% (up to $1.41\times$) compared to SWAPNK. These results demonstrate the effectiveness of symmetry for improving performance on NISQ devices, validating the advantage of AOQMAP’s symmetry-incorporated mapping approach.

For the demonstration on *ibmq_ehningen*, we directly compare AOQMAP-L with other methods, while for

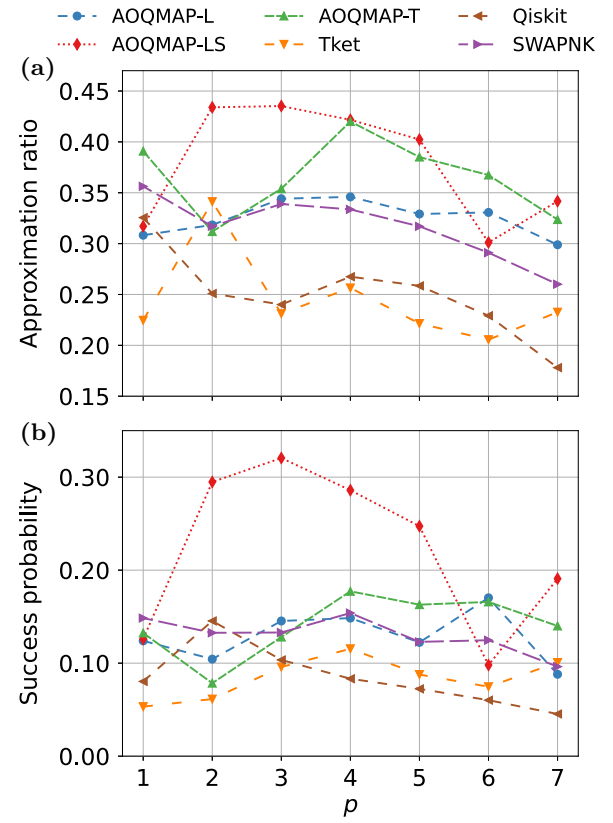


FIG. 12. Demonstration of four-qubit QAOA on a seven-qubit QPU *ibm_nairobi* with various qubit mapping methods. (a) Approximation ratio and (b) success probability. QAOA depths range from 1 to 7.

evaluations on other QPUs, a postselection process is utilized to generate the AOQMAP solution. Specifically, we construct the algorithms with AOQMAP-L, AOQMAP-LS, and AOQMAP-T. Each adapted circuit is mapped to the target device using the method described in Sec. III C. The demonstration solution corresponding to the minimum expectation value of the problem Hamiltonian is then chosen. As shown in Fig. 13(a), AOQMAP achieves the highest performance on all QPUs. Tket and Qiskit perform comparably on all QPUs, while SWAPNK performs better on *ibm_perth* and *ibmq_ehningen* than *ibmq_kolkata* and *ibm_nairobi*. The approximation ratio decreases with increased depth, indicating that the negative impact of noise dominates the improved performance at higher depths. For five-qubit QAOA [Fig. 13(b)], AOQMAP maintains the highest AR across all QPUs. Tket outperforms Qiskit on *ibmq_kolkata* and *ibmq_ehningen*, whereas Qiskit performs better on *ibm_cusco*. SWAPNK has a better performance on *ibmq_perth* and *ibmq_kolkata*, while it performs worse on *ibmq_ehningen*. For six-qubit QAOA on *ibmq_ehningen*, AOQMAP has a significantly higher AR value than Tket, Qiskit, and SWAPNK [Fig. 13(c)]. For the largest ten-qubit QAOA, AOQMAP

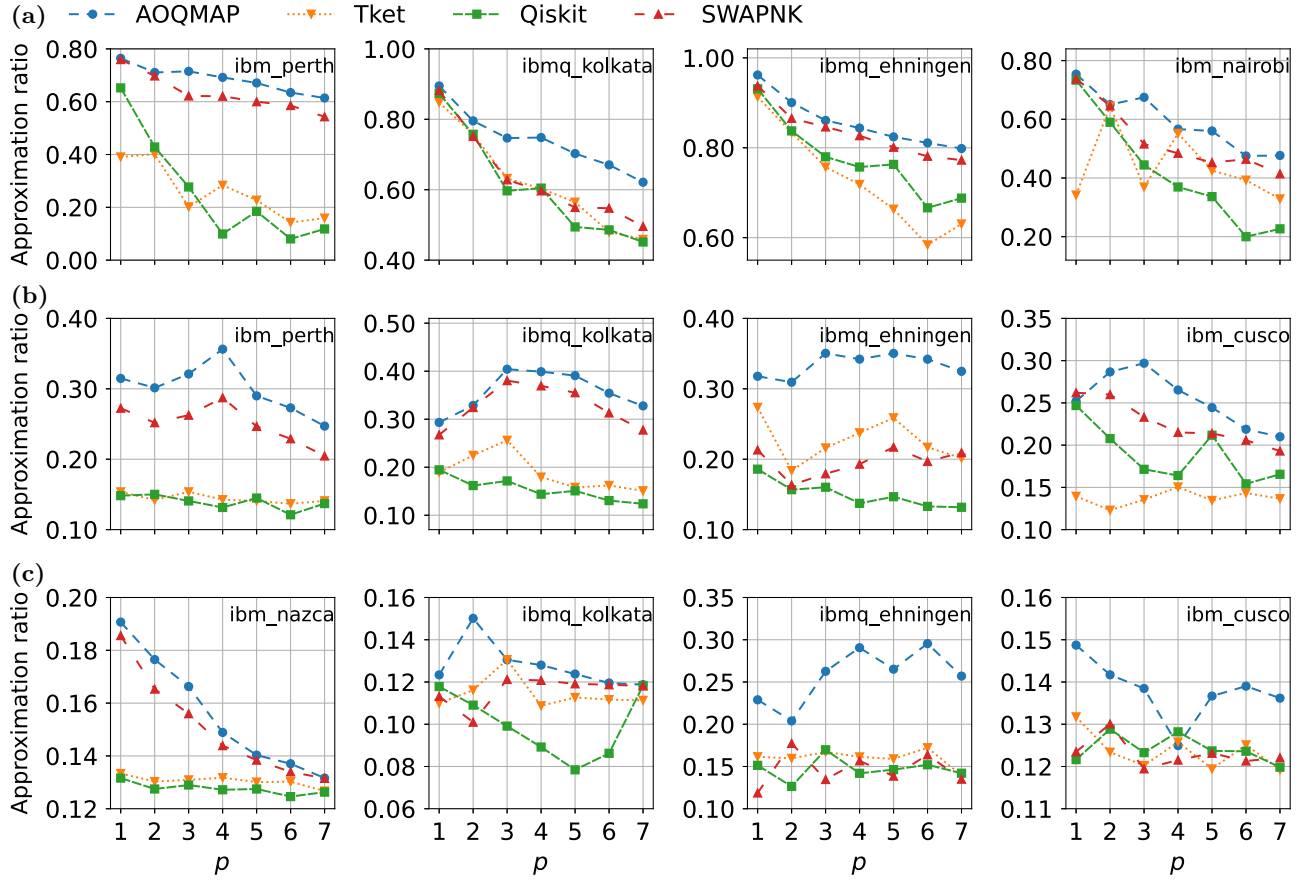


FIG. 13. Approximation ratio of QAOA obtained on various IBM QPUs using AOQMAP, Tket, Qiskit, and SWAPNK. The numbers of qubits are (a) three, (b) five, and (c) ten (six for ibmq_ehningen). AOQMAP demonstrates the highest performance on all QPUs in comparison to other approaches.

consistently achieves the highest AR across all tested QPUs. SWAPNK has a higher AR only on *ibm_nazca*, while Qiskit and Tket perform worse on all QPUs. On *ibmq_kolkata*, AOQMAP achieves the highest AR value at depth two, outperforming others. With increased depth, the noise in large-scale circuit increases significantly, resulting in a reduced AR value. For ten-qubit QAOA on *ibm_cusco*, AOQMAP has the highest AR value at depth $p = 1$. The presence of the second-highest AR at depth six demonstrates a trade-off between improved performance and introduced noise of the increased QAOA depth.

The success probability of QAOA with three, five, and ten qubits (six qubits for *ibmq_ehningen*) is displayed in Figs. 14(a)–14(c), respectively, following a similar trend as the AR. AOQMAP demonstrates the highest SP overall with different circuit sizes across all QPUs. For QAOA with five and ten qubits, the SP initially increases with depth and then decreases, peaking at different p depending on device noise. This phenomenon indicates a balance between performance improvement and noise impact in high-depth circuits. For five-qubit QAOA, Qiskit performs

better on *ibm_cusco*, Tket has higher performance on *ibmq_kolkata* and *ibmq_ehningen*, and SWAPNK has higher SP values on *ibmq_kolkata* and *ibm_cusco*. For ten-qubit QAOA, Tket and Qiskit have lower success probabilities on all QPUs, whereas SWAPNK performs better only on *ibm_nazca*.

These findings demonstrate that AOQMAP provides a significant improvement in approximation ratio and success probability, surpassing other popular qubit-mapping approaches. In particular, AOQMAP improves AR by an average of 54% and SP by an average of 138% compared to Qiskit, Tket, and SWAPNK, demonstrating robust high performance on NISQ devices. Furthermore, our experiments on various QPUs highlight the limitations of solely relying on noise model simulations for assessing circuit performance. While such simulations provide valuable insights, they often fail to accurately capture the complex and multifaceted noise characteristics inherent to real quantum hardware. Although different qubit-mapping strategies exhibit comparable behavior under simulated noise conditions, their performance diverges significantly when executed on physical devices. This disparity

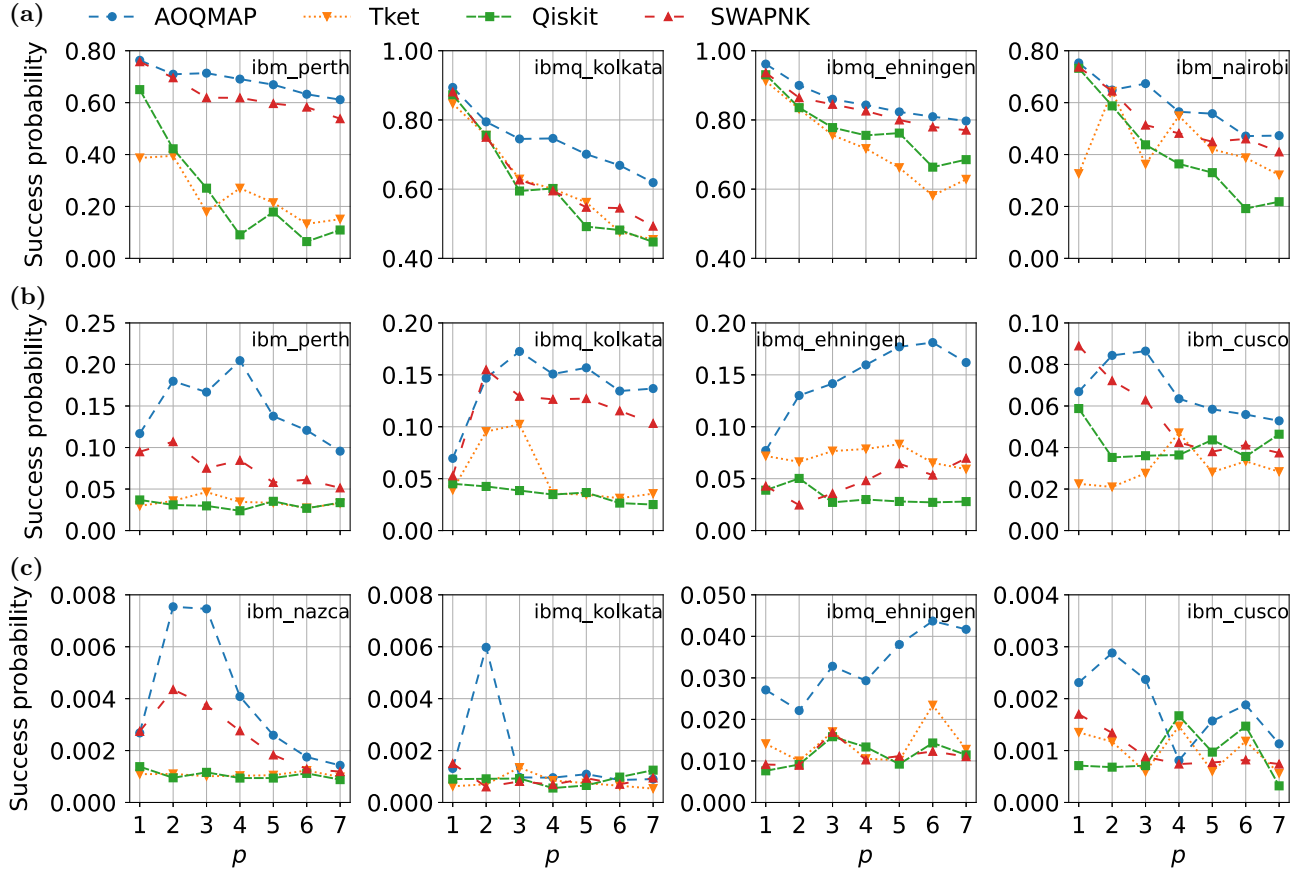


FIG. 14. Success probability of QAOA obtained on various QPUs with AOQMAP, Tket, Qiskit, and SWAPNK. The labels are the same as Fig. 13.

emphasizes the critical role of efficient qubit mapping in achieving high algorithm performance on NISQ devices.

VI. CONCLUSION

We present an efficient qubit-mapping methodology designed for VQAs on near-term quantum devices. The method involves two essential steps: adapting circuits to subtopologies and mapping adapted circuits onto target QPU, followed by postselecting execution results across different subtopologies. In the adaptation step, optimal routing solutions with the shortest circuit depth are provided for diverse subtopologies, including linear, T-, and H-shaped configurations. These solutions inherently scale to arbitrary depths and numbers of qubits for algorithms characterized by fully connected two qubit interactions. For partially connected interactions, optimizing initial qubit order enables the derivation of optimal depth-one solutions. Leveraging mirror symmetry, these solutions can be extended to higher QAOA depths without additional computational overhead. In the mapping step, the

adapted circuit is mapped onto target device, minimizing circuit error with a cost function. A postselection process is then employed to choose the optimal solution that minimizes the expectation value of problem Hamiltonian. Demonstrations of QAOA for portfolio optimization on six IBM QPUs with 7, 27, and 127 qubits exhibit improved performance of AOQMAP compared to existing methods. We also show that the symmetry used in the circuit structure can improve algorithm performance on QPUs. These results and solutions can be easily applied to other NISQ devices. Our qubit mapping strategy tailors mappings to algorithmic structures, promising to maximize the capabilities of near term devices for VQAs.

Future research directions include extending our method to handle more complex topologies, such as T- and H-shaped variants, and heavy-hex. Furthermore, exploring strategies to adapt Hamiltonian with partially connected two-qubit gate interactions presents a promising avenue of exploration. While mirror symmetry of swap layers enables scaling solutions to high depths, the combinatorial complexity associated with the depth-one circuits remains

a significant challenge. Developing heuristic optimization approaches to refine initial qubit mapping can accelerate the search process. The trade-off between computational cost and solution quality is essential for achieving efficient and effective optimization. Additionally, a comprehensive cost function and an accurate noise model, including crosstalk effects, are essential to determine the optimal subtopology for a given circuit. While linear subtopologies minimize circuit depth, T- and H-shaped subtopologies with increasing connectivity reduce SWAP gates at the expense of increased depth. The cost function should carefully balance circuit fidelity and schedule duration. The well-crafted error model, such as by integrating experimentally characterized information on real hardware [87,88], can assist in identifying the most suitable qubits. Designing cost functions and accurately modeling noise in QPUs, combined with a postselection process to choose between different subtopologies, represents a promising avenue for future research to significantly enhance algorithm performance on near-term quantum devices. Finally, extending AOQMAP to optimize circuits with many-body interactions, such as VQE with the unitary coupled cluster ansatz [57], by addressing increased connectivity requirements and operator noncommutativity represents an exciting avenue for broadening its applicability to more complex quantum algorithms.

The implementation of AOQMAP is publicly available on Github [89].

ACKNOWLEDGMENTS

The authors would like to thank Kathrin F. Koenig, Thomas Wellens, Joris Kattemölle, Sebastian Brandhofer, Andreas Ketterer, Koushik Paul, Pranav Chandarana, Julián Ferreira Vélez, and Pengcheng Zhu for their useful discussions and exchanges. We acknowledge the use of IBM Quantum services for this work and to advanced services provided by the IBM Quantum Researchers Program. The views expressed are those of the authors, and do not reflect the official policy or position of IBM or the IBM Quantum team. The authors acknowledge the financial support of the project QORA by Ministerium für Wirtschaft, Arbeit und Tourismus Baden-Württemberg in the frame of the Competence Center Quantum Computing Baden-Württemberg, the Basque Government through Grant No. IT1470-22, the Project Grant No. PID2021-126273NB-I00 funded by MCIN/AEI/10.13039/501100011033 and by “ERDF A way of making Europe” and “ERDF Invest in your Future,” Nanoscale NMR and complex systems (Grant No. PID2021-126694NB-C21), EU FET Open Grant EPIQUS (No. 899368), the ELKARTEK Program by the Basque Government under Grant KK-2022/00041, BRTA QUANTUM Hacia una especialización armonizada

en tecnologías cuánticas en BRTA. X.C. acknowledges ayudas para contratos Ramón y Cajal–2015-2020 (Grant No. RYC-2017-22482) and open access funding provided by UPV/EHU. Y.B. acknowledges the Ramón y Cajal (RYC2023-042699-I) research fellowship.

APPENDIX A: CIRCUIT VERIFICATION IN AOQMAP

Here we detail the circuit verification process in the AOQMAP. To maintain effectiveness, it is necessary to enforce the same or reduced number of CX gates during decomposition and optimization. Moreover, we verify the accuracy of adapted circuit by comparing output probability distributions of adapted and original circuits with Hellinger distance [90], which ranges from 0 to 1, with 0 indicating identical distributions. The Hellinger distance of two probability distributions is given by

$$H(P, Q) = \left(1 - \sum_j \sqrt{p_j q_j} \right)^{1/2}, \quad (\text{A1})$$

where p_j and q_j are probabilities of outcome j in distributions P and Q , respectively. The verification process begins by simulating each circuit using a simulator on a classical computer to obtain the expected output state in the absence of noise. In this study, we employ the QASM simulator provided by Qiskit to execute original and mapped circuits. We then calculate the Hellinger distance between probability distributions of simulated output states.

Table VIII reports the Hellinger distance between original and mapped circuits generated using AOQMAP on linear subtopology (AOQMAP-L). The benchmark circuits are QAOA for portfolio optimization with three, four, five, six, and ten qubits and depths ranging from 1 to 7. Each data point is averaged over 50 circuit repetitions, with a standard error of the means that is more than 15 orders of magnitude smaller than the mean. The Hellinger distance increases with qubit number but remains 2 orders of magnitude smaller than one. Values greater than 0 are due to measurement noise or shot noise generated when executing circuits using the QASM simulator. This observation confirms that the mapped circuits are consistent with the originals across the problem instances studied.

While we directly execute adapted and original circuits in the ideal case, without considering noise, and employ Hellinger distance to verify correctness, incorporating noise models is crucial for a more accurate assessment of AOQMAP’s performance in real-world scenarios. However, verifying quantum circuits under noise is challenging due to the exponential scaling of traditional strategies such as quantum state tomography [91–94], quantum process tomography [95,96], and randomized benchmarking [97–99] with the size of qubits to be characterized.

TABLE VIII. Hellinger distance between original and mapped circuits with AOQMAP-L using QAOA at a depth ranging from 1 to 7.

p	1	2	3	4	5	6	7
3-qubit	0.0045	0.0036	0.0033	0.005	0.0036	0.0048	0.0051
4-qubit	0.0044	0.0064	0.0057	0.0051	0.005	0.006	0.0044
5-qubit	0.0087	0.0067	0.0086	0.0103	0.0072	0.0095	0.0071
6-qubit	0.0138	0.0117	0.0138	0.0102	0.0123	0.0131	0.0124
10-qubit	0.0120	0.0127	0.0137	0.0127	0.0106	0.0158	0.0123

 TABLE IX. Qubits used in the demonstration of 4-qubit QAOA with depth p on ibm_nairobi in Fig. 12. Empty cells indicate that the value is the same as that in the previous row.

p	AOQMAP-L	AOQMAP-LS	AOQMAP-T	Tket	Qiskit	SWAPNK
1	[1, 3, 4, 5]	[1, 3, 4, 5]	[0, 1, 2, 3]	[0, 1, 2, 3]	[1, 2, 3, 5]	[1, 3, 4, 5]
2, ..., 7	[1, 2, 3, 5]	[1, 2, 3, 5]			[0, 1, 2, 3]	[1, 2, 3, 5]

Moreover, for large numbers of qubits, directly executing reference and original circuits with a QASM simulator and comparing their probability distributions becomes inaccurate and impractical. Current research on verifying quantum circuits typically involves decomposing the circuit into subcircuits and quantifying the difference between noisy and ideal subcircuit outputs using metrics such as total variation distance [87,100] and fidelity [101]. The resulting circuits produced by AOQMAP for VQAs on linear, T-, and H-shaped topologies maintain their structure with increasing qubit numbers and VQA depths, enabling efficient identification of subcircuits. These specifically structured circuits can also be employed to test the performance of NISQ devices as they minimize the impact of compilation quality on their performance and contain different types of topologies to capture more noise information such as crosstalk, providing valuable insights into the scalability of VQAs on real devices. Future research can verify subtopology-adapted VQAs on noisy quantum devices and benchmark the scalability of VQAs on real devices.

APPENDIX B: CLOUD PLATFORM DETAILS

Here we present calibration data for IBM QPUs used in the demonstration described in Sec. V C. Tables IX and

X summarize the qubits used for the QAOA implementation on ibm_nairobi (Fig. 12) along with their respective properties. In particular, T_1 and T_2 correspond to the relaxation and dephasing times, respectively. “Freq.” and “Anh.” denote the qubit frequency and anharmonicity, respectively. P_{01} and P_{10} represent the probability of measuring $|0\rangle$ after preparing $|1\rangle$ and measuring $|1\rangle$ after preparing $|0\rangle$, respectively. “RO err.” indicates the readout error, while “Gate err.” refers to the single-qubit gate error rate. “CX gate” and “ECR gate” specify interactions between qubits via the CX and ECR gates, respectively, with “CX err.” and “ECR err.” representing their associated error rates. Similarly, Table XI lists the qubits used for the QAOA implementation on IBM QPUs in Figs. 13 and 14. The calibration data of ibm_perth, ibm_nazca, ibmq_kolkata, ibmq_ehningen, ibm_nairobi, and ibm_cusco are detailed in Tables XII, XIII, XIV, XV, XVI, and XVII. We note that for the demonstration of QAOA with the same number of qubits on ibmq_ehningen, different transpilation methods are applied at different times, potentially leading to variations in the properties of the same qubits or CX gates. In contrast, for QAOA demonstrations on other QPUs, all transpilation methods are applied simultaneously, ensuring that the properties of the same qubits and two-qubit gates remain unchanged.

TABLE X. Calibration data for the 4-qubit QAOA demonstration on ibm_nairobi in Fig. 12.

Qubit	T_1 (μ s)	T_2 (μ s)	Freq. (GHz)	Anh. (GHz)	P_{01} (%)	P_{10} (%)	RO err. (%)	Gate err. (%)	CX gate	CX err. (%)
0	118.2	30.97	5.26	−0.3398	3.1	1.56	2.33	0.028	(0,1)	1.03
1	115.27	108.96	5.17	−0.3406	3.42	1.18	2.3	0.03	(1,2)	0.8
2	109.8	131.81	5.274	−0.3389	4.8	0.72	2.76	0.053	(1,3)	0.6
3	100.25	66.01	5.027	−0.3425	4.02	0.82	2.42	0.042	(3,5)	1.62
4	83.88	86.5	5.177	−0.3406	3.04	1.08	2.06	0.025	(4,5)	0.92
5	70.59	18.87	5.293	−0.3405	14.94	6.52	10.73	0.074		

TABLE XI. Qubits used in the demonstration of n -qubit QAOA with depth p , represented as D_n^p , in Figs. 13 and 14. Empty cells indicate that the value is the same as that in the previous row.

IBM QPU	QAOA	AOQMAP	Tket	Qiskit	SWAPNK
ibm_perth	D_3^1, \dots, D_3^7	[0, 1, 3]	[3, 5, 6]	[1, 3, 5]	[0, 1, 3]
	$D_5^1, D_5^3, \dots, D_5^7$	[0, 1, 3, 4, 5]	[1, 3, 4, 5, 6]	[0, 1, 2, 3, 5]	[0, 1, 3, 4, 5]
ibm_nazca	D_5^2			[1, 3, 4, 5, 6]	
	D_{10}^1	[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]	[60, 61, 62, 63, 64, 72, 80, 81, 82, 83]	[22, 23, 24, 34, 40, 41, 42, 43, 44, 45]	[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
	D_{10}^2		[60, 61, 62, 63, 64, 72, 79, 80, 81, 82]	[1, 2, 3, 4, 5, 15, 20, 21, 22, 23]	
	D_{10}^3			[20, 21, 22, 23, 24, 34, 39, 40, 42, 43]	
	$D_{10}^4, \dots, D_{10}^7$			[4, 15, 21, 22, 23, 24, 34, 42, 43, 44]	
ibmq_kolkata	D_3^1, \dots, D_3^7	[21, 23, 24]	[21, 23, 24]	[21, 23, 24]	[21, 23, 24]
	D_5^1	[12, 13, 15, 17, 18]	[7, 10, 12, 13, 15]	[10, 12, 13, 15, 18]	[12, 13, 15, 17, 18]
	D_5^2				[12, 13, 14, 15, 16]
	D_5^3	[12, 13, 14, 15, 16]			
	D_5^4, \dots, D_5^7		[16, 19, 20, 22, 25]		
	D_{10}^1	[1, 4, 7, 10, 12, 15, 18, 21, 23, 24]	[7, 8, 10, 11, 12, 13, 14, 15, 16, 19]	[7, 10, 12, 13, 14, 15, 17, 18, 21, 23]	[1, 4, 7, 10, 12, 15, 18, 21, 23, 24]
	D_{10}^2	[12, 13, 14, 15, 16, 18, 19, 21, 23, 24]	[7, 8, 10, 11, 12, 13, 14, 15, 16, 18]	[7, 10, 12, 13, 15, 17, 18, 21, 23, 24]	[0, 1, 4, 7, 10, 12, 13, 14, 16, 19]
	D_{10}^3	[0, 1, 4, 7, 10, 12, 13, 14, 16, 19]	[8, 10, 11, 12, 13, 14, 15, 16, 19, 20]	[10, 12, 13, 14, 15, 17, 18, 21, 23, 24]	
	D_{10}^4, D_{10}^5		[7, 8, 10, 11, 12, 13, 14, 15, 16, 18]	[7, 10, 12, 13, 14, 15, 17, 18, 21, 23]	
	D_{10}^6		[8, 10, 11, 12, 13, 14, 15, 16, 19, 20]		
	D_{10}^7		[7, 8, 10, 11, 12, 13, 14, 15, 16, 18]	[12, 13, 14, 15, 16, 19, 22, 23, 24, 25]	
ibmq_ehningen	D_3^1, \dots, D_3^7	[1, 4, 7]	[1, 4, 7]	[1, 4, 7]	[1, 4, 7]
	D_5^1, \dots, D_5^7	[10, 11, 12, 13, 14]	[13, 14, 16, 19, 20]	[16, 19, 20, 22, 25]	[19, 20, 22, 24, 25]
ibm_nairobi ibm_cusco	D_6^1, \dots, D_6^7	[1, 4, 7, 10, 12, 13]	[12, 13, 14, 16, 19, 20]	[14, 16, 19, 20, 22, 25]	[19, 20, 22, 23, 24, 25]
	D_3^1, \dots, D_3^7	[1, 2, 3]	[0, 1, 3]	[1, 2, 3]	[1, 2, 3]
	D_5^1, \dots, D_5^7	[0, 1, 2, 3, 4]	[44, 45, 46, 54, 64]	[4, 5, 6, 7, 15]	[0, 1, 2, 3, 4]
	D_{10}^1	[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]	[60, 61, 62, 63, 72, 79, 80, 81, 82, 83]	[1, 2, 3, 4, 5, 6, 7, 8, 15, 22]	[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
	D_{10}^2		[60, 61, 62, 63, 64, 72, 79, 80, 81, 82]	[3, 4, 5, 6, 7, 15, 21, 22, 23, 24]	
	D_{10}^3			[0, 1, 2, 3, 4, 14, 15, 19, 20, 22]	
	D_{10}^4, D_{10}^5			[3, 4, 5, 6, 7, 8, 9, 15, 16, 22]	
	D_{10}^6			[3, 4, 5, 6, 7, 15, 21, 22, 23, 24]	
	D_{10}^7			[3, 4, 5, 6, 15, 21, 22, 23, 24, 25]	

TABLE XII. Calibration data for the n -qubit QAOA demonstration on ibm_perth in Figs. 13 and 14.

n	Qubit	T_1 (μ s)	T_2 (μ s)	Freq. (GHz)	Anh. (GHz)	P_{01} (%)	P_{10} (%)	RO err. (%)	Gate err. (%)	CX gate	CX err. (%)
3	0	147.9	82.13	5.158	-0.3415	3.26	2.48	2.87	0.073	(0,1)	0.6
	1	166.58	51.27	5.034	-0.3444	2.82	2.84	2.83	0.031	(1,3)	0.46
	3	113.38	95.99	5.125	-0.3404	1.52	1.72	1.62	0.021	(3,5)	0.65
	5	144.73	108.09	4.979	-0.3460	3.08	2.74	2.91	0.026	(5,6)	1.14
	6	145.99	298.35	5.157	-0.3405	1.10	1.14	1.12	0.032		
5	0	145.05	89.48	5.158	-0.3415	2.80	2.44	2.62	0.035	(0,1)	0.73
	1	129.60	49.00	5.034	-0.3444	2.70	2.68	2.69	0.032	(1,2)	0.81
	2	132.88	76.48	4.863	-0.3473	2.84	11.90	7.37	0.036	(1,3)	0.48
	3	174.95	185.66	5.125	-0.3404	2.48	1.80	2.14	0.026	(3,5)	0.72
	4	138.13	98.61	5.159	-0.3334	3.00	2.46	2.73	0.036	(4,5)	0.83
	5	134.09	135.52	4.979	-0.3460	3.46	3.42	3.44	0.028	(5,6)	1.18
	6	158.21	237.22	5.157	-0.3405	1.56	0.98	1.27	0.030		

TABLE XIII. Calibration data for the 10-qubit QAOA demonstration on ibm_nazca in Figs. 13 and 14.

Qubit	T_1 (μ s)	T_2 (μ s)	Freq. (GHz)	Anh. (GHz)	P_{01} (%)	P_{10} (%)	RO err. (%)	Gate err. (%)	ECR gate	ECR err. (%)
0	251.62	65.56	5.092	-0.3064	1.28	1.14	1.21	0.044	(0, 1)	1.10
1	97.45	133.83	4.971	-0.3076	3.22	2.74	2.98	0.017	(1, 2)	0.43
2	289.35	224.95	4.891	-0.3089	2.80	2.90	2.85	0.015	(3, 2)	0.69
3	213.89	142.91	5.053	-0.3067	2.06	2.68	2.37	0.041	(3, 4)	0.63
4	293.74	267.30	4.978	-0.3078	3.82	3.72	3.77	0.015	(4, 15)	0.57
5	142.85	102.58	5.041	-0.3067	2.60	2.22	2.41	0.037	(5, 4)	0.67
6	155.86	63.35	5.170	-0.3038	3.78	2.50	3.14	0.149	(6, 5)	1.13
7	172.99	68.09	5.081	-0.3059	0.56	0.40	0.48	0.044	(6, 7)	1.04
8	228.30	64.16	5.003	-0.3076	0.56	0.46	0.51	0.039	(7, 8)	0.99
9	212.86	23.48	5.102	-0.3062	2.30	2.48	2.39	0.067	(9, 8)	1.61
15	201.55	112.02	5.153	-0.3054	1.00	0.68	0.84	0.021	(15, 22)	0.49
20	228.55	87.75	5.029	-0.3062	5.88	5.68	5.78	0.039	(20, 21)	0.63
21	213.32	236.54	5.169	-0.3041	4.98	5.62	5.30	0.021	(21, 22)	0.52
22	270.45	269.29	5.061	-0.3063	1.32	1.16	1.24	0.014	(23, 22)	0.53
23	227.09	21.28	5.005	-0.3072	3.92	3.38	3.65	0.021	(24, 23)	0.78
24	178.20	198.23	5.074	-0.3059	3.18	2.54	2.86	0.022	(25, 24)	1.14
25	145.51	135.14	5.164	-0.3054	2.48	2.38	2.43	0.049	(34, 24)	0.63
34	257.88	157.67	4.935	-0.3081	1.66	1.32	1.49	0.016	(34, 43)	0.59
39	217.16	147.13	5.019	-0.3071	3.66	3.26	3.46	0.027	(40, 39)	0.74
40	281.53	344.42	5.085	-0.3062	2.34	2.82	2.58	0.023	(40, 41)	0.55
41	229.28	265.23	4.953	-0.3083	1.66	2.38	2.02	0.016	(41, 42)	0.83
42	177.87	220.57	5.092	-0.3062	3.58	2.86	3.22	0.033	(42, 43)	0.85
43	372.59	244.69	4.870	-0.3094	2.26	1.92	2.09	0.015	(43, 44)	0.49
44	150.64	20.88	5.024	-0.3073	1.98	1.40	1.69	0.016	(45, 44)	0.67
45	165.29	91.74	5.112	-0.3051	5.76	6.40	6.08	0.036		
60	233.08	164.62	5.113	-0.3055	1.44	1.24	1.34	0.026	(61, 60)	0.73
61	293.83	194.05	4.912	-0.3085	6.72	7.00	6.86	0.026	(62, 61)	0.58
62	245.28	147.00	5.015	-0.3067	6.04	10.26	8.15	0.021	(63, 62)	1.40
63	129.04	179.81	5.080	-0.3068	1.16	0.66	0.91	0.059	(64, 63)	2.01
64	89.95	20.21	5.281	-0.3038	2.16	2.58	2.37	0.061	(72, 62)	0.81
72	214.77	118.50	5.078	-0.3049	1.44	1.28	1.36	0.043	(72, 81)	0.76
79	190.09	17.58	4.927	-0.3086	3.26	2.44	2.85	0.025	(80, 79)	0.57
80	185.49	160.87	4.993	-0.3077	2.40	2.42	2.41	0.019	(80, 81)	0.86
81	198.15	84.38	5.153	-0.3051	3.38	5.00	4.19	0.027	(81, 82)	0.76
82	212.53	235.84	5.205	-0.3046	0.86	1.28	1.07	0.019	(82, 83)	2.36
83	230.82	184.81	5.103	-0.3063	1.26	1.04	1.15	0.042		

TABLE XIV. Calibration data for the n -qubit QAOA demonstration on ibmq_kolkata in Figs. 13 and 14.

n	Qubit	T_1 (μ s)	T_2 (μ s)	Freq. (GHz)	Anh. (GHz)	P_{01} (%)	P_{10} (%)	RO err. (%)	Gate err. (%)	CX gate	CX err. (%)
3	21	79.08	19.49	5.274	-0.3408	0.38	0.54	0.46	0.022	(21, 23)	0.47
	23	108.35	49.07	5.138	-0.3431	0.80	0.34	0.57	0.016	(23, 24)	0.39
	24	134.94	83.96	5.005	-0.3459	1.04	0.68	0.86	0.015		
5	7	124.68	41.33	5.031	-0.3457	2.68	2.14	2.41	0.021	(7, 10)	2.25
	10	76.02	46.92	5.178	-0.3416	1.06	0.88	0.97	0.049	(10, 12)	2.10
	12	118.60	178.39	4.961	-0.3465	0.78	0.58	0.68	0.023	(12, 13)	0.70
	13	131.15	218.25	5.018	-0.3459	0.92	0.76	0.84	0.029	(12, 15)	0.53
	14	126.66	163.69	5.118	-0.3428	3.96	17.70	10.83	0.024	(13, 14)	0.65
	15	143.99	145.95	5.041	-0.3439	0.70	0.64	0.67	0.031	(14, 16)	0.65
	16	112.30	90.13	5.222	-0.3402	1.02	2.04	1.53	0.016	(15, 18)	0.76
	17	50.03	28.86	5.236	-0.3400	4.26	0.48	2.37	0.033	(16, 19)	0.76
	18	135.08	136.00	5.097	-0.3444	1.16	1.10	1.13	0.048	(17, 18)	1.20
	19	77.11	23.42	5.002	-0.3449	3.20	11.86	7.53	0.041	(19, 20)	1.19
	20	78.02	20.40	5.187	-0.3408	0.98	1.18	1.08	0.023	(19, 22)	0.97
	22	93.15	35.55	5.127	-0.3433	3.74	3.84	3.79	0.026	(22, 25)	0.60
	25	101.55	117.17	4.921	-0.3473	0.68	0.68	0.68	0.018		
10	0	90.1	92.67	5.204	-0.3415	1.04	0.74	0.89	0.021	(0, 1)	0.42
	1	128.73	221.16	4.991	-0.3453	1.28	0.86	1.07	0.018	(1, 4)	0.54
	4	89.1	134.19	5.225	-0.3410	2.58	1.92	2.25	0.026	(4, 7)	1.06
	7	88.36	40.82	5.031	-0.3457	2.42	2.36	2.39	0.033	(7, 10)	0.77
	8	119.69	55.53	4.928	-0.3454	5.26	2.80	4.03	0.045	(8, 11)	1.30
	10	116.95	45.47	5.178	-0.3416	1.00	0.88	0.94	0.020	(10, 12)	0.88
	11	44.99	29.61	4.868	-0.3734	13.96	14.20	14.08	0.025	(11, 14)	100
	12	84.02	156.07	4.961	-0.3465	0.80	0.72	0.76	0.016	(12, 13)	0.66
	13	84.04	202.42	5.018	-0.3459	0.92	0.66	0.79	0.024	(12, 15)	0.92
	14	134.37	279.81	5.118	-0.3428	3.76	12.62	8.19	0.017	(13, 14)	0.52
	15	121.91	287.98	5.041	-0.3439	0.64	0.66	0.65	0.027	(14, 16)	0.62
	16	116.7	91.08	5.222	-0.3402	0.78	1.32	1.05	0.019	(15, 18)	0.65
	17	78.53	31.69	5.236	-0.3400	0.80	0.44	0.62	0.024	(16, 19)	0.78
	18	118.3	104.6	5.097	-0.3444	1.02	0.56	0.79	0.019	(17, 18)	1.19
	19	112.14	25.47	5.002	-0.3449	5.34	13.90	9.62	0.028	(18, 21)	1.31
	20	124.65	19.87	5.187	-0.3408	0.74	0.60	0.67	0.019	(19, 20)	0.90
	21	69.31	17.05	5.274	-0.3408	0.52	0.56	0.54	0.023	(19, 22)	0.82
	22	85.69	36.12	5.127	-0.3433	3.82	4.22	4.02	0.028	(21, 23)	0.53
	23	139.81	52.81	5.138	-0.3431	0.48	0.52	0.50	0.018	(22, 25)	0.78
	24	108.62	76.32	5.005	-0.3459	0.94	1.00	0.97	0.014	(23, 24)	0.42
	25	146.49	105.09	4.921	-0.3473	1.02	0.40	0.71	0.028	(24, 25)	100

TABLE XV. Calibration data for the n -qubit QAOA demonstration on ibmq_ehningen in Figs. 13 and 14.

n	Method	Qubit	T_1 (μ s)	T_2 (μ s)	Freq. (GHz)	Anh. (GHz)	P_{01} (%)	P_{10} (%)	RO err. (%)	Gate err. (%)	CX gate	CX err. (%)
3	AOQMAP	1	185.32	219.79	5.182	-0.34	0.82	1.00	0.91	0.023	(1, 4)	0.44
		4	119.66	198.52	5.054	-0.3426	0.90	0.52	0.71	0.017	(4, 7)	0.55
		7	175.82	243.50	4.978	-0.344	0.76	0.66	0.71	0.017		
	Tket	1	185.32	219.79	5.182	-0.34	0.82	1.00	0.91	0.023	(1, 4)	0.44
		4	114.45	198.52	5.054	-0.3426	0.90	0.52	0.71	0.017	(4, 7)	0.55
		7	175.82	243.50	4.978	-0.344	0.76	0.66	0.71	0.017		
	Qiskit	1	185.32	219.79	5.182	-0.34	0.82	1.00	0.91	0.023	(1, 4)	0.44
		4	119.66	198.52	5.054	-0.3426	0.90	0.52	0.71	0.017	(4, 7)	0.55
		7	175.82	243.50	4.978	-0.344	0.76	0.66	0.71	0.017		
5	SWAPNK	1	185.32	219.79	5.182	-0.34	0.82	1.00	0.91	0.023	(1, 4)	0.44
		4	114.45	198.52	5.054	-0.3426	0.90	0.52	0.71	0.017	(4, 7)	0.55
		7	175.82	243.50	4.978	-0.344	0.76	0.66	0.71	0.017		
	AOQMAP	10	115.74	57.65	4.835	-0.3471	0.74	0.46	0.60	0.017	(10, 12)	0.50
		11	127.56	74.74	5.119	-0.3405	1.54	1.34	1.44	0.025	(11, 14)	0.84
		12	183.56	442.33	4.725	-0.3484	1.00	0.90	0.95	0.022	(12, 13)	0.66
	Tket	13	147.90	271.77	4.926	-0.3440	1.74	0.96	1.35	0.018	(13, 14)	0.59
		14	107.65	196.10	5.177	-0.3408	1.08	0.44	0.76	0.031		
		13	127.27	257.59	4.926	-0.3440	1.34	0.78	1.06	0.023	(13, 14)	0.59
	Qiskit	14	219.32	242.17	5.177	-0.3408	1.14	0.32	0.73	0.029	(14, 16)	0.59
		16	139.95	209.99	5.022	-0.3435	0.60	0.44	0.52	0.016	(16, 19)	0.56
		19	137.67	81.26	4.784	-0.3485	1.06	0.64	0.85	0.021	(19, 20)	0.47
	SWAPNK	20	219.74	227.51	5.042	-0.3426	1.76	2.62	2.19	0.031		
		16	168.67	213.73	5.022	-0.3435	0.70	0.42	0.56	0.016	(16, 19)	0.80
		19	130.23	79.19	4.784	-0.3485	1.10	0.52	0.81	0.022	(19, 20)	0.54
	Tket	20	153.35	275.77	5.042	-0.3426	1.36	2.26	1.81	0.036	(19, 22)	0.70
		22	187.89	32.12	4.725	-0.3464	2.00	0.64	1.32	0.020	(22, 25)	0.51
		25	210.17	76.32	4.950	-0.3457	0.90	0.80	0.85	0.016		
	Qiskit	19	130.23	79.19	4.784	-0.3485	1.10	0.52	0.81	0.022	(19, 20)	0.54
		20	153.35	275.77	5.042	-0.3426	1.36	2.26	1.81	0.036	(19, 22)	0.70
		22	187.89	32.12	4.725	-0.3464	2.00	0.64	1.32	0.020	(22, 25)	0.51
	SWAPNK	24	197.66	280.74	5.074	-0.3416	1.04	0.64	0.84	0.013	(24, 25)	0.76
		25	210.17	76.32	4.950	-0.3457	0.90	0.80	0.85	0.016		
6	AOQMAP	1	112.64	141.73	5.182	-0.3400	1.06	0.56	0.81	0.024	(1, 4)	0.76
		4	106.71	151.78	5.054	-0.3426	0.58	0.52	0.55	0.018	(4, 7)	0.40
		7	175.57	105.71	4.978	-0.3440	0.96	0.48	0.72	0.016	(7, 10)	1.59
	Tket	10	88.47	62.04	4.835	-0.3471	0.86	0.42	0.64	0.022	(10, 12)	0.60
		12	186.44	239.62	4.725	-0.3484	0.88	0.98	0.93	0.021	(12, 13)	0.52
		13	157.18	224.98	4.926	-0.3440	1.42	1.10	1.26	0.021		
	Qiskit	12	219.63	317.95	4.725	-0.3484	1.04	0.88	0.96	0.020	(12, 13)	0.66
		13	127.27	257.59	4.926	-0.3440	1.34	0.78	1.06	0.023	(13, 14)	0.59
		14	219.32	242.17	5.177	-0.3408	1.14	0.32	0.73	0.029	(14, 16)	0.59
	SWAPNK	16	139.95	209.99	5.022	-0.3435	0.60	0.44	0.52	0.016	(16, 19)	0.56
		19	137.67	81.26	4.784	-0.3485	1.06	0.64	0.85	0.021	(19, 20)	0.47
		20	219.74	227.51	5.042	-0.3426	1.76	2.62	2.19	0.031		
	Tket	14	217.62	256.21	5.177	-0.3408	1.00	0.44	0.72	0.030	(14, 16)	0.76
		16	168.67	213.73	5.022	-0.3435	0.70	0.42	0.56	0.016	(16, 19)	0.80
		19	130.23	79.19	4.784	-0.3485	1.10	0.52	0.81	0.022	(19, 20)	0.54
	Qiskit	20	153.35	275.77	5.042	-0.3426	1.36	2.26	1.81	0.036	(19, 22)	0.70
		22	187.89	32.12	4.725	-0.3464	2.00	0.64	1.32	0.020	(22, 25)	0.51
		25	210.17	76.32	4.950	-0.3457	0.90	0.80	0.85	0.016		
	SWAPNK	19	130.23	79.19	4.784	-0.3485	1.10	0.52	0.81	0.022	(19, 20)	0.54
		20	140.45	275.77	5.042	-0.3426	1.36	2.26	1.81	0.036	(19, 22)	0.70
		22	306.37	32.12	4.725	-0.3464	2.00	0.64	1.32	0.020	(22, 25)	0.51
	Tket	23	173.97	259.46	4.805	-0.3471	0.92	0.76	0.84	0.024	(23, 24)	0.59
		24	139.76	280.74	5.074	-0.3416	1.04	0.64	0.84	0.013	(24, 25)	0.76
		25	210.17	76.32	4.950	-0.3457	0.90	0.80	0.85	0.016		

TABLE XVI. Calibration data for the 3-qubit QAOA demonstration on ibm_nairobi in Figs. 13 and 14.

Qubit	T_1 (μ s)	T_2 (μ s)	Freq. (GHz)	Anh. (GHz)	P_{01} (%)	P_{10} (%)	RO err. (%)	Gate err. (%)	CX gate	CX err. (%)
0	118.2	30.97	5.26	-0.3398	3.10	1.56	2.33	0.028	(0, 1)	1.03
1	115.27	108.96	5.17	-0.3406	3.42	1.18	2.30	0.030	(1, 2)	0.80
2	109.8	131.81	5.274	-0.3389	4.80	0.72	2.76	0.053	(1, 3)	0.60
3	100.25	66.01	5.027	-0.3425	4.02	0.82	2.42	0.042		

TABLE XVII. Calibration data for the n -qubit QAOA demonstration on ibm_cusco in Figs. 13 and 14.

n	Qubit	T_1 (μ s)	T_2 (μ s)	Freq. (GHz)	Anh. (GHz)	P_{01} (%)	P_{10} (%)	RO err. (%)	Gate err. (%)	ECR gate	ECR err. (%)
5	0	189.11	132.77	5.015	-0.3074	0.94	1.14	1.04	0.016	(0,1)	0.90
	1	263.60	279.60	4.956	-0.3078	1.74	1.90	1.82	0.087	(2,1)	0.78
	2	281.81	378.48	4.807	-0.3104	0.58	0.16	0.37	0.013	(3,2)	0.70
	3	186.89	218.49	5.232	-0.3037	5.56	6.38	5.97	0.019	(4,3)	0.77
	4	97.79	145.27	5.138	-0.3054	0.72	1.26	0.99	0.013	(4,15)	0.42
	5	125.46	153.12	4.988	-0.3076	1.40	2.44	1.92	0.020	(5,4)	0.52
	6	293.68	109.06	4.899	-0.3075	2.06	1.54	1.80	0.017	(5,6)	0.47
	7	123.62	65.29	5.023	-0.3068	4.78	6.52	5.65	0.069	(7,6)	0.65
	15	192.38	63.09	5.015	-0.3070	1.50	1.38	1.44	0.015	(44,45)	0.85
	44	205.90	252.15	5.387	-0.3018	3.50	3.54	3.52	0.014	(46,45)	1.55
	45	211.66	263.52	4.960	-0.3071	9.32	3.20	6.26	0.047	(54,45)	0.57
	46	94.46	111.22	5.220	-0.3042	1.22	4.28	2.75	0.036	(54,64)	0.89
	54	232.49	235.38	5.127	-0.3045	1.64	0.88	1.26	0.015		
	64	103.23	126.04	5.225	-0.3044	2.44	2.18	2.31	0.067		
10	0	166.52	153.00	5.015	-0.3074	0.94	1.14	1.04	0.019	(0, 1)	1.34
	1	46.79	92.70	4.956	-0.3078	1.74	1.90	1.82	0.029	(2, 1)	1.34
	2	206.46	303.31	4.807	-0.3104	0.58	0.16	0.37	0.015	(3, 2)	0.80
	3	106.73	166.80	5.232	-0.3037	5.56	6.38	5.97	0.017	(4, 3)	0.82
	4	97.79	170.31	5.138	-0.3054	0.72	1.26	0.99	0.015	(4, 15)	0.44
	5	127.30	151.30	4.988	-0.3076	1.40	2.44	1.92	0.018	(5, 4)	0.57
	6	193.88	103.67	4.899	-0.3075	2.06	1.54	1.80	0.026	(5, 6)	0.51
	7	94.32	86.65	5.023	-0.3068	4.78	6.52	5.65	0.025	(7, 6)	0.54
	8	188.77	74.32	5.112	-0.3057	3.98	3.20	3.59	0.050	(8, 7)	0.80
	9	109.93	18.87	5.153	-0.3057	7.00	7.30	7.15	0.056	(9, 8)	1.34
	14	228.15	96.76	4.930	-0.3080	1.68	0.98	1.33	0.018	(14, 0)	0.56
	15	217.31	63.08	5.015	-0.3070	1.50	1.38	1.44	0.018	(16, 8)	2.46
	16	154.00	11.77	5.279	-0.3024	2.04	2.16	2.10	0.110	(20, 19)	0.77
	19	218.98	246.21	4.958	-0.3080	1.22	0.46	0.84	0.017	(21, 20)	1.51
	20	110.45	122.17	5.044	-0.3059	1.36	0.96	1.16	0.059	(21, 22)	0.83
	21	194.17	165.34	5.160	-0.3032	7.82	6.90	7.36	0.051	(22, 15)	1.04
	22	170.53	92.47	5.282	-0.3030	1.20	1.04	1.12	0.025	(23, 22)	1.49
	23	125.61	76.39	5.163	-0.3052	11.00	9.90	10.45	0.029	(23, 24)	0.71
	24	99.75	93.21	5.080	-0.3049	4.50	4.24	4.37	0.119	(25, 24)	0.75
	25	145.41	115.54	5.137	-0.3052	1.26	1.28	1.27	0.026	(60, 61)	1.69
	60	146.71	140.25	5.215	-0.3038	8.42	7.32	7.87	0.030	(62, 61)	0.69
	61	173.45	43.52	5.172	-0.2923	9.72	10.76	10.24	0.031	(62, 63)	2.35
	62	105.67	51.10	5.042	-0.3063	2.52	2.02	2.27	0.034	(64, 63)	4.04
	63	100.05	52.86	5.190	-0.3137	4.94	5.46	5.20	0.144	(72, 62)	0.95
	64	109.50	114.47	5.225	-0.3044	2.44	2.18	2.31	0.068	(72, 81)	2.20
	72	89.54	51.99	5.306	-0.3021	7.22	41.82	24.52	0.041	(80, 79)	1.14
	79	154.87	158.37	5.161	-0.3048	5.72	3.44	4.58	0.158	(80, 81)	4.26
	80	151.14	147.43	5.061	-0.3063	1.40	0.76	1.08	0.026	(81, 82)	2.02
	81	118.18	169.45	5.129	-0.3054	12.67	0.33	6.50	0.287	(83, 82)	1.29
	82	68.75	95.77	5.204	-0.3046	6.00	6.33	6.17	0.097		
	83	116.90	193.49	5.098	-0.3061	4.46	0.56	2.51	0.035		

- [1] M. Cerezo, A. Arrasmith, R. Babbush, S. C. Benjamin, S. Endo, K. Fujii, J. R. McClean, K. Mitarai, X. Yuan, L. Cincio *et al.*, Variational quantum algorithms, *Nat. Rev. Phys.* **3**, 625 (2021).
- [2] K. Bharti, A. Cervera-Lierta, T. H. Kyaw, T. Haug, S. Alperin-Lea, A. Anand, M. Degroote, H. Heimonen, J. S. Kottmann, T. Menke *et al.*, Noisy intermediate-scale quantum algorithms, *Rev. Mod. Phys.* **94**, 015004 (2022).
- [3] E. Farhi, J. Goldstone, and S. Gutmann, A quantum approximate optimization algorithm, *arXiv:1411.4028*.
- [4] H. Ma, M. Govoni, and G. Galli, Quantum simulations of materials on near-term quantum computers, *npj Comput. Mater.* **6**, 85 (2020).
- [5] Y. Hu, F. Meng, X. Wang, T. Luan, Y. Fu, Z. Zhang, X. Zhang, and X. Yu, Greedy algorithm based circuit optimization for near-term quantum simulation, *Quantum Sci. Technol.* **7**, 045001 (2022).
- [6] G. Li, Y. Ding, and Y. Xie, in *Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems*, ASPLOS '19 (Association for Computing Machinery, New York, NY, USA, 2019), pp. 1001–1014.
- [7] A. Cowtan, S. Dilkes, R. Duncan, A. Krajenbrink, W. Simmons, and S. Sivarajah, in *14th Conference on the Theory of Quantum Computation, Communication and Cryptography (TQC 2019)*, Leibniz International Proceedings in Informatics (LIPIcs), edited by W. van Dam and L. Mancinska (Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, Dagstuhl, Germany, 2019), Vol. 135, pp. 5:1–5:32.
- [8] P. Zhu, S. Feng, and Z. Guan, An iterated local search methodology for the qubit mapping problem, *IEEE Trans. Comput.-Aided Des. Integrated Circuits Syst.* **41**, 2587 (2022).
- [9] S. Niu, A. Suau, G. Staffelbach, and A. Todri-Sanial, A hardware-aware heuristic for the qubit mapping problem in the NISQ era, *IEEE Trans. Quantum Eng.* **1**, 1 (2020).
- [10] C. Zhang, A. B. Hayes, L. Qiu, Y. Jin, Y. Chen, and E. Z. Zhang, in *Proceedings of the 26th ACM International Conference on Architectural Support for Programming Languages and Operating Systems*, ASPLOS '21 (Association for Computing Machinery, New York, NY, USA, 2021), pp. 360–374.
- [11] M. Y. Siraichi, V. F. d. Santos, C. Collange, and F. M. Q. Pereira, in *Proceedings of the 2018 International Symposium on Code Generation and Optimization*, CGO 2018 (Association for Computing Machinery, New York, NY, USA, 2018), pp. 113–125.
- [12] D. Bhattacharjee and A. Chattopadhyay, Depth-optimal quantum circuit placement for arbitrary topologies, *arXiv:1703.08540*.
- [13] A. Shafaei, M. Saeedi, and M. Pedram, in *2014 19th Asia and South Pacific Design Automation Conference (ASP-DAC)* (IEEE, Piscataway, NJ, USA, 2014), pp. 495–500.
- [14] P. Murali, J. M. Baker, A. Javadi-Abhari, F. T. Chong, and M. Martonosi, in *Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems*, ASPLOS '19 (Association for Computing Machinery, New York, NY, USA, 2019), pp. 1015–1029.
- [15] M. Y. Siraichi, V. F. d. Santos, C. Collange, and F. M. Q. Pereira, Qubit allocation as a combination of subgraph isomorphism and token swapping, *Proc. ACM Program. Lang.* **3**, 120 (2019).
- [16] B. Tan and J. Cong, in *Proceedings of the 39th International Conference on Computer-Aided Design*, ICCAD '20 (Association for Computing Machinery, New York, NY, USA, 2020).
- [17] A. Zulehner, A. Paler, and R. Wille, An efficient methodology for mapping quantum circuits to the IBM QX architectures, *IEEE Trans. Comput.-Aided Des. Integrated Circuits Syst.* **38**, 1226 (2018).
- [18] A. M. Childs, E. Schoute, and C. M. Unsal, in *14th Conference on the Theory of Quantum Computation, Communication and Cryptography (TQC 2019)*, Leibniz International Proceedings in Informatics (LIPIcs), edited by W. van Dam and L. Mancinska (Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, Dagstuhl, Germany, 2019), Vol. 135, pp. 3:1–3:24.
- [19] Z.-T. Li, F.-X. Meng, Z.-C. Zhang, and X.-T. Yu, Qubits' mapping and routing for NISQ on variability of quantum gates, *Quantum Inf. Process.* **19**, 1 (2020).
- [20] M. P. Harrigan, K. J. Sung, M. Neeley, K. J. Satzinger, F. Arute, K. Arya, J. Atalaya, J. C. Bardin, R. Barends, S. Boixo *et al.*, Quantum approximate optimization of non-planar graph problems on a planar superconducting processor, *Nat. Phys.* **17**, 332 (2021).
- [21] I. D. Kivlichan, J. McClean, N. Wiebe, C. Gidney, A. Aspuru-Guzik, G. K.-L. Chan, and R. Babbush, Quantum simulation of electronic structure with linear depth and connectivity, *Phys. Rev. Lett.* **120**, 110501 (2018).
- [22] B. O'Gorman, W. J. Huggins, E. G. Rieffel, and K. B. Whaley, Generalized swap networks for near-term quantum computing, *arXiv:1905.05118*.
- [23] J. Weidenfeller, L. C. Valor, J. Gacon, C. Tornow, L. Bello, S. Woerner, and D. J. Egger, Scaling of the quantum approximate optimization algorithm on superconducting qubit based hardware, *Quantum* **6**, 870 (2022).
- [24] A. Hashim, R. Rines, V. Omole, R. K. Naik, J. M. Kreikebaum, D. I. Santiago, F. T. Chong, I. Siddiqi, and P. Gokhale, Optimized swap networks with equivalent circuit averaging for qaoa, *Phys. Rev. Res.* **4**, 033028 (2022).
- [25] A. Javadi-Abhari, M. Treinish, K. Krsulich, C. J. Wood, J. Lishman, J. Gacon, S. Martiel, P. D. Nation, L. S. Bishop, A. W. Cross, B. R. Johnson, and J. M. Gambetta, Quantum computing with Qiskit, *arXiv:2405.08810*.
- [26] S. Sivarajah, S. Dilkes, A. Cowtan, W. Simmons, A. Edgington, and R. Duncan, t|ket>: a retargetable compiler for NISQ devices, *Quantum Sci. Technol.* **6**, 014003 (2020).
- [27] D. Rosenberg, S. J. Weber, D. Conway, D.-R. W. Yost, J. Mallek, G. Calusine, R. Das, D. Kim, M. E. Schwartz, W. Woods, J. L. Yoder, and W. D. Oliver, Solid-state qubits: 3D integration and packaging, *IEEE Microw. Mag.* **21**, 72 (2020).
- [28] B. M. Niedzielski, D. K. Kim, M. E. Schwartz, D. Rosenberg, G. Calusine, R. Das, A. J. Melville, J. Plant, L. Racz,

- J. L. Yoder, D. Ruth-Yost, and W. D. Oliver, in *2019 IEEE International Electron Devices Meeting (IEDM)* (IEEE, Piscataway, NJ, USA, 2019), pp. 31.3.1–31.3.4.
- [29] J. Rahamim, T. Behrle, M. J. Peterer, A. Patterson, P. A. Spring, T. Tsunoda, R. Manenti, G. Tancredi, and P. J. Leek, Double-sided coaxial circuit QED with out-of-plane wiring, *Appl. Phys. Lett.* **110**, 222602 (2017).
- [30] P. Wang, C.-Y. Luan, M. Qiao, M. Um, J. Zhang, Y. Wang, X. Yuan, M. Gu, J. Zhang, and K. Kim, Single ion qubit with estimated coherence time exceeding one hour, *Nat. Commun.* **12**, 233 (2021).
- [31] G. Brida, I. P. Degiovanni, M. Genovese, F. Piacentini, P. Traina, A. Della Frera, A. Tosi, A. Bahgat Shehata, C. Scarcella, A. Gulinatti, M. Ghioni, S. V. Polyakov, A. Migdall, and A. Giudice, An extremely low-noise heralded single-photon source: A breakthrough for quantum technologies, *Appl. Phys. Lett.* **101**, 221112 (2012).
- [32] J. Preskill, Quantum computing in the NISQ era and beyond, *Quantum* **2**, 79 (2018).
- [33] P. Chandarana, P. S. Vieites, N. N. Hegade, E. Solano, Y. Ban, and X. Chen, Meta-learning digitized-counterdiabatic quantum optimization, *Quantum Sci. Technol.* **8**, 045007 (2023).
- [34] A. Kandala, A. Mezzacapo, K. Temme, M. Takita, M. Brink, J. M. Chow, and J. M. Gambetta, Hardware-efficient variational quantum eigensolver for small molecules and quantum magnets, *Nature* **549**, 242 (2017).
- [35] Y. Ji, S. Brandhofer, and I. Polian, in *2022 IEEE International Conference on Quantum Computing and Engineering (QCE)* (IEEE Computer Society, Los Alamitos, CA, USA, 2022), pp. 204–214.
- [36] P. Gokhale, A. Javadi-Abhari, N. Earnest, Y. Shi, and F. T. Chong, in *2020 53rd Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)* (IEEE Computer Society, Los Alamitos, CA, USA, 2020), pp. 186–200.
- [37] K. N. Smith, G. S. Ravi, T. Alexander, N. T. Bronn, A. Carvalho, A. Cervera-Lierta, F. T. Chong, J. M. Chow, M. Cubeddu, A. Hashim, L. Jiang, O. Lanes, M. J. Otten, D. I. Schuster, P. Gokhale, N. Earnest, and A. Galda, Summary: Chicago quantum exchange (CQE) pulse-level quantum control workshop, [arXiv:2202.13600](https://arxiv.org/abs/2202.13600).
- [38] P. Gokhale, Y. Ding, T. Propson, C. Winkler, N. Leung, Y. Shi, D. I. Schuster, H. Hoffmann, and F. T. Chong, in *Proceedings of the 52nd Annual IEEE/ACM International Symposium on Microarchitecture*, MICRO '52 (Association for Computing Machinery, New York, NY, USA, 2019), pp. 266–278.
- [39] Y. Shi, N. Leung, P. Gokhale, Z. Rossi, D. I. Schuster, H. Hoffmann, and F. T. Chong, in *Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems*, ASPLOS '19 (Association for Computing Machinery, New York, NY, USA, 2019), pp. 1031–1044.
- [40] P. Gokhale, T. Tomesh, M. Suchara, and F. Chong, in *Proceedings of the 2024 International Conference on Parallel Architectures and Compilation Techniques*, PACT '24 (Association for Computing Machinery, New York, NY, USA, 2024), pp. 351–362.
- [41] P. D. Nation and M. Treinish, Suppressing quantum circuit errors due to system variability, *PRX Quantum* **4**, 010327 (2023).
- [42] Y. Ji, K. F. Koenig, and I. Polian, Improving the performance of digitized counterdiabatic quantum optimization via algorithm-oriented qubit mapping, *Phys. Rev. A* **110**, 032421 (2024).
- [43] S. Brandhofer, D. Braun, V. Dehn, G. Hellstern, M. Hüls, Y. Ji, I. Polian, A. S. Bhatia, and T. Wellens, Benchmarking the performance of portfolio optimization with QAOA, *Quantum Inf. Process.* **22**, 25 (2023).
- [44] N. Earnest, C. Tornow, and D. J. Egger, Pulse-efficient circuit transpilation for quantum applications on cross-resonance-based hardware, *Phys. Rev. Res.* **3**, 043088 (2021).
- [45] Y. Ji, K. F. Koenig, and I. Polian, Optimizing quantum algorithms on bipotent architectures, *Phys. Rev. A* **108**, 022610 (2023).
- [46] A. M. Childs, A. Ostrander, and Y. Su, Faster quantum simulation by randomization, *Quantum* **3**, 182 (2019).
- [47] P. K. Faehrmann, M. Steudtner, R. Kueng, M. Kieferova, and J. Eisert, Randomizing multi-product formulas for Hamiltonian simulation, *Quantum* **6**, 806 (2022).
- [48] T. Hagge, Optimal fermionic swap networks for Hubbard models, [arXiv:2001.08324](https://arxiv.org/abs/2001.08324).
- [49] T. Peham, L. Burgholzer, and R. Wille, in *Proceedings of the 28th Asia and South Pacific Design Automation Conference*, ASPDAC '23 (Association for Computing Machinery, New York, NY, USA, 2023), pp. 702–708.
- [50] B. Mohar and S. Poljak, Eigenvalues and the max-cut problem, *Czech. Math. J.* **40**, 343 (1990).
- [51] A. Peruzzo, J. McClean, P. Shadbolt, M.-H. Yung, X.-Q. Zhou, P. J. Love, A. Aspuru-Guzik, and J. L. O'Brien, A variational eigenvalue solver on a photonic quantum processor, *Nat. Commun.* **5**, 1 (2014).
- [52] J. Tilly, H. Chen, S. Cao, D. Picozzi, K. Setia, Y. Li, E. Grant, L. Wossnig, I. Rungger, G. H. Booth *et al.*, The variational quantum eigensolver: A review of methods and best practices, *Phys. Rep.* **986**, 1 (2022).
- [53] D. A. Fedorov, B. Peng, N. Govind, and Y. Alexeev, VQE method: A short survey and recent developments, *Mater. Theory* **6**, 1 (2022).
- [54] Y. Cao, J. Romero, J. P. Olson, M. Degroote, P. D. Johnson, M. Kieferová, I. D. Kivlichan, T. Menke, B. Peropadre, N. P. Sawaya *et al.*, Quantum chemistry in the age of quantum computing, *Chem. Rev.* **119**, 10856 (2019).
- [55] B. Bauer, S. Bravyi, M. Motta, and G. K.-L. Chan, Quantum algorithms for quantum chemistry and quantum materials science, *Chem. Rev.* **120**, 12685 (2020).
- [56] D. Amaro, C. Modica, M. Rosenkranz, M. Fiorentini, M. Benedetti, and M. Lubasch, Filtering variational quantum algorithms for combinatorial optimization, *Quantum Sci. Technol.* **7**, 015021 (2022).
- [57] J. Romero, R. Babbush, J. R. McClean, C. Hempel, P. J. Love, and A. Aspuru-Guzik, Strategies for quantum computing molecular energies using the unitary coupled cluster ansatz, *Quantum Sci. Technol.* **4**, 014008 (2018).

- [58] D. Wecker, M. B. Hastings, and M. Troyer, Progress towards practical quantum variational algorithms, *Phys. Rev. A* **92**, 042303 (2015).
- [59] G. Nannicini, Performance of hybrid quantum-classical variational heuristics for combinatorial optimization, *Phys. Rev. E* **99**, 013304 (2019).
- [60] Y. Ji and I. Polian, Synergistic dynamical decoupling and circuit design for enhanced algorithm performance on near-term quantum devices, *Entropy* **26**, 586 (2024).
- [61] F. Arute, K. Arya, R. Babbush, D. Bacon, J. C. Bardin, R.arends, R. Biswas, S. Boixo, F. G. Brandao, D. A. Buell *et al.*, Quantum supremacy using a programmable superconducting processor, *Nature* **574**, 505 (2019).
- [62] J. S. Otterbach, R. Manenti, N. Alidoust, A. Bestwick, M. Block, B. Bloom, S. Caldwell, N. Didier, E. S. Fried, S. Hong *et al.*, Unsupervised machine learning on a hybrid quantum computer, [arXiv:1712.05771](https://arxiv.org/abs/1712.05771).
- [63] C. Monroe and J. Kim, Scaling the ion trap quantum processor, *Science* **339**, 1164 (2013).
- [64] I. Pogorelov, T. Feldker, C. D. Marciniak, L. Postler, G. Jacob, O. Kriegelsteiner, V. Podlesnic, M. Meth, V. Negnevitsky, M. Stadler, B. Höfer, C. Wächter, K. Lakhmanskiy, R. Blatt, P. Schindler, and T. Monz, Compact ion-trap quantum computing demonstrator, *PRX Quantum* **2**, 020343 (2021).
- [65] J. Ramette, J. Sinclair, Z. Vendeiro, A. Rudelis, M. Cetina, and V. Vuletić, Any-to-any connected cavity-mediated architecture for quantum computing with trapped ions or Rydberg arrays, *PRX Quantum* **3**, 010344 (2022).
- [66] M. Cetina, L. Egan, C. Noel, M. Goldman, D. Biswas, A. Risinger, D. Zhu, and C. Monroe, Control of transverse motion for quantum gates on individually addressed atomic qubits, *PRX Quantum* **3**, 010334 (2022).
- [67] A. Politi, M. J. Cryan, J. G. Rarity, S. Yu, and J. L. O'Brien, Silica-on-silicon waveguide quantum circuits, *Science* **320**, 646 (2008).
- [68] J. L. O'Brien, A. Furusawa, and J. Vučković, Photonic quantum technologies, *Nat. Photonics* **3**, 687 (2009).
- [69] J. Wang, F. Sciarrino, A. Laing, and M. G. Thompson, Integrated photonic quantum technologies, *Nat. Photonics* **14**, 273 (2020).
- [70] W. Luo, L. Cao, Y. Shi, L. Wan, H. Zhang, S. Li, G. Chen, Y. Li, S. Li, Y. Wang, S. Sun, M. F. Karim, H. Cai, L. C. Kwek, and A. Q. Liu, Recent progress in quantum photonic chips for quantum communication and internet, *Light Sci. Appl.* **12**, 175 (2023).
- [71] E. Knill, R. Laflamme, and G. J. Milburn, A scheme for efficient quantum computation with linear optics, *Nature* **409**, 46 (2001).
- [72] R. Raussendorf and H. J. Briegel, A one-way quantum computer, *Phys. Rev. Lett.* **86**, 5188 (2001).
- [73] P. Walther, K. J. Resch, T. Rudolph, E. Schenck, H. Weinfurter, V. Vedral, M. Aspelmeyer, and A. Zeilinger, Experimental one-way quantum computing, *Nature* **434**, 169 (2005).
- [74] N. C. Menicucci, S. T. Flammia, and O. Pfister, One-way quantum computing in the optical frequency comb, *Phys. Rev. Lett.* **101**, 130501 (2008).
- [75] F. Zilk, K. Staudacher, T. Guggemos, K. Förlinger, D. Kranzlmüller, and P. Walther, in *2022 IEEE/ACM Third International Workshop on Quantum Computing Software (QCS)* (IEEE Computer Society, Los Alamitos, CA, USA, 2022), pp. 57–67.
- [76] H. Zhang, A. Wu, Y. Wang, G. Li, H. Shapourian, A. Shabani, and Y. Ding, in *Proceedings of the 50th Annual International Symposium on Computer Architecture, ISCA '23* (Association for Computing Machinery, New York, NY, USA, 2023).
- [77] H. Zhang, J. Ruan, H. Shapourian, R. R. Kompella, and Y. Ding, in *Proceedings of the 29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 3, ASPLOS '24* (Association for Computing Machinery, New York, NY, USA, 2024), pp. 738–754.
- [78] I. Agresti, K. Paul, P. Schiansky, S. Steiner, Z. Yin, C. Pentangelo, S. Piacentini, A. Crespi, Y. Ban, F. Ceccarelli *et al.*, Demonstration of hardware efficient photonic variational quantum algorithm, [arXiv:2408.10339](https://arxiv.org/abs/2408.10339).
- [79] K. Mitarai, M. Negoro, M. Kitagawa, and K. Fujii, Quantum circuit learning, *Phys. Rev. A* **98**, 032309 (2018).
- [80] M. Benedetti, E. Lloyd, S. Sack, and M. Fiorentini, Parameterized quantum circuits as machine learning models, *Quantum Sci. Technol.* **4**, 043001 (2019).
- [81] M. Schuld, A. Bocharov, K. M. Svore, and N. Wiebe, Circuit-centric quantum classifiers, *Phys. Rev. A* **101**, 032308 (2020).
- [82] J. J. Meyer, M. Mularski, E. Gil-Fuster, A. A. Mele, F. Arzani, A. Wilms, and J. Eisert, Exploiting symmetry in variational quantum machine learning, *PRX Quantum* **4**, 010328 (2023).
- [83] L. K. Grover, in *Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing, STOC '96* (Association for Computing Machinery, New York, NY, USA, 1996), pp. 212–219.
- [84] M. J. Powell, *A Direct Search Optimization Method That Models the Objective and Constraint Functions by Linear Interpolation* (Springer Netherlands, Dordrecht, 1994).
- [85] K. Georgopoulos, C. Emary, and P. Zuliani, Modeling and simulating the noisy behavior of near-term quantum computers, *Phys. Rev. A* **104**, 062432 (2021).
- [86] J. Zeng, Z. Wu, C. Cao, C. Zhang, S.-Y. Hou, P. Xu, and B. Zeng, Simulating noisy variational quantum eigensolver with local noise models, *Quantum Eng.* **3**, e77 (2021).
- [87] M. L. Dahlhauser and T. S. Humble, Modeling noisy quantum circuits using experimental characterization, *Phys. Rev. A* **103**, 042603 (2021).
- [88] M. L. Dahlhauser and T. S. Humble, Benchmarking characterization methods for noisy quantum circuits, *Phys. Rev. A* **109**, 042620 (2024).
- [89] <https://github.com/QuantumYanJunJi/AOQMAP>.
- [90] E. Hellinger, Neue begründung der theorie quadratischer formen von unendlichvielen veränderlichen, *J. Reine Angew. Math.* **1909**, 210 (1909).
- [91] D. Leibfried, D. M. Meekhof, B. E. King, C. Monroe, W. M. Itano, and D. J. Wineland, Experimental determination of the motional quantum state of a trapped atom, *Phys. Rev. Lett.* **77**, 4281 (1996).
- [92] M. Cramer, M. B. Plenio, S. T. Flammia, R. Somma, D. Gross, S. D. Bartlett, O. Landon-Cardinal, D. Poulin, and Y.-K. Liu, Efficient quantum state tomography, *Nat. Commun.* **1**, 149 (2010).

- [93] D. Gross, Y.-K. Liu, S. T. Flammia, S. Becker, and J. Eisert, Quantum state tomography via compressed sensing, [Phys. Rev. Lett. **105**, 150401 \(2010\)](#).
- [94] M. Christandl and R. Renner, Reliable quantum state tomography, [Phys. Rev. Lett. **109**, 120403 \(2012\)](#).
- [95] J. F. Poyatos, J. I. Cirac, and P. Zoller, Complete characterization of a quantum process: The two-bit quantum gate, [Phys. Rev. Lett. **78**, 390 \(1997\)](#).
- [96] M. Mohseni, A. T. Rezakhani, and D. A. Lidar, Quantum-process tomography: Resource analysis of different strategies, [Phys. Rev. A **77**, 032322 \(2008\)](#).
- [97] E. Magesan, J. M. Gambetta, and J. Emerson, Scalable and robust randomized benchmarking of quantum processes, [Phys. Rev. Lett. **106**, 180504 \(2011\)](#).
- [98] E. Knill, D. Leibfried, R. Reichle, J. Britton, R. B. Blakestad, J. D. Jost, C. Langer, R. Ozeri, S. Seidelin, and D. J. Wineland, Randomized benchmarking of quantum gates, [Phys. Rev. A **77**, 012307 \(2008\)](#).
- [99] J. Helsen, I. Roth, E. Onorati, A. Werner, and J. Eisert, General framework for randomized benchmarking, [PRX Quantum **3**, 020357 \(2022\)](#).
- [100] F. B. Maciejewski, Z. Zimborás, and M. Oszmaniec, Mitigation of readout noise in near-term quantum devices by classical post-processing based on detector tomography, [Quantum **4**, 257 \(2020\)](#).
- [101] Y. Takeuchi, Y. Takahashi, T. Morimae, and S. Tani, Divide-and-conquer verification method for noisy intermediate-scale quantum computation, [Quantum **6**, 758 \(2022\)](#).