

Problem-specific Parameterized Quantum Circuits of the VQE Algorithm for Optimization Problems

Atsushi Matsuo^{1,3} Yudai Suzuki² Shigeru Yamashita³

概要 : The Variational Quantum Eigensolver (VQE) algorithm is attracting much attention to utilize current limited quantum devices. The VQE algorithm requires a quantum circuit with parameters, called a parameterized quantum circuit (PQC), to prepare a quantum state, and the quantum state is used to calculate the expectation value of a given Hamiltonian. Creating sophisticated PQCs is important from the perspective of the convergence speed. Thus, we propose problem-specific PQCs of the VQE algorithm for optimization problems. Our idea is to dynamically create a PQC that reflects the constraints of an optimization problem. With a problem-specific PQC, it is possible to reduce a search space by restricting unitary transformations in favor of the VQE algorithm. As a result, we can speed up the convergence of the VQE algorithm. Experimental results show that the convergence speed of the proposed PQCs is significantly faster than that of the state-of-the-art PQC.

1. Introduction

Many companies have been competing to develop quantum computers recently. Quantum computing promises advantages in solving certain tasks, e.g., integer factorization [1] and database search [2]. However, the number of errors in current quantum devices cannot be ignored, and they do not yet have the capability of the error correction. Thus, they have the limitation of the size of quantum circuits that can be executed [3]. Due to this limitation, we cannot yet execute quantum circuits for such complicated tasks.

The Variational Quantum Eigensolver (VQE) algorithm was proposed to utilize such limited quantum devices and it has been studied intensively [4–8]. The VQE algorithm is an algorithm to find the minimal eigenvalue and its eigenvector of a given Hamiltonian. It consists of two parts. One is executed on quantum computers, and the other on classical computers. The part executed on quantum computers has a shallow quantum circuit with parameters called a *parameterized quantum circuit (PQC)*. A PQC prepare a quantum state from an initial state, and it can also prepare various quantum states by changing the parameters. With the created quantum state, the expectation value of a given Hamiltonian is calculated by sampling outcomes. Since the VQE algorithm uses the variational method based on the results of sampling, making sophisticated PQCs is important from the perspective of the convergence speed.

The VQE algorithm can also be used to solve optimization problems by creating the corresponding Hamiltonian for an optimization problem [9, 10]. Formulations of the Hamiltonian for many NP-complete and NP-hard problems have been discussed in [11]. A converged expectation value corresponds to an answer to the optimization problem. Also, a quantum state for the converged expect-

tation value corresponds to an assignment of variables for the optimization problem.

Although the VQE algorithm is being studied intensively and PQCs of the VQE algorithm is important, there are a few researches considering PQCs of the VQE algorithm for the optimization problems. Hence, we would like to point out two problems in known PQCs. (1) Only a few types of PQCs are known. Even the state-of-the-art library for quantum computers [12] has only four types of PQCs such as *Ry*, *RyRz*, *SwapRz* and *UCCSD*. They are all general PQCs with static structures and can be used for any problems. (2) Existing PQCs do not take into account the feasibility of output answers, and they often output infeasible answers. We need to ensure that results are feasible answers to corresponding optimization problems when using the VQE algorithm for optimization problems.

In this paper, we propose novel PQCs for two types of optimization problems. In the proposed PQCs, we pay attention to the constraints of an optimization problem, and we dynamically create a PQC that reflects those constraints of the optimization problem. We call such a PQC for the specific problem as a *problem-specific PQC*. Since problem-specific PQCs reflect the constraints of optimization problems, they naturally take into account the feasibility of output answers. With problem-specific PQCs, it is possible to reduce search spaces significantly. Thus, we can speed up the convergence of the VQE algorithms.

The rest of the paper is organized as follows. Section 2 covers the background on quantum circuits and the VQE algorithm. In Section 3, we explain the proposed PQCs for two types of optimization problems. Section 4 summarizes the experimental results of the proposed PQCs. Finally, Section 5 concludes the paper.

2. Background

In this section, we introduce quantum circuits and the VQE algorithm.

¹ IBM Quantum, IBM Research - Tokyo

² Department of Mechanical Engineering, Keio University

³ College of Information Science and Engineering, Ritsumeikan University

2.1 Quantum Circuits

A quantum circuit is a model of quantum computation [13] and contains qubits and a sequence of quantum gates.

In quantum computation, we use qubits instead of bits. A bit in classical computers has to be either zero or one. However, a qubit can be $|0\rangle$, $|1\rangle$, or the superposition state. The superposition state is a linear combination of $|0\rangle$ and $|1\rangle$ such as $\alpha|0\rangle + \beta|1\rangle$, where $\alpha, \beta \in \mathbb{C}$ and $|\alpha|^2 + |\beta|^2 = 1$. These α and β are called amplitudes of the corresponding bases. We also represent an n -qubit state as $|\psi\rangle = \sum_{k \in \{0,1\}^n} \alpha_k |k\rangle$, where $\alpha_k \in \mathbb{C}$ and $\sum_{k \in \{0,1\}^n} |\alpha_k|^2 = 1$. It is represented with a 2^n -dimensional state vector such as $(\alpha_0, \alpha_1, \dots, \alpha_{2^n-1})^T$.

Each quantum gate has the functionality corresponding to the particular unitary operation. With qubits, a quantum gate represents what unitary operator is applied to which qubits. We explain the details of quantum gates used in the proposed PQC in Sec. 3.

2.2 The VQE Algorithm

The VQE algorithm is an algorithm to find the minimal eigenvalue and its eigenvector of a given Hamiltonian. To do this, the VQE algorithm uses the variational principle as shown in Eq. (1). H and $|\psi\rangle$ represent a given Hamiltonian and a quantum state, respectively in Eq. (1). λ_{min} represents the minimal eigenvalue of H .

$$\lambda_{min} \leq \langle \psi | H | \psi \rangle \quad (1)$$

The variational principle holds for an arbitrary quantum state. Thus, for an arbitrary quantum state $|\psi\rangle$, the expectation value $\langle \psi | H | \psi \rangle$ is greater than or equal to the minimal eigenvalue of H .

Based on the variational principle, the VQE algorithm consists of two parts. One is executed on quantum computers, and the other one is on classical computers. As we mentioned, the part executed on quantum computers has a shallow quantum circuit with parameters called a PQC. A PQC prepare a quantum state from an initial state, and it can also prepare various quantum state by changing the parameters. With the created quantum state, the expectation values of each term in a given Hamiltonian are obtained by sampling outcomes. Then, classical computers calculate the total of the expectation values by summing those of each term. After that, classical computers determine the next parameters for the PQC by using classical optimization algorithms such as the Nelder–Mead algorithm [14], the Powell algorithm [15], and many more [16–18]. The PQC creates a new quantum state with new parameters, and the expectation values of each term in the given Hamiltonian are obtained by sampling outcomes again with the new quantum state. This process is repeated until the expectation value of the given Hamiltonian converges.

3. The Proposed problem-specific PQCs

3.1 Overview of the problem-specific PQC

In this subsection, first, we introduce the general idea of the problem-specific PQC. After mapping binary variables x_i to qubits q_i , we pay attention to the constraints of an optimization problem. As always, constraints restrict the set of feasible answers for the optimization problem. We utilize the constraints to dynamically construct a problem-specific PQC that reflects those constraints of the optimization problem. Therefore, we can restrict a unitary transformation that is provided by the problem-

specific PQC while taking constraints into account. Then, it is possible to reduce the set of the bases of a state vector that is the output of the problem-specific PQC. As a result, we can make the search space smaller.

For example, suppose that a constraint of an optimization problem is $\sum_i x_i = 1$. The constraint represents that exactly one of the variable has to be one, while the other variables have to be zero. This type of constraint often appears in optimization problems, e.g., the traveling salesman problem and the job scheduling problem. Constraint $\sum_i x_i = 1$ restricts the set of the feasible answers to the set of the bases of the corresponding W state. A W state is a superposition of states that exactly one of the qubits is $|1\rangle$ while the other qubits are $|0\rangle$ with equal amplitudes. A W state of n qubits is represented as $|W\rangle = \frac{1}{\sqrt{2^n}}(|10\dots 0\rangle + |01\dots 0\rangle + |00\dots 1\rangle)$. Each base of $|W\rangle$ exactly corresponds to an assignment of variables that satisfies $\sum_i x_i = 1$. We do not need to consider other bases since all of them are obviously infeasible due to the constraint $\sum_i x_i = 1$.

The basic concept of the problem-specific PQC is as follows. Let \mathbb{S}_{all} be the set of all the bases of n qubits, so $|\mathbb{S}_{all}|$ is 2^n . Then, let $\mathbb{S}_{feasible}$ the a set of bases corresponding the feasible answers of an optimization problem after mapping variables to qubits. \mathbb{S}_{all} includes $\mathbb{S}_{feasible}$ from the definition. For example, when one of the feasible answers is $x_0 = 1, x_1 = 0$ and $x_2 = 0$, the corresponding base is $|q_0 q_1 q_2\rangle = |100\rangle$. Thus, $|100\rangle$ is in $\mathbb{S}_{feasible}$. With the problem-specific PQC, we consider set $\mathbb{S}_{proposed}$ that includes $\mathbb{S}_{feasible}$, but the size of the set is smaller than $|\mathbb{S}_{all}|$. The relation between each set is described as $\mathbb{S}_{feasible} \subseteq \mathbb{S}_{proposed} \subseteq \mathbb{S}_{all}$. By using such $\mathbb{S}_{proposed}$, the basic concept of the problem-specific PQC is written as Eq. (2). $U_{proposed}$ represents a unitary transformation that is provided by a problem-specific PQC. $|0\rangle$ represents a base whose index is all zeros. We use $|0\rangle$ as an initial state for the problem-specific PQC. α_i represents an amplitude of $|\psi_i\rangle$. These amplitudes are controlled by parameters of the problem-specific PQC. With a proper problem-specific PQC, we can change only α_i while keeping the amplitudes of the other states not included in $\mathbb{S}_{proposed}$ 0. We explain how the problem-specific PQC works with examples later.

$$U_{proposed} |0\rangle = \sum_i \alpha_i |\psi_i\rangle, \quad |\psi_i\rangle \in \mathbb{S}_{proposed} \quad (2)$$

Usually, an optimization problem has more than one constraint. For such cases, we create multiple problem-specific parameterized quantum sub-circuits each of which reflects the corresponding constraint. Then, by combining those sub-circuit properly, even though the optimization problem has more than one constraint, it is still possible to create a problem-specific PQC and reduce the search space.

3.2 Problem-specific PQCs for the TSP

In this subsection, we introduce problem-specific PQCs for the traveling salesman problem (TSP). The TSP is a well-known NP-hard problem in combinatorial optimization problems. The traveling salesman goes from city to city to sell products, and the objective is to find the shortest path that the salesman can visit all the cities once and return to his starting point. With an undirected graph $G = (V, E)$, we can formulate the TSP as follows. Each edge $(u, v) \in E$ in the graph has weight $W_{u,v}$, then find the Hamiltonian cycle such that the sum of the weights of each edge in the cycle is minimized. Let $N = |V|$ and let us label the vertices $1, \dots, N$. For a linear program, we

use N^2 variables $x_{v,p}$ where v represents the vertex and p represents its order in a prospective cycle. Then, the linear program of the TSP is formulated as Eq. (3). Note that $N + 1$ should be read as 1 in Eq. (3).

$$\begin{aligned} & \text{Minimize} \sum_{(u,v) \in E} W_{u,v} \sum_{p=1}^N x_{u,p} x_{v,p+1} \\ & \text{Subject to} \sum_{v=1}^N x_{v,p} = 1, \quad p = 1 \dots N \\ & \sum_{p=1}^N x_{v,p} = 1, \quad v = 1 \dots N \\ & x_{v,p} \in \{0, 1\} \end{aligned} \quad (3)$$

In this paper, we propose four PQCs for the TSP. Each of them has different characteristics such as the types of the constraints considered, the number of quantum gates, and the number of parameters. Their details will be explained in Sec. 3.2.1, Sec. 3.2.2, Sec. 3.2.3, and Sec. 3.2.4, respectively.

3.2.1 PQCs satisfying only the constraints on the first line

For the first proposed PQC, we take into account only the constraints on the first line. In each constraint of Eq.(3), exactly one variable has to be one while the other variables have to be zero. As we have already explained in this paper, this type of constraint restricts the set of the feasible answers to the set of the bases of the corresponding W state. The total number of the constraints represented by the first line in the constraints, $\sum_{v=1}^N x_{v,p} = 1$, is N since we have a constraint for each $p = 1, \dots, N$. Thus, after mapping binary variables to qubits, with the tensor product of the corresponding N W states, we can restrict a search space to $\bigotimes_{p=1}^N |W_p\rangle$. We do not need to consider other bases, not in $\bigotimes_{p=1}^N |W_p\rangle$, since they do not satisfy $\sum_{v=1}^N x_{v,p} = 1$, $p = 1 \dots N$. Note that we do not consider constraints represented by the second line in the constraints. Thus, some bases in $\mathbb{S}_{proposed}$ may not satisfy these constraints in the second line of constraints. However, the relation between each set, $\mathbb{S}_{feasible} \subseteq \mathbb{S}_{proposed} \subseteq \mathbb{S}_{all}$, still holds, and we can reduce the search space.

Therefore, we need to create quantum circuits that create W states. The deterministic methods for creating W states of arbitrary sizes are discussed in previous studies [19, 20]. However, a conventional W state has equal amplitudes for each base. For the VQE algorithm, we need to control the amplitudes of each base with parameters as shown in Eq. (4), and optimize them with a classical optimizer to find the minimum eigenvalue.

$$\begin{aligned} |W(\phi)\rangle &= \sum_i \alpha_{i(\phi)} |\psi_i\rangle, \\ \sum_i |\alpha_{i(\phi)}|^2 &= 1, \quad |\psi_i\rangle \in \{|10\dots 0\rangle, |01\dots 0\rangle, |00\dots 1\rangle\} \end{aligned} \quad (4)$$

In Eq. (4), $|\psi_i\rangle$ represents one of the bases in the corresponding W state where the i -th qubit is $|1\rangle$ while other qubits are $|0\rangle$. An amplitude α_i has the set of parameters, ϕ , to change its value. Note that ϕ can have multiple parameters such as $\{\theta_1, \theta_2, \dots\} \in \phi$. We call this $|W(\phi)\rangle$ in Eq. (4) as a parameterized W state.

Let us introduce quantum gates before explaining how

to create a quantum circuit for a parameterized W state. An X gate and a $R_y(\theta)$ gate act on a single qubit while a *Controlled Z* (CZ) gate and a *Controlled NOT* ($CNOT$) gate act on two qubits. A two-qubit gate has the control bit and the target bit. If the control bit of a two-qubit gate is $|1\rangle$, the two-qubit gate applies a particular operation to its target bit. If the control bit of a two-qubit gate is $|0\rangle$, the two-qubit gate does not apply any operations to its target bit. For example, in the case of a $CNOT$ gate, if the control bit of the $CNOT$ gate is $|1\rangle$, it applies an X gate to its target bit. If its control bit is $|0\rangle$, it does not apply any operations to its target bit. A *Controlled SWAP* ($CSWAP$) gate is a three-qubit gate, which acts on the control qubit and the two target qubits. If the control qubit is $|1\rangle$, then the two target qubits are swapped.

The unitary matrices of each gate are as follows.

$$X \equiv \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \quad (5)$$

$$R_y(\theta) \equiv \begin{bmatrix} \cos \frac{\theta}{2} & -\sin \frac{\theta}{2} \\ \sin \frac{\theta}{2} & \cos \frac{\theta}{2} \end{bmatrix}, \quad (6)$$

$$CZ \equiv \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix}, \quad (7)$$

$$CNOT \equiv \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}, \quad (8)$$

$$CSWAP \equiv \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}. \quad (9)$$

Note that a $R_y(\theta)$ gate has a parameter θ and its matrix elements can be changed dynamically by θ . On the other hand, the matrix elements of an X gate, a CZ gate, and a $CNOT$ gate do not change. We sometimes use an index for a gate to represent which qubit the gate was applied. For example, an X_i gate means an X gate for q_i . For a $R_y(\theta)$ gate, we also use an index for its parameter. A $R_{y_i}(\theta_p)$ gate means a $R_y(\theta)$ gate for q_i where its parameter is θ_p . Since two-qubit gates have control bits and target bits, we use two numbers for their index. The left number in an index represents the control bit of a two-qubit gate, and the right number represents its target bit. For example, a $CNOT_{i,j}$ gate means a $CNOT$ gate whose control bit is q_i and target bit is q_j . Note that which qubit is the control bit or the target bit of a CZ gate is not important since $CZ_{i,j} = CZ_{j,i}$. $CSWAP_{i,j,k}$ represents a $CSWAP$ gate whose control bit is q_i and target bits are q_j and q_k . Note that the target qubits in a $CSWAP$ gate is permutation invariant, i.e. $CSWAP_{i,j,k} = CSWAP_{i,k,j}$.

We use the above gates to create such parameterized W states and use existing methods [20] as the base. However, we do not determine the parameters of $R_y(\theta)$ gates yet for parameterized W states. For ease of explanation, we consider a case with three qubits, q_1, q_2 , and q_3 . We explain an algorithm for arbitrary sizes of qubits later. The initial state is $|q_1 q_2 q_3\rangle = |000\rangle$. Firstly, we apply an X gate to q_1 . Then, the state will change as $X_1 |000\rangle = |100\rangle$. Then we apply two $R_y(\theta)$ gates and a CZ gate in the following

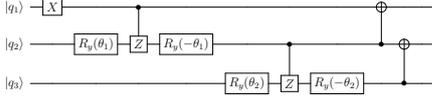


図 1 A quantum circuit for a parameterized W state of three qubits

order.

- (1) Apply a $R_{y_2}(\theta_1)$ gate.
 - (2) Apply a $CZ_{1,2}$ gate.
 - (3) Apply a $R_{y_2}(-\theta_1)$ gate. Note that the same parameter θ_1 is used in 1) and 3), but with a different sign.
- After that, we apply two $R_y(\theta)$ gates and a CZ gate in the same order. However, at this time, we apply a $R_{y_3}(\theta_2)$ gate, a $CZ_{2,3}$ gate, and a $R_{y_3}(-\theta_2)$ gate. The state will be as Eq. (10).

$$\alpha_{1(\phi)} |100\rangle + \alpha_{2(\phi)} |110\rangle + \alpha_{3(\phi)} |111\rangle,$$

$$\sum_{i=1}^3 |\alpha_{i(\phi)}|^2 = 1,$$

$$\alpha_{1(\phi)} = \cos \theta_1, \alpha_{2(\phi)} = -\sin \theta_1 \cos \theta_2, \alpha_{3(\phi)} = \sin \theta_1 \sin \theta_2 \quad (10)$$

Amplitude $\alpha_{i(\phi)}$ depends on the values of θ_1 and θ_2 . Then, we apply a $CNOT_{2,1}$ gate and a $CNOT_{3,2}$ gate. After applying $CNOT$ gates, the final state will be as Eq. (11).

$$\alpha_{1(\phi)} |100\rangle + \alpha_{2(\phi)} |010\rangle + \alpha_{3(\phi)} |001\rangle,$$

$$\sum_{i=1}^3 |\alpha_{i(\phi)}|^2 = 1,$$

$$\alpha_{1(\phi)} = \cos \theta_1, \alpha_{2(\phi)} = -\sin \theta_1 \cos \theta_2, \alpha_{3(\phi)} = \sin \theta_1 \sin \theta_2 \quad (11)$$

This state is the same as a parameterized W state of three qubits. Figure 1 shows a quantum circuit for a parameterized W state of three qubits. The text in the boxes of each quantum gate represents its unitary matrix. The leftmost gate in Fig. 1 represents that an X gate is applied to q_1 . The second gate from the left in Fig. 1 represents that a $R_y(\theta)$ gate is applied to q_2 with parameter θ_1 . The third gate from the left in Fig. 1 represents that a CZ gate is applied to q_1 and q_2 , and its control bit is q_1 and its target bit is q_2 . The rightmost gate in Fig. 1 represents a CNOT gate is applied to q_2 and q_3 , and its control bit is q_2 and its target bit is q_3 .

By combining quantum circuits to create parameterized W states, we can create a problem-specific PQC of the VQE algorithm for the TSP. As mentioned above, a linear program of the TSP is represented as Eq. (3). For the VQE algorithm, we need to map these variables to qubits. To do this, we prepare N^2 qubits $q_{v,p}$ and map each variable $x_{v,p}$ to the corresponding qubit $q_{v,p}$. Note that N is the number of vertices. We use N independent quantum circuits to create parameterized W states of N qubits. For qubits $q_{1,1}, q_{1,2}, \dots, q_{1,N}$, we insert the first quantum circuit to create a parameterized W state of N qubits. Then, for qubits $q_{2,1}, q_{2,2}, \dots, q_{2,N}$, we insert the second one. In the same manner, we keep inserting quantum circuits to create parameterized W states. The last one will be for $q_{N,1}, q_{N,2}, \dots, q_{N,N}$. After that, we obtain a quantum circuit as shown in Fig 2. Each box represents a quantum circuit to create a parameterized W state with the set of parameters ϕ_i for the corresponding qubits. Each $|W_i(\phi_i)\rangle$ ($i = 1, \dots, N$) on the right in Fig 2 represents the output of the corresponding circuit. Note

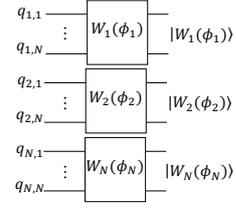


図 2 A problem-specific PQC of the VQE algorithm for the TSP

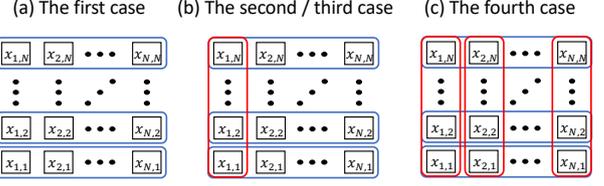


図 3 Schematic of the constraints considered for each case; (a) only the first line, (b) a constraint in the second line as well as the first line and (c) all constraints. The blue box represents a constraint in the first line, while the red box indicates a constraint in the second line of constraints of Eq. 3.

that each $|W_i(\phi_i)\rangle$ ($i = 1, \dots, N$) has the different set of parameters. With the circuit in Fig. 2, we can create a tensor product of the parameterized W states $\bigotimes_{p=1}^N |W_p(\phi_p)\rangle$

3.2.2 PQCs satisfying an L-shaped constraint with CNOT operations

For the second PQC, we take into account not only the first line but also taking into account constraint $\sum_{p=1}^N x_{1,p} = 1$ in the second line of the constraints of Eq. 3 to further reduce the search space, as is shown in Figure 3 (b). Unlike the first PQC, the situation requires more “correlations” among qubits being mapped from variables, since variables $x_{1,p}$, $p = 1 \dots N$ appear in both the first and the second line; it is no longer possible to realize the constraints by a tensor product of N quantum states. Hence, we utilize CNOT gates together with the parameterized W state gates to create such a quantum circuit.

The protocol of creating the PQC follows two steps. First, we construct a quantum circuit satisfying both two constraints, $\sum_{p=1}^N x_{1,p} = 1$ and $\sum_{v=1}^N x_{v,1} = 1$, which we call “an L-shaped constraint” because the involved variables form L-shape in Figure 3 (b). In the L-shaped constraint, one variable in each set of $\{x_{1,p'} | p' = 2 \dots N\}$ and $\{x_{v',1} | v' = 2 \dots N\}$ has to be one if $x_{1,1}$ is zero, while all variables in $\{x_{1,p'} | p' = 2 \dots N\}$ and $\{x_{v',1} | v' = 2 \dots N\}$ are zeros if $x_{1,1}$ is one. According to this, one can easily understand that the corresponding unitary operations on the initialized qubits $q_{i,j}$ being mapped from variables $x_{i,j}$ is realized by applying parameterized W state gates to each set of qubits $\{q_{1,p'} | p' = 2 \dots N\}$ and $\{q_{v',1} | v' = 2 \dots N\}$ if $|0\rangle_{q_{1,1}}$, and applying identity gates if $|1\rangle_{q_{1,1}}$. Thus, the PQC can be created with CNOT gates and parameterized W state gates, as depicted in Figure 4 (a). In a quantum circuit of Figure 4 (a), the leftmost R_y gate with the trainable parameter ϕ_0 and the following two CNOT gates, $CNOT_{q_{1,1}, q_{1,2}}$ and $CNOT_{q_{1,1}, q_{2,1}}$, determine whether $|0\rangle_{q_{1,1}}$ or $|1\rangle_{q_{1,1}}$, and whether W state gates or identity gates are applied to each set of qubits $\{q_{1,p'} | p' = 2 \dots N\}$ and $\{q_{v',1} | v' = 2 \dots N\}$ depending on the condition of $q_{1,1}$, respectively. Note that we here use the fact that a parameterized W state gate is exactly an identity gate if an X gate is applied to the first qubits beforehand, which can be checked readily by looking into

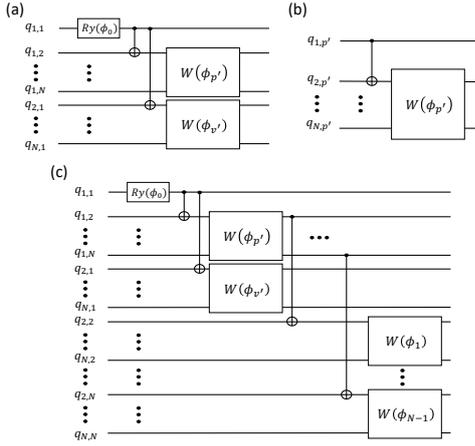


図 4 A PQC for the second case; (a) a quantum circuit for an L-shaped constraint, (b) a quantum circuit corresponding to the constraint, $\sum_{v=1}^N x_{v,p'} = 1$ and (c) a whole picture of the PQC for the second case.

Fig. 1. As a result, the quantum circuit can create the desired quantum state expressed as

$$\begin{aligned} & \cos \frac{\phi_0}{2} |0\rangle_{q_{1,1}} |W(\phi_{p'})\rangle_{\{q_{1,p'}|p'=2\dots N\}} |W(\phi_{v'})\rangle_{\{q_{v',1}|v'=2\dots N\}} \\ & + \sin \frac{\phi_0}{2} |1\rangle_{q_{1,1}} |0\rangle_{\{q_{1,p'}|p'=2\dots N\}} |0\rangle_{\{q_{v',1}|v'=2\dots N\}}, \end{aligned} \quad (12)$$

where ϕ_s are all trainable parameters.

Second, we apply unitary operations for the remaining constraints, $\sum_{v=1}^N x_{v,p'} = 1$, $p' = 2 \dots N$, to a resultant quantum state in Eq. 12. Here, since the qubits corresponding to the variables $\{x_{1,p'}|p' = 2 \dots N\}$ in the constraints have already been determined, the constraints can be read in the similar way to the first step as follows; if $x_{1,p'}$ is one, all variables $\{x_{v',p'}|v' = 2 \dots N\}$ are zeros, while if $x_{1,p'}$ is zero one variable in $\{x_{v',p'}|v' = 2 \dots N\}$ has to be one. As we have seen in the first step, the corresponding unitary operation can be realized by a $CNOT_{q_{1,p'},q_{2,p'}}$ gate followed by parameterized W state gates on the set of qubits $\{q_{v',p'}|v = 2 \dots N\}$ as represented in Figure 4 (b). Thus, using $N - 1$ CNOT gates and $N - 1$ circuits for parameterized W states, we can create the unitary operators that create a quantum state satisfying the remaining constraints.

Following the above two steps, we can create the problem-specific PQC in Figure 4 (c).

3.2.3 PQCs satisfying an L-shaped constraint with parameter sharing

For the third PQC, we modify the second PQC to reduce the cost of the implementation for the current quantum processors [21]. Since current noisy devices suffer from an exponential decay of quantum coherence, deep circuits would be problematic. In the second case, the circuit becomes deep due to the dependence in its own structure; firstly, a quantum circuit for an L-shaped constraint is constructed, and then other gates are applied for the remaining constraints.

To remedy this issue, we introduce the technique, *parameter sharing*, which makes the circuit shallower with fewer CNOT gates. The main point of this technique is as follows; CNOT gates (and also X gates in parameterized W state gates) used in Fig. 3 are replaced with R_y gates with the shared parameters such that the probability of obtaining $|1\rangle_{q_{1,p'}}$ is equal to that of $|0\rangle_{\{q_{v',p'}|v'=2\dots N\}}$.

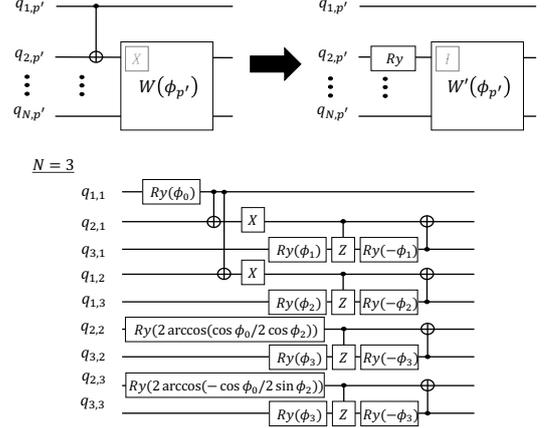


図 5 A PQC with $N = 3$ for the third case. To create the PQC, the top-left circuit appearing in the quantum circuit of the second case is replaced with the top-right circuit.

To demonstrate the parameter sharing in detail, we provide a simple example of the quantum circuit with $N = 3$ in Figure 5. In this scenario, parameters ϕ_0 and ϕ_2 are used for not only $q_{1,1}$ and $q_{1,2}$, but also $q_{2,2}$ and $q_{3,2}$ in unique ways such as $2 \arccos(\cos \phi_0 / 2 \cos \phi_2)$. Indeed, the trigonometric functions inside the inverse trigonometric function, $\cos \phi_0 / 2 \cos \phi_2$ and $-\cos \phi_0 / 2 \sin \phi_2$ are the amplitudes of $|1\rangle_{q_{1,2}}$ and $|1\rangle_{q_{1,3}}$, respectively. Therefore, the amplitude of $|00\rangle_{\{q_{v,2}|v=2,3\}}$ is always the same as that of $|1\rangle_{q_{1,2}}$ by the parameter sharing (similarly, this is true to $q_{1,3}$, $q_{2,3}$, and $q_{3,3}$). Note that this technique can be easily extended for PQCs with arbitrary N since the probability of obtaining $|1\rangle_{q_{1,p'}}$ for all $p' = 2 \dots N$ is analytically calculated in the similar way as shown in Eq.(11).

By utilizing such “classical correlation”, we can create a shallower PQC satisfying the constraints in Figure 3 (b), which is expected to be more suitable for current noisy devices. However, the technique has a limitation on the ability to restrict the set of bases compared to the second case. As we can see in Figure 5, the quantum state created by the PQC is not fully entangled, i.e. it can be written as the tensor product of small quantum states. Consequently, the set of the bases of the quantum state includes the bases that are not in the second case. However, the probability to obtain such extra bases is at most a half. This characteristic contributes to interesting results which we will discuss in Sec. 4

3.2.4 PQCs satisfying all constraints

For the fourth PQC, we consider all constraints of Eq. 3 to completely exclude the infeasible answers as shown in Fig. 3 (c). Thus, the set of the bases of the quantum states includes only feasible answers, i.e. $\mathbb{S}_{case 4} = \mathbb{S}_{feasible}$.

Such a PQC for arbitrary N can be constructed in a recursive manner. After a quantum circuit with $N = 2$ is exemplified, we will demonstrate that the PQC with $N = k$ can be constructed using the quantum circuit with $N = k - 1$. The basic idea is that the assignments of feasible answers on the 2D grid as shown in Fig. 3 (c) can be interpreted as permutation matrices. It is due to the fact that the constraints of Eq.(3) are exactly the same as the definition of permutation matrices. Note that a permutation matrix is a square matrix, every row and column of which has only one entry, 1 while the other entries are 0. Hence, with the equivalence of permutation matrices and the assignment of the feasible answers on the 2D grid, we construct the quantum circuit.

Firstly, we show the PQC with $N = 2$. For the num-

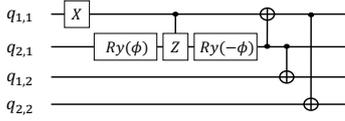


図 6 A PQC with $N = 2$ for the fourth case.

ber of city $N = 2$, two feasible answers exist, as there are two 2×2 permutation matrices, $\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$ and $\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$.

Thus, a quantum state we want to create can be described by the superposition of two bases, $|0110\rangle$ and $|1001\rangle$ with the order of qubits $|q_{1,1}q_{2,1}q_{1,2}q_{2,2}\rangle$. A quantum circuit for $N = 2$ can be created as shown in Fig. 6, where the quantum state is represented as $\cos \phi |1001\rangle - \sin \phi |0110\rangle$.

Secondly, we show that the PQCs with $N = k$ can be constructed by using the quantum circuit with $N = k - 1$. The conceptual overview to create PQCs for the fourth case is as shown in Fig. 7. Suppose that we have all the permutation matrices of size $k - 1$, so the total number of the permutation matrices is $(k - 1)!$. Then one can obtain $k!$ permutation matrices of size k in the following way.

- (1) Pad the additional zeros to all the $(k - 1) \times (k - 1)$ permutation matrices to form $k \times k$ square matrices.
- (2) Set the top right element in each of the matrices as 1 to make them permutation matrices.
- (3) Permute the k -th and the j -th column of the matrices for all $j = 1 \dots k - 1$.

Note that we here use the fact that exchanging the i -th and the j -th column of a permutation matrix results in also a permutation matrix. Consequently, with $(k - 1)!$ permutation matrices in the second step and $k(k - 1)!$ permutation matrices in the third step, we can obtain $k!$ permutation matrices of size k by the above steps. In analogous to the case of permutation matrices, we construct a quantum circuit with $N = k$. Let $|\Psi_{k-1}\rangle$ be a quantum state whose bases are all feasible answers for $N = k - 1$ with the order of qubits $|q_{1,1} \dots q_{1,k-1} q_{2,1} \dots q_{2,k-1} \dots q_{k-1,1} \dots q_{k-1,k-1}\rangle$. Then, in a similar way, we can create the desired quantum states as follows;

- (1) Prepare the initialized $2k - 1$ qubits, labeled as $q_{k,p'}, p' = 2 \dots k$ and $q_{v,k}, v = 1 \dots k$.
- (2) Apply a parameterized W state gate to the set of qubits, $\{q_{v,k} | v = 1 \dots k\}$.
- (3) Apply *CSWAP* gates to the corresponding qubits in $|\Psi_{k-1}\rangle$, $|W_k(\psi)\rangle_{\{q_{v,k} | v=1 \dots k\}}$, and $|0\rangle_{\{q_{k,p'} | p'=1 \dots k-1\}}$; the set of *CSWAP* operations, $\{CSWAP_{q_{v',k}, q_{k,p'}, q_{v',k}, q_{v',p'}} | p' = 1 \dots k - 1\}$ are applied for all $v' = 1 \dots k - 1$.

In this procedure, the parameterized W state gate is used to represent the additional k -th row of $k \times k$ matrix, which can be regarded as the permutation inside the k -th row. Then, *CSWAP* gates are used to serve as the permutation of the remaining rows depending on the state of k -th row; the states of $\{q_{k,p'} | p' = 1 \dots k - 1\}$ and $\{q_{v,p'} | p' = 1 \dots k - 1\}$ are exchanged if $|1\rangle_{q_{v,k}}$, while the states of $\{q_{k,p'} | p' = 1 \dots k - 1\}$ and $\{q_{v,p'} | p' = 1 \dots k - 1\}$ remain unchanged if $|0\rangle_{q_{v,k}}$. As a demonstration, we give a simple example of the PQC with $N = 3$ for the fourth case as shown in Fig. 8. Then the corresponding quantum state is represented as Eq.(13), with the order of qubits $|q_{1,1}q_{2,1}q_{3,1}q_{1,2}q_{2,2}q_{3,2}q_{1,3}q_{2,3}q_{3,3}\rangle$, which is exactly the superposition of bases of six feasible answers.

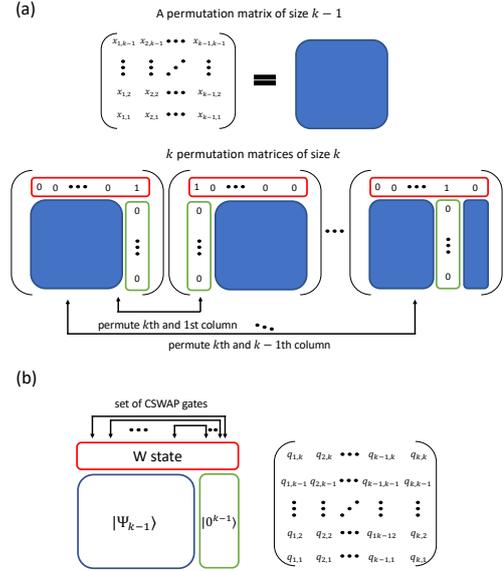


図 7 The conceptual overview to create PQCs for the fourth case. (a) illustrates the property of permutation matrices, which is used for constructing PQCs. (b) is a schematic view to create the desired quantum state for $N = k$ by using a quantum state for $N = k - 1$.

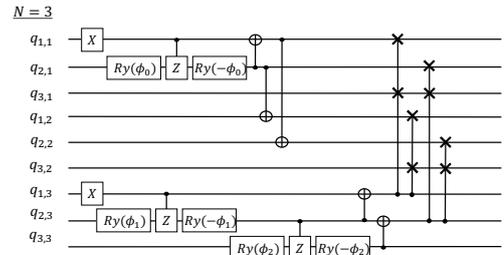


図 8 A PQC with $N = 3$ for the fourth case.

$$\begin{aligned}
 |\Psi_{k-1}\rangle = & -\cos \phi_0 \sin \phi_1 \cos \phi_2 |100001010\rangle \\
 & + \cos \phi_0 \sin \phi_1 \sin \phi_2 |001100010\rangle \\
 & + \cos \phi_0 \sin \phi_1 \sin \phi_2 |1000100001\rangle \\
 & - \sin \phi_0 \sin \phi_1 \cos \phi_2 |0101000001\rangle \\
 & + \cos \phi_0 \cos \phi_1 |001010100\rangle \\
 & - \sin \phi_0 \cos \phi_1 |010001100\rangle
 \end{aligned} \tag{13}$$

Therefore, we can construct the PQC for arbitrary N by recursively performing the procedure explained in the above starting from the quantum circuit with $N = 2$.

3.3 A problem-specific PQC for the Minimum Vertex Cover

In this subsection, we use the minimum vertex cover as another example of applying the proposed method. The minimum vertex cover is another well-known NP-hard problem in combinatorial optimization problems. When at least one of the endpoints of an edge e_i connects to a vertex v_j , it is said that e_i is covered by v_j . With an undirected graph $G = (V, E)$, the minimum vertex cover is to find the minimum number of vertices such that covers all the edges in G . Let $N = |V|$ and let us label the vertices $1, \dots, N$. We can then formulate the minimum vertex cover as Eq. (14) for linear programming. A constraint in Eq. (14) becomes 1, if and only if both x_u and x_v are zero, which means the edge (u, v) is not covered. The total number of constraints in Eq. (14) is equal to $|E|$.

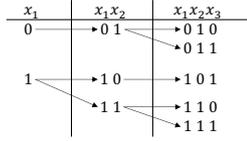


図 9 A process to enumerate possible assignments of variables for the vertex cover.

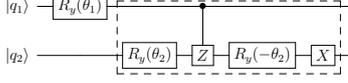


図 10 A quantum circuit to enumerate feasible bases of qubits for a constraint of the vertex cover.

$$\text{Minimize } \sum_{i=1}^N x_i \quad (14)$$

$$\text{Subject to } (1 - x_u)(1 - x_v) = 0, \quad \forall (u, v) \in E$$

$$x_i \in \{0, 1\}$$

A constraint in Eq. (14) can be read as if x_u is zero, x_v has to be one. Additionally, if x_v is one, x_u can be either one or zero. This process to enumerate feasible assignments of variables for the vertex cover can be written as in Fig. 9. In Fig. 9, we consider two constraints, $(1 - x_1)(1 - x_2) = 0$ and $(1 - x_2)(1 - x_3) = 0$. As Fig. 9 shows, when $x_1 = 0$, there is only one feasible assignment for x_1 and x_2 which is $(x_1 = 0, x_2 = 1)$. When $x_1 = 1$, there are two feasible assignments for x_1 and x_2 which are $(x_1 = 1, x_2 = 0)$ and $(x_1 = 1, x_2 = 1)$. We then move on to the second constraint $(1 - x_2)(1 - x_3) = 0$. In the same manner, when $x_2 = 0$, there is only one feasible assignment for x_2 and x_3 which is $(x_2 = 0, x_3 = 1)$. When $x_2 = 1$, there are two feasible assignments for x_2 and x_3 , which are $(x_2 = 1, x_3 = 0)$ and $(x_2 = 1, x_3 = 1)$. By combining the result of the first constraint and that of the second constraint, the feasible assignments of x_1, x_2 and x_3 can be written as the rightmost column in Fig. 9.

After mapping binary variables x_i to qubits q_i , we can realize the above process with a quantum circuit shown in Fig. 10. An initial state $|q_1 q_2\rangle = |00\rangle$ changes to the state as shown in Eq. (15) after applying the quantum circuit shown in Fig. 10. Similar to the case of the TSP, we can control the amplitudes of each base $\alpha_i(\phi)$ by changing parameters, θ_1 and θ_2 , in the quantum circuit.

$$\alpha_{1(\phi)} |01\rangle + \alpha_{2(\phi)} |10\rangle + \alpha_{3(\phi)} |11\rangle,$$

$$\sum_i |\alpha_{i(\phi)}|^2 = 1,$$

$$\alpha_{1(\phi)} = \cos \frac{\theta_1}{2}, \quad \alpha_{2(\phi)} = -\sin \frac{\theta_1}{2} \sin \theta_2, \quad \alpha_{3(\phi)} = \sin \frac{\theta_1}{2} \cos \theta_2 \quad (15)$$

The quantum circuit in Fig. 10 is for a single constraint of the minimum vertex cover. However, the vertex cover usually has more than one constraint. For multiple constraints, we use a sub-circuit in the dashed box of Fig. 10 for some of the constraints instead of every constraint. This is because that there is a possibility to break the relation $\mathbb{S}_{feasible} \subseteq \mathbb{S}_{proposed}$ if we use the sub-circuit in Fig. 10 for every constraint when a graph has *cycles*. To choose whether we use the sub-circuit in Fig. 10 for a constraint or not, we consider *spanning tree* T of graph G . Then, we consider the minimum vertex cover of T instead of G , and use the sub-circuit in Fig. 10 for every constraints of the minimum vertex cover of T sequentially. Some of the output bases of a problem-specific PQC for the minimum vertex cover may not satisfy the original

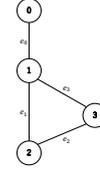
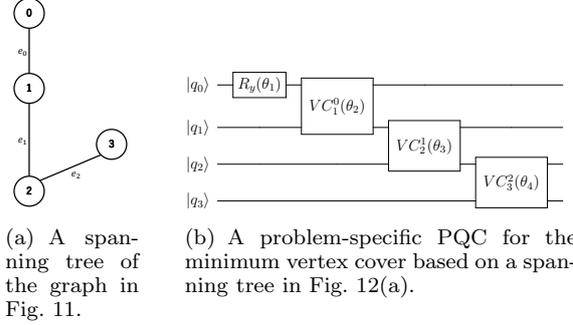


図 11 A Graph with four nodes that has a cycle.



(a) A spanning tree of the graph in Fig. 11.

(b) A problem-specific PQC for the minimum vertex cover based on a spanning tree in Fig. 12(a).

図 12 An example of a spanning tree of the graph in Fig 11 and a problem-specific PQC for the minimum vertex cover based on the spanning tree of the graph.

constraints as in the case of the TSP. However, by considering a spanning tree of a graph, we can keep the relation $\mathbb{S}_{feasible} \subseteq \mathbb{S}_{proposed}$ and reduce a search space.

We now explain an algorithm to create a problem-specific parameterized quantum for the minimum vertex cover of a graph with an example. Suppose that a graph in Fig 11 is given. A spanning tree of this graph is as shown in Fig. 12(a). Note that there are no edges between vertex 1 and vertex 3 in Fig. 12(a). Figure 12(b) shows an example of a problem-specific PQC for the minimum vertex cover of the graph based on the spanning tree in Fig. 12(a). A gate labeled with VC in Fig. 12(b) corresponds to the sub-circuit in the dashed box of Fig. 10. Thus, a VC gate consists of two $R_y(\theta)$ gates, a CZ gate, and a X gate. The top index of a VC gate represents an index of a qubit that is the control bit of the CZ gate. The bottom index represents an index of a qubit that is the target bit of the CZ gate. Also, the other gates, two $R_y(\theta)$ gates and the X gate, are applied to the qubit of the bottom index. We call the qubit of the top index the control bit of a VC gate. Similarly, we call the qubit of the bottom index the target bit of a VC gate.

Based on Eq. (14), after mapping binary variables x_i to qubits q_i , the number of bases is 2^N . Similarly to the case of the TSP, when we use existing PQCs, the number of bases remains as 2^N . However, if we use the problem-specific PQC created explained in the above, the number of bases will be $< 2^N$. This is because this circuit for the minimum vertex cover does not exactly double the number of bases for each additional qubit. Thus, it can reduce the number of bases of the search space compared to existing circuits.

4. Experimental Results

We conducted simulation experiments to compare the convergence speed of each proposed PQCs and the Ry PQCs using Python. Qiskit Aqua 0.7.5 was used to convert optimization problems to the corresponding Ising Hamiltonians. Firstly, we explain the experimental results for the TSP. For the TSP, we run the VQE algorithm in Qiskit with the QASM simulator. The number of the shots of the QASM simulator used in each experiments was 1024. We conduct 10 trials with different ini-

表 1 The comparison of necessary parameters and gates among the proposed problem-specific PQC and a Ry PQC for the TSP with n qubits

Necessary resources	Ry	Proposed 1	Proposed 2	Proposed 3	Proposed 4
# of Parameters	$(D+1)n$	$n - \sqrt{n}$	$n - \sqrt{n} - 1$	$n - \sqrt{n} - 1$	$\frac{1}{2}n - \frac{1}{2}\sqrt{n}$
# of one-qubit gates	$(D+1)n$	$2n - \sqrt{n}$	$2n - \sqrt{n} - 4$	$2n - \sqrt{n} - 4$	$n - 1$
# of two-qubit gates	$D(n-1)$	$2n - 2\sqrt{n}$	$2n - \sqrt{n} - 3$	$2n - 2\sqrt{n} - 2$	$n - \sqrt{n} + 2$
# of CSWAP gates	—	—	—	—	$\frac{1}{3}n\sqrt{n} - \frac{1}{2}n + \frac{1}{6}\sqrt{n} - 1$

tial parameters for each PQC except the Ry PQCs. The COBYLA algorithm was used as the classical optimizing algorithm of the VQE algorithm for the TSP. For the experiments of the TSP, we used a complete graph with four nodes as the graph of the TSP. The experiments were conducted on a MacBook Pro with 2.9 GHz Intel Core i5 and DDR3 8 GB memory running macOS 10.14.6.

Figure 13 shows the comparison between each proposed problem-specific PQC and Ry PQCs with depth one, two, and three. Proposed 1, Proposed 2, Proposed 3, and Proposed 4 correspond to the PQCs in Sec. 3.2.1, Sec. 3.2.2, Sec. 3.2.3, and Sec. 3.2.4, respectively. As we can see, the convergence of the proposed PQCs is significantly faster than that of the Ry PQCs. The average execution time of Proposed 1, Proposed 2, Proposed 3, and Proposed 4 was 98 sec, 78 sec, 101 sec, and 22 sec. The execution time of Ry PQCs with depth 1, 2, and 3 was 3606 sec, 4941 sec, and 8732 sec. The expectation values of the proposed PQCs are rapidly decreased in the first 60 iterations compared to the Ry PQCs. Also, the initial expectation values of the proposed PQCs are remarkably lower than that of the Ry PQCs. A graph in Fig. 14 is extracted from the graph in Fig. 13 to focus the experimental results of the proposed PQCs more. The order of the convergence speed was Proposed 4 < Proposed 2 < Proposed 3 < Proposed 1 < Ry. This is closely related to the set of the bases $|\mathcal{S}\rangle$, i.e. $\mathcal{S}_{feasible} = \mathcal{S}_{Proposed 4} \subseteq \mathcal{S}_{Proposed 2} \subseteq \mathcal{S}_{Proposed 3} \subseteq \mathcal{S}_{Proposed 1} \subseteq \mathcal{S}_{all}$. Note that the convergence speed of Proposed 3 was faster than Proposed 1 despite the fact the Proposed 3 has more bases than Proposed 1. It is because that the probability to obtain extra infeasible answers (bases) is at most a half due to the parameter sharing as we explained in Sec. 3.2.3. By utilizing such ‘‘classical correlation’’, the convergence speed of Proposed 3 is faster than Proposed 1 even though Proposed 3 has more bases than Proposed 1.

We also analyzed whether each PQC can reach to the global minimum. The result is as follows; For proposed 4, every trial reached to the global minimum while others did not. Proposed 1, 2, and 3 could find the feasible answers and they could sometime reach to the global minimum. More specifically, Proposed 1, 2, and 3 reached to the global minimum forth, four times, four times, and two times, respectively. Ry PQCs didn’t converge well and they produced infeasible answers even after 400 iterations. Of course, whether we can reach the global minimum depends on not only the PQCs, but also different factors such as problem configurations, the types of classical optimizers, and initial parameters. We will continue to study the convergence to the global minimum as our future work.

Table 1 shows the number of necessary gates and parameters for each PQC. In Table 1, the # of Parameters row, the # of one-qubit gates row, the # of two-qubit gates columns, and the # of CSWAP gates row correspond to the number of independent parameters used in $R_y(\theta)$ gates, the total number of X gates and $R_y(\theta)$ gates, the total number of CZ gates and $CNOT$ gates, and the total number of CSWAP gates, respectively. D in the Ry column corresponds to the depth of Ry PQCs. From the

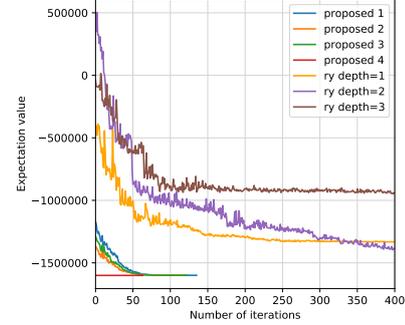


図 13 The comparison between each proposed problem-specific PQC and the Ry PQCs with depth one, two, and three for the TSP with four cities.

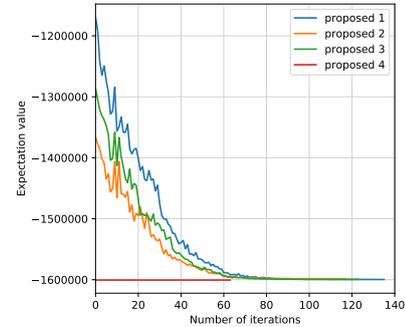


図 14 A graph extracted from the graph in Fig. 13 for the comparison between each proposed problem-specific PQC.

experimental results, we observed that the Proposed 4 is the best in terms of the convergence speed and the convergence to the global minimum. However, it requires a lot of CSWAP gates that realizing a CSWAP gate is expected to be difficult on current noisy devices. More specifically, a CSWAP gate requires 9 one-qubit gates and 8 two-qubit gates to be realized. Thus, the total number of the required one-qubit gates and two-qubit gates will be larger than other proposed PQCs. This will lead to challenges in the implementation on current noisy devices. To tackle this issue, we are considering to combine the parameter sharing and excitation preserving gates to replace high-cost CSWAP gates.

In contrast of the case of the TSP, for the minimum vertex cover, we used Numpy 1.18.4 to calculate the expectation values of Ising Hamiltonians and used Scipy 1.4.1 to optimize parameters for the VQE algorithm. The Nelder-Mead algorithm was used as a classical optimizer for the minimum vertex cover. For the experiments of the minimum vertex cover, we used a graph with six nodes that has a cycle in it. The experiments were conducted on a MacBook Air with 1.6 GHz Intel Core i5 and DDR3 8 GB memory running macOS 10.14.6.

Figure 15 shows the comparison between the proposed problem-specific PQC for the minimum vertex cover and Ry PQCs with depth one, two, and three. Similarly to the case of the TSP, the convergence of the proposed PQC is significantly faster than that of the Ry PQCs. Also, expectation values of the proposed PQC rapidly decreased. Specifically, an expectation value of our circuit after the

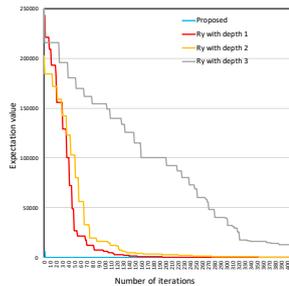


図 15 The comparison between the proposed problem-specific PQC and Ry PQC with depth 1, 2, and 3 for the minimum vertex cover.

表 2 The comparison of necessary parameters and gates between the proposed problem-specific PQC and a Ry PQC for the minimum vertex cover with n qubits

Necessary resources	Ry	Proposed
# of Parameters	$(D + 1)n$	n
# of one-qubit gates	$(D + 1)n$	$3n - 2$
# of two-qubit gates	$D(n - 1)$	$n - 1$

first iteration was 6342.657, and became 36.770226 after the second iteration. The answer for the minimum vertex cover was 3. The expectation value of the proposed PQC reached 3.0138958 after 150 iterations. On the other hand, the expectation value of the Ry PQC with depth 1 reached 1088.1005 after 150 iterations. Even after 400 iterations, it was still 4.729469.

Table 2 shows the comparison of necessary parameters and gates between the proposed problem-specific PQC and the Ry PQC for the minimum vertex cover. Similarly to the case of the TSP, the number of parameters of the proposed PQC is smaller than that of the Ry PQC. Also, when the depth of the Ry PQC is large, the numbers of one-qubit gates and two-qubit gates in the proposed PQC becomes less than those of the Ry PQC.

Each amplitude is not completely independent in the proposed problem-specific PQC. They have slight correlation between each other. However, it ensures that amplitudes of the bases that correspond to the answer of optimization problems can be 1. We need to carefully examine the relationship between the proposed method for the VQE algorithm and existing methods for classical computers.

5. Conclusions

In this paper, we proposed the problem-specific PQC of the VQE algorithm for optimization problems. In the proposed PQC, we pay attention to the constraints of an optimization problem, and we dynamically create a PQC that reflects those constraints of the optimization problem. By doing this, it is possible to significantly reduce search spaces. As a result, we can speed up the convergence of the VQE algorithms. We conducted the simulation experiments to compare the proposed PQC and the state-of-the-art PQC. In experiments, the proposed PQC could reduce the search spaces, and the convergence of the proposed PQC was significantly faster than that of the state-of-the-art PQC.

参考文献

[1] Shor, P. W.: Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer, *SIAM Journal on Computing*, Vol. 26, No. 5, pp. 1484–1509 (1997).
[2] Grover, L. K.: A fast quantum mechanical algorithm for database search, *STOC '96: Proceedings of the twenty-*

eight annual ACM symposium on Theory of Computing, July, pp. 212–219 (1996).

[3] Preskill, J.: Quantum Computing in the NISQ era and beyond (2018).
[4] Peruzzo, A., McClean, J., Shadbolt, P., Yung, M.-H., Zhou, X.-Q., Love, P. J., Aspuru-Guzik, A. and O’Brien, J. L.: .
[5] McClean, J. R., Romero, J., Babbush, R. and Aspuru-Guzik, A.: The theory of variational hybrid quantum-classical algorithms (2015).
[6] Wang, D., Higgott, O. and Brierley, S.: Accelerated Variational Quantum Eigensolver (2018).
[7] Parrish, R. M., Hohenstein, E. G., McMahon, P. L. and Martinez, T. J.: Quantum Computation of Electronic Transitions using a Variational Quantum Eigensolver (2019).
[8] Kandala, A., Mezzacapo, A., Temme, K., Takita, M., Brink, M., Chow, J. M. and Gambetta, J. M.: Hardware-efficient Variational Quantum Eigensolver for Small Molecules and Quantum Magnets (2017).
[9] Nikolaj Moll, e. a.: Quantum optimization using variational algorithms on near-term quantum devices, *Quantum Science and Technology*, Vol. 3, No. 3, p. 030503 (online), DOI: 10.1088/2058-9565/aab822 (2018).
[10] Barkoutsos, P. K., Nannicini, G., Robert, A., Tavernelli, I. and Woerner, S.: Improving Variational Quantum Optimization using CVaR (2019).
[11] Lucas, A.: Ising formulations of many NP problems, pp. 1–27 (online), DOI: 10.3389/fphy.2014.00005 (2013).
[12] Qiskit: Qiskit: An Open-source Framework for Quantum Computing, , available from (<https://www.qiskit.org/>) (accessed 09/16/2020).
[13] Nielsen, M. A. and Chuang, I. L.: *Quantum Computation and Quantum Information: 10th Anniversary Edition*, Cambridge University Press (2010).
[14] Nelder, J. A. and Mead, R.: A Simplex Method for Function Minimization, *The Computer Journal*, Vol. 7, No. 4, pp. 308–313 (1965).
[15] Powell, M. J. D.: An efficient method for finding the minimum of a function of several variables without calculating derivatives, *The Computer Journal*, Vol. 7, No. 2, pp. 155–162 (1964).
[16] Spall, J. C.: Multivariate stochastic approximation using a simultaneous perturbation gradient approximation, *IEEE Transactions on Automatic Control*, Vol. 37, No. 3, pp. 332–341 (1992).
[17] Hestenes, M. R. and Stiefel, E.: Methods of conjugate gradients for solving linear systems (1952).
[18] Nakanishi, K. M., Fujii, K. and Todo, S.: Sequential minimal optimization for quantum-classical hybrid algorithms (2019).
[19] Diker, F.: Deterministic construction of arbitrary W states with quadratically increasing number of two-qubit gates (2016).
[20] Qiskit-Community: W State 1 - Multi-Qubit Systems, , available from (https://github.com/Qiskit/qiskit-community-tutorials/blob/master/awards/teach_me_qiskit_2018/w_state/W%20State%201%20-%20Multi-Qubit%20Systems.ipynb) (accessed 04/30/2020).
[21] Preskill, J.: Quantum Computing in the NISQ era and beyond, *Quantum*, Vol. 2, p. 79 (2018).