Quantum circuit synthesis on noisy intermediate-scale quantum devices

Shuai Yang, Guojing Tian, Jialin Zhang, and Xiaoming Sun*

State Key Laboratory of Processors, Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100190, China and School of Computer Science and Technology, University of Chinese Academy of Sciences, Beijing 100049, China

(Received 28 July 2023; accepted 28 November 2023; published 2 January 2024)

In the near future of the noisy intermediate-scale quantum (NISQ) era, almost all quantum computing devices will be restricted to a specific fixed qubits connectivity architecture. Thus, the synthesis of quantum circuits with limited connectivity is urgent. We design quantum circuit synthesis algorithms for basic and essential synthesis problems, such as quantum state preparation, general unitary synthesis, and quantum isometries.For any architecture, the controlled NOT (CNOT) count is at most 5/3 times the state-of-the-art result on complete-graph architecture. For some specific architectures, such as square-grid ones, the ratio is reduced to 1.126. The numerical simulation result is confirmatory of theoretical conclusions. Our algorithms significantly reduce by more than 50% additional CNOT count compared to mapping algorithms. These algorithms help to implement the larger-scale algorithm in the physics device. Our results illustrate that well-designed synthesis algorithms can mitigate the problem of limited qubit connectivity in the NISQ era and may suggest the design of large-scale quantum devices.

DOI: 10.1103/PhysRevA.109.012602

I. INTRODUCTION

Quantum computation shows great potential to solve intractable problems. In [1], Shor introduces a quantum algorithm to solve the factoring problem in polynomial time. In [2], Grover gives a quantum unstructured search algorithm, quadratic speedup to the classical algorithm. Moreover, other quantum algorithms focus on solving linear equation systems, Hamiltonian simulation, quantum chemistry, etc. [3–6].

The crucial step is to synthesize the corresponding circuit to implement the quantum algorithms on the quantum devices. A quantum algorithm usually corresponds to a unitary matrix, and the quantum circuit synthesis problem is to decompose the given unitary matrix into a sequence of elementary gates [7]. There have been several studies on the quantum circuit synthesis problem since the concept of quantum computing began 40 years ago [7–15]. Studying quantum circuit synthesis problems is essential in computation science and physics. It helps to implement larger-scale meaningful algorithms in the NISQ era and inspires the connectivity restriction design of physical devices. Almost all the results focused on the complete-graph qubit connectivity architecture, or to say, no consideration was given to the restricted qubit connectivity architecture. However, the two-qubit gate is indeed restricted to operating on a fixed two-qubit pair for the present noisy intermediate-scale quantum (NISQ) devices, and it could go on for quite a long time. Two methods exist to conquer this restriction of the two-qubit gates. One is to map after synthesis, which adds SWAP or controlled NOT (CNOT) gates when the two-qubit gate cannot be implemented on the quantum device [16–18]. In the worst case, the additional CNOT count will reach $O(n)$ times the original CNOT count. The other, precisely what we will study in this paper, is to design a NISQ synthesis algorithm, which directly synthesizes the unitary matrices on a specific connectivity architecture [18–22].

In this paper, we introduce the phase gadget method and design quantum circuit synthesis algorithms for several essential synthesis problems, including diagonal unitary synthesis, uniformly controlled unitary synthesis, general unitary synthesis, general or sparse quantum state preparation (QSP), and quantum isometries. These problems all have their meanings. The quantum state preparation is the first step of quantum algorithms. Unitaries and isometries are usually used to describe the transformation of the quantum state in a quantum channel. The diagonal unitary and uniformly controlled unitary are some elementary quantum gates widely used in quantum circuit synthesis [13,20]. For all these problems, our algorithms work for all architectures, and the CNOT count of our synthesized circuit is only a maximum 5/3 times of the state-of-the-art results [12,20,23] on complete-graph architecture. The ratio can be further reduced for some specific architectures. For example, on the linear nearest neighbor (LNN), our algorithm improves the CNOT count upper bound of general unitary synthesis and QSP from $\frac{5}{6}4^n$, $\frac{10}{3}2^n$ to $\frac{3}{4}4^n$, $\frac{3}{2}2^n$, respectively. Our paper also analyzes the CNOT count of quantum isometries on NISQ architectures. To clearly understand the comparison, all the synthesized results of our algorithms (bold terms) and previous algorithms are listed in Table I, which illustrates that for the circuit synthesis problems, well-designed synthesis algorithms can mitigate the impact of NISQ connectivity architectures into an inconspicuous level. We also compare the CNOT count of our algorithms and the mapping (SABRE [24]) after the synthesis algorithm on the complete graph [20]. Our algorithm achieves ≈66% additional CNOT count reduction, compared to addi-

*sunxiaoming@ict.ac.cn

TABLE I. The CNOT count of uniformly controlled R_z gate $\mathcal{R}_{n-1}$, diagonal unitary Λ_n, uniformly controlled gate $\mathcal{U}_{n-1,1}$, general unitary U_n, quantum state preparation U_n^{QSP}, and quantum isometries $U_{m,n}^{\mathcal{ISO}}$ on different connectivity architecture.. We omit the low-order terms in the CNOT count.

	$\mathcal{R}_{n-1}$	Λ_n	$\mathcal{U}_{n-1,1}$	U_n	U_n^{QSP}	$U_{m,n}^{\mathcal{ISO}}$
Complete graph	$\frac{1}{2}2^n$ [20]	2^n [20]	$\frac{3}{2}2^n$ [20]	$\frac{23}{48}4^n$ [23]	$\frac{23}{24}2^n$ [23]	2^{m+n} [12]
LNN	$\frac{5}{6}2^n$ [20]	$\frac{5}{3}2^n$ [20]	$\frac{5}{2}2^n$ [20]	$\frac{3}{4}4^n$, $\frac{5}{6}4^n$ [20]	$\frac{3}{2}2^n$, $\frac{10}{3}2^n$ [20]	$\frac{5}{3}2^{m+n}$
General graph	$\frac{5}{6}2^n$	$\frac{5}{3}2^n$	$\frac{5}{2}2^n$	$\frac{3}{4}4^n$	$\frac{3}{2}2^n$	$\frac{5}{3}2^{m+n}$
Heavy-hex graph	0.641×2^n	1.282×2^n	1.923×2^n	0.606×4^n	1.212×2^n	$1.282 \times 2^{m+n}$
$2 \times n/2$ grid	$\frac{19}{30}2^n$	$\frac{19}{15}2^n$	$\frac{19}{10}2^n$	$\frac{3}{5}4^n$	$\frac{6}{5}2^n$	$\frac{19}{15}2^{m+n}$
Square $(\sqrt{n} \times \sqrt{n})$ grid	0.563×2^n	1.126×2^n	1.689×2^n	0.548×4^n	1.096×2^n	$1.126 \times 2^{m+n}$
Graph with max degree k	$(\frac{1}{2}+\frac{1}{2^{k-1}})2^n$	$(1+\frac{1}{2^{k-2}})2^n$	$(\frac{3}{2}+\frac{3}{2^{k-1}})2^n$	$(\frac{1}{2}+\frac{3}{2^{k+1}})4^n$	$(\mathbf{1}+\frac{3}{2^k})2^n$	$(1+\frac{1}{2^{k-2}})2^{m+n}$

tional CNOT gates used in the SABRE. When we choose the total CNOT count as the evaluation indicator, our algorithms reduce by 20–30% CNOT gates on the frequently used connection architecture. Our algorithms also perform well when $k < 10$ and the total CNOT count is available in the NISQ era. The numerical simulation results show that synthesis with the connectivity restriction is better when implementing a circuit on a specific device.

A. Preliminaries and related works

An n-qubit state in the quantum computation is usually described as a 2^n unit complex vector $|\psi\rangle$ belonging to the Hilbert space $\mathbb{C}^{2^n}$. The set of states $\{|00\cdots0\rangle, |00\cdots1\rangle, \ldots, |11\cdots1\rangle\}$ is a basis of the $\mathbb{C}^{2^n}$, which is usually named the computational basis. We use $[k]$ to denote the integer set $\{1, 2, \ldots, k\}$.

1. Quantum gate and quantum circuit

The quantum gate is the most critical component in the quantum circuit model of quantum computation. The elementary gate [7] denotes the gate, which is either a single-qubit gate or a CNOT gate. Every quantum gate corresponds to its form of a unitary matrix. For example,

$$R_z(\theta) = \begin{bmatrix} e^{-i\theta/2} & \\ & e^{i\theta/2} \end{bmatrix}, \quad \text{CNOT} = \begin{bmatrix} 1 & & & \\ & 1 & & \\ & & & 1 \\ & & 1 & \end{bmatrix},$$

where the parameter θ is real, and the blank elements denote zero. An n-qubit quantum circuit $\mathcal{C}$ consists of a sequence of the elementary gates, whose mathematical form is a unitary matrix $U \in \mathbb{C}^{2^n \times 2^n}$. Usually, the number of elementary gates, called circuit size, is used to measure the complexity of the quantum circuit. But in the NISQ era, the number of CNOT gates plays the leading role in circuit size rather than that of single-qubit gates. One reason is that the CNOT gate only operates on the connected two qubits of the NISQ device, while the single-qubit gate operates without this connectivity restriction. The other reason is that the fidelity of the CNOT gate is much lower than the single-qubit gate physically. Thus, in this paper, we use the number of CNOT gates, i.e., CNOT count, to measure the circuit complexity under the qubit connectivity architectures.

2. Quantum circuit synthesis problem on NISQ devices

We use the undirected graph $G = (V, E)$ to represent the qubit connectivity architecture of a specific NISQ device, where the vertices denote the qubits and the edges indicate the effective interactions between qubits. Now, we analogize the quantum circuit synthesis problem proposed in [7] to define the quantum circuit synthesis on the NISQ device. For any given $2^n \times 2^n$ unitary matrix U and the qubit connectivity architecture $G = (V, E)$, find a quantum circuit $\mathcal{C} = g_k g_{k-1} \cdots g_1$ satisfying the two conditions below.

(1) For any n-qubit state $|\psi\rangle$, we have $\mathcal{C}|\psi\rangle = U|\psi\rangle$.

(2) For any CNOT gate $g_i, i \in [k]$ operating on $u, v \in V$, we have $uv \in E$.

The target is to minimize the CNOT count of the circuit.

3. Lower bound of the quantum circuit synthesis problem on NISQ devices

In [23], Shende *et al.* prove the fact that *for almost all the n-qubit unitary $U \in \mathbb{C}^{2^n \times 2^n}$, there is no quantum circuit with $o(4^n)$ size that can synthesis U*.

Also, in [23], they show the lower bound of quantum state preparation: *For almost all the n-qubit quantum state $v \in \mathbb{C}^{2^n}$, there is no quantum circuit with $o(2^n)$ size that can prepare state $|\psi_v\rangle$*.

In [12], they give the lower bound for isometry using a similar method: *For almost all the $(n \to m)$-qubit quantum isometry $U^{\mathcal{ISO}} \in \mathbb{C}^{2^n \times 2^m}$, there is no quantum circuit with $o(2^{n+m})$ size that can synthesis $U^{\mathcal{ISO}}$*.

If we release the synthesis problem to the approximation synthesis problem, this means the resulting circuit $\tilde{\mathcal{C}}$ and the input unitary U are not exactly equal, but with an ϵ error. That is, $\max_{\psi \in 2^n} \|\tilde{\mathcal{C}}|\psi\rangle - U|\psi\rangle\| \leqslant \epsilon$. The lower bound of the approximation synthesis problem is still $\Omega(\frac{4^n \log \epsilon}{n})$. Similarly, the lower bound of the approximation synthesis problem of QSP and isometry is $\Omega(\frac{2^n \log \epsilon}{n})$ and $\Omega(\frac{2^{n+m} \log \epsilon}{n})$, respectively.

One CNOT gate can separate at most four single-qubit gates, and each new single qubit will provide two real free parameters. Thus, the CNOT-count lower bound of general unitary synthesis, quantum state preparation, and quantum isometry synthesis is $\frac{4^n}{4}$, $\frac{2^n}{4}$, and $\frac{2^{n+m}}{4}$.

4. Related works

The study of the circuit synthesis algorithm starts with the synthesis of the complete-graph architecture. In [7], Barenco *et al.* first give a synthesis algorithm to decompose any n-qubit unitary matrices, which takes $O(n^3 4^n)$ elementary gates. And then, in [23], Shende *et al.* design the synthesis algorithm for the general n-qubit unitary with only $\frac{23}{48}4^n$ CNOT gates and $O(4^n)$ elementary single-qubit gates. For n-qubit quantum state preparation, the best synthesis algorithm is also introduced in [23], which takes $\frac{23}{24}2^n$ CNOT gates and $O(2^n)$ elementary single-qubit gates. For m-qubit to n-qubit quantum isometries, in [12], Iten *et al.* give a synthesis algorithm with 2^{m+n} CNOT gates and $O(2^{n+m})$ single qubits. For uniformly controlled gates (UCGs), in [20], Bergholm *et al.* give the synthesis algorithms with $\frac{3}{2}2^n$ CNOT gate and $O(2^n)$ single-qubit gates.

With the development of quantum devices, some studies are focusing on the synthesis algorithm on specific connectivity architecture. In [20], Bergholm *et al.* consider the synthesis of uniformly controlled gates, general unitary synthesis, and quantum state preparation. The CNOT count in their works is $\frac{5}{2}2^n$, $\frac{5}{6}4^n$, and $\frac{10}{3}2^n$, respectively. Very recently, in an independent work [18], Allcock *et al.* also considers quantum synthesis algorithms for quantum state preparation and general unitary synthesis limited by connectivity architecture. Their work focuses on the circuit depth optimization using ancillary qubits. Compared to their result, the CNOT count in our paper achieves a ≈50–75% reduction.

Several heuristic algorithms synthesize quantum circuits under the connectivity restriction [21,22]. These studies' performance will be in some special cases, such as quantum Fourier transform and quantum arithmetic circuits. However, for the general unitary synthesis, these works do not bound the CNOT count of the resulting circuit. Besides, the running time of these algorithms will reach $O(\text{poly}(n)4^n)$ when the unitary is a random unitary. Moreover, these heuristics algorithms solve the synthesis problems approximately. The approximate synthesis problem differs from the standard synthesis problem; thus, in this paper, we do not design the comparative numerical simulation with these heuristics algorithms.

B. Organization

We start with designing an algorithm for the UCG synthesis problem in Sec. II because the UCG synthesis algorithm is the basis of all the other synthesis algorithms. Then, in Sec. III A, we show how to synthesize the general unitary with cosine-sine decomposition (CSD). In Sec. III B, we introduce two frameworks for QSP. Moreover, we design a more specific algorithm for sparse QSP. In Sec. III C, we present the synthesis algorithm for isometries, an extension of quantum state preparation and unitary synthesis. Next in Sec. IV, we design different numerical simulations to illustrate the performance of our algorithms. Finally, we conclude in Sec. V.

II. UNIFORMLY CONTROLLED GATE SYNTHESIS

A UCG is still a controlled gate, but for any values of the controlled qubits, there will be a corresponding unitary operated on the target qubits. We denote a uniformly k-controlled-m gate as $\mathcal{U}_{k,m}$, i.e.,

$$\mathcal{U}_{k,m} = \begin{bmatrix} U_1 & & & \\ & U_2 & & \\ & & \ddots & \\ & & & U_{2^k} \end{bmatrix} \in \mathbb{C}^{2^{k+m}\times 2^{k+m}}, \tag{1}$$

where $U_j \in \mathbb{C}^{2^m\times 2^m}$ is an m-qubit gate with $j \in [2^k]$. An important instance of $\mathcal{U}_{k,m}$ is the uniformly controlled R_z gate, denoted as

$$\mathcal{R}_k = \begin{bmatrix} R_z(\theta_1) & & & \\ & R_z(\theta_2) & & \\ & & \ddots & \\ & & & R_z(\theta_{2^k}) \end{bmatrix}, \tag{2}$$

where θ_i is a real number for any $i \in [2^k]$. $\mathcal{R}_k$ has been widely used in quantum circuit synthesis [20,23]. Specifically, we can use $\mathcal{R}_k$ to construct the uniformly controlled R_y or R_x gate, because $SHR_z(\theta)HS = R_y(\theta)$, $HR_z(\theta)H = R_x(\theta)$. The synthesis algorithm of UCG is crucial in this paper, since the synthesis of other types of unitary depends on it.

A. Algorithm

The synthesis algorithms for $\mathcal{R}_k$ and $\mathcal{R}_{k-1}$ ($\mathcal{U}_{k,1}$ and $\mathcal{U}_{k-1,1}$) are identical. To maintain consistency in the analysis in the following section, we will focus on the quantum cost of $\mathcal{R}_{k-1}$ ($\mathcal{U}_{k-1,1}$) since it is a k-qubit gate. We start with the synthesis algorithm for $\mathcal{R}_{k-1}$ on the NISQ device. In [20], Bergholm *et al.* introduce a decomposition for $\mathcal{R}_{k-1}$ on the complete graph and LNN with 2^{k-1} and $\frac{5}{6}2^k$ CNOT gates, respectively. In this subsection, we observe this algorithm from a higher perspective and then modify the algorithm to reduce the number of long-distance CNOT gates, which takes a large number of CNOT gates to implement on the specific device. Here, the distance of the CNOT gate is defined by the distance of two qubits in the connectivity graph. We decompose the diagonal unitary into several phase gadgets and carefully reorder the phase gadgets to minimize the cost of the long-distance CNOT. The algorithm, which makes full use of the phase gadget, can be divided into three parts.

(1) Center finding and vertex reordering: In this part, we will find a permutation $\pi : V \to V$ of vertices. All the R_z and almost all the CNOT gates will be implemented on the center qubit $\pi(q_n)$, minimizing the original long-distance CNOT gate after this part. This part can be finished in polynomial classical running time.

(2) Parameter transformation: We will transform the parameters of the diagonal unitary into those of R_z gates. We point out that the transformation between the parameter is the Walsh-Hadamard transformation, which is not mentioned in [20]. This part can be finished in $O(2^k)$ classical running time.

(3) Gate implementation: We will generate the final circuit in this part. The rotation angle is determined in Sec. II A 2, and the CNOT gate is generated by the result in Sec. II A 1. This part can be finished in $O(2^k)$ classical running time.

After showing the intuitive effect of each part, we will discuss the details in the following.

Algorithm 1: Center finding and vertex reordering.

input : A connected graph $G = (V, E)$, where $V = \{q_1, q_2, \cdots, q_n\}$ are the set of qubits. E denote the connectivity of qubits in the NISQ device. If a CNOT gate can be applied on q_u and q_v, then $q_u q_v \in E$.
output: A new graph $G' = (V', E')$ and the center vertex q'_n. There is a permutation $\pi : V \to V$ that $q'_i = \pi(q_i)$. And for all $q_u q_v \in E, \pi(q_u)\pi(q_v) = q'_u q'_v \in E'$.

mincost $= n2^{n+1}$, centervertex $= q_n$;
for $q_u \in V$**:**
  Vertexreorder(G,q_u) ;
  temcost $= \sum_{i=1}^{n-1}(2dis(q_i, q_n) - 1)2^{n-i}$;
  if $temcost \leq mincost$**:**
    mincost $=$ temcost;
    centervertex $= q_u$;
  Reset the graph;
Vertexreorder(G,$centervertex$) ;
Vertexreorder(G,q_u) **:**
  SWAP(q_u, q_n) ;
  Queue Q,$ind = 1$;
  Q.push(q_n);
  while $Q.empty()$== 0**:**
    $t = Q$.front();
    Q.pop();
    for $tq_v \in E$ *and* q_v *have not been pushed to* Q**:**
      Q.push(q_v)
    if $t \neq q_n$**:**
      SWAP(q_{ind}, t) ;
      $ind++$;

1. Center finding and vertex reordering

In the first part, we will find a given graph's "good center vertex." And then, we will reorder the other vertex indices by the distance to the center vertex. Almost all the gates will be implemented on the center vertex in the synthesis algorithm. For the other vertices, the frequency of interaction with the center vertex is negatively correlated with the distance to the center vertex. We show a pseudocode for the first part in Algorithm 1. Notice Algorithm 1 does not restrict the number of vertices of the graph. For $\mathcal{R}_{k-1}$, our algorithm can find a suitable subgraph of the given graph G to implement subsequent parts.

In this part, we can also calculate the shortest path from each vertex to the center vertex, which can be realized using breath first search (BFS) easily; the classical running time of the center finding algorithm is $O(|V|^2)$; this algorithm can be further optimized to $O[|V|\text{polylog}(|V|)]$ using the dynamic programming on the graph.

2. Parameter transformation

In this part, we will make pretreatment for the parameters in the UCG $\mathcal{R}_{k-1} = \text{diag}(R_z(\theta_0), R_z(\theta_1), \ldots, R_z(\theta_{2^{k-1}-1}))$. Letting $\theta = (\theta_0, \theta_1, \ldots, \theta_{2^{k-1}-1})$ be the input, we will derive a 2^{k-1}-length real vector α, which is the rotation angle of the R_z gate in the next part. The detailed calculating process will be presented in Algorithm 2. This step is actually the Walsh-Hadamard transformation and takes $O(2^{|V|})$ classical running time.

Algorithm 2: Parameter transformation.

input : A real vector $\theta = (\theta_0, \theta_1, \cdots, \theta_{2^{k-1}-1})$ which store the rotation angle of R_z in $\mathcal{R}_{k-1}$.
output: A new real vector $\alpha = (\alpha_0, \alpha_1, \cdots, \alpha_{2^{k-1}-1})$.

$tem = \theta[0]$;
for $i = 0; i \leq 2^{k-1} - 1; i++$**:**
  $\theta[i] = \theta[i] - tem$;
$H = \frac{1}{\sqrt{2}}[1, 1; 1, -1]$;
$\alpha = H^{\otimes k-1}\theta$;
$\alpha[0] = tem$;

3. Gate implementation

This section will introduce the synthesis algorithm for $\mathcal{R}_{k-1}$, shown in Algorithm 3. Here $\zeta(n)$ is the Ruler function, where $\zeta(n) = \max\{a : 2^{a-1} \mid n\}$. We use the $\alpha[s]$ to note the rotation angle of the phase gadget with string s. This step takes $O(2^{|V|})$ classical running time.

B. Correctness of the algorithm

This section will first introduce the phase gadget, a versatile tool commonly employed in circuit synthesis. Next, we will define the universal set of phase gadgets appropriate for different circuit classes. Finally, we demonstrate that our algorithm effectively encompasses all phase gadgets in the universal set for $\mathcal{R}_{k-1}$.

1. Phase gadget

The phase gadget is a quantum circuit $\mathcal{P}$ that $\mathcal{P}_{(s,\alpha)}|x\rangle \to e^{i\alpha\langle s,x\rangle}$, for any $(s, \alpha) \in \{0, 1\}^n \times \mathbb{R}$. A common implementation of the phase gadget is shown in Fig. 1, which contains two CNOT circuits and an R_z gate. There are some good properties of phase gadgets.

Algorithm 3: Gate implementation.

input : The graph G' generated in Alg. 1 and the parameter α in the Alg. 2.
output: The quantum circuit $\mathcal{C}$ for $\mathcal{R}_{k-1}$ on the graph G'.

s=0;
for $j = 1; j \leq 2^{k-1}; j++$**:**
  SP($\zeta(j)$) ;
  for *all vertex* i *in the shortest path from* $\zeta(j)$ *to the center vertex***:**
    $s = s \oplus 2^{i-1}$;
  Push the gate $R_z(\alpha[s])$ into $\mathcal{C}$;
SP(*vertex* q_u) **:**
  if $q_u == q_n$**:**
    return;
Find the vertex q_v that q_v which is on the shortest path from q_u to q_n and $q_u q_v \in E'$;
Push the gate CNOT_{q_u,q_v} into $\mathcal{C}$;
  SP(q_v) ;
  if $q_v \neq q_n$**:**
    Push the gate CNOT_{q_u,q_v} into $\mathcal{C}$;
  return;

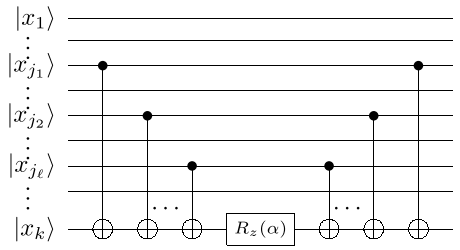

FIG. 1. The phase gadget.

(1) Since the unitary representation of the phase gadget is a diagonal unitary, any two phase gadgets commute.

(2) When we combine two phase gadgets, the CNOT gates between two R_z gates can be optimized. A simple example is shown in Fig. 2. Absolutely, the five CNOT gates in the dashed frame can be reduced to one CNOT gate on x_4 and x_5.

Thus, we can find a proper order for the circuit that consists of phase gadgets to minimize the CNOT count. More specifically, when two phase gadgets have the same s, then two phase gadgets can be merged into one.

2. Universal set

Due to these nice properties, the phase gadget is widely used in quantum circuit synthesis, especially in diagonal unitary synthesis. The ability to implement a diagonal unitary with a phase gadget circuit depends on the set of strings used in the phase gadget circuit. In [20], Bergholm *et al.* has proved that any $\mathcal{R}_{k-1}$ can be decomposed to $\Pi_{s\in S}C_{s,\alpha_s}$ and α_s can be calculated with θ, where $S := \{s'1|s' \in \{0,1\}^{k-1}\}$. We denote the set S as the universal set of phase gadgets for R_{k-1}. For general diagonal unitary Λ_k, the universal set is $\{s'|s' \in \{0,1\}^k, s' \neq 0^n\}$. Once a phase gadget circuit $\mathcal{C}$ includes phase gadgets with all the strings in the universal set for $\mathcal{F}$, then by choosing parameters appropriately, $\mathcal{C}$ can implement any unitary in $\mathcal{F}$.

3. Our algorithm reaches the universal set

We now prove the correctness of our algorithm. A stair circuit calculates the parity of all the variables in a path. As shown in Fig. 3, the result $|x_t \oplus_{j=1}^{k} x_j\rangle$ illustrates the parity of $(x_1, x_2, \ldots, x_k)$. In our algorithms, the circuit generated in the function $l_{\mathrm{SP}}()$ is a stair circuit, let $C_{ST}(\cdot)$ denote the stair circuit generate by $l_{\mathrm{SP}}(\cdot)$. Let $\zeta(n)$ be the Ruler function, where $\zeta(n) = \max\{a : 2^{a-1} \mid n\}$. We use the $code(i)$ to note the binary string in the i phase gadget.

Lemma 1. The algorithm synthesizes the circuit for $\mathcal{R}_{k-1}$.

Proof. Since we have shown the universal set for $\mathcal{R}_{k-1}$ in the previous section, we will finish the proof by proving that our algorithm enumerates all the pairs in the universal set. The choice of center vertex and the order of the vertices only affect the number of CNOT gates and will not affect the correctness of the algorithm. For convenience, we choose the $q_{|V|}$ as the center vertex in graph $G = (V, E)$, and the other vertices are reordered from small to large by the distance to the $q_{|V|}$.

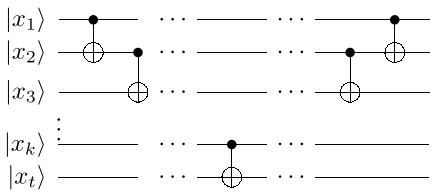

FIG. 3. The stair circuit.

To complete the proof, we first introduce the properties of Ruler function $\zeta(\cdot)$. According to the definition of $\zeta(\cdot)$, we could find that for any $t, m \in \mathbb{Z}^+, \ell \in [2^t - 1]$, $\zeta(\ell) = \zeta(m \cdot 2^t + \ell)$. Besides, for any $t \in \mathbb{Z}^+, \ell \in [t]$, $|\{x|\zeta(x) = \ell, x \in [2^t - 1]\}| = 2^{t-\ell}$. Thus in our algorithm, for any vertex q'_j, $j \in [|V| - 1]$, $C_{ST}(j)$ occurs an even number of times. Then $\mathrm{code}(2^{|V|-1}) = \mathrm{code}(0) = 0^{|V|-1}1$. We now prove the ergodicity of the code by induction on the length of prefix t. When $t = 1$, $C_{ST}[\zeta(m \cdot 2^1 + 1)] = \mathrm{CNOT}(q'_1, q'_{|V|})$, for any $m \in [2^{|V|-2} - 1]$. Thus $\mathrm{code}[m \cdot 2^1 + 1] \oplus \mathrm{code}(m \cdot 2^1) = 1(0)^{|V|-1}$. That is, $\mathrm{code}(m \cdot 2^1 + 1)$ and $\mathrm{code}(m \cdot 2^1)$ traverse all the prefixes of length 1 with a common suffix. Here, traversing all the prefix of length t with a common suffix s' means all items in $\{ps'|p \in \{0,1\}^t\}$ occur in the code. Suppose the ergodicity of our algorithm holds for $t \in [|V| - 2]$. Consider the code from $m' \cdots 2^{t+1}$ to $(m' + 1) \cdots 2^{t+1} - 1$. Without loss of generality, the $(t+1)$th letter in $\mathrm{code}(m' \cdots 2^{t+1})$ is zero and we denote $\mathrm{code}(m' \cdots 2^{t+1}) = x0y$, where $x \in \{0,1\}^t, y \in \{0,1\}^{|V|-t-1}$. According to the inductive hypothesis, the code from $m' \cdots 2^{t+1}$ to $m' \cdots 2^{t+1} + 2^t - 1$ traverses all the prefix of length t with a common suffix $0y$. And after the $C_{ST}(m' \cdots 2^{t+1} + 2^t) = C_{ST}(t+1)$, the $\mathrm{code}(\cdots 2^{t+1} + 2^t) = x'1y, x' \in \{0,1\}^t$, because $C_{ST}(t+1)$ is a stair circuit from q'_{t+1} to $q'_{|V|}$ and the vertex in the path is less than $t + 1$. According to the inductive hypothesis, the code from $m' \cdots 2^{t+1} + 2^t$ to $(m' + 1) \cdots 2^{t+1} - 1$ traverses all the prefix of length t with a common suffix $1y$. ■

The following equations can calculate the string of the phase gadget:

$$\mathrm{code}(j) = \begin{cases} 1(0)^{k-2}1 & j = 1 \\ \mathrm{code}(j-1) \oplus_{q'_t \in SP[\zeta(j-1)]} e_t & 2^{k-1} \geqslant j \geqslant 1 \end{cases}. \tag{3}$$

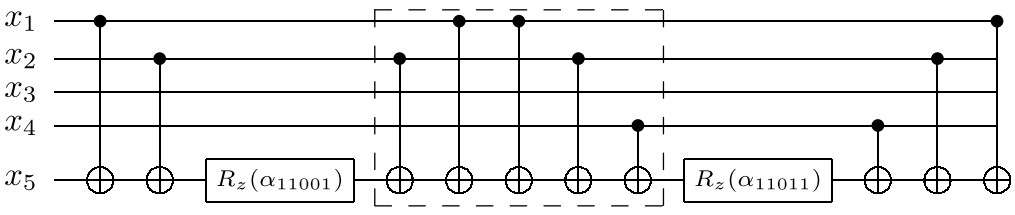

FIG. 2. The optimization of two phase gadgets.

Thus, in Algorithm 3, we calculated the correct code and implemented the correct R_z gate in the right place.

We first decompose the $\mathcal{R}_{k-1}$ into several CNOT gates and R_z gates. Then, we design the optimization algorithm for the universal set of $\mathcal{R}_{k-1}$ to minimize the CNOT count. For the CNOT gate applied on a long-distance qubit pair, we use a "stair circuit" to send the message, which may influence the order of the phase gadget. However, we then prove that the synthesis algorithm is still correct. Algorithm 2 calculated the corresponding rotation angle of each phase gadget, and Algorithm 1 found the minimal CNOT cost of our framework. Finally, in Algorithm 3, our algorithm gives the synthesis algorithm for $\mathcal{R}_{k-1}$.

C. Quantum cost

In this section, we will first analyze the total quantum cost for $\mathcal{R}_{k-1}$ synthesis on the general graph, and then we give a more detailed analysis of the cost on the specific graph. Later in this subsection, we will also show the quantum cost of other UCGs, which is the basis for general unitary synthesis, quantum state preparation circuit synthesis, and quantum isometries synthesis.

Lemma 2. For any connected graph restriction, our algorithm takes at most $\frac{5}{6}2^k + O(k)$ CNOT gates and 2^{k-1} R_z gates to synthesis any given $\mathcal{R}_{k-1}$.

Proof. In Algorithm 3, obviously, there are 2^{k-1} R_z gates used in the synthesis algorithm. As for the number of CNOT gates, it is equal to $2|l_{SP}(k-1)| - 1 + \sum_{j=1}^{2^{k-1}-1}\{2|l_{SP}[\zeta(j)]| - 1\}$, where $|l_{SP}(\cdot)|$ means the length of the shortest path from $(\cdot)$ to the center vertex:

$$2|l_{SP}(k-1)| - 1 + \sum_{j=1}^{2^{k-1}-1}\{2|l_{SP}[\zeta(j)]| - 1\} \tag{4}$$

$$= 2|l_{SP}(k-1)| - 1 + \sum_{j=1}^{k-1}[2|l_{SP}(j)| - 1]2^{k-j-1} \tag{5}$$

$$\leqslant 2(k-2) + \sum_{j=1}^{k-1}(2\lceil j/2\rceil - 1)2^{k-1-j} = \frac{5}{6}2^k + O(k). \tag{6}$$

The equal sign between Eqs. (4) and (5) holds from the definition of the Ruler function. Meanwhile, if we choose the middle vertex of the longest path in G as the center vertex, then $|l_{SP}(j)| \leqslant \lfloor j/2\rfloor$ holds for any j. Then, the CNOT count of our algorithm for arbitrary connectivity architecture is no more than the result in Eq. (6). ■

In Lemma 2, we show the upper bound of CNOT count for all connectivity architectures. In the NISQ era, LNN, the parallel ladder graph, and the square grid graph are the most popular architectures. Next, we will show that our algorithm works better for all these architectures. First, the case of LNN is the same as the proof of Lemma 2. Then we will discuss the other architectures. For the synthesis of $\mathcal{R}_{k-1}$ on the parallel ladder graph, when $k \mod 4 \equiv 0$, the center vertex is the middle vertex at the graph and the CNOT count is less than $k - 2 + 7 \times 2^{k-4} + \sum_{j=2}^{\lceil (k-3)/4\rceil} 15(2j-1)2^{k-4j} \leqslant \frac{19}{30}2^k + k - 2$. When $k \mod 4 \neq 0$, we need at most $28k$ additional CNOT gates. Second, for the synthesis of $\mathcal{R}_{k-1}$ on a two-dimensional grid graph, the center qubit is the qubit in row $\sqrt{k}/2$ and column $\sqrt{k}/2$. Similar to the analysis in the $2 \times k/2$ grid, the CNOT count is less than $0.563(2^k) + 2\sqrt{k-1} - 1$. Third, we will consider a more general graph, which is the graph with degree d. For the synthesis of $\mathcal{R}_{k-1}$ on this graph, the center qubit can be the vertex with the maximum degree. And $\mathrm{dis}(v, \pi(v_i)) \leqslant \max(1, i - d + 1)$, which leads to the result that the CNOT count is less than $(0.5 + 2^{1-d})2^k + O(k)$. In conclusion, the above synthesis results on the NISQ architectures can be summarized in the following corollary.

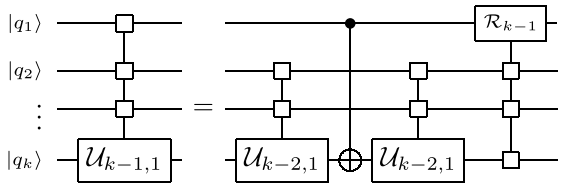

FIG. 4. The decomposition of $\mathcal{U}_{k-1,1}$. Here we use the square to represent the uniformly control qubits.

Corollary 1 (NISQ architectures). The CNOT counts for the synthesis of $\mathcal{R}_{k-1}$ on different NISQ architectures are at most (1) $\frac{19}{30}2^k + k - 2$ for the $2 \times k/2$ grid graph, (2) $0.563(2^k) + 2\sqrt{k-1} - 1$ for the square $(\sqrt{k} \times \sqrt{k})$ grid graph, and (3) $(0.5 + 2^{1-d})2^k + O(k)$ for the graph with degree d.

Based on the synthesis of $\mathcal{R}_{k-1}$, we can recursively decompose any general uniformly $k-1$ controlled single-qubit gate $\mathcal{U}_{k-1,1}$ into $2^k - 1$ CNOT gates, 2^k single-qubit gates, and a k-qubit diagonal unitary Λ_k using the circuit shown in Figs. 4 and 5. As shown in Fig. 4, each k qubit UCG $\mathcal{U}_{k-1,1}$ can be decomposed to a CNOT gate, two $\mathcal{U}_{k-2,1}$, and a $\mathcal{R}_{k-1}$. Recursively decompose the $\mathcal{U}$ until all $\mathcal{U}$s are decomposed to CNOT or single-qubit gates. Due to the transformation shown in Fig. 5, all the uniformly controlled R_z gates can be removed to the rightmost and merge to a diagonal unitary Λ_k with computable changes in the rotation angle. When the implementation of the CNOT gate is limited, we can replace the CNOT gate in Fig. 4 with a CNOT circuit generated by $SP(1)$ in Algorithm 3. The analysis of the number of CNOT gates is the same as the number of CNOT gates of $\mathcal{R}_{k-1}$. Let $c(\cdot)$ be the CNOT count of unitaries, $c(\mathcal{U}_{k-1,1}) = c(\mathcal{R}_{k-1}) + c(\Lambda_k)$, where Λ_k means the k-qubit diagonal unitary. As shown in Fig. 5, the diagonal unitary gate can always move to the rightmost of the circuit with some changes in the rotation angles. Thus, if

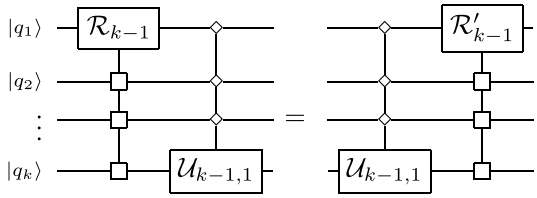

FIG. 5. The "commutativity" of $\mathcal{R}_{k-1}$ and $\mathcal{U}_{k-1,1}$. The diamond means for any 0/1 controlled, the equation holds. The prime symbol on $\mathcal{R}_{k-1}$ means the rotation angle may change.

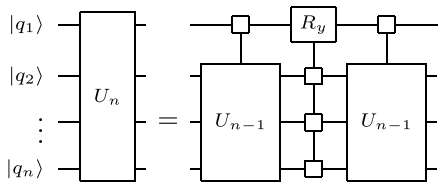

FIG. 6. The circuit of CSD.

there are t $\mathcal{U}_{k-1,1}$ used in the quantum circuit, the circuit can be decomposed to $t \cdot c(\mathcal{R}_k - 1)$ CNOT gates, $O(t2^k)$ single-qubit gates, and one diagonal circuit at the rightmost. The diagonal unitary gate Λ_k can be further realized by k $\mathcal{R}$ gates, such that $\Lambda_k = \mathcal{R}_0\mathcal{R}_1\mathcal{R}_2\cdots\mathcal{R}_{k-1}$.

III. CIRCUIT SYNTHESIS WITH UCGs

A. Unitary synthesis

We will first explain how to decompose the most general unitary matrix into elementary gates based on the synthesis of $\mathcal{U}_{k,1}$. Using CSD [20], as shown in Fig. 6, any n-qubit unitary matrix U_n can be decomposed into two uniformly controlled $(n-1)$-qubit gates and a uniformly controlled R_y gate. And the uniformly controlled $(n-1)$-qubit gate can be further decomposed into two $(n-1)$-qubit unitary matrices U_{n-1} and an $\mathcal{R}_{n-1}$, as shown in Fig. 7.

Let the CNOT count of a unitary matrix U be $c(U)$. Then we have

$$c(U_n) = 4c(U_{n-1}) + 3c(\mathcal{R}_{n-1}). \tag{7}$$

And it has been proven that any $U \in SU(4)$ can be implemented with two CNOT gates and several other single-qubit gates. Using some combinational skills, we derive

$$c(U_n) = \frac{1}{8}4^n + 3\sum_{j=0}^{n-3} 4^j c(\mathcal{R}_{n-j-1}). \tag{8}$$

B. Quantum state preparation

In this section, we will explain how to use the synthesis of $\mathcal{U}_{k,1}$ to prepare the corresponding quantum state $|\psi\rangle = \frac{1}{\|v\|_2}\sum_{j=0}^{2^n-1} v_j|j\rangle$ for a given complex vector $v = (v_0, v_1, \ldots, v_{2^n-1}) \in \mathbb{C}^{2^n}$. Actually the task is to design a quantum circuit $\mathcal{C}$ such that $\mathcal{C}|0\rangle^{\otimes n} = |\psi\rangle$. There are two well-known frameworks for general quantum state preparation, and UCG is indispensable in both.

The first framework, which first appeared in [25], is to prepare the quantum state qubit by qubit. Using a single-qubit gate U, we can easily transform the original $|0\rangle$ to any single-qubit state $|\phi\rangle = \alpha|0\rangle + \beta|1\rangle$. Next, any two-qubit

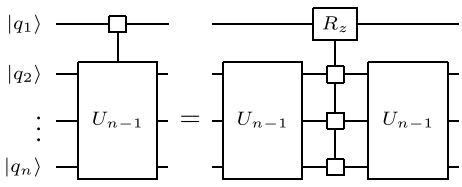

FIG. 7. The decomposition of $\mathcal{U}_{1,n-1}$.

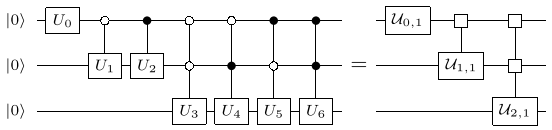

FIG. 8. The binary tree based framework for three-qubit QSP. Here $\mathcal{U}_{k,m}$ consists of two $\mathcal{R}_k$ and two single-qubit gates.

state can be derived by operating the corresponding uniformly 1-controlled-1 gates. Thus, the final step is to prepare the target n-qubit state from an $(n-1)$-qubit state, which can be completed by a uniformly controlled unitary. That is, for any $j \in \{0, 1\}^{n-1}$, the following equation holds:

$$\mathcal{U}_{n-1,1}\frac{\sqrt{v_{2j}^2 + v_{2j+1}^2}}{\|v\|_2}|j\rangle|0\rangle \to \frac{1}{\|v\|_2}|j\rangle(v_{2j}|0\rangle + v_{2j+1}|1\rangle).$$

The whole process has been demonstrated in Fig. 8 where we take a three-qubit state preparation as an example. The uniformly controlled single-qubit unitary $\mathcal{U}_{n-1,1}$ used in the first framework can be decomposed to an R_y gate and an R_z gate. Therefore the CNOT count

$$c\left(U_n^{QSP}\right) = \sum_{i=1}^{n} 2c(\mathcal{R}_{i-1}).$$

This framework is also suitable for sparse quantum state preparation. We still prepare the state qubit by qubit. Unlike the general QSP, we will apply some multicontrolled gates rather than a UCG when the leaf is sparse. Based on this idea, we design a framework for sparse QSP, which performs well on any connectivity architecture.

Lemma 3. Any s-sparse n-qubit quantum state can be prepared by an $O(sn)$-size quantum circuit on any connectivity architecture.

Proof. Since the circuit $\mathcal{C}'|\psi\rangle \to |0\rangle$ is the inverse of the circuit $\mathcal{C}|0\rangle \to |\psi\rangle$, we give an algorithm to generate the circuit $\mathcal{C}'$ rather than $\mathcal{C}$. Below is the detailed algorithm.

(1) Sort the qubit by topological order. Let the qubit set be $\mathcal{Q}$ and the binary tree be $\mathcal{T}$. The depth of $\mathcal{T}$ is $n+1$; the root node is at the depth 1.

(2) Count the number of nodes which have two leaves at the depth $\mathcal{Q}$; denote by t. Let $d = \|\mathcal{Q}\|$.

(3) For the nodes with two leaves, we first apply $O(n)$ CNOT gates and a multi-controlled-toffoli (MCT) to make two leaves only differ from the qubits d. If $t = O(2^d/n)$, apply t multicontrolled gates to merge the leaves pair by pair. And if $t = \omega(2^d/n)$, apply a UCG to merge $2t$ leaves into t leaves.

(4) Pop the qubit d in $\mathcal{Q}$.

(5) Do step 2 to step 4 until $\mathcal{Q} = \emptyset$.

The s-controlled Toffoli can be decomposed to $O(s)$ CNOT and single-qubit gates even on an LNN device. The cost of UCG is analyzed in the previous section. There are two situations in step 3. For the first one, we use the $O(n)$ CNOT gate and a single-qubit gate to merge two leaves. We use $O(2^d)$ gates in the second situation to merge $\omega(2^d/n)$ leaves. Thus, $O(sn)$ elementary gates are enough to merge s leaves into one. ■

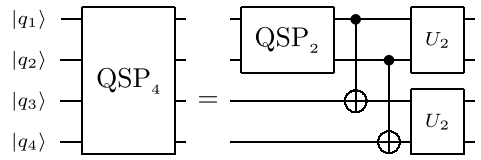

FIG. 9. The Schmidt decomposition based framework for four-qubit quantum state preparation.

We now introduce the second framework for quantum state preparation. When $n = 2k$ is even, we first use Schmidt decomposition. Then any n qubits state

$$|\psi\rangle = \sum_{j=0}^{2^k-1} \alpha_j |\psi(j)\rangle_A |\phi(j)\rangle_B.$$

A and B are k qubit state space. Then we use the k qubit state preparation circuit to generate the state $\sum_{j=0}^{2^k-1} |j\rangle_A |0\rangle_B$. Next the k CNOT gate is applied to get the state $\sum_{j=0}^{2^k-1} |j\rangle_A |j\rangle_B$. Finally, we apply the unitary $U_A|j\rangle \to |\psi(j)\rangle$ on the A subspace and $U_B|j\rangle \to |\phi(j)\rangle$ on the B subspace. We demonstrate in Fig. 9 to prepare a four-qubit state with Schmidt decomposition. The CNOT count

$$c\big(U_n^{QSP}\big) = c\big(U_{n/2}^{QSP}\big) + 2c(U_{n/2}) + 2.$$

Both frameworks can implement the quantum state preparation, but there still exist some differences between them. The first framework is easy to understand and can be used to synthesize quantum isometries and sparse quantum state preparation. The second framework takes less CNOT count for general quantum state preparation; we use the second framework to calculate the quantum costs, shown in Table I.

C. Quantum isometries

This section will present how to synthesize quantum isometry, which is the transformation between two Hilbert spaces [12]. Both quantum state preparation and unitary synthesis are special cases of quantum isometries. If we use $U_{m,n}^{\mathcal{ISO}}$ $(m \leqslant n)$ to describe an isometry from a 2^m-dimensional Hilbert space to a 2^n-dimensional Hilbert space, then $U_{0,n}^{\mathcal{ISO}} = U_n^{QSP}$ and $U_{m,n}^{\mathcal{ISO}} = U_n$. Mathematically, $U_{m,n}^{\mathcal{ISO}}$ can be represented by a $2^n \times 2^m$ matrix $U_{m,n}^{\mathcal{ISO}} = U_n I_{2^n \times 2^m}$, where $I_{2^n \times 2^m} = [{}^{I^{\otimes m}}_{0}]$. Notice that for a specific isometry, the U_n is not unique. The synthesis for quantum isometry is to find the quantum circuit for U_n with as few elementary gates as possible.

Since the reverse of the CNOT gate and single qubit is easy to calculate, to construct the isometry $U_{m,n}^{\mathcal{ISO}}$ is equal to finding a circuit $\mathcal{C}$ such that $\mathcal{C}U_{m_n}^{ISO} = I_{2^n \times 2^m}$. Usually, we synthesize a quantum isometry column by column [12]. We will find a sequence of $\mathcal{G}_k$, which satisfies the following conditions.

(1) For $j \in [2^m - 1]$, $\mathcal{G}_j \mathcal{G}_{j-1} \cdots \mathcal{G}_0 (U_{m,n}^{\mathcal{ISO}})|j\rangle_m = |j\rangle_n$.

(2) For any $k > j \in \{0, 1, \ldots, 2^m - 1\}$, $\mathcal{G}_k |j\rangle_n = |j\rangle_n$.

The first condition is easily met using a quantum state preparation circuit. However, the second condition cannot be met using the quantum state preparation introduced in the previous section. In [12], they modify the quantum state preparation circuit to meet the second condition by slightly increasing the number of CNOT gates.

The design of $\mathcal{G}_0$ is the same as the first framework in the quantum state preparation section. The CNOT count is $\sum_{k=1}^{n} c(\mathcal{R}_{k-1}) + c(\Lambda_n)$. We now design the circuit for $\mathcal{G}_j$. After $\mathcal{G}_{j-1} \cdots \mathcal{G}_1$, the first j column and the first j row (which we will call "fixed block") are the first j column and the first j row of an n-qubit identity unitary $I^{\otimes n}$. When we want to apply a uniformly controlled gate, the unitary effect on the fixed block should be identity unitary I. And then apply a n-qubit controlled single-quit unitary. Since both the uniformly controlled gate and the n-qubit controlled gate do not affect the first j column, the $\mathcal{G}_j$ can transform the column j into $(0, 0, \ldots, \overbrace{1}^{j\text{th}}, \ldots, 0)^T$, without affecting the first j column. Notice that all the diagonal unitary can be removed to the rightmost without an increase in the number of CNOT gates. The CNOT count for $U_{m,n}^{\mathcal{ISO}} = 2^m \sum_{k=1}^{n} c(\mathcal{R}_{k-1}) + c(\Lambda_n) + (2^m - 1)(n-1)c(U_n^{MCG}) = 2^m \sum_{k=1}^{n} c(\mathcal{R}_{k-1}) + o(2^{m+n})$.

IV. EVALUATION

In this section, we implement our algorithms on the specific unitary to illustrate the quantum cost of our algorithm. We also compare our algorithm with the resulting circuit generated by mapping [24] after synthesis with the algorithm on the complete graph [20]. Notice that the circuit from [20] has a good property. Thus, it can be easily proved that the mapping by SABRE is optimal, which means our algorithms have an advantage over any mapping algorithms.

A. Numerical simulation setup

1. Evaluation indicator

Since the number of single-qubit gates is the same in different synthesis algorithms and the difference between different algorithms lies in the CNOT count of the resulting circuit, we choose the CNOT count as the evaluation indicator.

2. Benchmark setup

The CNOT count of general unitary, state preparation unitary, and quantum isometries can be easily computed by the formula in the Sec. III; we will choose random $\mathcal{R}_{k-1}$ to evaluate the performance of different methods. We choose random $\mathcal{R}_{k-1}, k = 5, 10, 15, 20$ as the benchmarks. The following steps generate the k-qubit random $\mathcal{R}_{k-1}$. First we random sample 2^{k-1} random number $x_1, x_2, \ldots, x_{2^{k-1}}$ in [0,1] as the corresponding angles in $\mathcal{R}_{k-1}$, which means $\mathrm{diag}(R_z(2x_1\pi), R_z(2x_1\pi), \ldots, R_z(2x_{2^k-1}\pi))$ is the random $\mathcal{R}_{k-1}$. We mainly choose five connectivity architectures of different devices: the IBM Q device [26] (20-qubit device and heavy-hex device), a ladderlike 24 qubits device [27], and the LNN device; the connectivity restrictions of each device are shown in Fig. 10.

B. Compared to theoretical analysis

We start the evaluation by comparing the theoretical analysis and the CNOT count of the resulting circuit. We list the

TABLE II. The CNOT count of our algorithm on different quantum devices for $\mathcal{R}_{k-1}$. The column "Theoretical" records the CNOT count upper bound calculated by the formula in Table I, and the "Experiment" column records the CNOT count of the result circuit of our algorithms.

	IBMQ 20-qubit device		IBMQ heavy-hex device		Ladder		LNN		Complete graph	
k	Theoretical	Experiment	Theoretical	Experiment	Theoretical	Experiment	Theoretical	Experiment	Theoretical	Experiment
5	17	16	21	20	20	20	26	24	16	16
10	544	528	656	656	648	648	853	852	512	512
15	17408	16896	21004	21000	21173	20752	27306	27304	16384	16384
20	557056	540680	672137	672004	664098	664096	873813	873812	524288	524288

CNOT count of $\mathcal{R}_{k-1}$ on different architectures in Table II. Due to the design of our algorithms, the CNOT count is a fixed number with a high probability. The theoretical result is shown in the column "Theoretical," calculated by the formulas in Table I. For the IBMQ 20-qubit device, we choose the formula of the graph with max degree 6, that is, $(\frac{17}{32})2^n$ CNOT gates. For the ladderlike architecture, the formula of CNOT count is on the "$2 \times n/2$-grid" row in Table I. The formula for LNN and the complete graph are shown in the corresponding rows in the Table I. The "Experiment" column records the numerical simulation result of the CNOT count of the circuit generated by our algorithms. The average error between the theoretical and numerical simulation results is about 2%. The additional CNOT count of different connectivity architecture, calculated by the CNOT count of corresponding connectivity architecture minus the CNOT count of the complete graph, is different. The additional CNOT count of IBMQ20 is significantly less than the ladderlike architecture due to IBMQ20 containing a qubit that can entangle with six other qubits. The numerical simulation result shows that the CNOT cost of our algorithm is consistent with the theoretical analyses. The differentiation of the CNOT count under different connectivity restrictions may suggest the design of the large-scale quantum device.

C. Compared to mapping after synthesis

Then, we compare the performance of our algorithm and mapping after synthesis. The CNOT counts of different methods are shown in Table III. The column "Additional" records the CNOT of the additional CNOT count of the corresponding architecture. For convenience, let the additional CNOT count of mapping after synthesis be $\text{AddCNOT}_{\text{Map}}$ and the additional CNOT count of our algorithm be $\text{AddCNOT}_{\text{Syn}}$. The reduce ratio shows the ratio of

$$\frac{\text{AddCNOT}_{\text{Map}} - \text{AddCNOT}_{\text{Syn}}}{\text{AddCNOT}_{\text{Map}}}.$$

For the $\mathcal{R}_{k-1}$ synthesis on the IBMQ 20 qubits device and ladderlike devices, we count the CNOT cost of the synthesis algorithm and mapping algorithm (SABRE). When $k = 5$, the CNOT count generated by applying the SABRE on the circuit resulting from the synthesis algorithm on the complete graph is 25 (16 CNOT gates and four swap gates) on a ladderlike device. However, our algorithm synthesizes a 20 CNOT gates circuit on the ladderlike device, which saves more than 55.6% additional CNOT gates. For larger-scale cases, our algorithm achieves ≈66.6% additional CNOT count reduction compared to additional CNOT gates used in the SABRE. Even if the mapping algorithm can add a SWAP gate or CNOT gate, which means it may use a bridgelike CNOT circuit to save the CNOT count, our algorithm still achieves a ≈50% reduction on additional CNOT gates. The results when $k \leqslant 10$ reduce by about 20–30% total CNOT count on the gridlike architecture, such as ladderlike and heavy-hex devices. The total CNOT count is available in the NISQ era. Our algorithms also showcase their generalization in the coming fault-tolerant quantum era. The results show that synthesis with the connectivity architecture is sound when synthesizing the unitary.

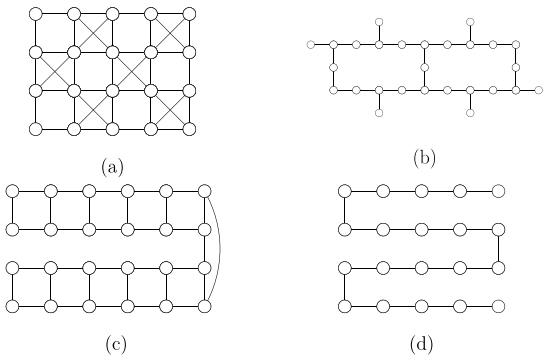

FIG. 10. The connectivity restriction graphs (a) IBMQ 20 qubits device (partial), (b) IBMQ heavy-hex device, (c) a ladderlike 24 qubits graph, and (d) 20 qubits LNN.

V. CONCLUSION AND FUTURE WORK

In this paper, we design several synthesis algorithms for different kinds of unitary matrices under any qubit-connectivity architecture. For any architecture, the CNOT count is about 5/3 times the state-of-the-art (SOTA) result on complete-graph architecture. The ratio can be further reduced for some specific architecture. We also give the relationship between the max degree of the graph and the ratio. The algorithm reaches a ≈50 to 75% reduction on additional CNOT count compared to the mapping after the synthesis algorithm. Our results quantitatively illustrate in theory that the impact of NISQ connectivity architectures can be mitigated to an inconspicuous level by well-designed synthesis algorithms. We also compare the additional CNOT count of different architectures, which may suggest the design of

TABLE III. The CNOT count of our algorithm and the synthesis algorithm [20] + mapping algorithm on different quantum devices for $\mathcal{R}_{k-1}$. The column "Complete graph" records the CNOT count of the algorithm [20] on the complete graph, the "Synthesis + mapping" column records the CNOT count of the resulting circuit after mapping by SABRE, and the "Our algorithm" column records the CNOT count of our algorithms shown in Algorithm 3. The "Total" column records the total CNOT count of the circuit, and the "Additional" column records the CNOT count of the additional SWAP gates or the CNOT gates. The reduce ratio means the ratio of CNOT gates we save compared to the synthesis + mapping.

		IBMQ 20-qubit device					Ladder				
		Synthesis+mapping		Our algorithm		Reduce	Synthesis+mapping		Our algorithm		Reduce
k	Complete graph	Total	Additional	Total	Additional	ratio (%)	Total	Additional	Total	Additional	ratio (%)
5	16	16	0	16	0	0.0	25	9	20	4	55.6
10	512	557	45	528	16	64.4	914	402	648	136	66.1
15	16384	17917	1533	16896	512	66.6	29479	13095	20752	4368	66.6
20	524288	573458	49170	540680	16392	66.7	943700	419412	664096	139808	66.7

quantum devices. In the future, we wonder whether there is a synthesis algorithm that can eliminate the impact of connectivity architectures. Or is there an essential gap? Can we give the lower bound of the CNOT count for specific connectivity architecture?

ACKNOWLEDGMENT

This work was supported in part by the National Natural Science Foundation of China Grants No. 62325210 and No. 62272441, and the Strategic Priority Research Program of Chinese Academy of Sciences Grant No. XDB28000000.

[1] P. W. Shor, Polynominal time algorithms for discrete logarithms and factoring on a quantum computer, in *Proceedings of the Algorithmic Number Theory, First International Symposium, ANTS-I*, 1994 (unpublished).

[2] L. K. Grover, A fast quantum mechanical algorithm for database search, in *Proceedings of the 28th Annual ACM Symposium on Theory of Computing* (ACM Press, New York, NY, 1996), pp. 212–219.

[3] A. W. Harrow, A. Hassidim, and S. Lloyd, Quantum algorithm for linear systems of equations, Phys. Rev. Lett. **103**, 150502 (2009).

[4] A. M. Childs and N. Wiebe, Hamiltonian simulation using linear combinations of unitary operations, Quantum Inf. Comput. **12**, 901 (2012).

[5] Y. Cao, J. Romero, J. P. Olson, M. Degroote, P. D. Johnson, M. Kieferová, I. D. Kivlichan, T. Menke, B. Peropadre, N. P. D. Sawaya *et al.*, Quantum chemistry in the age of quantum computing, Chem. Rev. **119**, 10856 (2019).

[6] Y. Wang, G. Li, and X. Wang, A hybrid quantum-classical hamiltonian learning algorithm, Sci. China Inf. Sci. **66**, 129502 (2023).

[7] A. Barenco, C. H. Bennett, R. Cleve, D. P. DiVincenzo, N. Margolus, P. Shor, T. Sleator, J. A. Smolin, and H. Weinfurter, Elementary gates for quantum computation, Phys. Rev. A **52**, 3457 (1995).

[8] Z. Sasanian, R. Wille, and D. M. Miller, Realizing reversible circuits using a new class of quantum gates, in *DAC Design Automation Conference 2012* (IEEE, New York, 2012), pp. 36–41.

[9] M. Soeken, M. Roetteler, N. Wiebe, and G. De Micheli, Hierarchical reversible logic synthesis using luts, in *2017 54th ACM/EDAC/IEEE Design Automation Conference (DAC)* (IEEE, New York, 2017), pp. 1–6.

[10] G. Li, Y. Ding, and Y. Xie, Eliminating redundant computation in noisy quantum computing simulation, in *2020 57th ACM/IEEE Design Automation Conference (DAC)* (IEEE, New York, 2020), pp. 1–6.

[11] N. Gleinig and T. Hoefler, An efficient algorithm for sparse quantum state preparation, in *2021 58th ACM/IEEE Design Automation Conference (DAC)* (IEEE, New York, 2021), pp. 433–438.

[12] R. Iten, R. Colbeck, I. Kukuljan, J. Home, and M. Christandl, Quantum circuits for isometries, Phys. Rev. A **93**, 032318 (2016).

[13] X. Sun, G. Tian, S. Yang, P. Yuan, and S. Zhang, Asymptotically optimal circuit depth for quantum state preparation and general unitary synthesis, in *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* (IEEE, New York, 2023).

[14] L. Li, C. Guo, Q. Li, and X. Sun, Fast exact synthesis of two-qubit unitaries using a near-minimum number of t gates, Phys. Rev. A **107**, 042424 (2023).

[15] S. Ashhab, N. Yamamoto, F. Yoshihara, and K. Semba, Numerical analysis of quantum circuits for state preparation and unitary operator synthesis, Phys. Rev. A **106**, 022426 (2022).

[16] A. Shafaei, M. Saeedi, and M. Pedram, Optimization of quantum circuits for interaction distance in linear nearest neighbor architectures, in *2013 50th ACM/EDAC/IEEE Design Automation Conference (DAC)* (IEEE, New York, 2013), pp. 1–6.

[17] R. Wille, L. Burgholzer, and A. Zulehner, Mapping quantum circuits to IBM QX architectures using the minimal number of swap and H operations, in *2019 56th ACM/IEEE Design Automation Conference (DAC)* (IEEE, New York, 2019), pp. 1–6.

[18] J. Allcock, P. Yuan, and S. Zhang, Does qubit connectivity impact quantum circuit complexity? arXiv:2211.05413 (2022).

[19] B. Wu, X. He, S. Yang, L. Shou, G. Tian, J. Zhang, and X. Sun, Optimization of CNOT circuits on topological superconducting processors, Phys. Rev. Res. **5**, 013065 (2023).

[20] V. Bergholm, J. J. Vartiainen, M. Möttönen, and M. M. Salomaa, Quantum circuits with uniformly controlled one-qubit gates, Phys. Rev. A **71**, 052330 (2005).

[21] M. G. Davis, E. Smith, A. Tudor, K. Sen, I. Siddiqi, and C. Iancu, Towards optimal topology aware quantum circuit synthesis, in *2020 IEEE International Conference on Quantum Computing and Engineering (QCE)* (IEEE, New York, 2020), pp. 223–234.

[22] M. Weiden, J. Kalloor, J. Kubiatowicz, Ed. Younis, and C. Iancu, Wide quantum circuit optimization with topology aware synthesis, in *2022 IEEE/ACM Third International Workshop on Quantum Computing Software (QCS)* (IEEE, New York, 2022), pp. 1–11.

[23] V. V. Shende, S. S. Bullock, and I. L. Markov, Synthesis of quantum-logic circuits, IEEE Trans. Computer-Aided Design Integr. Circuits Syst. **25**, 1000 (2006).

[24] G. Li, Y. Ding, and Y. Xie, Tackling the qubit mapping problem for nisq-era quantum devices, in *Proceedings of the 24th International Conference on Architectural Support for Programming Languages and Operating Systems*, 2018 (unpublished).

[25] L. Grover and T. Rudolph, Creating superpositions that correspond to efficiently integrable probability distributions, arXiv:quant-ph/0208112 (2002).

[26] IBM Q, IBM quantum device, https://quantum-computing.ibm.com/.

[27] Y. Ye, Z.-Y. Ge, Y. Wu, S. Wang, M. Gong, Y.-R. Zhang, Q. Zhu, R. Yang, S. Li, F. Liang, J. Lin, Y. Xu, C. Guo, L. Sun, C. Cheng, N. Ma, Z. Y. Meng, H. Deng, H. Rong, C.-Y. Lu *et al.*, Propagation and localization of collective excitations on a 24-qubit superconducting processor, Phys. Rev. Lett. **123**, 050502 (2019).