Divide-and-Conquer Quantum Algorithm for Hybrid *de novo* Genome Assembly of Short and Long Reads

Jing-Kai Fang,[1,*] Yue-Feng Lin,[2,*] Jun-Han Huang,[1,*] Yibo Chen,[1] Gao-Ming Fan,[2] Yuhui Sun,[1] Guanru Feng,[2] Cong Guo,[2] Tiejun Meng,[2] Yong Zhang,[1,3,4] Xun Xu,[1,3,†] Jingen Xiang,[2,‡] and Yuxiang Li[1,3,4,§]

[1]*BGI Research, Shenzhen 518083, China*
[2]*Shenzhen SpinQ Technology Co., Ltd, Shenzhen 518048, China*
[3]*BGI Research, Wuhan 430047, China*
[4]*Guangdong Bigdata Engineering Technology Research Center for Life Sciences, BGI Research, Shenzhen 518083, China*

(Received 27 October 2023; accepted 26 March 2024; published 23 April 2024)

Computational biology holds immense promise as a domain that can leverage quantum advantages due to its involvement in a wide range of challenging computational tasks. Researchers have recently explored the applications of quantum computing in genome assembly implementation. However, the issue of repetitive sequences remains unresolved. In this paper, we propose a hybrid assembly quantum algorithm using high-accuracy short reads and error-prone long reads to deal with sequencing errors and repetitive sequences. The proposed algorithm builds upon the variational quantum eigensolver and utilizes divide-and-conquer strategies to approximate the ground state of larger Hamiltonian while conserving quantum resources. Using simulations of ten-qubit quantum computers, we address problems as large as 140 qubits, yielding optimal assembly results. The convergence speed is significantly improved via the problem-inspired *Ansatz* based on the known information about the assembly problem. In addition, we qualitatively verify that entanglement within quantum circuits may accelerate the assembly path optimization.

DOI: 10.1103/PRXLife.2.023006

I. INTRODUCTION

DNA sequencing technology has dramatically transformed the fields of biology and medicine in the past few decades. This revolutionary tool allows researchers to decode the genetic blueprints of living organisms, leading to breakthroughs such as early cancer diagnosis and detection of inherited diseases. The throughput and speed of DNA sequencing have increased exponentially over the years, surpassing even Moore's law that predicts the growth of computational power [1]. The Sanger sequencing platform, back in 1987, was only capable of sequencing approximately 1000 nucleotides per day [1–3]. This limitation rendered the sequencing and assembly of the complete human genome by the Human Genome Project a 13-year endeavor. In contrast, present technologies allow for the sequencing of an entire human genome within mere hours. However, sequencing is just one facet of the challenge. Genome reconstruction is an indispensable subsequent step to pave the way for comprehensive studies. With the rapid development of sequencing technology, there has been a corresponding surge in data volume that amplifies computational demands, prompting the evolution of diverse assembly algorithms.

One of the early assembly algorithms is the overlap-layout-consensus (OLC) algorithm [4–8]. The OLC algorithm transforms the genome assembly problem into a graph problem where each vertex represents a read and edges represent overlaps among all reads, aiming to find a Hamiltonian path in the graph, which is equivalent to the traveling salesman problem, a famous NP-hard combinatorial optimization problem [9]. The OLC approach has been widely used in various genome assemblers for long DNA fragments. Another different algorithm, based on de Bruijn graphs, has been applied in a series of assemblers, called Eulerian assemblers [10], such as SOAPdenovo [11], Spades [12], and ABySS [13]. In this algorithm, vertices in the graph no longer represent reads, but rather k-mers generated from the reads, and the assembly process can be reformulated as an Eulerian path problem. Unlike the intractable complexity of the OLC algorithm, the de Bruijn graph assemblers are suitable for the assembly based on next-generation sequencing data that produces a vast number of high-accuracy short reads.

The next-generation sequencing has been extensively used due to its accuracy and low cost. Nevertheless, the limited length of short reads makes it difficult to complete the assembly of highly repetitive complex regions in genomes. The advent of the third-generation sequencing platform such as Oxford Nanopore Technology [14] and Pacific Biosciences [15–17] has enabled the generation of long reads with a length

[*] These authors contributed equally to this work.
[†] xuxun@genomics.cn
[‡] jxiang@spinq.cn
[§] liyuxiang@genomics.cn

of more than 10 kbp, which can span multiple genomic repeats and improve the contiguity of assembly. However, these long reads also have a high error rate and are not cost-friendly, limiting their applicability for large-scale genome projects.

To overcome these challenges, hybrid assembly approaches combining short and long reads have been developed. These methods have been shown to present unique benefits by complementing the strengths and weaknesses of each read type. For example, Koren *et al.* [18] and Au *et al.* [19] applied short reads to correct long reads and then assembled the corrected long reads. These approaches require high coverage and usually huge computing resources. Deshpande *et al.* [20], Antipov *et al.* [21], and Wick *et al.* [22] used another approach that assembled the short reads first to generate precise contigs and employed long reads for scaffolding. The hybrid assembly provides an alternative cost-effective way since it requires fewer long reads than long-read-only methods. Some recent studies [23,24] have shown that hybrid assembly is superior to long-read-only methods in terms of correctness, contiguity, and completeness. Nevertheless, hybrid assembly is still computationally demanding, requiring powerful computational resources and storage capacity, especially for large and complex genomes.

In contrast, concerns have been raised regarding the ability of supercomputers to handle the explosive growth of sequencing data, particularly as classical computers are already approaching their physical limits. Given these limitations of classical computing, it is crucial to explore alternative computational paradigms. Quantum computers promise a new solution to this challenge, especially in terms of enhanced computational efficiency and capability to handle complex optimization problems [25,26]. Numerous quantum algorithms have been proposed and demonstrated, promising quantum advantages. Some of them have been applied to various bioinformatics problems, such as *de novo* assembly [27–29], sequence alignment [30,31], protein folding [32–34], and phylogenetic tree inference [35], which inspired us to achieve hybrid assembly using quantum computing.

The previous studies [27–29] proposed several OLC-based assembly quantum algorithms, including quantum annealing and variational quantum algorithms, and provided a proof of the application of quantum computing to the assembly problem, with sequencing error also taken into account in Ref. [29]. However, there are inevitable repeats in DNA sequences and they cannot be solved before assembly. The classical assemblers solve the problem by indexing reads near the repeats and seeking out the reads that contain the entire repetitive region. However, it is worth mentioning that none of the quantum algorithms have addressed the issue of repetitive sequences.

In this paper, we propose a quantum algorithm for hybrid *de novo* genome assembly using short and long reads. In our algorithm, short reads and long reads are utilized to deal with the assembly path conflicts caused by sequencing errors or repetitive sequences. Concretely, we first map the hybrid *de novo* assembly problem with repeats and sequencing errors onto a combinatorial optimization problem. Then we utilize the variational quantum eigensolver (VQE) framework and introduce the divide-and-conquer strategy to obtain optimal assembly sequences with fewer quantum resources in the noisy intermediate-scale quantum (NISQ) era, characterized by quantum devices that are susceptible to errors but can still provide computational advantages [36]. Furthermore, a problem-inspired *Ansatz* is proposed to significantly improve the hybrid assembly quantum algorithm.

II. RESULTS

A. Hybrid assembly quantum algorithm

The workflow of the hybrid assembly quantum algorithm is presented in Fig. 1. The sequencing data of short reads and long reads are processed on quantum computers or quantum simulators after an encoding step. This generates a state that can be decoded to obtain the assembly sequences. The details are shown below.

We first propose a mapping model of the hybrid *de novo* assembly problem involving repetitive sequences and sequencing errors. Due to the path conflicts introduced by repetitive sequences and sequencing errors, it is not feasible to directly determine an Eulerian path within the de Bruijn graph [10] that faithfully reconstructs the original sequences. To overcome this obstacle, we introduce long-read data of third-generation sequencing and encode the information into the weight of the cost function. Then we can construct the cost function of the hybrid *de novo* assembly problem.

First, we need to encode the de Bruijn graph. Let binary variables $x_{i,p} \in \{0, 1\}$ be the occupation number of every node at each position, where $i \in \{1, 2, \dots, n\}$ represents the ith node and $p \in \{1, 2, \dots, p_{\max}\}$ represents the position in the assembly path. Here $x_{i,p} = 1$ means that the ith node is chosen at position p in the path. The possible assembly path can be expressed as a quantum state $|\Psi\rangle = |x_{1,1}\, x_{2,1} \cdots x_{i,p} \cdots x_{n,p_{\max}}\rangle$. Note that for a graph with n nodes, the maximum of p must be larger than n if there are repeats in DNA sequences. Conservatively, we can take $p_{\max} = 2n$. Then the cost function of the hybrid *de novo* assembly problem is formulated as

$$\begin{aligned}
&\text{minimize} \quad \sum_{\Delta} \sum_{p} \sum_{i,j} \omega_{\Delta,i,j} x_{i,p} x_{j,p+\Delta} \\
&\text{subject to} \quad \sum_{i} x_{i,1} = 1, \\
&\sum_{i} x_{i,p} \leqslant 1 \,\forall p \\
&\text{if} \ \sum_{i} x_{i,p-1} = 0 : \sum_{i} x_{i,p} = 0 \,\forall p, \\
&x_{i,p} \in \{0, 1\},
\end{aligned} \tag{1}$$

where $\omega_{\Delta,i,j}$ is the weight of k-mers i and j separated by an internucleotide distance Δ. For the weight of two adjacent nodes, we set $\omega_{1,i,j} = -\alpha$ to indicate that two k-mers are correctly sequenced and $\omega_{1,i,j} = -\beta$ if the frequency of k-mers is less than 10% (considered as sequencing errors), where $0 < \beta \ll \alpha$. For nodes that are not connected, a large positive weight γ is given as a penalty. Additionally, by introducing the long-read data from third-generation sequencing as shown in Fig. 1, we can determine whether any two nodes are separated by a distance Δ The long-range links between the nodes

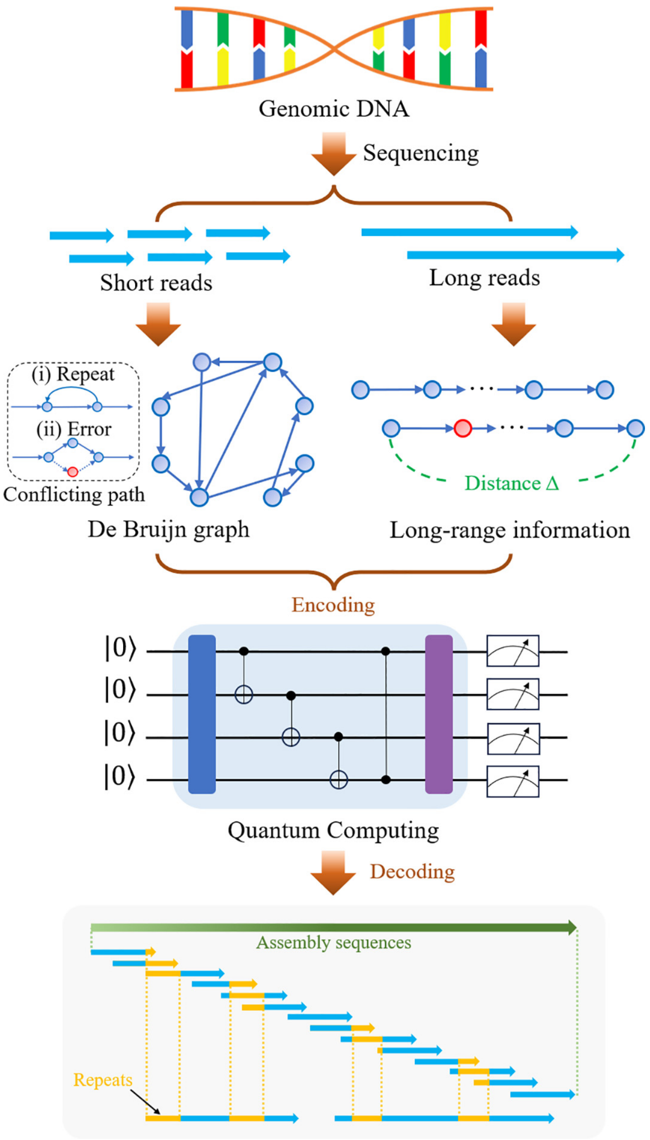

FIG. 1. Schematic overview of the quantum algorithm for hybrid *de novo* genome assembly. The short reads are used to construct the de Bruijn graphs, while the long reads are used to provide long-range linking between nodes. Here the nodes represent k-mers and the red ones mean low-frequency k-mers from sequencing errors. Connections between nodes are based on the common $k-1$ suffix and prefix. Inevitable path conflicts are caused by sequencing errors or repetitive sequences. The short-read and long-read data are encoded together into a optimization model solvable on a quantum device. The results are then decoded to obtain the assembly sequences.

depend on the length of long-read sequences that generally exceed 10 kbp. If two nodes are identified with a distance of Δ in the long read, we assign the corresponding term with a negative weight $\omega_{\Delta,i,j} = -\eta$ to contribute to the cost function. Otherwise we set it equal to zero. In order to solve the problem of repetitive sequences, the distance Δ must be greater than the sum of the lengths of the repeats, and the absolute value of $\omega_{\Delta,i,j}$ should be as large as possible compared to the total weight of the cyclic paths caused by repetitive sequences.

Next we construct the Hamiltonian of the hybrid *de novo* assembly problem according to the cost function in Eq. (2). The objective term can be directly written as

$$H_1 = \sum_{\Delta}\sum_{p}\sum_{i,j} \omega_{\Delta,i,j} x_{i,p} x_{j,p+\Delta}. \tag{2}$$

Then we consider the constraints of the cost function. The first constraint term of Eq. (2) ensures that the first position ($p = 1$) of the assembly path must be one and only one node. Thus, the Hamiltonian of this constraint term can be formulated as

$$H_2 = A\left(1 - \sum_{i} x_{i,1}\right)^2, \tag{3}$$

where the coefficient A is the penalty parameter. The second and third constraint terms ensure the contiguity of the assembly sequence, that is, the assembly path is unique and continuous. Mathematically, for all p, the term $\sum_i x_{i,p}\sum_j x_{j,p+1}(\sum_k x_{k,p+1} - 1) = 0$ ensures an unforked path and the term $(1-\sum_i x_{i,p})\sum_j x_{j,p+1} = 0$ prevents the path from being discontinuous. Thus, the Hamiltonian of the second and third constraint terms can be combined as

$$H_3 = B\sum_{p}\left(\sum_{i} x_{i,p}\sum_{j} x_{j,p+1}\sum_{k} x_{k,p+1} - 2\sum_{i,j} x_{i,p}x_{j,p+1} + \sum_{j} x_{j,p+1}\right). \tag{4}$$

Here the parameters must satisfy $0 < B < A$ to ensure that the contribution of all penalty terms is not less than zero. Finally, the whole Hamiltonian of the hybrid *de novo* assembly problem is expressed as

$$H_C = H_1 + H_2 + H_3. \tag{5}$$

Thus, the optimal assembly path with a cycle caused by repeats in the de Bruijn graph can be effectively represented as a continuous and unforked path. According to this Hamiltonian, parametrized quantum circuits are constructed and applied in the VQE approach to calculate the minimum eigenvalue and the corresponding eigenvector of H_C that exactly matches the optimal assembly path (Sec. IV). Eventually, a decoding process is added to yield the desired assembly sequence as shown in Fig. 1.

B. Numerical simulations

The proposed hybrid assembly quantum algorithm is implemented in SpinQit with Torch simulator. We analyze the convergence properties of various quantum circuits, including two distinct types of hardware-efficient *Ansätze* (HEA) as well as a problem-inspired *Ansatz* based on the known information about the assembly problem. The experiments are conducted on Dell equipped with 2.90 GHz Intel Core i7-10700 and 16.0 GB memory and operated under the Windows 11 operating system.

1. Toy model

We illustrate the hybrid assembly quantum algorithm for the toy model as shown in Fig. 2. The nodes in the toy model

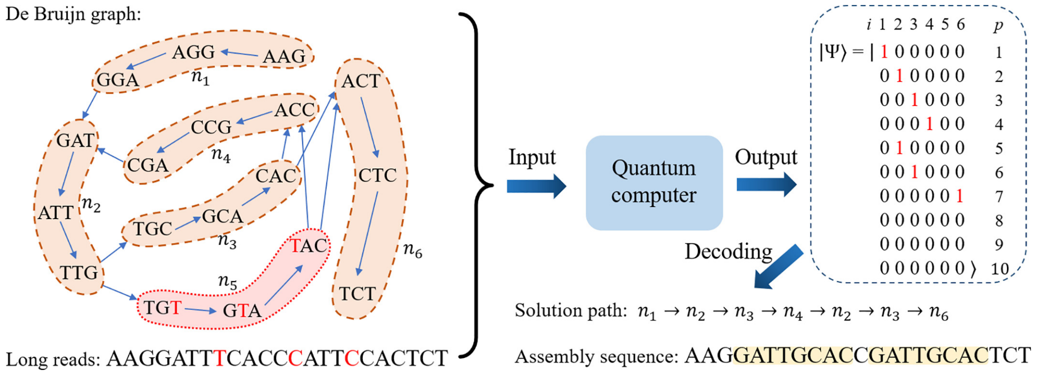

FIG. 2. Schematic diagram of the toy model. Both short-read and long-read sequencing data are used, where the red characters represent sequencing errors. To optimize qubit utilization and streamline the model, a strategy is implemented where three consecutive k-mers are consolidated into a single node. The nodes in the red dotted circle are low-frequency k-mers from sequencing errors. The connection of the nodes is established based on the shared $k-1$ suffix and prefix. The nodes n_2 and n_3 represent repeat regions and the node n_5 represents sequencing errors. The toy model is input into a quantum computer for simulation, resulting in the generation of the final quantum state of the binary string $|\Psi\rangle$. The optimal assembly sequence is then obtained through the process of decoding the binary string.

represent k-mers or a group of k-mers generated by splitting short reads. Note that we treat three consecutive k-mers as a single node in the de Bruijn graph to further simplify the model. Long-read data are introduced as weights $\omega_{\Delta,i,j}$ to aid assembly.

In this toy model, we assume that the correct assembly path is $n_1 \to n_2 \to n_3 \to n_4 \to n_2 \to n_3 \to n_6$. Here we set $p_{\max} = 10$ to allow more wrong assembly solutions to verify the reliability of our approach. Moreover, we utilize the long-range linking information to ensure that the loop $n_2 \to n_3 \to n_4 \to n_2$ does not occur twice by setting $\alpha = 2$, $\beta = 1$, $\gamma = 20$, and $\eta = 10$. Considering the quantum resources required are quadratic to the problem size, a divide-and-conquer strategy is designed in the VQE framework (Sec. IV). Thus, the VQE algorithm is used to obtain an optimal assembly result by simulations of six-qubit quantum computers. In this case, the exact loss value is -27 and it is verified that the corresponding assembly sequence AAGGATTGCACCGATTGCACTCT is correctly extracted.

2. *Entangling capability*

In practice, the final results of the hybrid assembly significantly rely on the expressibility and entangling capability of *Ansätze*. Hardware-efficient *Ansätze* [37] are one of the widely used parametrized quantum circuits in the VQE algorithm, as they are designed based on several factors of the near-term quantum devices, such as qubit connectivity and restricted gate sets. Normally, they are constructed by single-qubit gates with parameters that can be optimized, including R_x, R_y, and R_z gates as well as two-qubit entangling gates, such as controlled-NOT (CNOT), controlled-Z (CZ), and iSWAP gates.

In this work, we simulate the Hamiltonian of the toy model by the VQE approach with two different hardware-efficient *Ansätze*. Our goal is to examine the performance of optimization convergence with and without quantum entanglement. In Figs. 3(a) and 3(b), the former, called HEA1, consists of the R_y gate, R_z gate, and CZ gate; the latter, called HEA2, is constructed with single-qubit rotations R_y and R_z only. In the experiment, we set the depth of quantum circuit D_e to be 2 and randomly generate 100 sets of initial parameters for HEA1 and HEA2. As illustrated in Figs. 4(a) and 4(b), our findings indicate a significant correlation between the initial parameters and the convergence speed of various *Ansätze*. More importantly, we show that the average convergence speed of HEA1 is faster than that of HEA2. In Fig. 4(c), the average iteration number for the convergence of HEA1 is 348, while it is 393 for HEA2. This indicates that the entangling capability of *Ansätze* may have a positive impact on the convergence of the hybrid *de novo* assembly quantum algorithm using short and long reads.

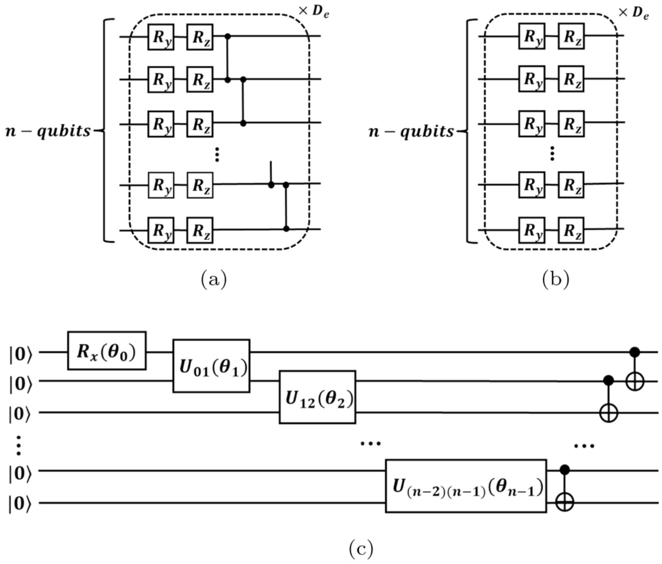

FIG. 3. Variational *Ansätze* for the VQE algorithm. (a) HEA1 implemented by R_y, R_z, and CZ gates. (b) HEA2 implemented by R_y and R_z gates. The quantum circuits in the dashed boxes repeat D_e times. (c) Problem-inspired *Ansatz* based on the assembly problem (Sec. IV).

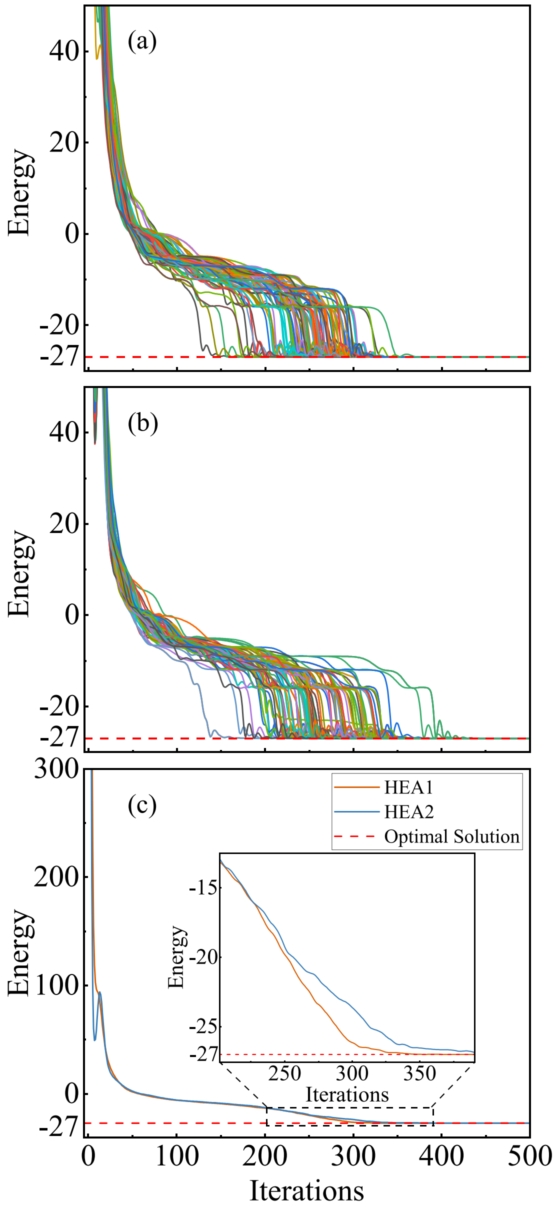

FIG. 4. Simulation results of HEA. Convergence of (a) HEA1 and (b) HEA2 is shown with 100 sets of random initial parameters. (c) Comparison of the average convergence between HEA1 and HEA2.

3. Problem-inspired Ansatz

Unlike HEA, the problem-inspired *Ansatz* is designed with unique attributes of assembly problems. In the divide-and-conquer quantum algorithm, because there is at most one variable of 1 in the binary string $|x_{1,p}\cdots x_{i,p}\cdots x_{n,p}\rangle$, the trial wave function of n qubits can be restricted to $|\psi\rangle = \sum_k \lambda_k|k\rangle$, $k\in\{0, 2^0, 2^1, \ldots, 2^{n-1}\}$, where n is the number of nodes in de Bruijn graphs. The problem-inspired *Ansatz* is constructed to prepare the candidate n-qubit trial wave function $|\psi\rangle$ [more details can be found in Fig. 3(c) and Sec. IV].

As in the case of HEA, we also randomly generate 100 sets of initial parameters for the proposed problem-inspired *Ansatz*. As shown in Fig. 5(a), we show that the convergence speed of the problem-inspired *Ansatz* is significantly faster than that of HEA. Furthermore, the average iteration number required for the convergence of the problem-inspired *Ansatz* is 168 in Fig. 5(b). The results verify that the problem-inspired *Ansatz* can accelerate the convergence speed of the hybrid assembly quantum algorithm due to a reduced search space and fewer optimization parameters. Table I presents the quantum resources required for the three quantum circuits. In contrast to HEA, the problem-inspired *Ansatz* employs a fixed circuit depth, yielding a streamlined computational process and fewer quantum gate requirements.

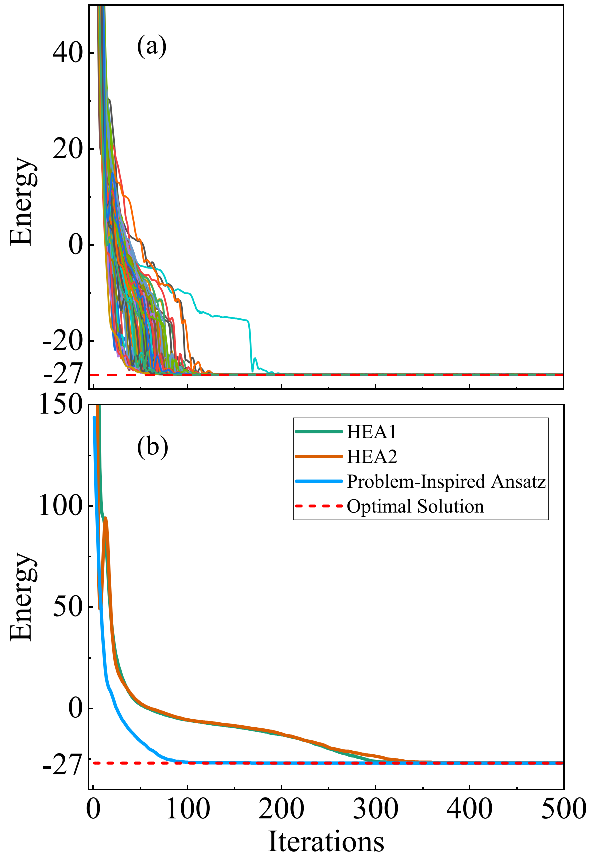

FIG. 5. Simulation results of the problem-inspired *Ansatz*. (a) Evaluation value for 100 groups of random initial parameters with the problem-inspired *Ansatz*. (b) Comparison of the average convergence of HEA1, HEA2, and the problem-inspired *Ansatz*.

TABLE I. Cost estimates for the problem-inspired *Ansatz* and HEA. Note that $N = p_{\max}n$, where n denotes the number of qubits, i.e., the number of nodes in the de Bruijn graphs. The depth of the problem-inspired *Ansatz* is fixed, but the blocks of variational quantum circuits in HEA are repeated D_e times.

Ansatz	Number of parameters	Single-qubit gates	Entangling gates
Problem-inspired	N	$2N - p_{\max}$	$2N - 2p_{\max}$
HEA1	$2D_eN$	$2D_eN$	$D_e(N - p_{\max})$
HEA2	$2D_eN$	$2D_eN$	0

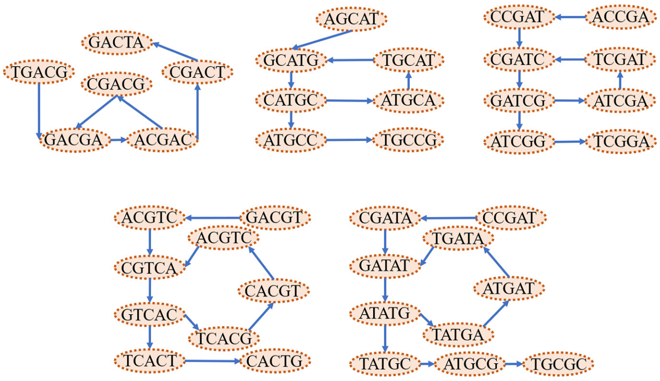

FIG. 6. The de Bruijn graphs for five different sizes of assembly problems, from six nodes to ten nodes, calculated by the problem-inspired *Ansatz*.

4. Problem sizes of hundreds of qubits

Further, we evaluate the performance of the problem-inspired *Ansatz* for larger problems. Five different sizes of assembly problems, ranging from six nodes to ten nodes, are illustrated in Fig. 6 with the numbers of qubits required for the corresponding Hamiltonians being 60, 77, 96, 117, and 140, respectively. Note that a divide-and-conquer strategy is employed in the VQE framework in order to calculate on quantum simulators. This approach enables more efficient calculations on quantum simulators by breaking down larger problems into more manageable subproblems. In addition, we restrict the initial parameters of the problem-inspired *Ansatz* to the range $[0.3\pi, 0.5\pi]$, as opposed to the more common range of $[0, \pi]$. This restriction leads to a significant improvement in both the convergence speed and success rate of the optimization process. As evidenced by Fig. 7(a), the ground-state energy of each Hamiltonian and the assembly path for each problem size are correctly optimized, serving as a validation of the problem-inspired *Ansatz* for larger problems.

Finally, we measure the impact of problem sizes on computational cost by evaluating the average iteration number for convergence and R_{99} [27] of the five different sizes. Here the parameter R_{99} represents the number of experiments required to find the optimal assembly sequence (ground state) with a probability of 99%. The algebraic expression of R_{99} is defined as

$$R_{99} = \frac{\log(1 - 0.99)}{\log(1 - P)}, \tag{6}$$

where P is the estimated success probability of each simulation. As shown in Fig. 7(b), both the average iteration number for convergence and R_{99} increase linearly with the size of the problem. The computational cost is directly proportional to the number of qubits involved in the hybrid *de novo* assembly problem. This suggests that the proposed problem-inspired *Ansatz* scales reasonably well for larger problem sizes, which is optimistic for the large-scale practical applications of genome assembly using quantum computing. Therefore, the rapid advancement in quantum computing hardware and further improved divide-and-conquer quantum algorithms can facilitate the demonstration of quantum utility on the genome assembly problem.

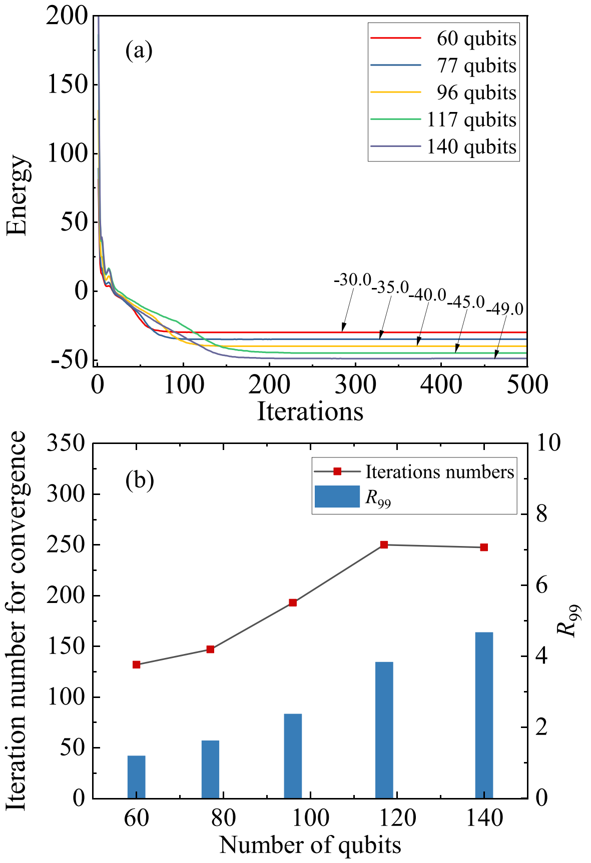

FIG. 7. Simulation results of the problem-inspired *Ansatz* for five different problem sizes. (a) Average convergence of different problem sizes. (b) Iteration number for convergence and the number of experiments required to find the optimal assembly results with a probability of 99%, called R_{99}.

III. DISCUSSION

In this work, we propose a divide-and-conquer quantum algorithm for hybrid *de novo* genome assembly of short and long reads. We demonstrate that the optimized assembly sequences can be obtained by decoding quantum computing results for the ground states based on the VQE framework. Furthermore, we adopt divide-and-conquer strategies to extract the optimal assembly results with problem sizes up to hundreds of qubits, by employing simulations on few-qubit quantum computers.

The hybrid assembly quantum algorithm offers several advantages. First, it integrates short reads from next-generation sequencing technology and long reads from third-generation sequencing technology to address assembly path conflicts. This hybrid sequencing strategy can leverage the complementary strengths of both sequencing platforms to improve the correctness and completeness of genome assembly and annotation. Second, it has the potential to significantly reduce the computational expense of simulating intricate quantum systems, which could enable practical applications of quantum computing in life sciences during the NISQ era. The *de novo* genome assembly problem is solved by mapping the DNA sequences into graph models. However, encoding a binary variable with multidimensional information requires a number of qubits that scales as $O(n^2)$, where n denotes the number of nodes in de Bruijn graphs. This presents a major challenge in optimizing assembly sequences when using the original VQE without any preprocessing, especially on NISQ devices. Consequently, in light of quantum resources, the proposed algorithm utilizes a divide-and-conquer strategy to alleviate the requirements of logical qubits, that is, to scale as $O(n)$. Third, our research findings indicate that quantum entanglement plays a constructive role in optimizing the genome assembly problem. Finally, our proposed problem-inspired *Ansatz* significantly accelerates the optimization of assembly paths in the hybrid assembly quantum algorithm.

However, some issues may emerge in practical applications of the hybrid assembly quantum algorithm and deserve further investigation. Above all, a considerable number of qubits is required in practical assembly tasks even with divide-and-conquer strategies, which poses a huge challenge for near-term quantum devices. In theory, the count of k-mers obtained through sequencing, disregarding repeats, aligns with the formula $L - k + 1$, where L represents the total genome size. This results in a substantial number, exemplified by the approximately 3×10^9 base pairs in the human genome. Traditional assemblers are incapable of concurrently assembling such extensive sequences. Typically, these assemblers compile k-mers into shorter contigs, which are then linked to generate longer scaffolds. Presently, the longest contig achievable through next-generation sequencing reaches approximately the order of 10^6 [24]. Hence, for our algorithm to match the performance of classical assemblers, it would necessitate approximately 10^6 qubits, which is a challenging feat in the near term. Nonetheless, addressing this challenge is not overly arduous; here we propose two distinct solutions. First, we can partition the task into multiple subtasks, assembling them into short contigs before merging them into longer contigs. An intuitive approach involves amalgamating N consecutive k-mers into a single node, with the implementation necessitating consideration for cases where the number of k-mers is not divisible by N. Second, an alternate methodology frequently employed involves leveraging quantum superposition states for information encoding, consequently reducing the requisite quantum bit resources, representing the focus of our forthcoming endeavors. Furthermore, given that our algorithm adeptly handles repeat sequences during assembly, we anticipate it will demonstrate enhanced performance in assembling duplicate genes located in highly repetitive regions.

Another issue that can be considered is that, as the problem size increases, the parameter optimization of variational quantum algorithms becomes unpredictable and intractable. Fortunately, we find that narrowing the range of initial parameters can significantly improve the parameter optimization and the probability of obtaining the optimal assembly results. This also provides insights into solving other combinatorial optimization problems that are prone to local minima or barren plateaus as the number of qubits increases. In the near future, improving variational quantum algorithms and exploring quantum advantages will be primary goals in the field of quantum computing applications, especially in bioinformatics.

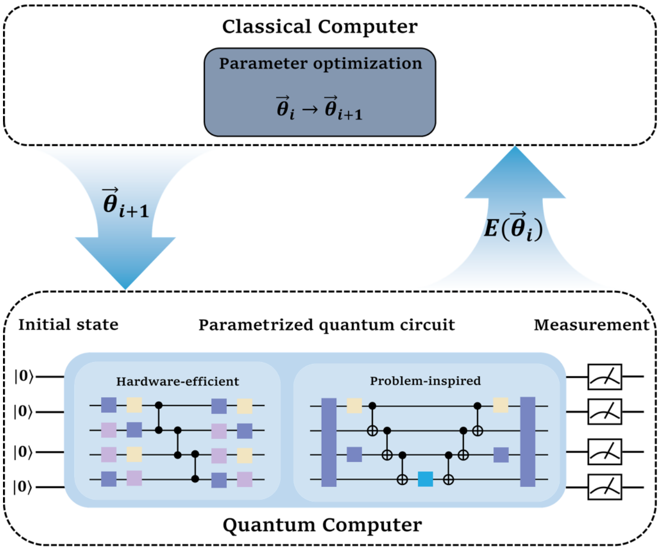

FIG. 8. Scheme of the VQE method. The VQE is a hybrid quantum-classical algorithm, where quantum processing units are used to prepare the trial wave function by parametrized quantum circuits and measure the corresponding expectation value of the Hamiltonian and classical processing units are used to update the parameters.

IV. METHODS

A. Variational quantum eigensolver

The VQE is a hybrid quantum-classical algorithm for solving approximate minimum eigenvalues and eigenvectors of the Hamiltonian. It relies on the Rayleigh-Ritz variational principle as follows:

$$E = \langle\psi(\vec{\theta})|H|\psi(\vec{\theta})\rangle \geqslant E_0 = \langle\Psi|H|\Psi\rangle. \tag{7}$$

Here $|\psi(\vec{\theta})\rangle$ is the trial wave function prepared to extract the optimal assembly result by applying a parametrized quantum circuit to the initial state $|\psi_0\rangle$, E is the expectation value of the Hamiltonian with $|\psi(\vec{\theta})\rangle$, and E_0 and $|\Psi\rangle$ are the exact minimum eigenvalue and the corresponding eigenvector of the Hamiltonian H, respectively.

As shown in Fig. 8, the procedure of the VQE is implemented with two devices, quantum and classical computers. Typically, quantum computers are used to implement parametrized quantum circuits constructed of a set of quantum gates with adjustable parameters in order to prepare the trial wave function and measure the corresponding expectation value of the Hamiltonian. Meanwhile, the classical computer optimizes the parameters of quantum gates and transmits them to the quantum computer to initiate the next iteration. This iterative process continues until a predefined convergence criterion is satisfied. Explicitly, we introduce the key components of the VQE algorithm below.

1. Hamiltonian construction

For the VQE, the first step is to construct the Hamiltonian for describing the various attributes of the systems, which are in general chemical molecules [38], combinatorial optimization problems [39], many-body physics [40], and so on. Since quantum computers only measure observables that can be expressed as linear combinations of Pauli operators $\{I, X, Y, Z\}$, the Hamiltonian needs to be transformed to the corresponding terms.

2. Parametrized quantum circuits

The parametrized quantum circuit, also known as an *Ansatz*, is designed to prepare the trial wave function $|\psi(\vec{\theta})\rangle$. Undeniably, since the VQE was first proposed by Peruzzo *et al.* [41], the research on constructing a good *Ansatz* for specific problems has attracted considerable interest. Now, various forms of parametrized quantum circuits have been proposed, such as hardware-efficient *Ansätze* [37], problem-inspired *Ansätze* [38,39,42–47], and variable-structure *Ansätze* [48,49]. Generally speaking, a good *Ansatz* includes the following two attributes: (i) Sufficient expressibility ensures that the trial wave function contains the ground-state wave function and (ii) the parameter search space is reduced as much as possible to speed up the convergence of the VQE algorithm. In our work, we design a problem-inspired *Ansatz* to reduce the search space and boost the converging process for hybrid *de novo* genome assembly.

3. Parameter optimization

After transforming the Hamiltonian and preparing the trial wave function, the expectation value of the Hamiltonian can be measured by a quantum computer. The measurement results are subsequently transmitted to the classical computer to update the parameters in the quantum circuits for the next iteration. Parameter optimization is necessary for any variational algorithms to obtain an approximate solution. The parameters of quantum circuits will be updated iteratively with some efficient optimization strategies, such as gradient descent methods [50,51], as well as gradient-free methods [52,53], until convergence.

B. Implementation

As mentioned above, the value of the binary variable $x_{i,p}$ indicates whether or not to choose the node i at the position p in the assembly path. In addition, the possible assembly path can be expressed as a quantum state $|\Psi\rangle = |x_{1,1}\, x_{2,1} \cdots x_{i,p} \cdots x_{n,p_{\max}}\rangle$.

1. Qubit mapping

To construct the qubit Hamiltonian of the hybrid *de novo* assembly problem from Eq. (5), the common approach is to map the binary variable $x_{i,p}$ to the Pauli Z and identity matrix I as

$$x_{i,p} \rightarrow \frac{I - Z_{i,p}}{2}, \tag{8}$$

where $Z_{i,p}$ is a Pauli Z acting on the $[p(n-1)+i]$th qubit. Correspondingly, Eq. (5) can be rewritten as

$$H_q = H_{1q} + H_{2q} + H_{3q}. \tag{9}$$

It is apparent that the dimension of the Hamiltonian system represented by H_q is 2^N, where $N = np_{\max}$ denotes the required number of qubits.

In the NISQ era, it is challenging to employ the original VQE algorithm to simulate and estimate the approximate minimum eigenvalue of the Hamiltonian H_q without any preprocessing. Consequently, in light of the original VQE algorithm, we propose a divide-and-conquer strategy to alleviate the requirements of logical qubits. To be specific, we leverage Hamiltonian decomposition and state decomposition to estimate the approximate ground-state energy of an N-qubit Hamiltonian on $p_{\max}$ quantum processing units with n qubits.

2. Hamiltonian decomposition

Once qubit mapping is performed, the Hamiltonian H_q can be expressed in the form

$$H_q = \sum_t c_t H_t, \tag{10}$$

where H_t denotes a component of the Hamiltonian H_q. Here H_t can be constructed using tensor products of the Pauli Z operator and the identity operator I. Its dimension is 2^N. We further decompose H_t into $p_{\max}$ Hamiltonians denoted by $\hat{H}_t^{(\omega)}$,

$$H_t = \bigotimes_{\omega=1}^{p_{\max}} \hat{H}_t^{(\omega)}, \tag{11}$$

where the dimension of each Hamiltonian $H_t^{(\omega)}$ is 2^n. Therefore, the Hamiltonian H_q can be rewritten as

$$H_q = \sum_t c_t \bigotimes_{\omega=1}^{p_{\max}} \hat{H}_t^{(\omega)}. \tag{12}$$

3. State decomposition

The quantum state of the assembly path can be expressed using the tensor product notation

$$|\Psi\rangle = \bigotimes_{\omega=1}^{p_{\max}} |x_{1,\omega} \cdots x_{i,\omega} \cdots x_{n,\omega}\rangle. \tag{13}$$

In the VQE, each quantum state $|x_{1,\omega} \cdots x_{i,\omega} \cdots x_{n,\omega}\rangle$ can be obtained from a trial wave function $|\psi^{(\omega)}(\theta)\rangle$,

$$|\psi^{(\omega)}(\theta)\rangle = \sum_{k=0}^{2^n-1} \lambda_k^{(\omega)}(\theta)|k\rangle, \tag{14}$$

where $|k\rangle$ are the n-qubit computational bases of $|\psi^{(\omega)}(\theta)\rangle$ and the parameters $\lambda_k^{(\omega)}(\theta)$ satisfy $\sum_{k=0}^{2^n-1}[\lambda_k^{(\omega)}(\theta)]^2 = 1$. In general, the optimal assembly path corresponds to a nonsuperposed quantum state. Therefore, the trial wave function can be reformulated as follows:

$$|\psi(\vec{\theta})\rangle = \bigotimes_{\omega=1}^{p_{\max}} \sum_{k=0}^{2^n-1} \lambda_k^{(\omega)}(\vec{\theta})|k\rangle. \tag{15}$$

4. Ground-state calculation

By decomposing the Hamiltonian and state, leveraging the VQE algorithm, we can obtain an approximate ground state and its corresponding eigenvector of the large quantum system. From Eq. (7), we aim to minimize the expectation value

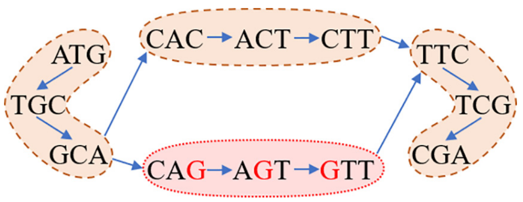

FIG. 9. Schematic diagram of the toy model with four nodes.

with the parameter $\vec{\theta}$,

$$\min_{\vec{\theta}} E(\vec{\theta}) = \min_{\vec{\theta}} \langle \psi(\vec{\theta})|H_q|\psi(\vec{\theta})\rangle$$
$$= \min_{\vec{\theta}} \sum_t c_t \prod_{\omega=1}^{p_{\max}} \sum_{k=0}^{2^n-1} \left[\lambda_k^{(\omega)}(\vec{\theta})\right]^2 \langle k|\hat{H}_t^{(\omega)}|k\rangle. \quad (16)$$

By utilizing the proposed approach, we can produce N-qubit quantum states, where there is at most one qubit at state $|1\rangle$, and estimate the approximate ground-state energy of a 2^N-dimensional Hamiltonian using $p_{\max}$ n-qubit quantum processing units.

5. Benchmarking of divide-and-conquer strategy

Here we conduct simulations in the model with four nodes as illustrated in Fig. 9 and benchmark the performance of two algorithms: the VQE without the divide-and-conquer strategy and the distributed VQE (DVQE) with the divide-and-conquer strategy. The metrics evaluated include the average running time per iteration, the average iteration number, and the total time for convergence. As depicted in Table II, the disparity in the average iteration number for convergence between the VQE and DVQE is marginal, with the DVQE exhibiting a slightly superior performance. However, the DVQE demonstrates a significant advantage over the VQE in terms of iteration time. This outcome aligns with expectations given the DVQE's strategy of employing fewer qubits.

C. Problem-inspired *Ansatz*

The constraints of the optimization problem offer an idea to exclude some infeasible answers and narrow the search spaces significantly, as well as speed up the convergence of the VQE algorithm for the hybrid *de novo* assembly problem.

Considering a specific value of p, the binary variable $x_{i,p}$ of the Hamiltonian H_q can be encoded by the state $|\psi^{(p)}\rangle = |x_{1,p}\cdots x_{i,p}\cdots x_{n,p}\rangle$. Because at most one node can be chosen for a position in the assembly path, one of the binary variables $x_{i,p}$ is limited to 1 or 0 while the remaining variables must be set to 0. After eliminating the incorrect solutions from the entire 2^n-dimensional Hilbert space, the feasible answers are restricted to a set $\mathbb{S}$ of the $n+1$ elements. The feasible answers are similar to a W state, except for an extra $|0\rangle^{\otimes n}$ state. Hence, it suffices to construct the trial wave function

$$|\psi^{(p)}(\vec{\theta})\rangle = \alpha_0|00\cdots0\rangle + \alpha_1|10\cdots0\rangle + \alpha_2|01\cdots0\rangle + \cdots + \alpha_n|00\cdots1\rangle, \quad (17)$$

where $\sum_{k=0}^{n}|\alpha_k|^2 = 1$.

TABLE II. Performance of both the DVQE and VQE, utilizing a hardware-efficient *Ansatz*.

Algorithm	Average time per iteration (s)	Average iteration number for convergence	Total time for convergence (s)
VQE	115.83	157	18185.31
DVQE	0.35	147	51.45

Here we construct the problem-inspired *Ansatz* with R_x, R_y, and CZ gates, shown in Fig. 10, and apply it to the initial state $|0\rangle^{\otimes n}$ to generate a trial wave function $|\psi^{(p)}(\vec{\theta})\rangle$. To provide a more comprehensive understanding of the function of the quantum circuit, we decompose the process of preparing the trial wave function into two parts.

The first part involves the preparation of a pseudotarget state $|\psi_n\rangle$, which satisfies that the number of bases is equal to $n+1$. In this part, our first step will be to initialize the state $|\psi_0\rangle = |0\rangle^{\otimes n}$. Then an $R_x(\theta_0)$ gate is applied on the first qubit, thus giving us

$$|\psi_1\rangle = (\alpha_0|0\rangle + \beta_0|1\rangle) \otimes |0\rangle^{\otimes(n-1)}, \quad (18)$$

where $\alpha_0 = \cos(\theta_0/2)$ and $\beta_0 = -i\sin(\theta_0/2)$.

Next we apply a series of $U_{cq,tq}(\theta_j)$ gates in the next step, where $j \in [1, n-1]$. As Fig. 10(b) shows, the $U_{cq,tq}(\theta_j)$ is implemented with $R_y(\theta_j)$, $\mathrm{CZ}_{cq,tq}$, and $R_y(\theta_j+\pi)$ gates. Here we set the cqth qubit as a control qubit and the tqth as a target qubit. When the state of the target qubit is $|0\rangle$, the functions of $U_{cq,tq}(\theta_j)$ are as follows:

$$|00\rangle \xrightarrow{U_{cq,tq}(\theta_j)} |0\rangle \otimes [-\sin(\theta_j)|0\rangle + \cos(\theta_j)|1\rangle],$$
$$|10\rangle \xrightarrow{U_{cq,tq}(\theta_j)} |11\rangle. \quad (19)$$

Thus, after a sets of $U_{cq,tq}(\theta_j)$ gates applied on $|\psi_1\rangle$, we obtain

$$|\psi_n\rangle = \alpha_0\alpha_1\cdots\alpha_{n-1}|0\rangle^{\otimes n} + \alpha_0\alpha_1\cdots\alpha_{n-2}\beta_{n-1}|0\rangle^{\otimes(n-1)}\otimes|1\rangle + \cdots + \alpha_0\beta_1|0\rangle\otimes|1\rangle^{\otimes(n-1)} + \beta_0|1\rangle^{\otimes n}, \quad (20)$$

where $\alpha_j = -\sin(\theta_j)$ and $\beta_j = \cos(\theta_j)$.

The second part involves a set of CNOT gates applied on the pseudotarget state $|\psi_n\rangle$ to flip partial bases and prepare the trial wave function $|\psi^{(p)}(\vec{\theta})\rangle \equiv |\psi_{n+1}\rangle$. In order to better understand how a set of CNOT gates work, we consider a case with three qubits; the quantum circuit is shown in Fig. 10(c). The input state $|\psi_0\rangle = |000\rangle$ is sent through $R_x(\theta_0)$, $U_{01}(\theta_1)$,

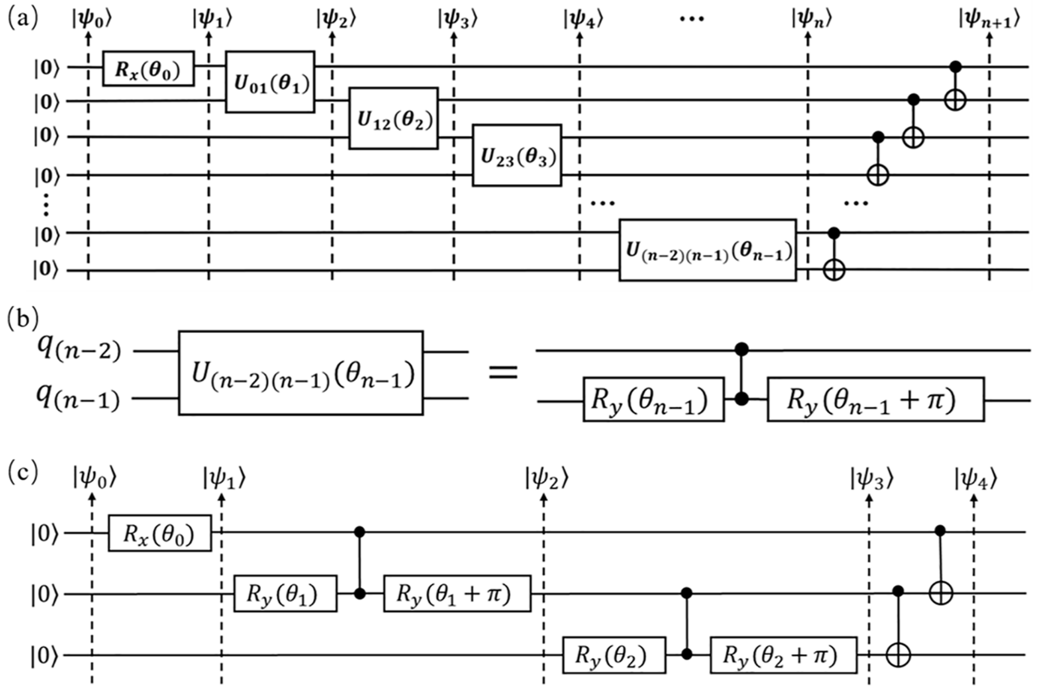

FIG. 10. Schematic of the problem-inspired *Ansatz* for preparing the *n*-qubit wave function $|\psi^{(p)}(\vec{\theta})\rangle$. (a) Problem-inspired *Ansatz* implemented by $R_x(\theta_0)$, $U_{cq,tq}(\theta_j)$, and CNOT gates. (b) Implementation of the $U_{cq,tq}(\theta_j)$ gate using $R_y(\theta_j)$, CZ, and $R_y(\theta_j + \pi)$ gates. (c) Quantum circuit for the three-qubit wave function.

and $U_{12}(\theta_2)$ gates to give

$$\begin{aligned}|\psi_3\rangle = &\cos(\theta_0/2)\sin(\theta_1)\sin(\theta_2)|000\rangle\\ &-\cos(\theta_0/2)\sin(\theta_1)\cos(\theta_2)|001\rangle\\ &+\cos(\theta_0/2)\cos(\theta_1)|011\rangle\\ &-i\sin(\theta_0/2)|111\rangle.\end{aligned} \tag{21}$$

Compared to the wave function $|\psi_4\rangle$, we need to flip the bases $|011\rangle$ and $|111\rangle$ in the pseudotarget state $|\psi_3\rangle$ to $|010\rangle$ and $|100\rangle$ while keep other bases unchanged. Specifically,

$$\begin{aligned}&|000\rangle \xrightarrow{\text{CNOT}_{1,2}} |000\rangle \xrightarrow{\text{CNOT}_{0,1}} |000\rangle,\\ &|001\rangle \xrightarrow{\text{CNOT}_{1,2}} |001\rangle \xrightarrow{\text{CNOT}_{0,1}} |001\rangle,\\ &|011\rangle \xrightarrow{\text{CNOT}_{1,2}} |010\rangle \xrightarrow{\text{CNOT}_{0,1}} |010\rangle,\\ &|111\rangle \xrightarrow{\text{CNOT}_{1,2}} |110\rangle \xrightarrow{\text{CNOT}_{0,1}} |100\rangle,\end{aligned} \tag{22}$$

and

$$\begin{aligned}|\psi_4\rangle = &\cos(\theta_0/2)\sin(\theta_1)\sin(\theta_2)|000\rangle\\ &-\cos(\theta_0/2)\sin(\theta_1)\cos(\theta_2)|001\rangle\\ &+\cos(\theta_0/2)\cos(\theta_1)|010\rangle\\ &-i\sin(\theta_0/2)|100\rangle.\end{aligned} \tag{23}$$

As shown in the example of three qubits, we can prepare $|\psi^{(p)}(\vec{\theta})\rangle$ of any qubits by this problem-inspired *Ansatz*.

Here we give the matrix forms of the gates used in this *Ansatz*. The $R_x(\theta_j)$ and $R_y(\theta_j)$ are rotation gates acting on the jth qubit. The $\text{CZ}_{cq,tq}$ and $\text{CNOT}_{cq,tq}$ gates are the CZ and CNOT gates, respectively. The unitary matrices of each gate are

$$R_x(\theta) \equiv \begin{pmatrix}\cos\frac{\theta}{2} & -i\sin\frac{\theta}{2}\\ -i\sin\frac{\theta}{2} & \cos\frac{\theta}{2}\end{pmatrix}, \tag{24}$$

$$R_y(\theta) \equiv \begin{pmatrix}\cos\frac{\theta}{2} & -\sin\frac{\theta}{2}\\ \sin\frac{\theta}{2} & \cos\frac{\theta}{2}\end{pmatrix}, \tag{25}$$

$$\text{CZ} \equiv \begin{pmatrix}1 & 0 & 0 & 0\\ 0 & 1 & 0 & 0\\ 0 & 0 & 1 & 0\\ 0 & 0 & 0 & -1\end{pmatrix}, \tag{26}$$

$$\text{CNOT} \equiv \begin{pmatrix}1 & 0 & 0 & 0\\ 0 & 1 & 0 & 0\\ 0 & 0 & 0 & 1\\ 0 & 0 & 1 & 0\end{pmatrix}. \tag{27}$$

The data that support the findings of this study are available from the authors upon reasonable request. The code that supports the findings of this study is available from the authors upon reasonable request.

ACKNOWLEDGMENTS

This work was supported by the National Key R&D Program of China (Grant No. 2022YFC3400400). We acknowledge support from China National GeneBank and the Guangdong Bigdata Engineering Technology Research Center for Life Sciences.

[1] J. Shendure, S. Balasubramanian, G. M. Church, W. Gilbert, J. Rogers, J. A. Schloss, and R. H. Waterston, Nature (London) **550**, 345 (2017).

[2] L. M. Smith, J. Z. Sanders, R. J. Kaiser, P. Hughes, C. Dodd, C. R. Connell, C. Heiner, S. B. H. Kent, and L. E. Hood, Nature (London) **321**, 674 (1986).

[3] C. Connell, S. Fung, C. Heiner, J. Bridgham, V. Chakerian, E. Heron, B. Jones, S. Menchen, W. Mordan, M. Raff *et al.*, BioTechniques **5**, 342 (1987).

[4] J. K. Bonfield, K. F. Smith, and R. Staden, Nucleic Acids Res. **23**, 4992 (1995).

[5] G. G. Sutton, O. White, M. D. Adams, and A. R. Kerlavage, Genome Sci. Technol. **1**, 9 (1995).

[6] J. D. Kececioglu and E. W. Myers, Algorithmica **13**, 7 (1995).

[7] E. W. Myers, J. Comput. Biol. **2**, 275 (1995).

[8] E. W. Myers, G. G. Sutton, A. L. Delcher, I. M. Dew, D. P. Fasulo, M. J. Flanigan, S. A. Kravitz, C. M. Mobarry, K. H. Reinert, K. A. Remington *et al.*, Science **287**, 2196 (2000).

[9] E. L. Lawler, in *The Traveling Salesman Problem: A Guided Tour of Combinatorial Optimization*, edited by E. L. Lawler, J. K. Lenstra, A. H G. Rinnooy Kan, and D. B. Shmoys (Wiley, New York, 1985).

[10] P. A. Pevzner, H. Tang, and M. S. Waterman, Proc. Natl. Acad. Sci. USA **98**, 9748 (2001).

[11] R. Li, H. Zhu, J. Ruan, W. Qian, X. Fang, Z. Shi, Y. Li, S. Li, G. Shan, K. Kristiansen, S. Li, H. Yang, J. Wang, and J. Wang, Genome Res. **20**, 265 (2010).

[12] A. Bankevich, S. Nurk, D. Antipov, A. A. Gurevich, M. Dvorkin, A. S. Kulikov, V. M. Lesin, S. I. Nikolenko, S. Pham, A. D. Prjibelski, A. V. Pyshkin, A. V. Sirotkin, N. Vyahhi, G. Tesler, M. A. Alekseyev, and P. A. Pevzner, J. Comput. Biol. **19**, 455 (2012).

[13] J. T. Simpson, K. Wong, S. D. Jackman, J. E. Schein, S. J. M. Jones, and İ. Birol, Genome Res. **19**, 1117 (2009).

[14] A. S. Mikheyev and M. M. Y. Tin, Mol. Ecol. Res. **14**, 1097 (2014).

[15] M. Foquet, J. Korlach, W. Zipfel, W. W. Webb, and H. G. Craighead, Anal. Chem. **74**, 1415 (2002).

[16] M. J. Levene, J. Korlach, S. W. Turner, M. Foquet, H. G. Craighead, and W. W. Webb, Science **299**, 682 (2003).

[17] J. Eid, A. Fehr, J. Gray, K. Luong, J. Lyle, G. Otto, P. Peluso, D. Rank, P. Baybayan, B. Bettman, A. Bibillo, K. Bjornson, B. Chaudhuri, F. Christians, R. Cicero, S. Clark, R. Dalal, A. deWinter, J. Dixon, M. Foquet *et al.*, Science **323**, 133 (2009).

[18] S. Koren, M. C. Schatz, B. P. Walenz, J. Martin, J. T. Howard, G. Ganapathy, Z. Wang, D. A. Rasko, W. R. McCombie, E. D. Jarvis *et al.*, Nat. Biotechnol. **30**, 693 (2012).

[19] K. F. Au, J. G. Underwood, L. Lee, and W. H. Wong, PLoS ONE **7**, e46679 (2012).

[20] V. Deshpande, E. D. K. Fung, S. Pham, and V. Bafna, in *Algorithms in Bioinformatics*, edited by A. Darling and J. Stoye, Lecture Notes in Computer Science Vol. 8126 (Springer, Berlin, 2013), pp. 349–363.

[21] D. Antipov, A. Korobeynikov, J. S. McLean, and P. A. Pevzner, Bioinformatics **32**, 1009 (2016).

[22] R. R. Wick, L. M. Judd, C. L. Gorrie, and K. E. Holt, PLoS Comput. Biol. **13**, e1005595 (2017).

[23] N. De Maio, L. P. Shaw, A. Hubbard, S. George, N. D. Sanderson, J. Swann, R. Wick, M. AbuOun, E. Stubberfield, S. J. Hoosdally, D. W. Crook, T. E. A. Peto, A. E. Sheppard, M. J. Bailey, D. S. Read, M. F. Anjum, A. S. Walker, and N. Stoesser, Microb. Genomics **5**, e000294 (2019).

[24] H. A. Craddock, Y. Motro, B. Zilberman, B. Khalfin, S. Bardenstein, and J. Moran-Gilad, Microorganisms **10**, 619 (2022).

[25] A. W. Harrow and A. Montanaro, Nature (London) **549**, 203 (2017).

[26] C. Neill, P. Roushan, K. Kechedzhi, S. Boixo, S. V. Isakov, V. Smelyanskiy, A. Megrant, B. Chiaro, A. Dunsworth, K. Arya, R. Barends, B. Burkett, Y. Chen, Z. Chen, A. Fowler, B. Foxen, M. Giustina, R. Graff, E. Jeffrey, T. Huang *et al.*, Science **360**, 195 (2018).

[27] A. S. Boev, A. S. Rakitko, S. R. Usmanov, A. N. Kobzeva, I. V. Popov, V. V. Ilinsky, E. O. Kiktenko, and A. K. Fedorov, Sci. Rep. **11**, 13183 (2021).

[28] A. Sarkar, Z. Al-Ars, and K. Bertels, PLoS One **16**, e0249850 (2021).

[29] K. Nałęcz-Charkiewicz and R. M. Nowak, BMC Bioinf. **23**, 122 (2022).

[30] L. C. L. Hollenberg, Phys. Rev. E **62**, 7532 (2000).

[31] P. Niroula and Y. Nam, npj Quantum Inf. **7**, 37 (2021).

[32] A. Perdomo, C. Truncik, I. Tubert-Brohman, G. Rose, and A. Aspuru-Guzik, Phys. Rev. A **78**, 012320 (2008).

[33] A. Robert, P. K. Barkoutsos, S. Woerner, and I. Tavernelli, npj Quantum Inf. **7**, 38 (2021).

[34] P. Hauke, G. Mattiotti, and P. Faccioli, Phys. Rev. Lett. **126**, 028104 (2021).

[35] D. Ellinas and P. D. Jarvis, J. Phys. A: Math. Theor. **52**, 115601 (2019).

[36] F. Arute, K. Arya, R. Babbush, D. Bacon, J. C. Bardin, R. Barends, R. Biswas, S. Boixo, F. G. S. L. Brandao, D. A. Buell, B. Burkett, Y. Chen, Z. Chen, B. Chiaro, R. Collins, W. Courtney, A. Dunsworth, E. Farhi, B. Foxen, A. Fowler *et al.*, Nature (London) **574**, 505 (2019).

[37] A. Kandala, A. Mezzacapo, K. Temme, M. Takita, M. Brink, J. M. Chow, and J. M. Gambetta, Nature (London) **549**, 242 (2017).

[38] J. Romero, R. Babbush, J. R. McClean, C. Hempel, P. J. Love, and A. Aspuru-Guzik, Quantum Sci. Technol. **4**, 014008 (2018).

[39] E. Farhi, J. Goldstone, and S. Gutmann, arXiv:1411.4028.

[40] M. Benedetti, M. Fiorentini, and M. Lubasch, Phys. Rev. Res. **3**, 033083 (2021).

[41] A. Peruzzo, J. McClean, P. Shadbolt, M.-H. Yung, X.-Q. Zhou, P. J. Love, A. Aspuru-Guzik, and J. L. O'Brien, Nat. Commun. **5**, 4213 (2014).

[42] G. A. Quantum, Collaborators, F. Arute, K. Arya, R. Babbush, D. Bacon, J. C. Bardin, R. Barends, S. Boixo, M. Broughton, B. B. Buckley *et al.*, Science **369**, 1084 (2020).

[43] A. Matsuo, Y. Suzuki, I. Hamamura, and S. Yamashita, IEICE Transactions on Information and Systems **E106.D**, 1772 (2023).

[44] I. G. Ryabinkin, R. A. Lang, S. N. Genin, and A. F. Izmaylov, J. Chem. Theory Comput. **16**, 1055 (2020).

[45] P.-L. Dallaire-Demers, J. Romero, L. Veis, S. Sim, and A. Aspuru-Guzik, Quantum Sci. Technol. **4**, 045005 (2019).

[46] J. Lee, W. J. Huggins, M. Head-Gordon, and K. B. Whaley, J. Chem. Theory Comput. **15**, 311 (2018).

[47] Y. S. Yordanov, D. R. Arvidsson-Shukur, and C. H. Barnes, Phys. Rev. A **102**, 062612 (2020).

[48] H. R. Grimsley, S. E. Economou, E. Barnes, and N. J. Mayhall, Nat. Commun. **10**, 3007 (2019).

[49] H. L. Tang, V. Shkolnikov, G. S. Barron, H. R. Grimsley, N. J. Mayhall, E. Barnes, and S. E. Economou, PRX Quantum **2**, 020310 (2021).

[50] J. Li, X. Yang, X. Peng, and C.-P. Sun, Phys. Rev. Lett. **118**, 150503 (2017).

[51] J. Stokes, J. Izaac, N. Killoran, and G. Carleo, Quantum **4**, 269 (2020).

[52] J. A. Nelder and R. Mead, Comput. J. **7**, 308 (1965).

[53] D. J. A. Welsh and M. B. Powell, Comput. J. **10**, 85 (1967).