RESEARCH ARTICLE | JANUARY 22 2025

Enhancing computational accuracy with parallel parameter optimization in variational quantum eigensolver

Daisuke Tsukayama; Jun-ichi Shirakashi ; Tetsuo Shibuya; Hiroshi Imai

AIP Advances 15, 015226 (2025)
https://doi.org/10.1063/5.0236028

Enhancing computational accuracy with parallel parameter optimization in variational quantum eigensolver

Cite as: AIP Advances 15, 015226 (2025); doi: 10.1063/5.0236028
Submitted: 28 November 2024 • Accepted: 30 December 2024 •
Published Online: 22 January 2025

Daisuke Tsukayama,[1] Jun-ichi Shirakashi,[1,a)] Tetsuo Shibuya,[2] and Hiroshi Imai[3]

AFFILIATIONS

[1] Department of Electrical Engineering and Computer Science, Tokyo University of Agriculture and Technology, Koganei, Tokyo 184-8588, Japan
[2] Division of Medical Data Informatics, Human Genome Center, The Institute of Medical Science, The University of Tokyo, Minato, Tokyo 108-8639, Japan
[3] The Graduate School of Information Science and Technology, The University of Tokyo, Bunkyo, Tokyo 113-8656, Japan

[a)] **Author to whom correspondence should be addressed:** shrakash@cc.tuat.ac.jp

ABSTRACT

Variational quantum algorithms have promising applications in noisy intermediate-scale quantum (NISQ) devices. These algorithms rely on a classical optimization outer loop that minimizes a parameterized quantum circuit function. The optimization in variational quantum eigensolver (VQE) is NP-hard, meaning that finding the optimal solution is infeasible in the worst-case scenario. One way to address this challenge is through parallel optimization of parameters using multiple-parameterized quantum circuits. However, this approach is unsuitable for cloud-based quantum processing unit utilization due to the increased number of quantum circuit executions. Although NISQ devices have limitations in terms of gate depth, their size has been growing in recent years. Therefore, implementing multiple-parameterized quantum circuits in NISQ devices can suppress the increase in the number of executions. In this study, we propose a parallel-VQE, which leverages the parallel execution of parameterized quantum circuits to perform parallel parameter optimization in VQE, achieving convergence to solutions closer to the ground state. We validate the effectiveness of parallel-VQE in solving the random weighted max-cut problem using numerical simulations and a real quantum device. We present the results of running up to six circuits in parallel (120 qubits) and demonstrate the advantages of using multiple units to improve computational accuracy. This study provides a potential method for solving eigenvalue problems and combinatorial optimization problems for future quantum devices.

I. INTRODUCTION

Quantum computing has rapidly advanced in recent years.[1,2] Despite their infancy, rapid progress in quantum hardware and substantial global investment have led several experts to believe that noisy intermediate-scale quantum (NISQ) devices[3] will soon surpass classical computational capabilities. Current NISQ devices feature hundreds of qubits and are expected to expand to thousands or even tens of thousands of qubits. One of the promising hybrid classical–quantum algorithms that leverages this quantum advantage is the variational quantum algorithm (VQA),[4] notably the variational quantum eigensolver (VQE).[5–8] VQE is designed to determine a quantum state or a set of states that minimizes the energy of a given Hamiltonian, making it a prime candidate within NISQ algorithmic strategies. However, the optimization challenges inherent in the VQA framework are NP-hard,[9] highlighting the computational difficulties in finding optimal solutions. The effectiveness of VQE in identifying the ground state of a Hamiltonian depends on several factors, such as the initial state, *Ansatz*, initial parameters, and optimizer used. Several methodologies have been proposed to address these factors; however, selecting the initial parameters remains challenging. In addition, repeatedly running the VQE to explore potential initial parameters is resource-intensive. Inspired by the strategies in particle swarm optimization (PSO),[10]

parallelly processing multiple parameters to select those leading to lower-energy states is a potential solution. However, this approach raises concerns regarding the increased usage quotas and execution times of quantum processing units (QPUs) available through cloud services. To address this issue, we propose introducing parallel computing to execute multiple-parameterized quantum circuits simultaneously on a single QPU. Parallel computing presents a viable strategy to mitigate these challenges. Parallel quantum annealing has been reported to solve different problems simultaneously on a single QPU.[11,12] Similarly, PSO has evolved to improve optimizer accuracy through parallel parameter processing in classical computing. This study proposes a parallel-VQE, which implements parallel execution of multiple-parameterized quantum circuits and parallel parameter optimization to address the training parameter issues in VQAs.

Although several studies have addressed the theory and practice of parallel processing in NISQ devices,[13–17] the relationship between parallel processing and parameter optimization remains largely unexplored. Niu and Todri-Sanial proposed a multi-programming mechanism to optimize resource utilization in NISQ devices.[13,14] Their research focuses on enabling the parallel execution of multiple quantum circuits while addressing hardware constraints, such as topology, calibration data, and crosstalk effects. Although their work provides a valuable contribution to improving hardware-level resource management and execution efficiency, it does not directly address algorithmic challenges in quantum computation. In contrast, our research adopts a fundamentally different approach by focusing on enhancing the performance and practicality of quantum algorithms. By leveraging algorithmic advancements, we explore new opportunities to improve the scalability and efficiency of large-scale parallel executions, enabling innovative methods to fully utilize NISQ devices for solving complex computational problems. In comparison to the work of Mineh and Montanaro, which focused on the parallel execution of quantum circuits with identical parameters and the use of surrogate models to assist classical optimization,[15] our study introduces a distinct approach aimed at improving the efficiency and accuracy of the VQE. Specifically, we enable the parallel execution of quantum circuits with distinct parameter sets, facilitating a broader and more diverse exploration of the parameter space. This not only accelerates convergence toward solutions closer to the ground state but also mitigates the limitations of local minima, a common challenge in VQE optimization. Unlike the reliance of Mineh and Montanaroon surrogate models, our method directly utilizes the inherent parallelism of quantum hardware, eliminating dependence on classical computational resources and making it more aligned with the capabilities and constraints of current NISQ devices. Although Ohkura *et al.* focused on optimizing the parallel allocation of quantum circuits to minimize crosstalk and improve hardware utilization efficiency,[16] their research remains primarily centered on mitigating hardware-level interference. In contrast, our study directly addresses algorithmic challenges by leveraging quantum hardware's parallel capabilities to optimize the execution of VQE. Through the simultaneous execution of parameterized quantum circuits with distinct parameter sets, we improve the algorithmic efficiency and enhance solution quality, offering a complementary yet fundamentally different perspective from hardware-centric studies. Baker *et al.* introduced quantum multi-programming to the field of quantum–classical hybrid machine learning, specifically focusing on kernelized time-series classification tasks.[17] Their approach leverages quantum kernel computation to parallelize the evaluation of multiple kernel elements, achieving a reported speedup of over 35 times compared to classical methods. While their work demonstrates significant advancements in machine learning applications, our study adopts a broader approach aimed at tackling the inherent optimization challenges in VQE. By directly integrating the parallel execution capabilities of quantum hardware into the optimization process, we provide a novel pathway that extends beyond specific machine learning tasks to improve the efficiency and scalability of quantum algorithms. To investigate the computational accuracy of the parallel-VQE compared to the conventional VQE, we conducted benchmarks across 20 instances of the max-cut problem involving up to 120 qubits (20 qubits × 6 units).

The contributions of this study are threefold. First, we introduce the parallel-VQE approach, which improves computational efficiency by executing multiple-parameterized quantum circuits in parallel on a single QPU. Second, we comprehensively evaluate the convergence properties and computational accuracy of both single- and parallel-VQEs, utilizing simulators and real QPU implementations, thereby highlighting the advantages of parallel-VQE in mitigating local minima issues and enhancing solution quality. Third, we analyze the impact of hardware noise on computational accuracy, providing insights into the practical implementation of VQE methods on current quantum hardware. Our work is structured as follows: Sec. II analyzes the VQE and introduces the parallel-VQE approach, which enhances the computational efficiency by running multiple quantum circuits in parallel on a single QPU. The problem setup for evaluating parallel-VQE is also outlined. Section III compares the convergence properties and computational accuracy of single- and parallel-VQE, using both simulators and QPU implementations, and examines the effects of hardware noise. Section IV presents the conclusions and future directions.

II. EXPERIMENTAL DETAILS

A. Variational quantum eigensolver

VQE is a method for ground-state energy calculations that fundamentally utilizes quantum circuits based on the variational principle. Specifically, energy expectation derived from a parameterized trial wavefunction $\psi(\vec{\theta})$, governed by parameters $\vec{\theta}$, adheres to the following principle:

$$\left\langle \psi(\vec{\theta})|H|\psi(\vec{\theta})\right\rangle \geq E_0, \tag{1}$$

where H represents the given Hamiltonian and E_0 denotes the minimum eigenvalue. This inequality becomes equal only if the trial wavefunction is the exact eigenstate of the Hamiltonian. Consequently, the ground state energy and wavefunction can be determined by optimizing $\vec{\theta}$ to minimize the energy expectation value. In VQE, a trial wavefunction is generated using a quantum circuit called *Ansatz*, and the energy expectation value is calculated through quantum measurements. The measurement outcomes and parameters are provided to a classical optimizer, which updates the parameters to decrease the energy. This iterative procedure continues until the energy converges, yielding the ground state.

VQE faces significant challenges due to factors such as the Hamiltonian, *Ansatz* design, initial parameters, and optimization algorithm, which may result in the algorithm becoming trapped in local minima. Bittel and Kliesch demonstrated that the parameter optimization in VQAs is NP-Hard,[9] indicating that the difficulties in classical optimization are not merely incidental to the complexity of solving the ground-state problem. One approach to addressing this issue involves adopting a strategy similar to PSO, where multiple candidate solutions are used to achieve an optimal solution. However, when utilizing QPUs available on cloud platforms, the need to execute multiple-parameterized quantum circuits raises concerns regarding the increased access frequency and execution time owing to queue delays. To address this issue, we propose the parallel-VQE method that executes parallel optimization of multiple-parameterized quantum circuits concurrently.

B. Parallel-VQE

Figure 1 illustrates the proposed parallel-VQE system. To deal with an n-qubit Hamiltonian, we prepared parameterized quantum circuits for p units on n qubits. If $p = 1$, it is equivalent to the conventional VQE, hereafter referred to as a single-VQE. We assign different parameters, $\vec{\theta}_1$ through $\vec{\theta}_p$, to each unit. By implementing a quantum circuit with n qubits per unit, totaling np qubits, on a QPU and executing it for a predetermined number of shots, we can parallelize the execution of the *Ansatz* for p units. This allows for the simultaneous estimation of the expectation values for each unit with a single QPU invocation.

On a classical computer, each parameter $\vec{\theta}$ is independently updated based on the expectation values of the corresponding units. This approach facilitates parallel exploration of multiple candidate solutions in the same parameter space, leading to a solution closer to the ground state. After parameter optimization, the unit with the minimum expectation value is selected to obtain the solution closest to the ground state.

To simulate a quantum circuit, we used Qiskit.[18] The results were obtained from the quantum circuit simulation using the aer_simulator_matrix_product_state. All experiments were conducted in the 127-qubit IBM Eagle processor, "ibm_kawasaki."[19] The expectation values were calculated by sampling the quantum circuit with $n \times p$ qubits on both the simulator and ibm_kawasaki. Specifically, in the case of the simulator, we used the run method of the aer simulator, whereas for ibm_kawasaki, we employed the Sampler primitive in Qiskit.[18] In this setup, CVaR-VQE,[20] which enables efficient solutions for combinatorial optimization, is also applicable. In parallel-VQE, arbitrary quantum circuits can be configured. Furthermore, the QPU results in this study are raw outputs, with no error mitigation applied. In our *Ansatz*, we applied a single R_Y gate to each qubit. This approach avoids using two-qubit gates, such as CNOT gates, which are susceptible to higher error rates.[21] Additionally, Nannicini reported that when solving combinatorial optimization problems with VQE, quantum circuits employing two-qubit gates to generate entanglement did not show a clear advantage over circuits that generate only product states.[22] It is also worth noting that entanglement is not necessarily a crucial component when solving quadratic unconstrained binary optimization (QUBO) or combinatorial optimization problems using VQE, as well as when performing binary classification tasks in quantum machine learning.[23–26] Consequently, the trial state $|\psi(\vec{\theta})\rangle$ for each unit can be expressed as

$$|\psi(\vec{\theta})\rangle = \prod_{i=0}^{n-1} e^{-i\frac{\theta_i}{2}Y}|0\rangle^{\otimes n}, \tag{2}$$

where n is the number of qubits, θ_i is the rotation angle of the R_Y gate[18] applied to the ith qubit, and Y is the Pauli-Y matrix. The *Ansatz* includes the ground state for combinatorial optimization problems by generating any binary string based on the rotation angle with a single variational parameter per qubit.[22] According to Cerezo, quantum circuits are shown to be trainable when using a local cost function, provided the circuit depth D satisfies $D \in O(poly(n))$ or less. In our study, the quantum circuit employed has a depth of $D = 1$.[27] The initial circuit parameter values were randomly sampled from a uniform distribution $[-\pi, \pi)$. For each iteration k, the energy expectation value was estimated using 8192 shots. We used the Nakanishi–Fujii–Todo method (NFT)[28] with S (=100)

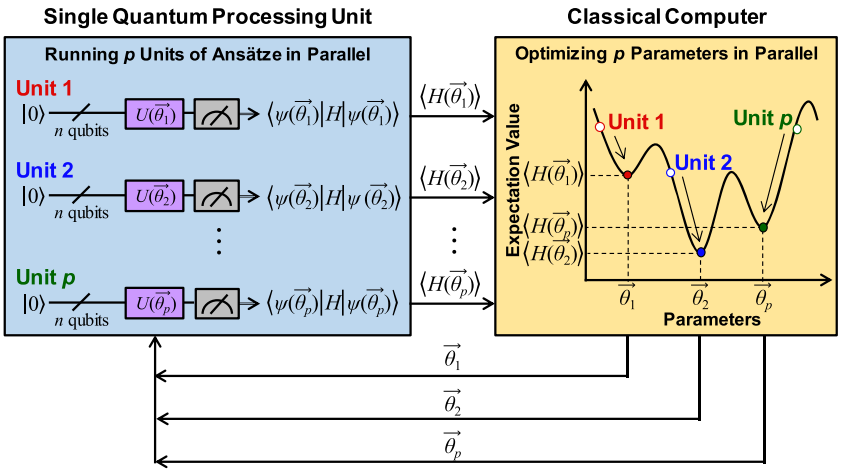

FIG. 1. Architecture of the parallel-VQE. In quantum computing, a quantum circuit of p units, each equipped with n qubits, is executed on the same QPU. The p units of quantum circuits are run on a single QPU. Based on the measured outcomes, the expectation values for each unit are estimated. On classical computers, the parameters corresponding to each unit and their expectation values are utilized to update the parameters concurrently. The updated parameters are applied to the quantum circuits corresponding to each unit.

TABLE I. Pseudocode of the NFT algorithm for parallel-VQE. p refers to the number of parallel units of parameterized quantum circuits. The modulo operation "%" yields the remainder of the division of the first number by the second. $\vec{e_j}$ is the unit vector along the θ_j axis.

1. *Require*: $\overrightarrow{\theta_1}, \overrightarrow{\theta_2}, \ldots, \overrightarrow{\theta_p}$ (Initial Parameter Set)
2. *Set*: $k = 0$ (Number of Iterations)
3. **While** $k \leq S$ **Do**: (Loop over S Iterations)
4. **If** k % 32 = 0
   Measure: $\langle\psi(\overrightarrow{\theta_1})|H|\psi(\overrightarrow{\theta_1})\rangle, \langle\psi(\overrightarrow{\theta_2})|H|\psi(\overrightarrow{\theta_2})\rangle, \ldots\langle\psi(\overrightarrow{\theta_p})|H|\psi(\overrightarrow{\theta_p})\rangle$
   (Reset Expectation Values Needed for Optimization in Line 7)
   Else
   $\langle\psi(\overrightarrow{\theta_1})|H|\psi(\overrightarrow{\theta_1})\rangle \leftarrow \min_{\theta_{1,j}} \langle\psi(\overrightarrow{\theta_1})|H|\psi(\overrightarrow{\theta_1})\rangle,$
   $\langle\psi(\overrightarrow{\theta_2})|H|\psi(\overrightarrow{\theta_2})\rangle \leftarrow \min_{\theta_{2,j}} \langle\psi(\overrightarrow{\theta_2})|H|\psi(\overrightarrow{\theta_2})\rangle, \ldots,$
   $\langle\psi(\overrightarrow{\theta_p})|H|\psi(\overrightarrow{\theta_p})\rangle \leftarrow \min_{\theta_{p,j}} \langle\psi(\overrightarrow{\theta_p})|H|\psi(\overrightarrow{\theta_p})\rangle$
   (Reuse Expectation Values Needed for Optimization in Line 7)
5. $\theta_{1,j} \in \overrightarrow{\theta_1}, \theta_{2,j} \in \overrightarrow{\theta_2}, \ldots, \theta_{p,j} \in \overrightarrow{\theta_p}$ (Choose Index j of Parameters)
6. *Measure*:
   $\left\langle\psi(\overrightarrow{\theta_1} \pm \frac{\pi}{2}\overrightarrow{e_j})\middle|H\middle|\psi(\overrightarrow{\theta_1} \pm \frac{\pi}{2}\overrightarrow{e_j})\right\rangle,$
   $\left\langle\psi(\overrightarrow{\theta_2} \pm \frac{\pi}{2}\overrightarrow{e_j})\middle|H\middle|\psi(\overrightarrow{\theta_2} \pm \frac{\pi}{2}\overrightarrow{e_j})\right\rangle, \ldots,$
   $\left\langle\psi(\overrightarrow{\theta_p} \pm \frac{\pi}{2}\overrightarrow{e_j})\middle|H\middle|\psi(\overrightarrow{\theta_p} \pm \frac{\pi}{2}\overrightarrow{e_j})\right\rangle$
   (Expectation Values Needed for Optimization in Line 7)
7. $\theta_{1,j} \leftarrow \arg\min_{\theta_{1,j}} \langle\psi(\overrightarrow{\theta_1})|H|\psi(\overrightarrow{\theta_1})\rangle,$
   $\theta_{2,j} \leftarrow \arg\min_{\theta_{2,j}} \langle\psi(\overrightarrow{\theta_2})|H|\psi(\overrightarrow{\theta_2})\rangle, \ldots,$
   $\theta_{p,j} \leftarrow \arg\min_{\theta_{p,j}} \langle\psi(\overrightarrow{\theta_p})|H|\psi(\overrightarrow{\theta_p})\rangle$ (Update Parameters)
   $k \leftarrow k + 1$ (Update Number of Iterations)
8. **End While**
9. **Return** $\overrightarrow{\theta_1}, \overrightarrow{\theta_2}, \ldots, \overrightarrow{\theta_p}$ (Resulting Parameters)

iterations because of its fast convergence and hyperparameter-free feature. NFT is a sequential optimization method that utilizes function fitting along the parameter axis using function fitting rather than gradient details. In accordance with Nakanishi *et al.*,[28] we re-estimated the energy expectation value once every 32 iterations using the NFT optimizer. In addition, we applied the NFT method to the parallel-VQE, as delineated in Table I. A quantum circuit was executed to estimate the expectation values for each unit parallelly, with each parameter $\vec{\theta}$ being independently updated for the corresponding unit. To assess the convergence of gradient-based optimizers in parallel-VQE, we employed the CoolMomentum[29] method in a subset of simulations. In our previous study, we found that while CoolMomentum requires a higher number of iterations to converge compared to NFT, adaptive moment estimation,[30] and simultaneous perturbation stochastic approximation,[31] it tends to yield solutions closer to the optimal.[32] For the hyperparameters in the CoolMomentum method, we set S = 200, an initial momentum coefficient ρ_0 = 0.999, and an initial learning rate η_0 = 0.02.

This study used the simplest method of sequentially assigning qubits, starting from q_0 to each unit. Figure 2 shows the assignment of qubits to each unit, with color coding representing the qubit map of the QPU employed in this study. In Fig. 2(a), for a 10-qubit Hamiltonian, q_0 to q_9 is assigned to unit 1, q_{10} to q_{19} to unit 2, and finally, q_{50} to q_{59} to unit 6. Similarly, for the 20-qubit Hamiltonian in Fig. 2(b), q_0 to q_{19} are assigned to unit 1, q_{20} to q_{39} to unit 2, and finally, q_{100} to q_{119} to unit 6. Figure 3 illustrates the readout error and single-qubit gate error rates for ibm_kawasaki, as measured on January 26, 2024. When transpiling the R_Y gate on ibm_kawasaki, it is implemented using a combination of SX and R_Z gates,[18] with the error rate of the SX gate designated as the single-qubit gate error due to the R_Z gate's zero error rate.[19] Microwave pulses drive rotations on the Bloch sphere in superconducting qubits, and by leveraging the phase of these drives, arbitrary zero-duration virtual Z gates can be implemented.[33,34] The x-axis, labeled "index of qubit", corresponds to the qubit numbering shown in the gate map of Fig. 2. The readout error rates ranged from 0.41% to 33%, while single-qubit gate error rates ranged from 0.011% to 0.45%. Although qubit error rates fluctuate daily, in this experiment, readout error is anticipated to be the primary contributor to computational accuracy degradation.

C. Problem setup

We numerically demonstrated the performance of the proposed method using the max-cut problem. The Hamiltonian H is given by

$$H = -\frac{1}{2}\sum_{(i,j)\in E} w_{ij}(I - Z_iZ_j), \tag{3}$$

where I is the identity matrix, Z_i denotes the Pauli Z operator on qubit i, and w_{ij} corresponds to integer weights selected uniformly at random in the interval [−10, 10]. For each problem size $q \in \{10, 20\}$, the experiment was repeated 20 times with different random seeds. Here, q corresponds to the number of nodes in the fully connected graph generated for the max-cut problem, which also matches the number of qubits n required for the solution.

When evaluating the performance of parallel-VQE on max-cut problems, the residual energy serves as a metric to quantify how well the trial state approximates the optimization problem's solution. The residual energy (r),[35,36] which measures the closeness of the final output state energy to that of the ground state, is calculated as follows:

$$r = \frac{\left\langle\psi(\vec{\theta_p})\middle|H\middle|\psi(\vec{\theta_p})\right\rangle - E_{\min}}{E_{\max} - E_{\min}}, \tag{4}$$

which $\left|\psi(\vec{\theta_p})\right\rangle$ represents the trial state of the p-th unit, $E_{\min}$ denotes the ground state energy of the problem, and $E_{\max}$ is the energy of the highest excited state. $E_{\max}$ and $E_{\min}$ for each instance were predetermined using a brute-force approach on classical computers. With the given normalization, r = 0 holds if and only if $\left|\psi(\vec{\theta_p})\right\rangle$ corresponds to the quantum state that solves the optimization problem.

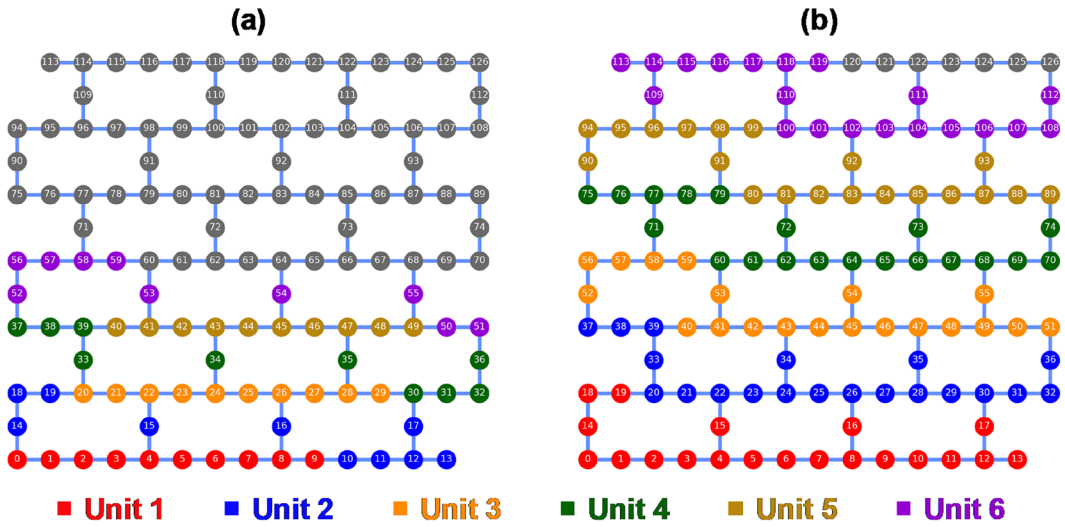

FIG. 2. Gate map of the ibm_kawasaki quantum processor visualized using Qiskit's qiskit.visualization.plot_gate_map function[18] and the qubit allocations used in the parallel-VQE experiments. Allocation of qubits for (a) 10-qubit and (b) 20-qubit Hamiltonian ground state computation using up to six units of quantum circuits. The color of each node represents the unit to which the qubit is assigned, with gray nodes indicating unused qubits. Edges represent the available two-qubit gates between the qubits. The Eagle quantum processor installed in the ibm_kawasaki system features 127 qubits and is characterized by a 6 × 3 heavy hexagonal lattice structure.

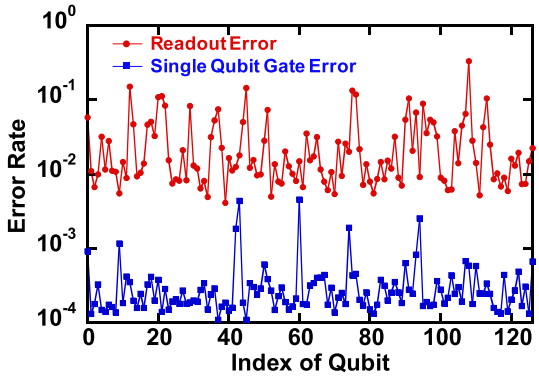

FIG. 3. Error rates for readout error and single-qubit gate error for the qubits on ibm_kawasaki, as measured on January 26, 2024.

For a general state, $r \in [0, 1]$ represents the relative approximation error.

III. RESULTS AND DISCUSSION

A. Benchmarks of QPU performance at initial parameter

First, we investigated how shot noise and qubit error rates on QPUs influence the error in expected values estimated from parallel executions of quantum circuits. Specifically, we measured expected values for initial parameters with varying numbers of parallel units, p, and compared the residual energy of unit 1 as a reference. The reference value was defined as the residual energy calculated from the ideal Hamiltonian expectation value obtained via the statevector estimator in qiskit.primitive for the single-VQE quantum circuit using unit 1's initial parameters.

Figure 4 presents the mean absolute error (MAE) of 20 instances with respect to the number of units p in the parallel-VQE setup. For each unit count, the MAE represents the absolute difference between the expectation value of unit 1 and the ideal single-VQE expectation value obtained using the same initial parameters. Figure 4(a) shows results from a 10-node max-cut problem using both a simulator and a QPU (ibm_kawasaki). For the simulator, the MAE remained within a range from 9.3×10^{-4} to 1.7×10^{-3} regardless of p, likely due to sampling error inherent in the sampling-based expectation value estimation method of the aer simulator. Indeed, the MAE for single-VQE, when measured with the same sampling-based method as parallel-VQE, was 1.2×10^{-3}. In contrast, the MAE for the QPU ranged from 1.2×10^{-2} to 1.3×10^{-2}, which we attribute to shot noise in addition to hardware-specific gate errors, readout errors, and crosstalk resulting from the use of adjacent qubits across parallel units. Notably, variations in parallelism did not impact the MAE. The MAE for single-VQE on the QPU was also 1.2×10^{-2}. Figure 4(b) shows the results for a 20-node max-cut problem using the same simulator and QPU. Here, the simulator's MAE remained between 7.4×10^{-4} and 1.1×10^{-3}, with the single-VQE measurement resulting in an MAE of 8.0×10^{-3}. For the QPU, the MAE ranged from 1.5×10^{-2} to 1.6×10^{-2}, and the single-VQE MAE was 1.6×10^{-2}. The characteristic MAE trend with respect to p

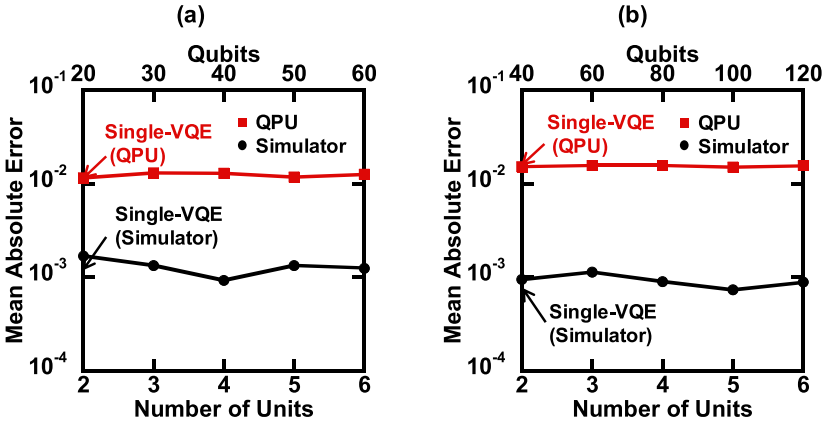

FIG. 4. Mean absolute error across 20 instances using (a) a 10-qubit Hamiltonian and (b) a 20-qubit Hamiltonian. The absolute error is defined as the absolute difference between the expectation value of parallel-VQE (unit 1) and that of single-VQE, calculated at the initial parameters used for single-VQE.

was consistent even as the number of nodes increased. These results suggest that for initial parameters likely to generate random superposition states, increasing parallelism does not affect the scale of the error in expected values.

B. Convergence properties of parallel-VQE

Next, we examine the optimization process of the conventional single-VQE and proposed parallel-VQE methods. Figures 5(a)–5(f) show the transition of the residual energy with respect to the number of iterations k when solving the max-cut problem with 10 nodes. Because this problem corresponds to searching for the ground state of a Hamiltonian with 10 qubits, we used 10 qubits per unit. We used a simulator to determine the quantum circuit output to verify computational performance in a noiseless environment. Figure 5(a) depicts the convergence of the residual energy in the single-VQE, whereas Figs. 5(b)–5(f) show the convergence in the parallel-VQE with varying numbers of units. Instances

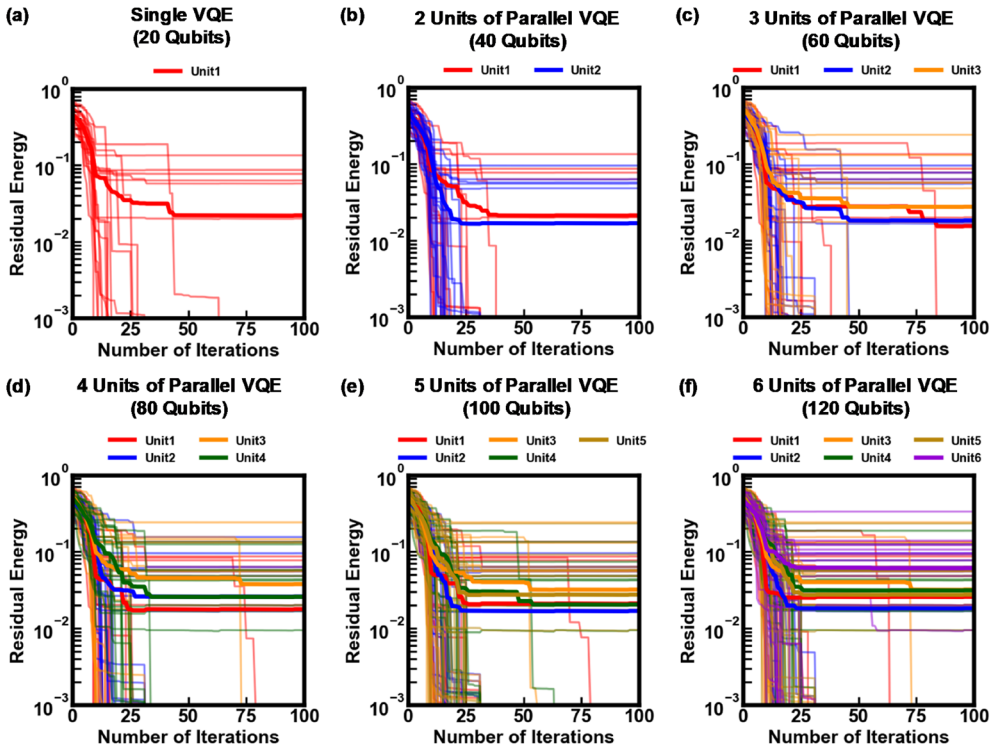

FIG. 5. Residual energy convergence of parallel- and single-VQE across different numbers of units as a function of iteration count using a simulator. Results when solving the max-cut problem for a 10-qubit system for (a) single-VQE and (b) 2-unit, (c) 3-unit, (d) 4-unit, (e) 5-unit, and (f) 6-unit parallel-VQE configurations. Different initial parameters are used for each unit. Thin lines represent individual experimental outcomes for each instance, whereas thick lines denote the mean performance across 20 instances.

where r reached below 10^{-3} before reaching the maximum number of iterations (100) indicate that the ground state has been obtained. The results show a decrease in residual energy with parameter parallel optimization for all parallel-VQE implementations. The number of instances in which at least one unit achieved a residual energy below 10^{-3} was 14 for the single-VQE, 18 for the 2-unit parallel-VQE, and 19 for the parallel-VQE implementations with 3–6 units. Because different initial parameters were used for each unit, the distribution of instances reaching the ground state vs those falling into the local minima varied. In addition, changes in the number of units lead to slower convergence characteristics and increasing instances of converging to different solutions. This may be attributed to shot noise affecting the expectation values, owing to the limited number of samples.

Figure 6 illustrates the residual energy distribution for the max-cut problem with 10 nodes solved using the single- and parallel-VQE methods. For single- and parallel-VQE, the residual energy is concentrated around $r \approx 0.5$ before optimization ($k = 0$) and around $r = 0$ after optimization ($k = 100$), indicating no significant differences in computational characteristics between different units. However, parallel-VQE benefits from the availability of multiple units, allowing for the selection of a unit that provides a lower residual energy for each instance.

Figure 7 shows the progression of residual energy r with respect to the number of iterations k for solving the max-cut problem with 20 nodes. This problem corresponds to searching for the ground state of a 20-qubit Hamiltonian using 20 qubits per unit. Simulations were conducted using a simulator to assess performance in the absence of noise. Figure 7(a) illustrates the convergence of the residual energy for a single-VQE, whereas Figs. 7(b)–7(f) show the convergence for a parallel-VQE with varying numbers of units. Instances where the residual energy r was below 10^{-3} before reaching the maximum iteration count of 100 indicated successful retrieval of the ground state. Compared to Fig. 5, there is an

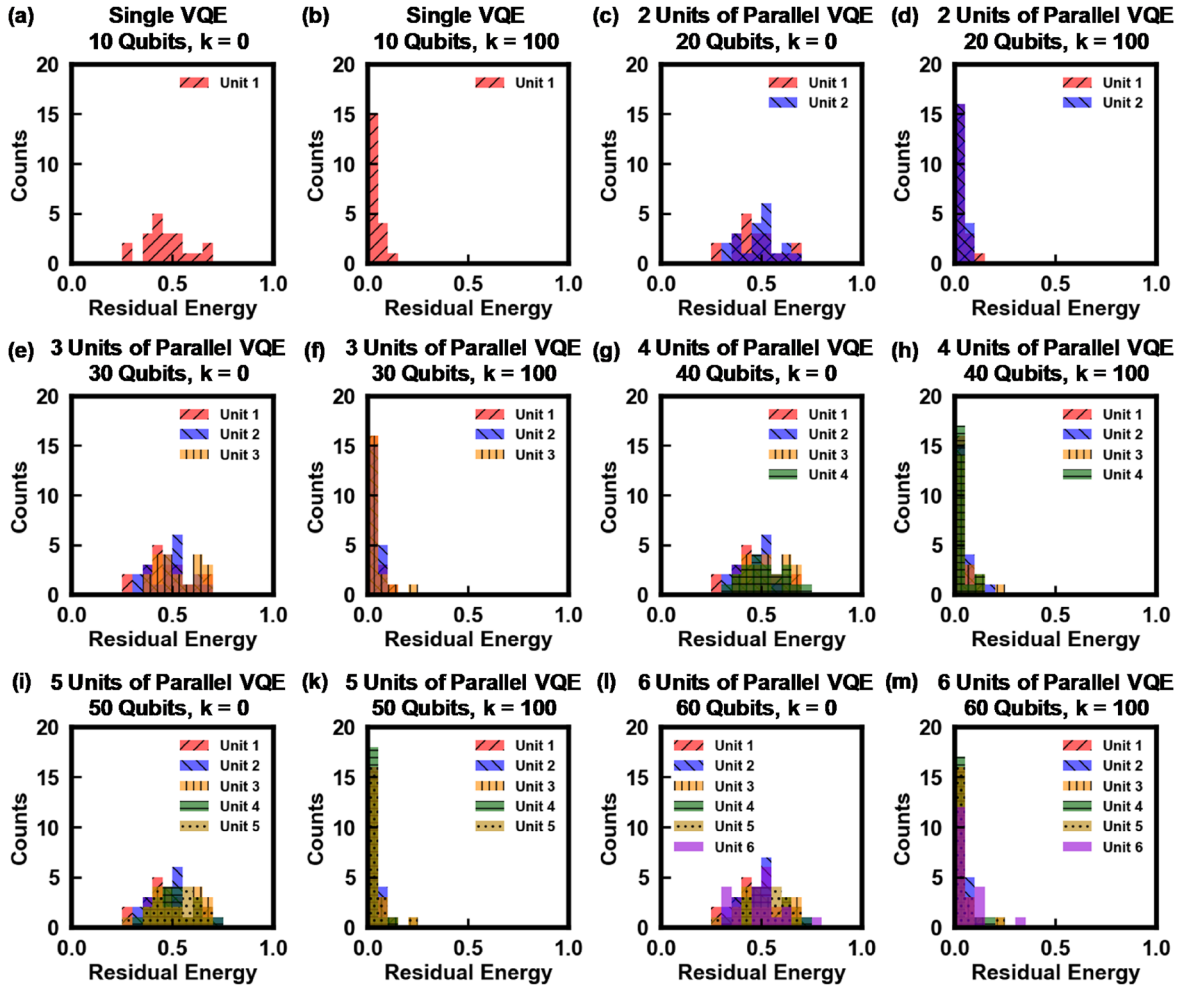

FIG. 6. Distribution of residual energy before and after optimization using (a) and (b) single- and (c)–(m) parallel-VQE across 20 instances of a 10-node max-cut problem solved using a simulator. The horizontal axis represents residual energy, and the vertical axis represents the count.

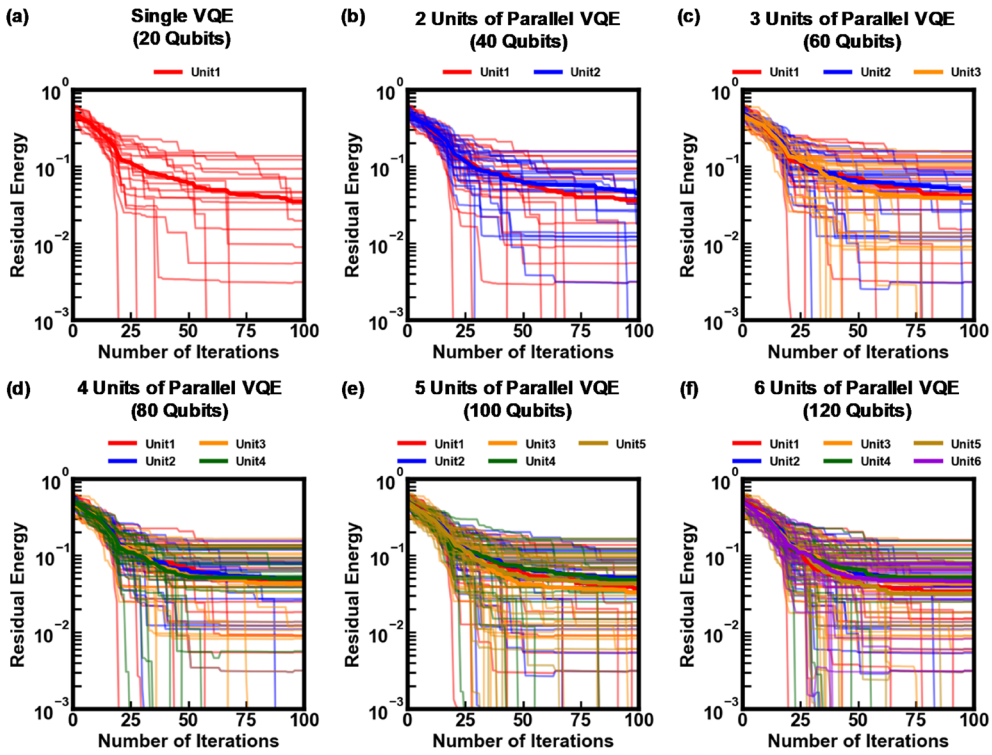

FIG. 7. Residual energy convergence of parallel- and single-VQE across different numbers of units as a function of iteration count using a simulator. Results for (a) single-VQE and (b) 2-unit, (c) 3-unit, (d) 4-unit, (e) 5-unit, and (f) 6-unit parallel-VQE configurations, solving the max-cut problem for a 20-qubit system. Different initial parameters are used for each unit. Thin lines represent individual experimental outcomes for each instance, whereas thick lines denote the mean performance across 20 instances.

increasing trend in instances falling into local minima, likely owing to the exponential increase in local minima with the number of nodes. The number of instances in which at least one unit achieved a residual energy below 10^{-3} was as follows: 5 for the single-VQE, 9 for the 2-unit parallel-VQE, 11 for the 3-unit parallel-VQE, 12 for the 4-unit parallel-VQE, 14 for the 5-unit parallel-VQE, and 13 for the 6-unit parallel-VQE. These results confirm that the reduction in residual energy owing to the parallel optimization of parameters persists even with increased nodes.

Figures 8(a), 8(b), and 8(c)–(m) depict the distribution of residual energy for solving the 20-node max-cut problem with a single-VQE and parallel-VQE, respectively, at $k = 0$ (before optimization) and $k = 100$ (after optimization). Similar to the results in Fig. 6, single- and parallel-VQE show a concentration of residual energy around $r \approx 0.5$ before optimization and around $r = 0$ after optimization. No significant differences were observed in computational characteristics between different units. However, an increased number of nodes was associated with a higher proportion of instances falling into local minima, leading to a higher residual energy after optimization.

Figure 9 illustrates the progression of residual energy r with respect to the number of iterations k for solving the max-cut problem with 10 nodes. This problem corresponds to the search for the ground state of a 10-qubit Hamiltonian utilizing 10 qubits per unit. The experiment was conducted using a 127-qubit QPU (ibm_kawasaki). Figure 7(a) shows the convergence of the residual energy for a single-VQE, whereas Figs. 9(b)–9(f) depict the convergence for a parallel-VQE with varying numbers of units. The increase in the residual energy at $k = 32$ and $k = 64$ is due to the NFT method performing measurements of the expectation values every 32 iterations of the current parameters. These results confirm the reduction in the residual energy owing to the parallel optimization of the parameters, even in a noisy environment. However, compared with Fig. 5, the converged residual energies for each instance tend to be more than 10 times worse. This degradation is attributed to noise, which causes high-energy solutions to be output alongside optimal or local solutions. Parallel-VQE, which can utilize multiple units, allows for selecting a unit with lower residual energy for each instance, thus avoiding high-energy solutions.

Figures 10(a), 10(b), and 10(c)–(m) present the distribution of residual energy for solving the 10-node max-cut problem using single-VQE and parallel-VQE, respectively, at $k = 0$ (before optimization) and $k = 100$ (after optimization). Compared to Fig. 6, both VQEs show a concentration of residual energy around $r \approx 0.5$ before optimization and around $r = 0.1$ after optimization. This indicates a trend of deterioration in the residual energy when using actual QPUs. The observed increase in the residual energy after optimization is likely due to noise affecting the output, resulting in high-energy optimal or local solutions.

Figure 11 shows the progression of residual energy r with respect to the number of iterations k for solving the max-cut problem with 20 nodes. This problem corresponds to the search for the ground state of a 20-qubit Hamiltonian utilizing 20 qubits per unit. The experiment was conducted using a 127-qubit QPU

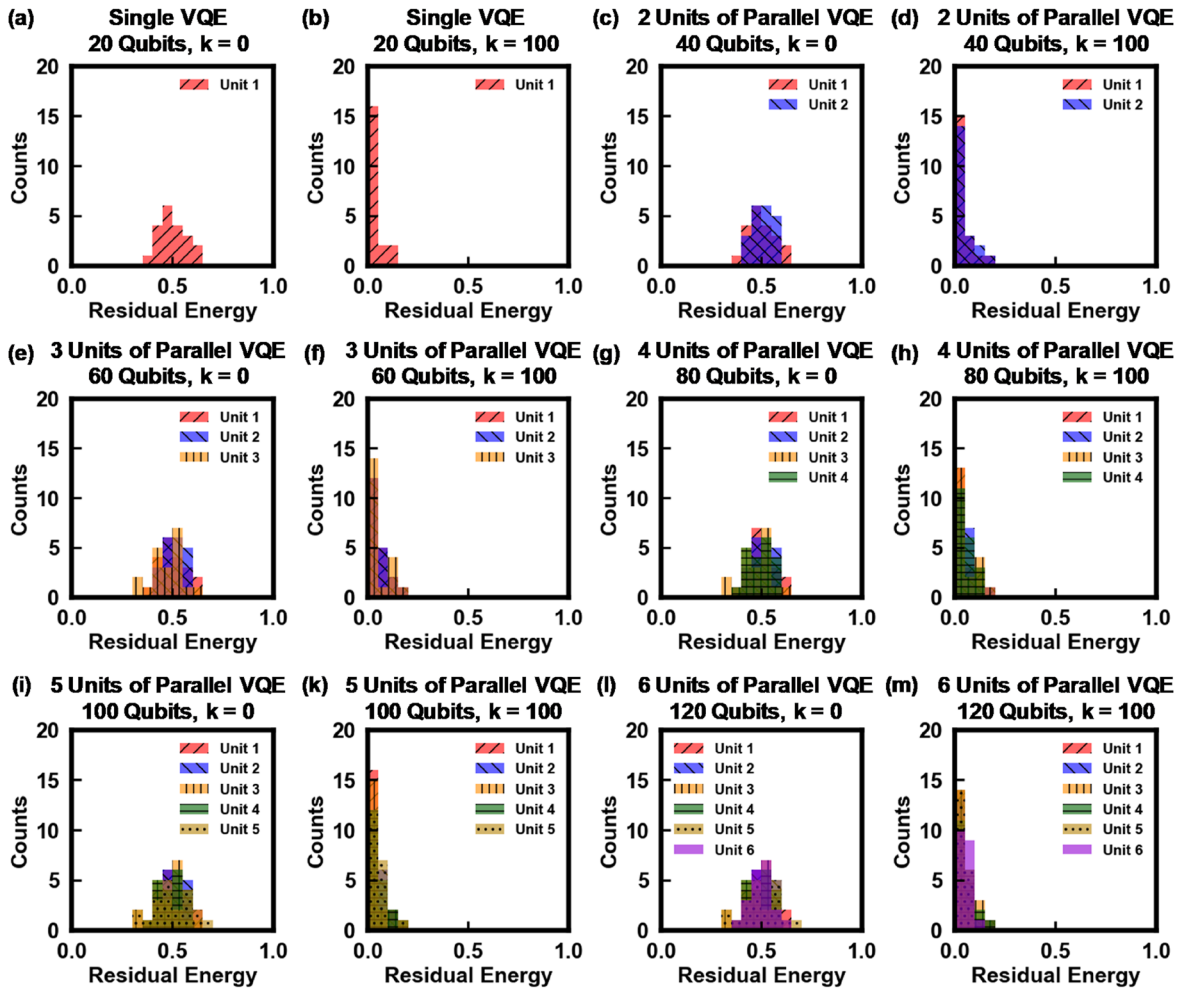

FIG. 8. Distribution of residual energy before and after optimization using (a) and (b) single-VQE, and (c)–(m) parallel-VQE across 20 instances of a 20-node max-cut problem solved using a simulator. The horizontal axis represents the residual energy, and the vertical axis represents the count.

(ibm_kawasaki). Figure 11(a) illustrates the convergence of the residual energy for a single-VQE, whereas Figs. 11(b)–11(f) depict the convergence for a parallel-VQE with varying numbers of units. These results confirm the reduction in the residual energy owing to the parallel optimization of the parameters, even in a noisy environment. Compared to Fig. 9, there is a trend of worsening residual energy for each instance. This deterioration is attributed to increased noise effects and more instances of falling into the local minima. Despite this, parallel-VQE allows for selecting units with lower residual energy for each instance, helping to avoid high-energy solutions. However, because ibm_kawasaki has 127 qubits, it is not possible to implement more than seven units.

Figures 12(a), 12(b), and 12(c)–(m) present the distribution of residual energy for solving the 20-node max-cut problem using single- and parallel-VQEs, respectively, at $k = 0$ (before optimization) and $k = 100$ (after optimization). Similar to Fig. 10, both VQEs show residual energy concentrated around $r \approx 0.5$ before optimization and around $r = 0.1$ after optimization. This indicates that although there are different convergence processes for instances sharing the same initial parameter values, parallel-VQE tends to converge to optimal or local solutions, similar to single-VQE.

Compared to the traditional single-VQE method, the parallel-VQE approach maintains nearly equivalent convergence properties. By utilizing multiple units, the parallel-VQE method enables more efficient parameter space exploration, thereby reducing the likelihood of becoming trapped in local optima. This improvement is particularly pronounced as the number of qubits and problem size increase. Our findings suggest that parallel-VQE offers significant advantages for larger and more complex problems. However, the parallel-VQE method also has inherent limitations. The maximum number of units that can be utilized is constrained by the number of qubits available on the QPU and

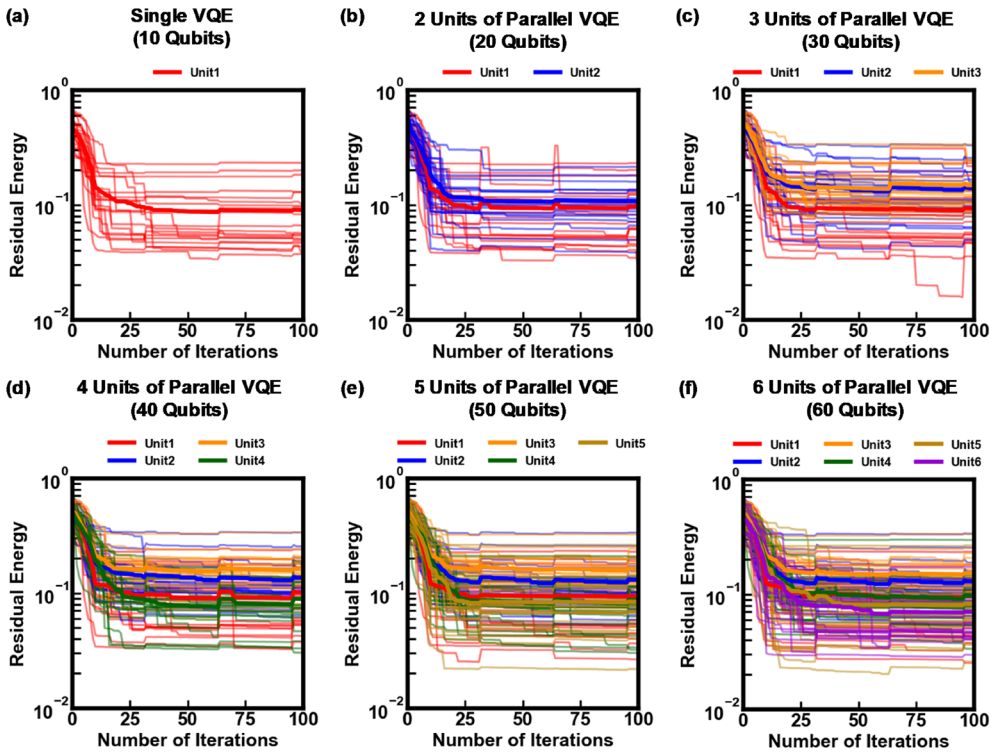

FIG. 9. Residual energy convergence of parallel- and single-VQE across different numbers of units as a function of iteration count using ibm_kawasaki. Results for (a) single-VQE and (b) 2-unit, (c) 3-unit, (d) 4-unit, (e) 5-unit, and (f) 6-unit parallel-VQE configurations, solving the max-cut problem for a 10-qubit system. Different initial parameters are used for each unit. Thin lines represent individual experimental outcomes for each instance, whereas thick lines denote the mean performance across 20 instances.

the qubits required to represent the Hamiltonian. As the problem size increases, the efficiency of parallel-VQE may be impacted by increased noise and the presence of local optima. These factors should be considered when interpreting results and planning future experiments.

Figure 13 demonstrates the residual energy convergence process during optimization using the CoolMomentum method on a simulator, comparing single-VQE and parallel-VQE for different qubit configurations. Figures 13(a) and 13(b) show the progression of residual energy with respect to the iteration number k when solving a 10-node max-cut problem, where each unit utilizes 10 qubits. Figure 13(a) represents the convergence for single-VQE, while Fig. 13(b) illustrates the case of parallel-VQE with two units. Figures 13(c) and 13(d) extend the analysis to a 20-node max-cut problem for single-VQE and parallel-VQE with two units, respectively.

Instances achieving residual energy r below 10^{-3} before reaching the maximum number of iterations (200) are considered to have reached the ground state. A characteristic feature of the CoolMomentum method, as observed in Fig. 13(a), is its convergence process: during the initial phase of optimization, large momentum coefficients and learning rates facilitate exploration of the solution space, which are gradually reduced as the optimization progress, allowing convergence to a solution. This dynamic adjustment leads to a tendency to achieve solutions closer to the global minimum compared to the NFT method, albeit with a higher number of iterations required for convergence. In practice, the number of instances in which at least one unit achieved a residual energy below 10^{-3} was 15 for the single-VQE and 20 for the 2-unit parallel-VQE.

This trend is also evident in the implementation of parallel-VQE with two units, as depicted in Fig. 13(b), as well as in larger problem instances such as the 20-node max-cut problem shown in Figs. 13(c) and 13(d), where the use of parameter parallelization effectively reduces residual energy. Additionally, the number of instances in which at least one unit achieved a residual energy below 10^{-3} was 8 for the single-VQE and 9 for the 2-unit parallel-VQE. Similar to the NFT method, an increase in the number of units leads to a greater number of instances converging to distinct solutions. This behavior is attributed to the impact of shot noise on expectation values, which arises due to limitations in the number of samples.

Furthermore, the results suggest that the use of gradient-based optimizers is feasible, achieving convergence characteristics comparable to those of single-VQE. A trade-off between the number of iterations required for convergence and the residual energy achieved is observed when comparing CoolMomentum with the NFT method, further indicating the effectiveness of the CoolMomentum approach for obtaining solutions close to the global minimum.

C. Evaluation of computational accuracy in parallel-VQE

Here, we present the results obtained after the optimization using a parallel-VQE. Figure 14 illustrates the relationship between the residual energy and the number of units in parallel-VQE,

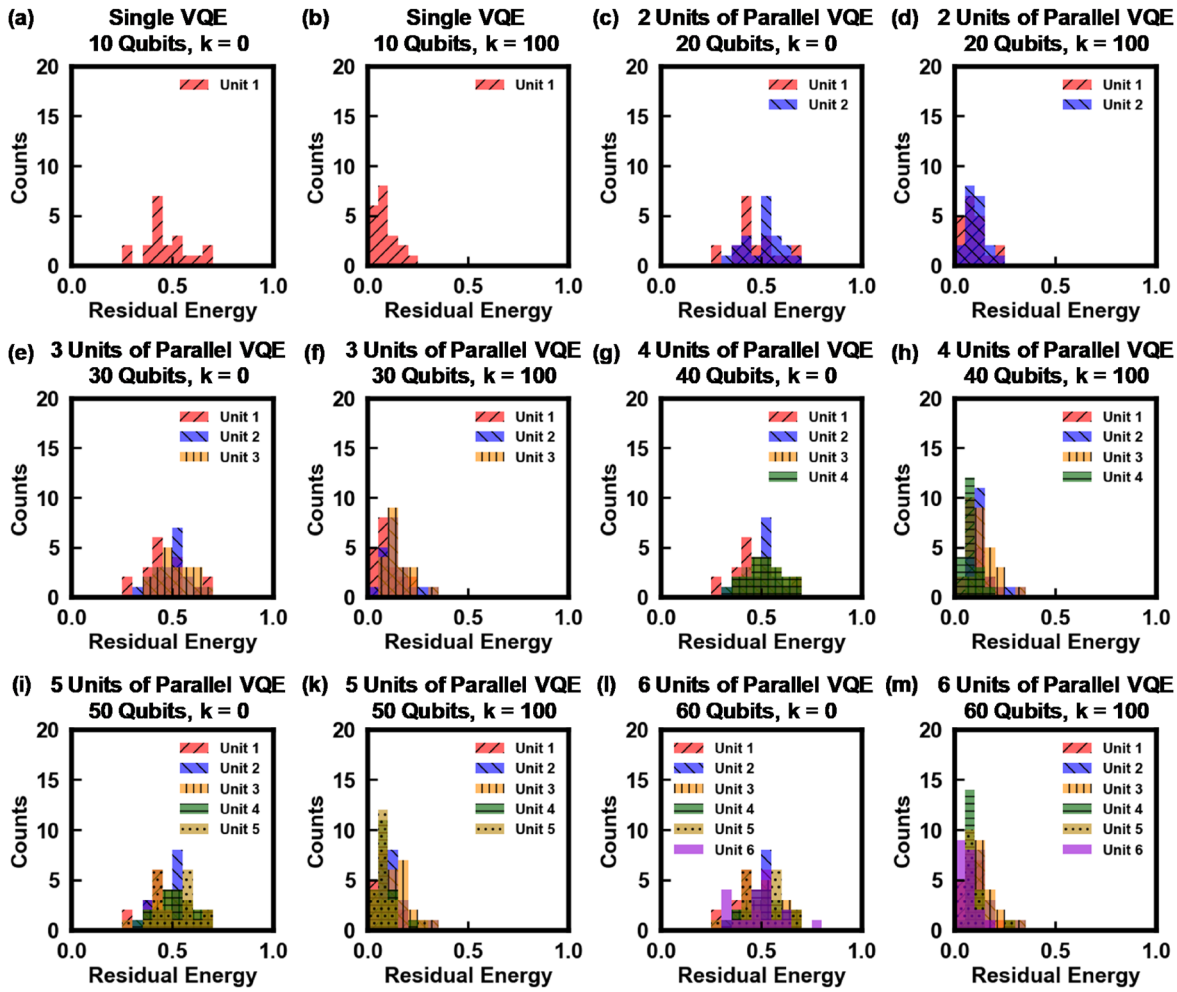

FIG. 10. Distribution of residual energy before and after optimization using (a) and (b) single-VQE and (c)–(m) parallel-VQE across 20 instances of a 10-node max-cut problem solved using ibm_kawasaki. The horizontal axis represents residual energy, and the vertical axis represents the count.

highlighting that combinations yielding lower expectation values are selected for each instance. For details on the selected units, refer to Tables II–V. Figures 14(a) and 14(b) show the results for the max-cut problem with 10 and 20 nodes (10-qubit and 20-qubit Hamiltonians, respectively) using the simulator, whereas Figs. 14(c) and 14(d) display the results obtained using the QPU (ibm_kawasaki). Results from single-VQE are indicated by arrows, with residual energy values of 2.19×10^{-2} (10 nodes) and 3.49×10^{-2} (20 nodes) for the simulator and 8.95×10^{-2} (10 nodes) and 1.33×10^{-1} (20 nodes) for ibm_kawasaki.

Figures 14(a) and 14(b) show the degree to which the residual energy reflects the local minima because they do not include hardware noise from the QPU. When comparing the 20-qubit Hamiltonian to the 10-qubit Hamiltonian, both VQEs showed increased residual energy. This trend highlights the characteristic of the VQE to fall more readily into local minima as the computational basis increases exponentially with the number of nodes. Comparing single- and parallel-VQE showed that the residual energy improved by a factor of 9.1 and 6.1 for the 10- and 20-qubit cases when using a 6-unit parallel-VQE configuration, respectively. By contrast, Figs. 14(c) and 14(d) include hardware noise from the QPU. Consequently, the residual energy serves as an indicator of how much the solutions have fallen into local minima and as a measure of how much the solution quality has deteriorated owing to noise. When noise is present in the measurement results, the residual energy increases by approximately three to four times compared with that of the simulator results. This makes it more challenging to observe the trend of increasing residual energy with a 20-qubit Hamiltonian compared with a 10-qubit Hamiltonian in single- and parallel-VQE. Nonetheless, the residual energy improved by a factor of 1.9 and 2.0 for the 10- and 20-qubit cases, respectively, when using a 6-unit parallel-VQE configuration. These findings suggest

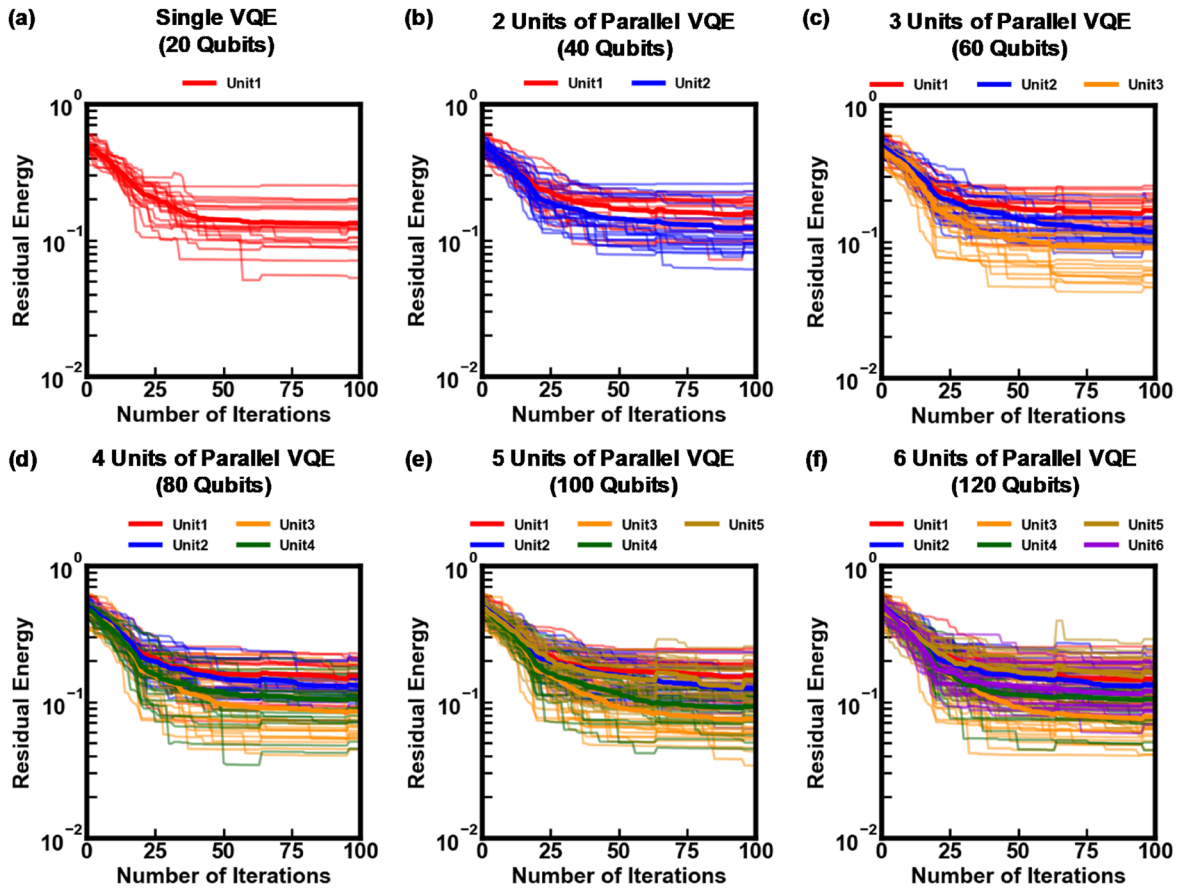

FIG. 11. Residual energy convergence of parallel- and single-VQEs across different numbers of units as a function of iteration count using ibm_kawasaki. Results for (a) single-VQE, (b) 2-unit parallel-VQE, and (c) 3-unit parallel-VQE, whereas the bottom row shows results for (d) 4-unit, (e) 5-unit, and (f) 6-unit parallel-VQE configurations, solving the max-cut problem for a 20-qubit system. Different initial parameters are used for each unit. Thin lines represent individual experimental outcomes for each instance, whereas thick lines denote the mean performance across 20 instances.

that the parallel-VQE can improve the residual energy in the simulator and QPU implementations. Additionally, in the simulator, an increase in the number of units reduces the residual energy, improving the solution quality. This improvement can be attributed to the availability of multiple initial parameters, which increases the likelihood of exploring solutions closer to the ground state. A similar trend is observed for the QPU. However, the overall performance of qubits plays a significant role. For instance, higher-performance qubits can yield results similar to those observed with the 10-qubit with 2 units, whereas qubits with a relatively lower performance may produce outcomes similar to those observed with the 20-qubit with 6 units. Therefore, it may be necessary to set the appropriate number of units based on the performance characteristics of the qubits.

Then, we examined the accuracy of convergence to the ground state. Figure 15 illustrates the relationship between the ground state probability (GSP) and the number of units in the parallel-VQE. As shown in Fig. 14, the combinations yielding lower expectation values were preferentially selected for each instance (refer to Tables II–V for details on the selected units). The GSP was calculated by determining the proportion of instances that achieved the ground state out of the 8192 shots used for expectation value estimation and averaging this across 20 instances. Figures 15(a) and 15(b) show the results for the max-cut problem with 10 and 20 nodes, respectively, using the simulator, whereas Figs. 15(c) and 15(d) display the corresponding results obtained using QPU (ibm_kawasaki). The results obtained with single-VQE are indicated by arrows, with a GSP of 0.700 (10 nodes) and 0.250 (20 nodes) for the simulator and 4.01×10^{-3} (10 nodes) and 3.21×10^{-2} (20 nodes) for ibm_kawasaki. Figures 15(a) and 15(b) show GSP without the inclusion of hardware noise from the QPU, indicating the rate of successful ground-state discovery. When using a 20-qubit Hamiltonian, a reduced GSP was observed for single- and parallel-VQE compared with a 10-qubit Hamiltonian. Moreover, parallel-VQE using two units resulted in a 20% improvement in the GSP for the 10- and 20-qubit Hamiltonians than the single-VQE. For the 10-qubit Hamiltonian, a maximum

TABLE II. Units selected to determine the solution of the max-cut problem with 10 nodes across 20 instances using parallel-VQE on the aer_simulator_matrix_product_state.

No. of instances	Backend	No. of nodes	Selected unit (2 units)	Selected unit (3 units)	Selected unit (4 units)	Selected unit (5 units)	Selected unit (6 units)
1	Simulator	10	2	2	2	3	3
2	Simulator	10	2	3	3	2	4
3	Simulator	10	1	1	1	2	1
4	Simulator	10	1	2	2	1	5
5	Simulator	10	1	1	4	5	6
6	Simulator	10	1	2	1	4	1
7	Simulator	10	2	3	2	2	5
8	Simulator	10	2	3	3	2	2
9	Simulator	10	2	3	3	5	4
10	Simulator	10	2	1	2	1	5
11	Simulator	10	1	1	4	2	1
12	Simulator	10	1	3	4	3	1
13	Simulator	10	1	3	4	5	1
14	Simulator	10	2	3	3	2	5
15	Simulator	10	1	1	1	1	1
16	Simulator	10	1	1	1	3	1
17	Simulator	10	2	3	4	3	3
18	Simulator	10	2	3	2	3	3
19	Simulator	10	1	1	1	5	1
20	Simulator	10	2	1	4	1	3

TABLE III. Units selected to determine the solution of the max-cut problem with 20 nodes across 20 instances using parallel-VQE on the aer_simulator_matrix_product_state.

No. of instances	Backend	No. of nodes	Selected unit (2 units)	Selected unit (3 units)	Selected unit (4 units)	Selected unit (5 units)	Selected unit (6 units)
1	Simulator	20	2	3	3	2	2
2	Simulator	20	1	3	1	1	1
3	Simulator	20	2	3	3	3	3
4	Simulator	20	1	1	4	5	4
5	Simulator	20	2	2	3	3	2
6	Simulator	20	1	3	3	5	5
7	Simulator	20	1	1	1	4	1
8	Simulator	20	2	3	4	4	4
9	Simulator	20	1	1	1	5	5
10	Simulator	20	2	2	4	3	4
11	Simulator	20	1	1	3	3	5
12	Simulator	20	1	1	1	4	4
13	Simulator	20	1	2	2	2	2
14	Simulator	20	1	1	3	3	3
15	Simulator	20	2	3	3	3	3
16	Simulator	20	2	2	4	2	4
17	Simulator	20	1	3	4	1	1
18	Simulator	20	2	3	3	5	5
19	Simulator	20	1	3	3	3	2
20	Simulator	20	1	3	1	5	5

TABLE IV. Units selected to determine the solution of the max-cut problem with 10 nodes across 20 instances using parallel-VQE on the ibm_kawasaki.

No. of instances	Backend	No. of nodes	Selected unit (2 units)	Selected unit (3 units)	Selected unit (4 units)	Selected unit (5 units)	Selected unit (6 units)
1	Kawasaki	10	2	3	4	5	1
2	Kawasaki	10	1	1	1	1	6
3	Kawasaki	10	2	1	4	4	5
4	Kawasaki	10	2	1	1	5	5
5	Kawasaki	10	1	1	1	1	6
6	Kawasaki	10	2	1	4	5	6
7	Kawasaki	10	1	1	1	5	6
8	Kawasaki	10	1	1	4	4	6
9	Kawasaki	10	1	1	4	5	5
10	Kawasaki	10	1	1	4	4	5
11	Kawasaki	10	2	2	4	2	5
12	Kawasaki	10	1	1	1	1	6
13	Kawasaki	10	1	1	1	1	1
14	Kawasaki	10	1	1	1	1	6
15	Kawasaki	10	2	1	1	1	1
16	Kawasaki	10	1	1	4	4	6
17	Kawasaki	10	2	3	4	4	3
18	Kawasaki	10	2	2	3	4	5
19	Kawasaki	10	2	2	4	5	5
20	Kawasaki	10	1	2	1	1	6

TABLE V. Units selected to determine the solution of the max-cut problem with 20 nodes across 20 instances using parallel-VQE on the ibm_kawasaki.

No. of instances	Backend	No. of nodes	Selected unit (2 units)	Selected unit (3 units)	Selected unit (4 units)	Selected unit (5 units)	Selected unit (6 units)
1	Kawasaki	20	2	3	3	3	3
2	Kawasaki	20	1	3	3	3	3
3	Kawasaki	20	1	3	3	3	3
4	Kawasaki	20	2	3	4	4	4
5	Kawasaki	20	2	3	3	3	3
6	Kawasaki	20	1	3	3	3	3
7	Kawasaki	20	2	3	3	4	3
8	Kawasaki	20	2	3	4	4	4
9	Kawasaki	20	1	2	4	4	6
10	Kawasaki	20	2	3	4	3	3
11	Kawasaki	20	2	3	3	3	3
12	Kawasaki	20	1	3	4	5	3
13	Kawasaki	20	2	2	4	3	4
14	Kawasaki	20	2	2	3	3	3
15	Kawasaki	20	2	3	3	3	6
16	Kawasaki	20	2	3	3	3	3
17	Kawasaki	20	2	2	3	4	3
18	Kawasaki	20	1	3	3	3	3
19	Kawasaki	20	1	3	3	3	4
20	Kawasaki	20	1	3	3	3	3

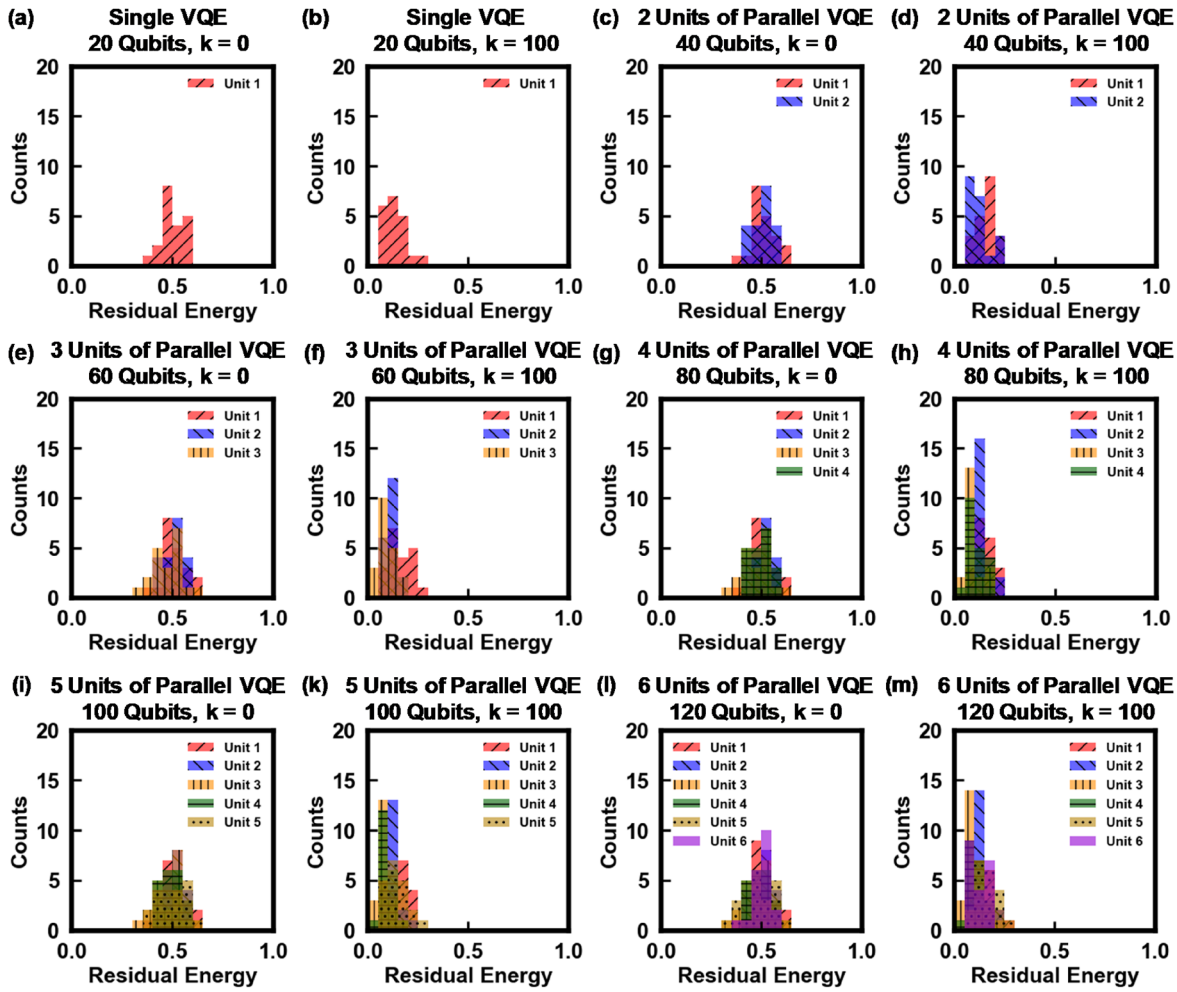

FIG. 12. Distribution of residual energy before and after optimization using (a) and (b) single-VQE, and (c)–(m) parallel-VQE across 20 instances of a 20-node max-cut problem solved using ibm_kawasaki. The horizontal axis represents the residual energy, and the vertical axis represents the count.

GSP of 0.95 was achieved with three or more units. In the 20-qubit Hamiltonian, a maximum GSP of 0.70 was observed with five units. The decrease in GSP with six units may be attributed to the delayed convergence caused by the increased number of qubits. However, Figs. 15(c) and 15(d) include hardware noise from QPU. Thus, GSP indicates the extent to which the quality of the ground-state solutions deteriorates because of noise. Comparing single- and parallel-VQEs, the latter using two units showed an improvement of 10% and 3.6% in the GSP for the 10- and 20-qubit cases, respectively. These results suggest that parallel-VQE can improve the GSP in the simulator and QPU implementations. Additionally, an increased number of units corresponded to an increase in the GSP, indicating improved solution quality. This improvement was likely due to the higher probability of exploring solutions closer to the ground state, facilitated by the multiple initial parameters. A similar trend was observed for the QPU although the overall performance of the qubits played a crucial role. For instance, high-performance qubits may yield results similar to those of 10-qubit with two units. Conversely, when lower-performing qubits are included, the results may resemble those of the 20-qubit with six units. Therefore, it may be necessary to set the appropriate number of units based on the performance characteristics of the qubits used.

Compared to the traditional single-VQE method, the parallel-VQE approach demonstrates improved accuracy, particularly in the calculation of residual energy and ground state probabilities. Results from parallel-VQE indicate a significant performance enhancement as the number of units increases. This improvement is reflected in both simulator and QPU implementations, highlighting the effectiveness of parallel-VQE in enhancing the quality of solutions for large-scale problems. However, a primary limitation of parallel-VQE is its dependence on the number of qubits available on the QPU and the qubits required to represent the Hamiltonian. Additionally, similar to single-VQE, the

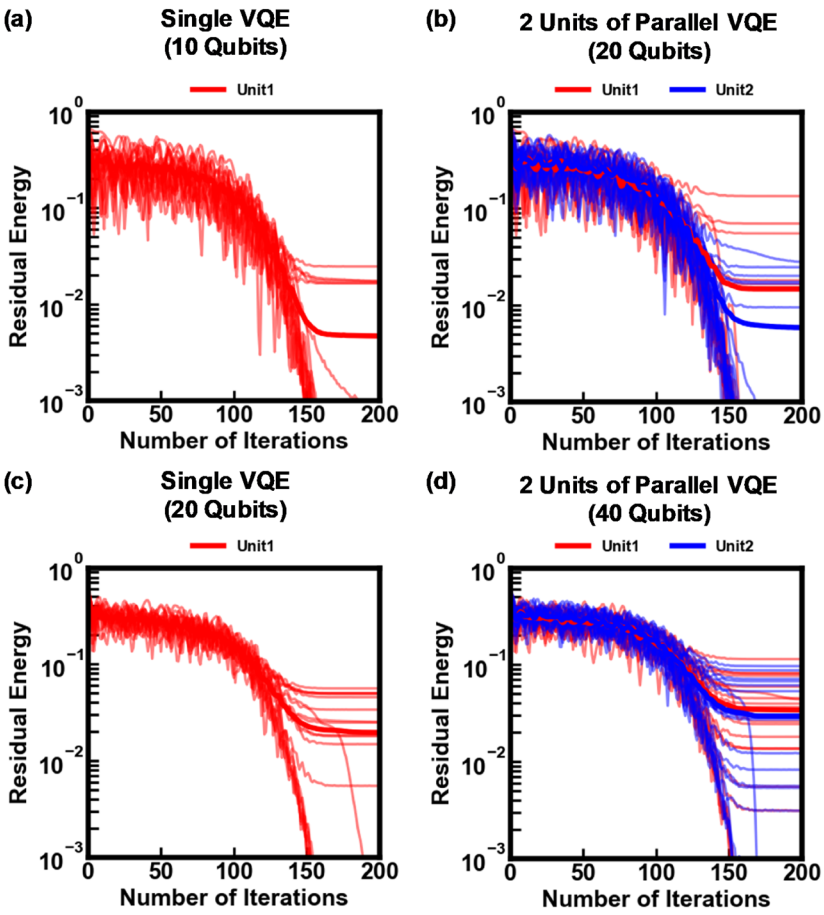

FIG. 13. Residual energy convergence of parallel-VQE and single-VQE with varying numbers of units, using the CoolMomentum optimizer on a simulator, is shown as a function of iteration count. Panels (a) and (b) present results for single-VQE and 2-unit parallel-VQE, respectively, for solving the max-cut problem on a 10-qubit system. Panels (c) and (d) show results for single-VQE and 2-unit parallel-VQE, respectively, on a 20-qubit system for the same problem. Different initial parameters are applied to each unit. Thin lines represent individual experimental outcomes for each instance, while thick lines indicate the mean performance across 20 instances.

performance of parallel-VQE is affected by inherent noise in QPU measurements and increased computational complexity due to large Hamiltonians. These limitations should be considered when evaluating the effectiveness of the parallel-VQE and designing future research.

D. Effect of error mitigation

Finally, we investigated the effect of the mthree (M3) error mitigation technique[37] when applied to parallel VQE. In this experiment, we used a simulator to optimize parameters under the same experimental conditions as in the previous study, employing the NFT method with the maximum number of iterations set to 200 to improve convergence. After applying the optimized parameters to the parameterized quantum circuit, the generate_preset_pass_manager from qiskit.transpiler.preset_passmanagers was utilized to transpile the quantum circuit at an optimization level of 3.[18] As a result, quantum bits were sequentially assigned to unit 1, unit 2, . . ., unit 6, starting from those with the lowest error rates. The transpiled circuit was executed on the ibm_kawasaki quantum processor, and raw data (without M3 error mitigation) were collected. The raw data were then processed using the mthree package[38] for error mitigation with the M3 method, and the results were compared across three conditions: the simulator, raw data (without M3) from ibm_kawasaki, and the error-mitigated results.

Figure 16(a) shows the average residual energy for 20 instances of a 10-node max-cut problem across the three methods. Compared with the results in Fig. 14(a), the residual energy from the simulator is nearly identical, suggesting that the NFT method converges to a local or global optimum within 100 iterations. When using raw data from ibm_kawasaki, the residual energy is lower than that in Fig. 14(c), highlighting the importance of leveraging low-error-rate qubits via circuit transpilation. While this result reflects a comparison of residual energy under simulator-optimized parameters, rather than parameters directly optimized on ibm_kawasaki, it underscores the potential for improving outcomes by targeting low-error-rate qubits. However, no significant improvement in residual energy was observed for cases with more than two units, likely due to the higher error rates of additional qubits as parallelism increases. Assigning lower-expectation-value units to lower-error-rate qubits may further enhance performance on NISQ devices.

Applying M3 error mitigation yielded residual energy values close to those obtained from the simulator in cases with a small number of units, suggesting that error mitigation techniques can help recover accuracy in parallel VQE. Nevertheless, the use of quasiprobabilities in M3 occasionally resulted in negative residual

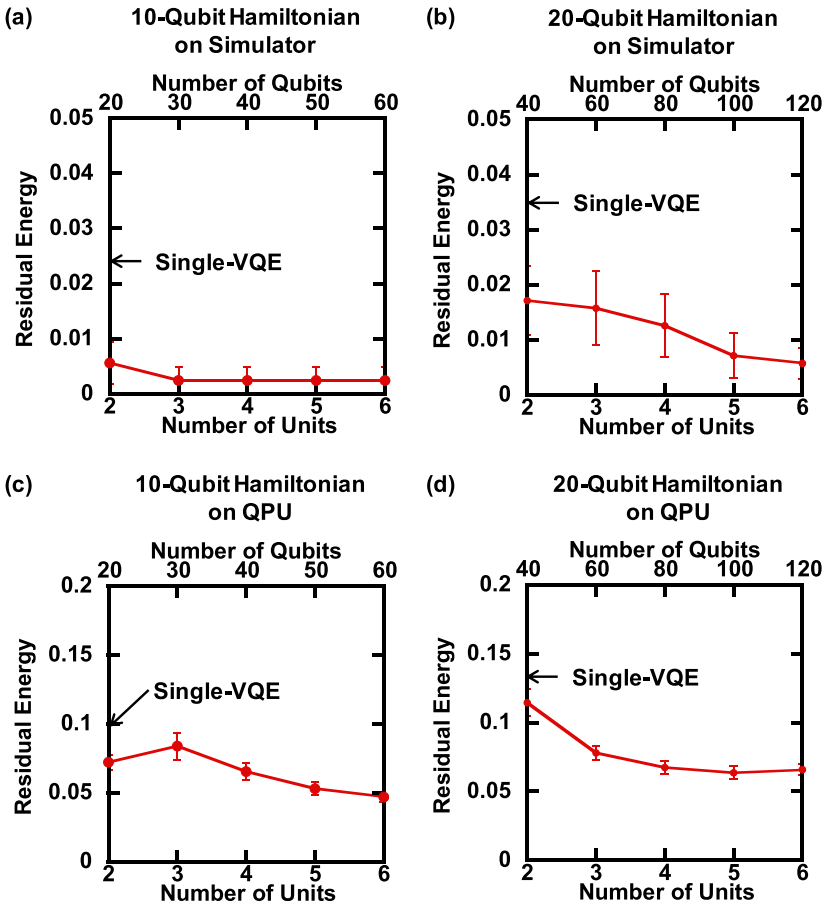

FIG. 14. Average residual energy of selected units post-parallel-VQE optimization across different qubit counts. Solutions for (a) 10-node and (b) 20-node max-cut problems over 20 instances on a simulator. Corresponding solutions were obtained using ibm_kawasaki for (c) 10-node and (d) 20-node. Error bars indicate standard errors across instances.

energy values as the expectation values exceeded the ground state energy. Furthermore, the residual energy increased with the number of units, implying that the error mitigation effect of M3 diminishes as the number of qubits increases.

Figure 16(b) illustrates the average residual energy for 20 instances of a 20-node max-cut problem across the three methods. Compared with Fig. 14(b), the simulator results show a lower overall residual energy, suggesting that the NFT method may not have fully converged to a local or global optimum within 100 iterations for larger instances. As with the 10-node case, raw data from ibm_kawasaki yielded lower residual energy compared to Fig. 14(d), despite no observable improvement in residual energy for cases involving more than two units. M3 error mitigation produced residual energies close to those of the simulator for cases with up to two units, confirming its utility for systems involving ~40 qubits or fewer. However, as the number of units or qubits increased, the results converged toward those of the raw data from ibm_kawasaki, suggesting a diminishing error mitigation effect of M3 with increasing qubit counts.

IV. CONCLUSION AND OUTLOOK

This study proposed a parallel-VQE method to explore solutions closer to the ground state and compared its computational accuracy with that of the conventional single-VQE method for solving the max-cut problem. In experiments conducted using identical initial parameters, it was confirmed that the absolute error between the residual energy and the ideal expected value remained on the order of 10^{-3} for the simulator and 10^{-2} for the QPU, regardless of the number of units used. Then, we investigated the parallel optimization characteristics of the parameters in the parallel-VQE. When using simulators, parallel-VQE outperformed single-VQE in reducing the residual energy. By leveraging multiple units, the parallel-VQE can explore different parameter settings and select the best result for each problem. However, when using actual quantum hardware (ibm_kawasaki), the single- and parallel-VQE exhibited higher residual energies due to noise. This noise makes it more challenging to obtain precise results than when using simulators. For the 10-node and 20-node max-cut problems, the residual energy was approximately $r \approx 0.5$ before optimization and decreased to around $r = 0$ or $r = 0.1$ after optimization. However, as the problem size and noise levels increased, more instances fell into the local minima, resulting in a higher residual energy. Nonetheless, the ability of the parallel-VQE to use multiple units aids in finding better solutions, even under noisy conditions, by selecting units with lower residual energy. In this study, in addition to NFT, the gradient-based global optimization method CoolMomentum was applied to parallel-VQE, demonstrating that the fundamental convergence

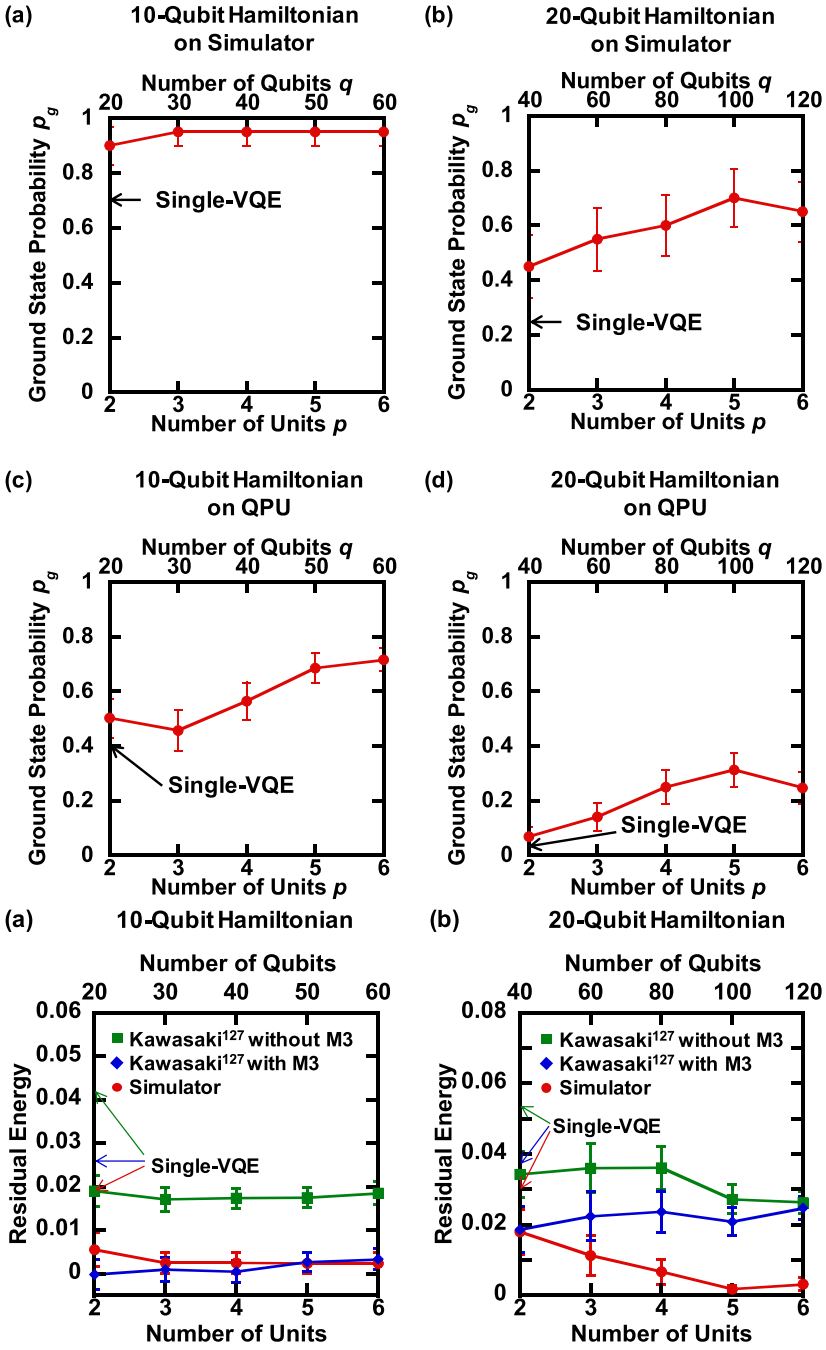

FIG. 15. Ground state probability of selected units post-parallel-VQE optimization across different qubit counts. Solutions for (a) 10-node and (b) 20-node max-cut problems over 20 instances on a simulator. Corresponding solutions were obtained using ibm_kawasaki for (c) 10-node and (d) 20-node. Error bars indicate standard errors across instances.

FIG. 16. Average residual energy of each unit post-parallel-VQE optimization with the NFT method (using $S = 200$) is shown for different qubit counts on a simulator. Results on ibm_kawasaki, with and without error mitigation, were obtained by applying the optimal parameters determined from the simulator. Panels (a) and (b) present solutions for the 10-node and 20-node max-cut problems, respectively, across 20 instances on the simulator. Error bars indicate the standard errors across instances.

characteristics remained consistent and suggesting the applicability of various optimizers.

We explored the optimization outcomes of the proposed parallel-VQE method and compared them with those of the conventional single-VQE approach. Our parallel-VQE demonstrated a significant reduction in residual energy by factors of 9.1 and 6.1 for 10-qubit and 20-qubit Hamiltonians, respectively, in a noise-free simulator environment. When considering the presence of hardware noise in QPU experiments, the method showed a substantial enhancement, improving the residual energy by factors of 1.9 and 2.0 for 10-qubit and 20-qubit Hamiltonians, respectively. Moreover, the GSP analysis revealed that parallel-VQE enhances the likelihood of successfully finding the ground state, particularly as the number of units increases. For instance, using a 6-unit configuration, we observed GSP improvements of up to 20% in the simulator and enhancements of 10% and 3.6% for 10- and

20-qubit Hamiltonians on the QPU, respectively. The observed trends suggest that using parallel-VQE enables enhanced solution space exploration, increasing the probability of converging to the ground state. Furthermore, it was suggested that constructing units from low-error-rate qubits and incorporating error mitigation techniques, such as M3, could achieve accuracy closer to that of the simulator.

In conclusion, parallel-VQE shows great promise for enhancing the efficiency and accuracy of VQAs. Unlike previous parallel execution approaches for quantum circuits, this study focuses on parallel computations with different initial values, demonstrating an increased probability of reaching the true optimal solution not only in simulations but also in experiments conducted on actual quantum computers. This is a notable advantage of the proposed method as it is expected to efficiently explore the energy landscape of interest. Moreover, the sequential application of this method may allow for a gradual narrowing of the search space, further enhancing its potential for solving complex optimization problems. The findings of this study have significant implications for the development and practical implementation of VQAs. In particular, by demonstrating the effectiveness of the parallel-VQE approach in improving the computational efficiency and accuracy in noisy quantum environments, this work suggests that parallel optimization strategies could play a crucial role in overcoming the limitations of current quantum hardware in VQAs. Additionally, the insights gained from analyzing the effects of hardware noise provide valuable guidance for optimizing the implementation of parallel-VQE, paving the way for more reliable and scalable quantum computations in the near future. It offers a robust approach for mitigating local minima and managing the impact of hardware noise, advancing quantum computing methodologies. Further research may focus on optimizing the unit configurations and exploring the scalability of the method for larger qubit systems. In addition, integrating classical PSO concepts and sharing optimization information between different units holds promise for improving computational accuracy while reducing the execution count on QPUs.[39,40] Typically, *Ansätze*, such as hardware-efficient[7] and entanglement-variational hardware-efficient *Ansätze* ,[41] used in VQE involve two-qubit gates to induce quantum entanglement. However, it is essential to consider quantum circuits that account for crosstalk and the connectivity constraints of two-qubit gates on QPUs, particularly when using identical circuits or when the gate connectivity differs. Thus, employing parallel-VQE might yield higher-quality solutions more efficiently. Although the execution count of the quantum computer remains equivalent to that of the traditional VQE, our research focuses on the relationship between execution time and unit count when operating parallel-VQE, along with the performance in terms of solution acquisition time. Finally, a thorough investigation of the scalability with respect to the number of qubits and the applicability of similar ideas to other VQAs, such as the quantum approximate optimization algorithm,[42] is essential.

AUTHOR DECLARATIONS

Conflict of Interest

The authors have no conflicts to disclose.

Author Contributions

This work is the result of teamwork and discussions among all authors. All the authors have read and agreed to publish the manuscript.

Daisuke Tsukayama: Conceptualization (lead); Data curation (lead); Formal analysis (lead); Investigation (lead); Methodology (lead); Visualization (lead); Writing – original draft (lead). **Jun-ichi Shirakashi**: Funding acquisition (lead); Supervision (equal); Writing – review & editing (equal). **Tetsuo Shibuya**: Supervision (equal); Writing – review & editing (equal). **Hiroshi Imai**: Supervision (equal); Writing – review & editing (equal).

DATA AVAILABILITY

The data that support the findings of this study are available from the corresponding author upon reasonable request.

REFERENCES

[1] Y. Kim, A. Eddins, S. Anand, K. X. Wei, E. Van Den Berg, S. Rosenblatt, H. Nayfeh, Y. Wu, M. Zaletel, K. Temme, and A. Kandala, "Evidence for the utility of quantum computing before fault tolerance," Nature **618**, 500 (2023).
[2] E. Pelofske, A. Bärtschi, and S. Eidenbenz, "Short-depth QAOA circuits and quantum annealing on higher-order Ising models," npj Quantum Inf. **10**, 30 (2024).
[3] J. Preskill, "Quantum computing in the NISQ era and beyond," Quantum **2**, 79 (2018).
[4] M. Cerezo, A. Arrasmith, R. Babbush, S. C. Benjamin, S. Endo, K. Fujii, J. R. McClean, K. Mitarai, X. Yuan, L. Cincio, and P. J. Coles, "Variational quantum algorithms," Nat. Rev. Phys. **3**, 625 (2021).
[5] 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**, 4213 (2014).
[6] J. R. McClean, J. Romero, R. Babbush, and A. Aspuru-Guzik, "The theory of variational hybrid quantum-classical algorithms," New J. Phys. **18**, 023023 (2016).
[7] 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).
[8] N. Moll, P. Barkoutsos, L. S. Bishop, J. M. Chow, A. Cross, D. J. Egger, S. Filipp, A. Fuhrer, J. M. Gambetta, M. Ganzhorn, A. Kandala, A. Mezzacapo, P. Müller, W. Riess, G. Salis, J. Smolin, I. Tavernelli, and K. Temme, "Quantum optimization using variational algorithms on near-term quantum devices," Quantum Sci. Technol. **3**, 030503 (2018).
[9] L. Bittel and M. Kliesch, "Training variational quantum algorithms is NP-hard," Phys. Rev. Lett. **127**, 120502 (2021).
[10] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of ICNN'95 - International Conference on Neural Networks* (IEEE, 1995), Vol. 4, p. 1942.
[11] E. Pelofske, G. Hahn, and H. N. Djidjev, "Parallel quantum annealing," Sci. Rep. **12**, 4499 (2022).
[12] J. Artag, M. Shimada, and J. Shirakashi, "Parallel quantum annealing: A novel approach to solving multiple NP-hard problems concurrently," in *2023 IEEE International Conference on Quantum Computing and Engineering (QCE)* (IEEE, 2023), p. 328.
[13] S. Niu and A. Todri-Sanial, "How parallel circuit execution can be useful for NISQ computing?," arXiv:2112.00387v1 (2021).
[14] S. Niu and A. Todri-Sanial, "Enabling multi-programming mechanism for quantum computing in the NISQ era," Quantum **7**, 925 (2023).
[15] L. Mineh and A. Montanaro, "Accelerating the variational quantum eigensolver using parallelism," Quantum Sci. Technol. **8**, 035012 (2023).

[16]Y. Ohkura, T. Satoh, and R. V. Meter, "Simultaneous execution of quantum circuits on current and near-future NISQ systems," IEEE Trans. Quantum Eng. **3**, 2500210 (2022).
[17]J. S. Baker, G. Park, K. Yu, A. Ghukasyan, O. Goktas, and S. K. Radha, "Parallel hybrid quantum-classical machine learning for kernelized time-series classification," Quantum Mach. Intell. **6**, 18 (2024).
[18]G. Aleksandrowicz *et al.* (2019). "Qiskit: Open-source framework for quantum computing," Zenodo. https://doi.org/10.5281/zenodo.2562111.
[19]See https://quantum-computing.ibm.com/ for IBM Quantum.
[20]P. K. Barkoutsos, G. Nannicini, A. Robert, I. Tavernelli, and S. Woerner, "Improving variational quantum optimization using CVaR," Quantum **4**, 256 (2020).
[21]T. Miki, R. Okita, M. Shimada, D. Tsukayama, and J. Shirakashi, "Variational ansatz preparation to avoid CNOT-gates on noisy quantum devices for combinatorial optimizations," AIP Adv. **12**, 035247 (2022).
[22]G. Nannicini, "Performance of hybrid quantum-classical variational heuristics for combinatorial optimization," Phys. Rev. E **99**, 013304 (2019).
[23]P. Díez-Valle, D. Porras, and J. J. García-Ripoll, "Quantum variational optimization: The role of entanglement and problem hardness," Phys. Rev. A **104**, 062426 (2021).
[24]Y. Chai, L. Funcke, T. Hartung, K. Jansen, S. Kühn, P. Stornati, and T. Stollenwerk, "Optimal flight-gate assignment on a digital quantum computer," Phys. Rev. Appl. **20**, 064025 (2023).
[25]J. Bowles, S. Ahmed, and M. Schuld, "Better than classical? The subtle art of benchmarking quantum machine learning models," arXiv:2403.07059v2 (2024).
[26]T. Rohe, D. Schuman, J. Nüßlein, L. Sünkel, J. Stein, and C. Linnhoff-Popien, "The questionable influence of entanglement in quantum optimisation algorithms," arXiv:2407.17204v1 (2024).
[27]M. Cerezo, A. Sone, T. Volkoff, L. Cincio, and P. J. Coles, "Cost function dependent barren plateaus in shallow parametrized quantum circuits," Nat. Commun. **12**, 1791 (2021).
[28]K. M. Nakanishi, K. Fujii, and S. Todo, "Sequential minimal optimization for quantum-classical hybrid algorithms," Phys. Rev. Res. **2**, 043158 (2020).
[29]O. Borysenko and M. Byshkin, "CoolMomentum: A method for stochastic optimization by Langevin dynamics with simulated annealing," Sci. Rep. **11**, 10705 (2021).
[30]D. P. Kingma and J. L. Ba, "Adam: A method for stochastic optimization," arXiv:1412.6980v9 (2014).
[31]J. C. Spall, "Multivariate stochastic approximation using a simultaneous perturbation gradient approximation," IEEE Trans. Autom. Control **37**, 332 (1992).
[32]D. Tsukayama, J. Shirakashi, and H. Imai, "CoolMomentum mitigating local minima in variational quantum eigensolvers," Jpn. J. Appl. Phys. **62**, 088003 (2023).
[33]D. C. McKay, C. J. Wood, S. Sheldon, J. M. Chow, and J. M. Gambetta, "Efficient Z gates for quantum computing," Phys. Rev. A **96**, 022330 (2017).
[34]P. Krantz, M. Kjaergaard, F. Yan, T. P. Orlando, S. Gustavsson, and W. D. Oliver, "A quantum engineer's guide to superconducting qubits," Appl. Phys. Rev. **6**, 021318 (2019).
[35]G. B. Mbeng, R. Fazio, and G. E. Santoro, "Quantum annealing: A journey through digitalization, control, and hybrid quantum variational schemes," arXiv:1906.08948v3 (2019).
[36]M. M. Wauters, G. B. Mbeng, and G. E. Santoro, "Polynomial scaling of QAOA for ground-state preparation of the fully-connected p-spin ferromagnet," arXiv:2003.07419v2 (2020).
[37]P. D. Nation, H. Kang, N. Sundaresan, and J. M. Gambetta, "Scalable mitigation of measurement errors on quantum computers," PRX Quantum **2**, 040326 (2021).
[38]mthree (2.8.1) user guide available on-line at https://qiskit.github.io/qiskit-addon-mthree/index.html.
[39]C. N. Self, K. E. Khosla, A. W. R. Smith, F. Sauvage, P. D. Haynes, J. Knolle, F. Mintert, and M. S. Kim, "Variational quantum algorithm with information sharing," npj Quantum Inf. **7**, 116 (2021).
[40]P. Díez-Valle, J. Luis-Hita, S. Hernández-Santana, F. Martínez-García, A. Díaz-Fernández, E. Andrés, J. José García-Ripoll, E. Sánchez-Martínez, and D. Porras, "Multiobjective variational quantum optimization for constrained problems: An application to cash handling," Quantum Sci. Technol. **8**, 045009 (2023).
[41]X. Wang, B. Qi, Y. Wang, and D. Dong, "Entanglement-variational hardware-efficient ansatz for eigensolvers," Phys. Rev. Appl. **21**, 034059 (2024).
[42]E. Farhi, J. Goldstone, and S. Gutmann, "A quantum approximate optimization algorithm," arXiv:1411.4028v1 (2014).