

Distributed quantum circuit partitioning and optimization based on combined spectral clustering and search tree strategies

Zilu Chen(陈子禄), Zhijin Guan(管致锦), Shuxian Zhao(赵书娴), and Xueyun Cheng(程学云)

Citation: Chin. Phys. B, 2025, 34 (**5**): 050305. DOI: 10.1088/1674-1056/adbada
Journal homepage: http://cpb.iphy.ac.cn; http://iopscience.iop.org/cpb

What follows is a list of articles you may be interested in

Asymmetric mode-pairing quantum key distribution with advantage distillation

Hai-Tao Wang(汪海涛), Chun Zhou(周淳), Yi-Fei Lu(陆宜飞), Chen-Peng Hao(郝辰鹏), Yan-Mei Zhao(赵燕美), Yan-Yang Zhou(周砚扬), Hong-Wei Li(李宏伟), and Wan-Su Bao(鲍皖苏)

Chin. Phys. B, 2025, 34 (**4**): 040305. DOI: 10.1088/1674-1056/adbb5c

Global receptive field transformer decoder method on quantum surface code data and syndrome error correction

Ao-Qing Li(李熬庆), Ce-Wen Tian(田策文), Xiao-Xuan Xu(徐晓璇), Hong-Yang Ma(马鸿洋), and Jun-Qing Liang(梁俊卿)

Chin. Phys. B, 2025, 34 (**3**): 030306. DOI: 10.1088/1674-1056/adab63

Automatic architecture design for distributed quantum computing

Ting-Yu Luo(骆挺宇), Yu-Zhen Zheng(郑宇真), Xiang Fu(付祥), and Yu-Xin Deng(邓玉欣)

Chin. Phys. B, 2024, 33 (**12**): 120302. DOI: 10.1088/1674-1056/ad7c2c

Hierarchical simultaneous entanglement swapping for multi-hop quantum communication based on multi-particle entangled states

Guang Yang(杨光, Lei Xing(邢磊), Min Nie(聂敏), Yuan-Hua Liu(刘原华), and Mei-Ling Zhang(张美玲)

Chin. Phys. B, 2021, 30 (**3**): 030301. DOI: 10.1088/1674-1056/abcf3d

Probabilistic direct counterfactual quantum communication

Sheng Zhang(张盛)

Chin. Phys. B, 2017, 26 (**2**): 020304. DOI: 10.1088/1674-1056/26/2/020304

Distributed quantum circuit partitioning and optimization based on combined spectral clustering and search tree strategies

Zilu Chen(陈子禄)[1], Zhijin Guan(管致锦)[1,2,†], Shuxian Zhao(赵书娴)[3], and Xueyun Cheng(程学云)[1,‡]

[1] *School of Artificial Intelligence and Computer Science, Nantong University, Nantong* 226019, *China*
[2] *College of Engineering the Internet of Things, Taihu University, Wuxi* 214063, *China*
[3] *Jiangsu Huihuan Environmental Protection Technology Co., Ltd., Nantong* 226010, *China*

(Received 24 December 2024; revised manuscript received 24 February 2025; accepted manuscript online 27 February 2025)

In the current noisy intermediate-scale quantum (NISQ) era, a single quantum processing unit (QPU) is insufficient to implement large-scale quantum algorithms; this has driven extensive research into distributed quantum computing (DQC). DQC involves the cooperative operation of multiple QPUs but is concurrently challenged by excessive communication complexity. To address this issue, this paper proposes a quantum circuit partitioning method based on spectral clustering. The approach transforms quantum circuits into weighted graphs and, through computation of the Laplacian matrix and clustering techniques, identifies candidate partition schemes that minimize the total weight of the cut. Additionally, a global gate search tree strategy is introduced to meticulously explore opportunities for merged transfer of global gates, thereby minimizing the transmission cost of distributed quantum circuits and selecting the optimal partition scheme from the candidates. Finally, the proposed method is evaluated through various comparative experiments. The experimental results demonstrate that spectral clustering-based partitioning exhibits robust stability and efficiency in runtime in quantum circuits of different scales. In experiments involving the quantum Fourier transform algorithm and Revlib quantum circuits, the transmission cost achieved by the global gate search tree strategy is significantly optimized.

Keywords: NISQ era, distributed quantum computing, quantum circuit partitioning, transmission cost

PACS: 03.67.–a, 03.67.Lx, 03.67.Ac

DOI: 10.1088/1674-1056/adbada **CSTR:** 32038.14.CPB.adbada

1. Introduction

Quantum computing represents a revolution in the field of computational science, leveraging the superposition and entanglement properties of qubits to demonstrate tremendous potential in solving complex problems that are challenging for classical computing.[1,2] Although current quantum computers are making continuous progress in the number and performance of qubits, single devices remain limited in scale and performance due to experimental and engineering constraints,[3,4] making it difficult to meet the demands of large-scale quantum algorithms. To overcome these limitations, distributed quantum computing has been proposed as a viable solution.[5] Distributed quantum computing achieves the collaborative completion of large-scale quantum computational tasks by assigning quantum tasks to multiple quantum processing units (QPUs).[6] Establishing classical or quantum channels between multiple QPUs can optimize the interactions and control of qubits. This enables efficient quantum information transmission and entanglement management in a distributed environment, enhancing system stability and reliability. The distributed quantum computing model provides a key pathway to surpass the qubit number limitations, promoting the broader development of quantum computing in practical applications.[7–9]

Distributed quantum computing (DQC) holds significant potential for enhancing the scale and performance of quantum computations, but it faces a major challenge in the form of high communication costs.[10] Since distributed quantum computing requires the transmission of qubits and global gate operations between multiple QPUs, it necessitates the establishment and maintenance of entangled states to ensure accurate quantum information transfer. However, these operations are often accompanied by substantial generation delays and the risk of coherence loss,[11] significantly increasing the system's communication cost. Therefore, reducing communication cost in DQC has become a crucial research direction.

Numerous studies have proposed various DQC partitioning methods to reduce the frequency of quantum communication and enhance computational efficiency. Reference [12] proposed the two-step DQC algorithm, which solves the balanced k-min-cut problem on an edge-weighted graph and then calculates communication cost using an approximation algorithm. Building upon Ref. [12], Ref. [13] further considered the allocation of quantum circuits on quantum networks of heterogeneous quantum computers, using a tabu search algorithm to find the allocation of qubits. Reference [14] converted quan-

†Corresponding author. E-mail: guan.zj@ntu.edu.cn
‡Corresponding author. E-mail: chen.xy@ntu.edu.cn

© 2025 Chinese Physical Society and IOP Publishing Ltd. All rights, including for text and data mining, AI training, and similar technologies, are reserved.

tum circuits into bipartite graph models and then used dynamic programming methods to partition the models into smaller-capacity quantum circuits. The automatic window partitioning method introduced in Ref. [15] utilized temporal information, adjusting partitioning based on changes in qubit connection patterns during circuit execution. Reference [16] introduced a genetic algorithm-based circuit partitioning method aimed at minimizing the communication cost between two partitions in distributed quantum circuit design. Reference [17] proposed a connection matrix model for quantum circuits and used NSGA-III to divide the quantum circuit into two balanced parts with an equal number of qubits and the fewest global gates. In Ref. [18], quantum circuits were reordered based on qubit positions and combined genetic algorithms with an improved tabu search algorithm to partition the quantum circuit graph model. Reference [19] employed an integer linear programming model to distribute a single quantum system into K balanced partitions. Reference [20] proposed an automated design framework called AutoArch for quantum chip networks (QCNs). Its aim is to reduce the number of remote Einstein–Podolsky–Rosen (EPR) pairs and additional local SWAP gates required during the execution of quantum algorithms. Furthermore, in Ref. [21], the focus was on how to effectively assign logical qubits to different QPUs in compiler design, utilizing a multi-level hybrid simulated annealing algorithm for qubit allocation. However, existing studies still have some limitations, such as unbalanced qubit distributions after partitioning or they rely solely on minimum cut results in graph partitioning, making it difficult to obtain more optimal distributed quantum circuit partitioning schemes.

This paper employs direct quantum state transmission as the transmission technique within distributed quantum circuits. The number of state transmissions is used as the transmission cost and serves as the optimization objective. The primary contributions of this work are as follows.

(i) The problem of partitioning distributed quantum circuits is transformed into a minimum cut problem on a graph. A quantum circuit partitioning method based on spectral clustering is proposed, which selects several partition schemes with different total cut edge weights.

(ii) An analysis of the obstacles encountered during the search for merged transmissions is conducted. Based on this analysis, a global gate search tree strategy is introduced to identify the optimal partition scheme.

(iii) Various partitioning methods are simulated and compared in terms of runtime under different partitioning scenarios. Through experimental numerical simulations, the transmission cost of distributed quantum circuits is evaluated, and the impact of different partition cut edge weights on transmission cost is investigated. Comparative experimental results validate the effectiveness of the proposed method.

The remaining sections of this paper are organized as follows. Section 2 introduces the basic concepts; Section 3 proposes quantum circuit partitioning based on spectral clustering methods; Section 4 presents the global gate search tree merged transmission optimization strategy; Section 5 verifies the effectiveness of the proposed method through experiments; Section 6 concludes the paper.

2. Background

This section briefly introduces distributed quantum circuits and quantum state transmission.

2.1. Distributed quantum circuit

Distributed quantum circuits are composed of multiple sub-circuits with limited capacities, which collaborate to achieve the overall functionality of large-scale quantum circuits. Each sub-circuit can be regarded as an independent partition within the distributed quantum circuit, and each partition is mapped to a QPU. However, this allocation method may result in the execution of two-qubit gate operations where the two qubits are distributed across different QPUs. In such cases, these two-qubit gates are referred to as global gates; otherwise, they are considered as local gates.

In distributed quantum circuits, local gates and single-qubit gates can be executed directly within their respective partitions. However, for global gate operations, the quantum states of the involved qubits must be moved to the same QPU to ensure that both quantum states can be operated on within the same QPU. To adhere to the no-cloning theorem[22] and ensure the integrity of quantum information, when a quantum state is transferred from the original QPU to the target QPU it must be returned to the original QPU after the operation to avoid interfering with subsequent operations in the original QPU and to maintain the consistency of the quantum state.

In the example of a distributed quantum circuit illustrated in Fig. 1, a six-qubit quantum circuit is divided into two partitions, P_1 and P_2. The target and control qubits of gates G_1 and G_2 are within the same partition, so G_1 and G_2 are local gates. The H gate is located in P_2 and can be executed within its respective partition alongside G_1 and G_2. Gates G_3 and G_4 have their target and control qubits in P_1 and P_2, respectively, classifying them as global gates. Therefore, the quantum states of the qubits involved in G_3 and G_4 need to be transferred to the same partition for execution. Additionally, since both global gates G_3 and G_4 act on q_2 and the other qubits are all within P_2, the merged transfer model[23] can be utilized to reconstruct them into local gates through a single transmission and execute them accordingly.

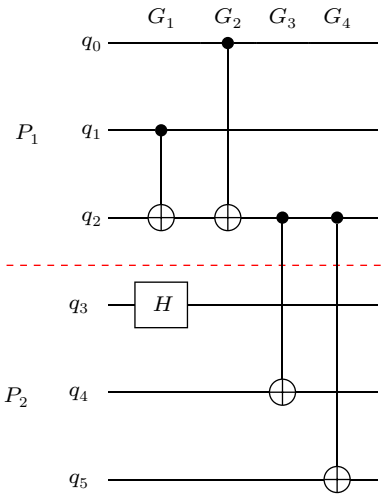


Fig. 1. An example of a distributed quantum circuit, where qubits q_0 to q_5 are divided into two partitions.

2.2. Quantum state transmission

Global gate transmission operations require quantum communication to transfer the relevant qubit information to a single QPU for computation. Unlike classical distributed computing systems, quantum information cannot be easily shared between QPUs due to the constraints of the no-cloning theorem. This means that the transfer of quantum information must ensure that no irrecoverable errors are introduced. To overcome this challenge, various methods have been proposed to effectively transfer qubits between network nodes, including techniques such as quantum teleportation,[24] cat-state entanglement[25] and direct transmission.[26]

Quantum teleportation is a method of transferring qubit states through quantum entanglement rather than physical transfer. This process requires the pre-sharing of an EPR pair between two QPUs. By distributing the EPR pair and combining classical communication with local gate operations, the state of a qubit can be transferred between different QPUs. This process not only transfers the state of the qubit but also consumes and destroys the original EPR pair.

Cat-state entanglement operations (such as cat-entanglers and cat-disentanglers) transfer qubit states by creating 'linked copies' of qubits. Similar to quantum teleportation, cat-state entanglement operations also require the pre-sharing of an EPR pair, allowing the state of a qubit to be shared in a read-only manner across multiple QPUs.

Direct transfer involves connecting QPUs through devices such as superconducting coaxial lines, optical fibers and microwave resonators, establishing quantum channels between qubits separated by certain distances to facilitate interaction and transfer of quantum states. However, quantum communication technologies face numerous challenges, such as qubit stability, transmission loss and environmental noise. These limitations mean that excessive quantum communication during the execution of distributed quantum circuits can have a negative impact on the overall performance of the system. Therefore, reducing the number of quantum communications is an effective means of enhancing performance metrics such as system latency and fidelity in the design of distributed quantum circuits.

3. Circuit partitioning based on spectral clustering methods

To reduce the transmission cost of distributed quantum circuits, researchers have been dedicated to developing quantum circuit partitioning methods. The core issue lies in efficiently dividing qubits into multiple partitions while minimizing the number of quantum gates that span across partitions, namely global gates. The number of global gates has a direct impact on the complexity and computational overhead of distributed quantum computing. Therefore, this section focuses on the quantum circuit partitioning problem and employs spectral clustering methods to partition quantum circuits, aiming to achieve more optimal quantum circuit partitioning schemes.

3.1. Problem discussion

Quantum circuit partitioning is the initial step in distributed quantum computing, and different partitioning methods often result in varying numbers of global gates. An excessive number of global gates can lead to frequent quantum state transmission between different quantum processing nodes, increasing the complexity and computational overhead of distributed quantum circuits. This manifests specifically as increased communication latency and a higher probability of data transmission errors. Partitioning a quantum circuit involves dividing the entire circuit into multiple sub-circuits, which is essentially a combinatorial problem in classical combinatorics, specifically the problem of how to partition a set into multiple subsets.

Theorem 1 Let n denote the number of logical qubits in the quantum circuit and k represent the number of partitions. If the number of qubits in each pair of partitions is either equal or differs by one, then the total number of possible partitioning schemes $W(n,k)$ is given by

$$W(n,k) = \frac{1}{r!\times(k-r)!} \times \binom{n}{\underbrace{m+1,\ldots,m+1}_{r},\underbrace{m,\ldots,m}_{k-r}}, \quad (1)$$

where $m = \lfloor \frac{n}{k} \rfloor$ represents the base number of qubits per group and $r = (n \mod k)$ denotes the number of partitions that contain $m+1$ qubits.

Proof Consider the case of partitioning n qubits into k partitions, where the number of qubits in each partition is approximately equal. To simplify the problem, assume that the number of qubits in each partition is either m or $m+1$, where $m = \lfloor \frac{n}{k} \rfloor$ represents the base number of qubits per partition.

To calculate the total number of such partitioning schemes, two parts must be considered: first, select r partitions from n qubits, each containing $m+1$ qubits; second, distribute the remaining $n-r(m+1)$ qubits into $k-r$ partitions, each containing m qubits.

Since partitions are interchangeable, meaning the order of partitions does not affect the partitioning result, the total number of partitioning schemes must also be divided by $r!\times(k-r)!$ to eliminate duplicate counts. Specifically, $r!$ is used to eliminate the repetition in the order of partitions assigned to $m+1$ qubits, and $(k-r)!$ is used to eliminate the repetition in the order of partitions assigned to m qubits.

If $r=0$, meaning n is divisible by k, then all partitions have the same number of qubits, and $W(n,k)=\frac{n!}{(m!)^k}$.

If $r>0$, meaning n is not divisible by k, then r partitions contain $m+1$ qubits and the remaining $k-r$ partitions contain m qubits. In this case, multiplying by $\frac{1}{r!\times(k-r)!}$ is necessary to adjust for the order of the groups.

As demonstrated by Eq. (1), when the number of logical qubits n in the quantum circuit increases and there exists a remainder r (i.e., n is not divisible by k), the number of ways to partition the sub-circuits becomes more complex. This complexity and diversity also reflect that, in practical distributed quantum circuit design, different partitioning methods lead to significantly different computational performance and resource utilization. Through the combinatorial mathematical analysis of the quantum circuit partitioning problem, it can be observed that as the number of qubits and the number of partitions increase, the search space grows rapidly, making the partitioning process exceptionally complex. This necessitates the consideration of more factors when seeking the optimal solution, thereby further enhancing the efficiency of distributed quantum computing.

After a quantum circuit has been partitioned into multiple segments, the transmission cost between different partitions impacts computational efficiency. During the execution of global gates, because their qubits are allocated to different partitions, additional quantum state transmission is often required to complete the computation. These transmission operations not only increase the complexity of distributed quantum computing but may also lead to the accumulation of computational errors and delays. Therefore, devising a rational circuit partitioning strategy to reduce the number of global gates and the frequency of quantum state transmission becomes crucial for improving the performance of distributed quantum computing.

The six-qubit quantum circuit shown in Fig. 2, which includes eight two-qubit gates, is partitioned into two sections as illustrated in Fig. 3. In Fig. 3(a), q_3, q_5 and q_2 of the original circuit are grouped and assigned to partition P_1, while q_1, q_4 and q_0 are grouped and assigned to partition P_2. After partitioning, there are three global gates: G_2, G_3 and G_5. Due to the absence of merged transfer conditions, this configuration requires six quantum state transmission operations to complete the execution of the global gates and to return the quantum states. In Fig. 3(b), q_1, q_5 and q_2 of the original circuit are assigned to partition P_1, and q_3, q_4 and q_0 are assigned to partition P_2. After this partitioning, there are still three global gates: G_1, G_3 and G_4. However, this grouping allows the global gates G_1 and G_3 to both act on q_5, satisfying the conditions for merged transfer. Consequently, only two transmission operations are needed to complete the transfer and execution of gates G_1 and G_3, resulting in a total of four quantum state transmission operations.

Although reducing the number of global gates is an important objective in quantum circuit partitioning, it does not necessarily mean that fewer global gates will always lead to a reduced transmission cost. In practice, the transmission cost depends not only on the number of global gates but also on the qubits involved in these global gates and their distribution. Through clever partitioning strategies, it is possible to reduce the transmission cost by utilizing a merged transfer model, even if this results in a greater number of global gates.

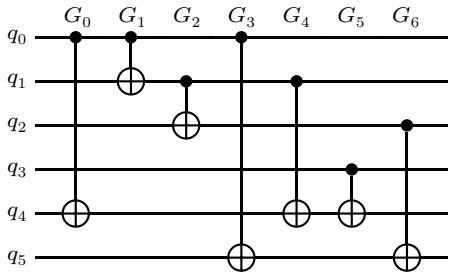


Fig. 2. An example of a six-qubit quantum circuit.

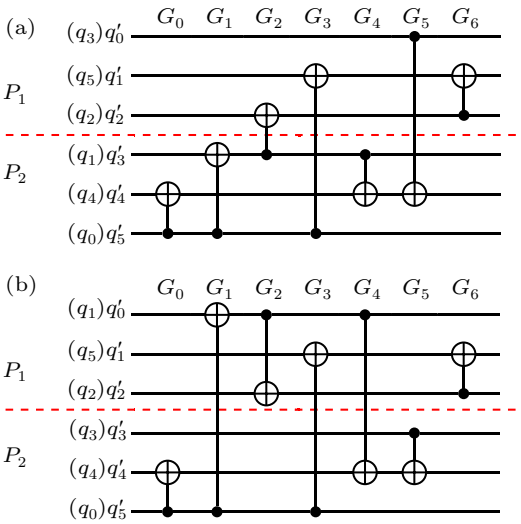


Fig. 3. Different distributed quantum circuit partitioning schemes for Fig. 2 (same number of global gates, different transmission operations): (a) three global gates, six transmission operations; (b) three global gates, four transmission operations.

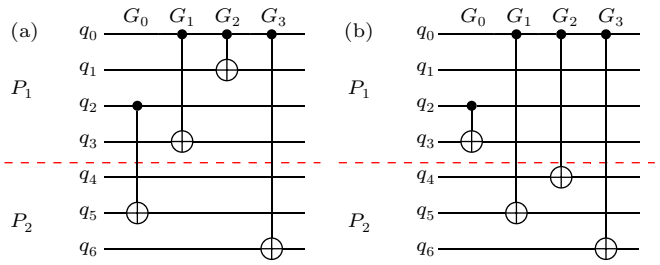


Fig. 4. Fewer global gates, more transmission operations: (a) two global gates, four transmission operations; (b) three global gates, two transmission operations.

Figure 4 shows two different partitioning methods for the same circuit. In Fig. 4(a), the partitioning approach results in two global gates, G_0 and G_3, which do not satisfy the conditions for merged transfer, requiring a total of four transmission operations. In Fig. 4(b), three global gates, G_1, G_2 and G_3, are generated. However, since all three gates act on q_0, a merged transfer model can be utilized, requiring only two transmission operations.

In summary, different quantum circuit partitioning strategies have a significant impact on the number of global gates and the transmission cost. Simply reducing the number of global gates is not sufficient to optimize the transmission cost. To further enhance the efficiency of distributed quantum computing, it is necessary to explore more intelligent and efficient partitioning methods.

3.2. Optimized spectral clustering partitioning method

To effectively reduce the number of transmissions in distributed quantum computing it is often necessary to find an optimal qubit partitioning scheme. The minimum cut method for quantum circuits is one such common strategy, which minimizes the number of qubit transmissions across partitions by finding the set of minimum cut edge weights in the graph structure. The objective of the minimum cut problem is to identify a set of edges whose total weight represents the number of global gates generated by the partitioning, such that removing this set of edges divides the node set in the graph into several independent subsets, and the total weight of the removed edges is minimized. A minimum cut implies optimizing the partitioning of the quantum circuit by minimizing the transmission operations across partitions, specifically minimizing the number of global gates.

However, the minimum cut method also has some limitations. For example, the minimum cut method only focuses on minimizing a single set of cut edges and does not fully consider the global characteristics of the overall structure of the quantum circuit. Therefore, in some complex quantum circuits the minimum cut method may not provide the best partitioning scheme. To address this issue, a spectral clustering-based partitioning method is introduced. This method maps the quantum circuit partitioning problem to the eigenvector space of the Laplacian matrix. This simplifies the high-dimensional structure of complex graphs into the analysis of a few eigenvectors. As a result, it avoids the need for repeated probing in a vast search space as traditional methods do, significantly reducing the amount of information that needs to be processed. Furthermore, the second smallest eigenvalue of the Laplacian matrix and its corresponding eigenvector provide key information about the global structure of the graph, avoiding the potential problem of getting trapped in locally optimal solutions.

A graph G is generated based on the topological structure of the quantum circuit, where each node represents a qubit and edges represent the associated operations between two qubits (i.e., two-qubit gates). The weight of an edge reflects the number of two-qubit gates acting on the two qubits.

Based on the generated graph G, its adjacency matrix A and degree matrix D are constructed. The elements A_{ij} in the adjacency matrix represent the edge weights between nodes i and j in graph G. If there is no edge between node i and node j, then $A_{ij} = 0$; otherwise, A_{ij} equals the weight of the edge. The degree matrix D is a diagonal matrix, where D_{ii} represents the degree of node i, i.e., the number of edges connected to node i. For the adjacency matrix A, the degree matrix D can be calculated using

$$D_{ii} = \sum_j A_{ij}. \tag{2}$$

The Laplacian matrix L reflects the structure of the graph and is obtained by subtracting the adjacency matrix from the degree matrix, specifically, $L = D - A$. By performing eigenvalue decomposition on L, its eigenvalues and eigenvectors are computed. Suppose the eigenvalues of L are $\lambda_1, \lambda_2, \ldots, \lambda_n$ and the corresponding eigenvectors are $\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_n$. The spectral clustering algorithm requires selecting the eigenvector $\boldsymbol{v}_2$ corresponding to the second smallest eigenvalue λ_2 (also known as the Fiedler value) as the basis for partitioning. The eigenvector corresponding to the second smallest eigenvalue reflects the 'cutting' properties of the graph, enabling effective graph partitioning.

Ultimately, by clustering the data points in the eigenvector $\boldsymbol{v}_2$, they are divided into k clusters, each representing a set of qubits in the partitioned sub-circuits. The k-way spectral clustering partitioning algorithm is shown in Algorithm 1.

Figure 5 illustrates the result of converting the quantum circuit from Fig. 2 into a graph structure. Based on the weighted graph of the quantum circuit shown in Fig. 5, the process of the spectral clustering method is detailed as follows. First, construct the adjacency matrix A of the graph, where each row and column represent a node and the elements in the matrix denote the weights of the edges between two nodes. Then, based on the adjacency matrix A, generate the

degree matrix D.

$$A=\begin{pmatrix}0&1&0&1&1&0\\1&0&1&0&0&0\\0&1&0&0&0&1\\1&0&0&0&1&0\\1&0&0&1&0&1\\0&0&1&0&1&0\end{pmatrix},\quad D=\begin{pmatrix}3&0&0&0&0&0\\0&2&0&0&0&0\\0&0&2&0&0&0\\0&0&0&2&0&0\\0&0&0&0&3&0\\0&0&0&0&0&2\end{pmatrix}.$$

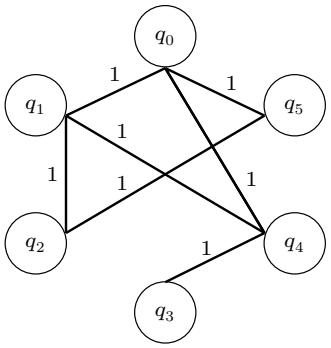


Fig. 5. Weighted graph of the quantum circuit generated from Fig. 2.

Algorithm 1: k-way spectral clustering partitioning (KSCP).

Input: quantum circuit $circ=[g_1,g_2,\ldots]$, number of desired partitions k.
Output: partition of vertex set V into k subsets $\{V_1,V_2,\ldots,V_k\}$.

begin
  $G(V,E)\leftarrow circ$// Convert the quantum circuit circ into graph G
  $A\leftarrow$ AdjacencyMatrix(G)// Construct the adjacency matrix A of the graph
  $D\leftarrow$ DegreeMatrix(A)// Compute the degree matrix D
  $L\leftarrow D-A$// Compute the Laplacian matrix L
  $\{\lambda,V\}\leftarrow$ EigenDecomposition(L)// Calculate the eigenvalues and eigenvectors of L
  $F\leftarrow$ emptymatrixofsize$(|V|,k-1)$// Initialize an empty matrix F of size $(|V|,k-1)$
  for $j\leftarrow 2$ **to** k **do**
    $F[:,j-1]\leftarrow V[:,j]$// Assign the j-th eigenvector to $F[:,j-1]$
  end
  $\{V_1,V_2,\ldots,V_k\}\leftarrow K$-means$(F,k)$ // Use K-means on the rows of F for k partitions
  return $\{V_1,V_2,\ldots,V_k\}$
end

The Laplacian matrix L is obtained by subtracting the adjacency matrix from the degree matrix. The importance of the Laplacian matrix lies in the fact that its eigenvalues and eigenvectors contain structural information for the graph.

$$L=D-A=\begin{pmatrix}3&-1&0&-1&-1&0\\-1&2&-1&0&0&0\\0&-1&2&0&0&-1\\-1&0&0&2&-1&0\\-1&0&0&-1&3&-1\\0&0&-1&0&-1&2\end{pmatrix}.$$

Next, calculate the eigenvalues and eigenvectors of L. The eigenvalues are $\lambda_1\approx 0$, $\lambda_2=1$, $\lambda_3=1.58578644$, $\lambda_4=3$, $\lambda_5=4$, $\lambda_6=4.41421356$. The corresponding eigenvector matrix (represented by columns) is as follows:

$$V=\begin{pmatrix}-0.408&0.289&-0.271&-0.289&0.408&-0.653\\-0.408&-0.289&-0.653&-0.289&-0.408&0.271\\-0.408&-0.577&0.000&0.577&0.408&-0.000\\-0.408&0.577&-0.000&0.577&-0.408&0.000\\-0.408&0.288&0.271&-0.289&0.408&0.653\\-0.408&-0.288&0.653&-0.289&-0.408&-0.271\end{pmatrix}.$$

According to the theory of spectral clustering, the eigenvector corresponding to the second smallest eigenvalue is chosen as the basis for partitioning. Based on the calculation results, the second smallest eigenvalue is 1, and its corresponding eigenvector is $(0.289,-0.289,-0.577,0.577,0.288,-0.288)^{\mathrm{T}}$. Finally, by applying K-means clustering, two clusters are obtained: $\{0.289,0.577,0.288\}$ and $\{-0.289,-0.577,-0.288\}$. These correspond to the node partitions $\{0,3,4\}$ and $\{1,2,5\}$, and the resulting circuits are shown in Fig. 3(b).

However, this preliminary partitioning method may present the following issues in practical applications: (i) imbalance in the number of logical qubits in each partition; (ii) a relatively small number of global gates but still a high number of transmission operations.

To address these issues, the spectral clustering algorithm is optimized. This optimized algorithm further adjusts the subset partitions based on the initial partitioning, making the size of each subset more balanced while striving to minimize the weight of the cut edges. Additionally, in multiple iterations, candidate partitioning schemes with different cut edge weights are proportionally selected. The pseudocode for the proposed optimized k-way spectral clustering partitioning algorithm is presented in Algorithm 2.

Algorithm 2 obtains the initial vertex set partition through the K-SGP algorithm, but K-SGP may cause a large difference in the number of qubits allocated between partitions. If this is abstracted as a weighted graph of the circuit, it is equivalent to having a large difference in the number of nodes between the partitions. The balance_partitions function is used to balance the initial vertex set, gradually reallocating nodes from partitions with a larger number of nodes to those with fewer nodes, ensuring that the size of each partition is balanced. The calculate_cut_weight function computes the cut edge weights for each partition to evaluate the quality of the partitioning. After completing all iterations, the algorithm uses the filter_partition function to select n unique partitioning solutions from the results, where the optimal weight accounts for 50%, the second-best for 30% and the third-best for 20%. This optimization algorithm combines the advantages of initial spectral clustering with subsequent optimization strategies, making it suitable for finding multiple relatively optimal partitioning solutions in complex networks while ensuring that the number of nodes in each partition is as balanced as possible and the cut edge weights are minimized.

Algorithm 2: Optimized k-way spectral clustering partitioning (O-KSCP).

Input: quantum circuit $circ = [g_1, g_2, \ldots]$, number of iterations for optimization *num_iterations*, total number of partition schemes to output n.

Output: a set of finally optimized partition methods *Fin_Partitions*.

```
begin
  for iter ← 1 to num_iterations do
    {V_1,...,V_k} ← KSCP(circ,k)// Invoke the KSCP to partition the circuit
    {P_1,...,P_k} ← balance_partitions({V_1,...,V_k})// Balance the nodes across partitions
    cut_weight ← Σ_{i=1}^{k-1} Σ_{j=i+1}^{k} calculate_cut_weight(G,P_i,P_j)
    all_partitions.append({P_1,...,P_k})// Store partition
    all_cut_weights.append(cut_weight)// Store cut edge weights
  end
  num_best ← n//2
  second_best ← n*3//10
  third_best ← n − num_best − second_best
  Fin_Partitions ← filter_partition(num_best, second_best, third_best)
  Return Fin_Partitions
end
```

4. Transmission cost optimization of the global gate search tree strategy

In distributed quantum computing, the number of transmission operations for global gates across partitions has a significant impact on computational efficiency. This section presents a global gate search tree strategy that generates nodes based on the qubits involved in global gate operations and systematically identifies opportunities for merged transfers, facilitating the selection of the optimal partitioning.

4.1. Determination and search for merged transmission

In distributed quantum computing, because qubits are distributed across multiple QPUs, quantum information transmission between QPUs is required to complete computational tasks. The scale of transmission operations is given by

$$T_{\text{merged}} + T_{\text{unmerged}} \leq TC \leq 2\text{Num}_{\text{global}}, \tag{3}$$

where TC denotes the number of transmission operations required for executing global gates in a distributed quantum circuit. T_{merged} refers to the number of transmission operations using the merged transfer model, Specifically, this refers to global gates in the circuit that can utilize the merged transmission model, which subsequently reduces the transmission cost. T_{unmerged} represents the number of transmission operations required for the remaining global gates that do not use the merged transmission model. These gates cannot be merged transfer and must be executed through single transmission operations. The sum of $T_{\text{merged}} + T_{\text{unmerged}}$ stands for the total number of transmission operations obtained by searching for as many merged transfer opportunities as possible. $\text{Num}_{\text{global}}$ represents the number of global gates present in the distributed quantum circuit. The significance of merged transfer lies in its ability to reduce the total transmission cost by combining multiple global gate transmission requests into a single transmission operation. In contrast, not using merged transfer requires multiple independent transmission operations, leading to a significant increase in transmission cost, with an upper bound reaching twice the number of global gates. Therefore, identifying global gates that can be merged for transmission in the partitioned distributed quantum circuit is of great importance for reducing the transmission cost of computational tasks.

From the merged transfer model, it can be observed that global gates acting on the same qubit and whose remaining qubits belong to the same partition can be reconstructed into local gates and executed through a single transmission operation. However, when searching for global gates that can be merged for transfer in actual circuits, the following situations often arise: (i) The subsequent quantum gate is a local gate. (ii) The subsequent quantum gate is a global gate, and the qubits it acts on are different from those of the global gate for which merged transfer is being sought.

These two scenarios cause the search for merged transfer to terminate prematurely, missing many opportunities for merged transfer. As shown in Fig. 6(a), when transferring the qubit q_0 for the global gate G_0, because gate G_1 is a local gate, even if global gate G_2 also acts on q_0, the conditions for merged transfer are not met. In Fig. 6(b), the transfer of q_0 for the global gate G_0 is still required, but the subsequent gate is a global gate that acts on different qubits from those of gate G_0, hindering the opportunity to merge transfer G_2 and G_0.

Therefore, it is necessary to utilize the quantum gate movement rules to move subsequent global gates that act on the same qubits behind the required global gates for transmission as much as possible, so that merged transfers can cover more global gates. Figure 7 illustrates the movement rules that need to be used. In Fig. 7(a), if two two-qubit gates act on different qubits, the positions of these two gates can be swapped. In Fig. 7(b), if two two-qubit gates have the same control qubit but different target qubits, the positions of these two gates can be swapped. In Fig. 7(c), if two two-qubit gates have the same target qubit but different control qubits, the positions of these two gates can be swapped.

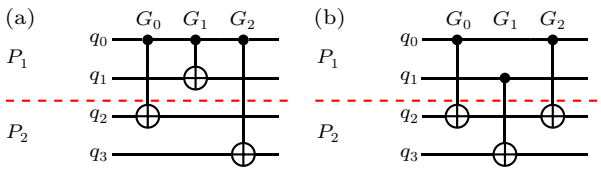


Fig. 6. Obstacles in merged transfer search. (a) When transferring q_0 for global gate G_0, gate G_1 blocks the merged transfer with G_2. (b) The subsequent global gate G_1 hinders the merged transfer between G_0 and G_2.

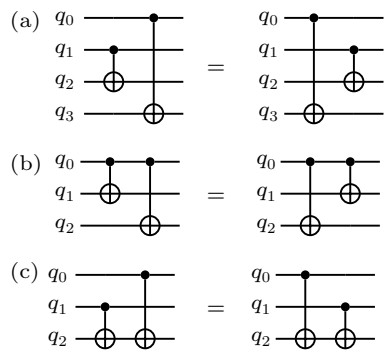


Fig. 7. Quantum circuit movement rules: (a), (b) and (c) respectively represent the movement rules that may be used during the global gate search tree process.

According to the aforementioned gate movement rules, more opportunities for merged transfer can be created. In Fig. 6(a), gate G_1 can be swapped with gate G_2 based on the rule in Fig. 7(b). At this point, if qubit q_0 is transferred, G_0 and G_2 can be merged for transfer. In Fig. 6(b), the rule in Fig. 7(a) is satisfied. After swapping the positions of gates G_1 and G_2, gates G_0 and G_2 meet the conditions for merged transfer.

4.2. Global gate search tree strategy

In distributed quantum computing, effectively reducing the number of quantum state transmissions is crucial for optimizing computational efficiency. To this end, this paper proposes a method using a global gate search tree to calculate the number of transfers. This method generates corresponding nodes based on the qubits involved in global gate operations and utilizes movement rules to identify as many subsequent global gates that can be merged for transfer as possible. The distributed quantum circuit is continuously updated until no global gates remain in the circuit.

Definition 1 A tree generated by using the global gates in the quantum circuit as root nodes and by progressively moving and updating the states of qubits is called a global gate search tree.

Each node in the global gate search tree contains the following information: (i) Current global gate (Global_gate): represents the first global gate that appears among all quantum gates. (ii) Qubit allocation (Trans_qubit): records the qubits on which the current global gate acts. (iii) Merged transfer list (Trans_list): stores the set of global gates that meet the conditions for merged transfer.

Through this progressive search strategy, the layout of quantum gates in the circuit can be analyzed in greater detail, effectively reducing the number of transmission operations and ultimately achieving the optimal number of transmission operations.

The method for finding merged transfers is proposed as described in Algorithm 3. It starts at the index position of the first global gate in the circuit and traverses to the last global gate that acts on the transmission qubit. It then retrieves the partition of the other qubits involved in the gate, referred to as other_partition. When encountering a global gate that acts on Trans_qubit, the function qubit_interacts_partition is used to determine whether the other qubits involved in that global gate belong to the same partition as other_partition. If they are the same, it initially satisfies the conditions for merged transfer, and it can be further determined whether to use the movement rules to move it to the next position after the first global gate index. If the conditions are met, it is added to Trans_list. It should be noted that the function can_move_to_position ignores the global gates in Trans_list to prevent gates already added to the list from affecting the movement of subsequent global gates. After the traversal is complete, the first global gate is also added to Trans_list. This list provides the index of quantum gates that need to be removed for updating the quantum circuit in the node.

Algorithm 3: Find global gates that can be merged transfer.

Input: quantum circuit $circ = [g_1, g_2, \ldots]$, partition list $V = \{V_1, \ldots, V_k\}$, $Trans_qubit$ that need to be transferred.
Output: the merged transfer list $Trans_list$.

```
begin
  start_index ← find_first_global_gate(circ, V)
  end_index ← find_last_global_gate(circ, V)
  // Retrieve other partitions to which the current global gate belongs
  other_partition ← get_other_partition(Trans_qubit, V)
  for i = start_index to end_index do
      // This quantum gate is a global gate
      if gate_is_global(circ[i], V) then
          // This quantum gate acts on Trans_qubit
          if Trans_qubit in circ[i] then
              // The other qubits of this gate belong to the same
                 partition as other_partition
              if qubit_interacts_partition(circ[i], other_partition) then
                 // This quantum gate can be moved to the position at
                    start_index + 1
                 if can_move_to_position(circ[i], start_index) then
                     Trans_list.append(circ[i])// Add this gate to Trans_list
                 end
              end
          end
      end
  end
  Trans_list.append(circ[start_index])// Add the first global gate
  to Trans_list
  return Trans_list
end
```

To prevent the global gate search tree from growing excessively during the search process, which would lead to high computational complexity, the following pruning measures are adopted: (i) Construct a global variable min_depth, and stop generating nodes once the depth exceeds its value. (ii) Use a dictionary to record the depth of each circuit relative to the root node and the current circuit. When constructing a new node, if the current circuit already exists and has a greater depth, the node is not generated. These measures can limit the size of

the tree and prevent its overexpansion while ensuring search efficiency.

4.3. Example

Taking the six-qubit circuit shown in Fig. 8 as an example, the global gate search tree strategy is elaborated, and the resulting tree diagram is shown in Fig. 9.

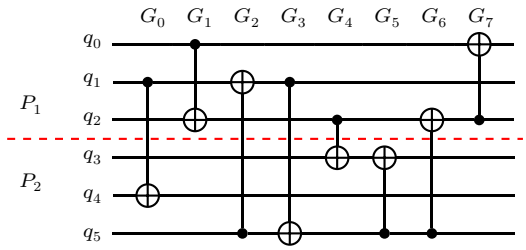


Fig. 8. The six-qubit distributed quantum circuit is partitioned into two partitions P_1 and P_2.

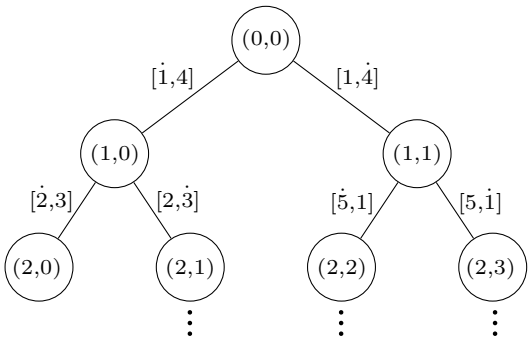


Fig. 9. Process diagram of the global gate search strategy.

Step 1 The current circuit is taken as the root node $(0,0)$, and the first global gate G_0 in the circuit is identified, with its associated qubits being q_1 and q_4, meaning the Trans_qubit are q_1 and q_4. Next, the subsequent quantum circuit is traversed to determine whether each global gate satisfies the conditions for merged transfer with G_0, generating child nodes $(1,0)$ and $(1,1)$. Each node stores the updated circuit based on the transfer of the current qubit.

In node $(1,0)$, considering the transfer of qubit q_1, i.e., Trans_qubit is q_1, the other qubit of G_0 belongs to the partition P_2(other_partition). Gate G_1 is a local gate, and G_2 is a global gate acting on q_1. The other qubit affected by G_2 is in the same partition as other_partition. According to the movement rules, G_2 can be moved before G_1 to meet the merged transmission condition. Similarly, G_3 also acts on q_1, and its other_partition is P_2, so it can be swapped with G_1. Continuing the traversal, after no more global gates act on qubit q_1, gates G_0, G_2 and G_3 are added to Trans_list. These gates are then removed from the original circuit as shown in Fig. 10(a), and the circuit in node $(1,0)$ is updated. Subsequently, in node $(1,1)$, considering the transfer of qubit q_4, i.e., Trans_qubit is q_4, the other_partition of the qubit affected by G_0 is P_1. Since there are no global gates acting on qubit q_4 in the circuit, G_0 is added to Trans_list and removed, and the circuit is updated accordingly as shown in Fig. 10(b).

Step 2 Taking node (1,0) as the root, as shown in Fig. 10(a), the first global gate in the circuit is gate G_1. It acts on qubits q_2 and q_3, leading to the generation of nodes (2,0) and (2,1). In node (2,0), when transferring q_2, i.e., Trans_qubit is q_2, the other_partition of the qubit affected by G_1 is P_2. The movement rules allow the global gate G_3 to be swapped with the local gate G_2, adding G_1 and G_3 to the Trans_list, removing them and updating the circuit. After the update, there are no remaining global gates in the circuit, thus completing the path composed of nodes (0,0), (1,0), and (2,0) with a path length of 2. If subsequent path lengths exceed 2, no further nodes are generated. In node (2,1), when considering the transfer of q_3, i.e., Trans_qubit is q_3 and the other_partition is P_1, there are no global gates that can be merged for transfer, so G_1 is added to the list and the circuit is updated. As shown in Fig. 10(a), the global gate G_3 still exists, but the path has already been completed, so node (2,1) does not need to generate new child nodes by repeating the above steps.

Step 3 Continuing with node (1,1), as shown in Fig. 10(b), global gate G_1 acts on qubits q_5 and q_1, generating nodes (2,2) and (2,3). In node (2,2), when transferring q_5, i.e., Trans_qubit is q_5 and the other_partition is P_1, only the target qubit of global gate G_2 acts on q_5, and its other_partition is P_1, which is adjacent to G_1, thus satisfying the conditions for merged transfer. After removing these two gates and updating the circuit, global gates G_3 and G_5 still exist in the updated circuit, so no new child nodes are generated. In node (2,3), global gate G_2 meets the conditions for merged transfer, but the target qubit of global gate G_5 and the control qubit of G_3 both act on q_2, preventing G_5 from being moved to the left of G_3, thus making merged transfer impossible. Similarly, no new nodes are generated for this node.

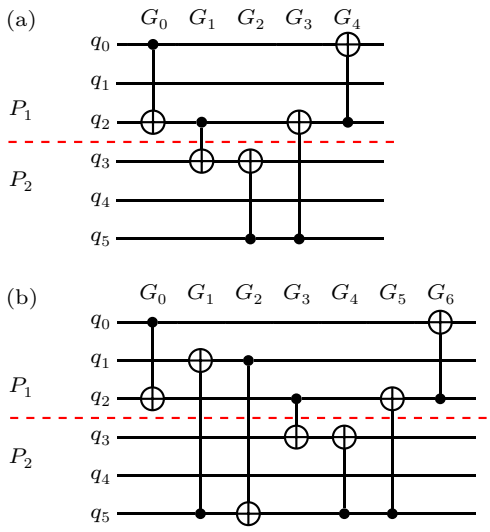


Fig. 10. Distributed quantum circuits updated after considering different transmission qubits: (a) and (b) respectively represent the circuits updated by transferring qubits q_1 and q_4 of global gate G_0.

Table 1. Process table of the global gate search strategy.

Node	Circuit	Global_gate	Trans_qubit	Trans_list
(0,0)	[[1,4],[0,2],[5,1],[1,5],[2,3],[5,3],[5,2],[2,0]]			
(1,0)	[[0,2],[2,3],[5,3],[5,2],[2,0]]	[1,4]	q_1	[[1,4],[5,1],[1,5]]
(1,1)	[[0,2],[5,1],[1,5],[2,3],[5,3],[5,2],[2,0]]	[1,4]	q_4	[[1,4]]
(2,0)	[[0,2],[5,3],[2,0]]	[2,3]	q_2	[[2,3],[5,2]]
(2,1)	[[0,2],[5,3],[5,2],[2,0]]	[2,3]	q_3	[[2,3]]
(2,2)	[[0,2],[2,3],[5,3],[5,2],[2,0]]	[5,1]	q_5	[[5,1],[1,5]]
(2,3)	[[0,2],[2,3],[5,3],[5,2],[2,0]]	[5,1]	q_1	[[5,1],[1,5]]

Ultimately, the path composed of nodes (0,0), (1,0) and (2,0) is the shortest path, with a total of four transmission operations including the transfer back of quantum states. The process information of the global gate search strategy is shown in Table 1.

5. Experiential results and analysis

The proposed method is implemented using Python, with experiments conducted on a system equipped with an Intel i5-8300H CPU @ 2.30 GHz and 16 GB of memory. The experiments selected circuits from the reversible circuit test set RevLib, quantum circuits of the QFT algorithm and randomly generated circuits encompassing various sizes and structures as experimental subjects. In the experiment, the algorithm iterates 1000 times in different test circuits. The number of quantum state transmissions between partitions is used as the transmission cost, serving as a performance metric for comparing distributed quantum circuits. This study is based on a superconducting distributed architecture and employs direct quantum state transmission for transferring quantum states, using the number of quantum state transmissions as the transmission cost. Therefore, even though different methods of quantum state transmission may be used, the counting method remains consistent with the comparative references, ensuring the comparability of experimental data.

5.1. Transmission cost optimization for QFT circuits

As shown in Table 2, the proposed global gate search tree strategy is compared with those in Refs. [27,28] regarding the optimization of transmission cost for QFT circuits under different partitioning scenarios. In the table, Circuit denotes the name of the quantum circuit, Qubits represents the number of qubits involved in the circuit, Gates indicates the number of gates present in the circuit, K signifies the number of partitions, Global_gates refers to the number of global gates generated after partitioning, TC denotes the transmission cost required to execute the distributed quantum circuit and Imp represents the improvement rate of the proposed method compared to the comparative results in terms of transmission cost metrics.

Reference [27] employs a genetic algorithm to find the optimal allocation scheme of quantum gates across different partitions. This is achieved by encoding possible allocation schemes as chromosomes and applying genetic operations such as selection, crossover and mutation to reduce the number of required transmission operations. Reference [28] adopts a lookahead strategy based on gate commutativity, identifying commutable global gates within the distributed quantum circuit and optimizing the execution order of these gates to lower the transmission cost.

As observed in Table 2, the global gate search tree strategy proposed in this paper outperforms the methods in Refs. [27,28] in reducing transmission cost, achieving an average improvement rate of 10.2%. This optimized performance is attributed to the proposed method's utilization of gate movement rules combined with continuous optimization and updating of the distributed quantum circuit through tree generation. This progressive search strategy allows for a more detailed analysis of the distribution of global gates within the distributed quantum circuit, thereby attaining the optimal number of transmission operations.

Table 2. Transmission cost comparison of QFT circuits under different partitions with Refs. [27,28].

Circuit	Qubits	Gates	K	Global_gates	TC			Imp
					[27]	[28]	Ours	
4-qubit QFT	4	16	2	4	8	4	4	0.00%
			3	5	NA	10	6	40.00%
			4	6	NA	14	12	14.29%
8-qubit QFT	8	48	2	8	38	8	8	0.00%
			3	21	NA	20	14	30.00%
			4	24	NA	24	24	0.00%
16-qubit QFT	16	160	2	16	133	16	16	0.00%
			3	85	NA	40	30	25.00%
			4	96	NA	48	48	0.00%
32-qubit QFT	32	576	2	30	532	32	30	0.00%
			3	62	NA	80	62	22.50%
			4	384	NA	96	96	0.00%
64-qubit QFT	64	2176	2	62	2250	64	62	0.00%
			3	1365	NA	160	126	21.25%
			4	1536	NA	192	192	0.00%

5.2. Transmission cost optimization for Revlib circuits

This experiment selected quantum circuits from the RevLib library for testing, comparing the impact of different cut edge weights under two partitions on the transmission cost of distributed quantum circuits. Table 3 presents a compari-

son of transmission cost between the method proposed in this paper and those in Refs. [17,19]. In this table, *Min_weight* denotes the minimum sum of cut edge weights during the quantum circuit partitioning process, reflecting the number of global gates generated, while *Weight* represents the sum of cut edge weights corresponding to achieving the optimal transmission cost. Reference [19] introduced a hierarchical method for constructing distributed quantum systems, applying recursive functions to minimize the number of transmission operations. Reference [17], on the other hand, abstracted distributed quantum circuits into matrix models and scheduled quantum gates by optimizing these matrices.

Table 3. Transmission cost comparison with Refs. [17,19].

Circuit	Qubits	Min_weight	Weight	*TC*			Imp
				[19]	[17]	Ours	
4gt5_76	5	7	7	4	4	4	0.00%
min_alu_305	10	18	32	NA	16	6	62.50%
sym9_147	12	48	65	22	8	10	−25.00%
sym6_316	14	27	51	NA	14	6	57.14%
parity_247	17	8	8	6	2	2	0.00%
ham7_299	21	43	71	NA	34	14	58.82%

Prior to the experiment, because the quantum circuits under comparison included multi-controlled Toffoli (MCT) gates, which are unfavorable for subsequent routing tasks, the decomposition algorithm provided by Qiskit is employed to break down MCT gates into several two-qubit gates. Although this process increased the number of global gates, Table 3 shows that the proposed method still significantly reduced the transmission cost compared with Refs. [17,19] when handling distributed quantum circuits with greater depth after decomposition, achieving an average improvement rate of 25.6% and a maximum improvement rate of 62.5%. This result demonstrates that the global gate search tree strategy proposed in this paper can meticulously analyze each opportunity for merged transfer, allowing more global gates to be completed through a single transmission, effectively reducing the overall transmission cost.

Further analysis reveals that although the *Weight* corresponding to the optimal transmission cost may be higher than *Min_weight*, solely relying on the minimum sum of cut edge weights for circuit partitioning does not necessarily achieve the lowest transmission cost. This highlights the advantage of the proposed algorithm in comprehensively considering both the number of global gates and the optimization of transmission cost.

5.3. Runtime comparison of partitioning methods

As shown in Table 4, the partitioning times required for quantum circuits of different sizes under two partitions are compared with the partitioning methods in Refs. [29–32]. The experiment randomly generated quantum circuits ranging from 5 to 1000 qubits, each size containing 1000 two-qubit gates, to compare the performance of the proposed quantum circuit partitioning method with other graph partitioning methods in terms of runtime.

As the number of qubits increases, the runtime of all methods also grows. The KL method, however, shows a rapid increase in runtime for circuits with more than 300 qubits, indicating its poor scalability for large-scale quantum circuit partitioning. The *K*-means method incurs higher time overhead for small-scale qubits but maintains a relatively stable growth rate as the number of qubits increases. The hypergraph method performs consistently across different qubit scales, but its runtime is generally longer compared with other methods. In contrast, the partitioning method proposed in this paper exhibits a relatively moderate growth rate in runtime, demonstrating better partitioning efficiency for large-scale circuits. It maintains a low runtime across various qubit scales, offering a more efficient solution for quantum circuit partitioning.

Table 4. Partitioning runtime comparison under bipartition with Refs. [29–32].

Qubits	FM[29]	KL[30]	*K*-means[31]	Hypergraph[32]	The proposed method
5	0.000997543	0.001031399	0.068432915	0.012964964	0.002994061
10	0.000998974	0.000997066	0.071159339	0.012965679	0.002991915
20	0.00105381	0.000998974	0.070941232	0.01362984	0.002994776
50	0.0010674	0.007984638	0.073692042	0.014960289	0.004986763
100	0.002055168	0.063453913	0.076984814	0.01946814	0.007978201
200	0.002485209	0.560501099	0.081880004	0.030917168	0.01000762
300	0.002572096	1.399894476	0.086251282	0.042885065	0.013994217
400	0.002958536	3.482462454	0.090319073	0.056847572	0.017952442
500	0.204488754	5.482630968	0.099824299	0.075796366	0.027066708
600	0.273268938	9.267543793	0.114939346	0.11862384	0.041171312
700	0.314194679	13.58206201	0.122612362	0.130704184	0.061133146
800	0.365586281	18.73419366	0.145042524	0.198788743	0.066094875
900	0.449824104	29.57125031	0.169331951	0.226820607	0.082159996
1000	0.556922197	35.77057076	0.187444906	0.244667961	0.110704184

6. Conclusion

In distributed quantum computing, the number of quantum state transmissions required for global gates operating across partitions is a critical factor affecting computational efficiency. This paper introduces a quantum circuit partitioning method based on spectral clustering, which efficiently balances the number of qubits in each partition while reducing the number of global gates, thereby selecting candidate partition schemes. By employing a global gate search tree strategy, the distribution of global gates within distributed quantum circuits is analyzed in greater detail, facilitating the identification of more opportunities for merged transfers. This approach effectively reduces transmission cost and enables the selection of the most optimal partitioning scheme for distributed quantum circuits. Experimental results demonstrate that the proposed method achieves optimization across various metrics. Future research may explore partitioning schemes that involve cutting qubit lines, integrate advanced quantum state transmission technologies and consider the impact of noise on the fidelity of distributed quantum computing.

Acknowledgments

Project supported by the National Natural Science Foundation of China (Grant No. 62072259), in part by the Natural Science Foundation of Jiangsu Province (Grant No. BK20221411), the PhD Start-up Fund of Nantong University (Grant No. 23B03), and the Postgraduate Research & Practice Innovation Program of School of Information Science and Technology, Nantong University (Grant No. NTU-SISTPR24_05).

References

[1] Feynman R P 2018 *Feynman and Computation* (cRc Press) pp. 133–153
[2] Preskill J 2018 *Quantum* **2** 79
[3] Qin D, Xu X and Li Y 2022 *Chin. Phys. B* **31** 090306
[4] Jin Y X, Xu H Z, Wang Z A, Zhuang W F, Huang K X, Shi Y H, Ma W G, Li T M, Chen C T, Xu K, *et al.* 2024 *Chin. Phys. B* **33** 050301
[5] Van Meter R and Devitt S J 2016 *Computer* **49** 31
[6] Buhrman H and Röhrig H 2003 *International Symposium on Mathematical Foundations of Computer Science* (Springer) pp. 1–20
[7] Beals R, Brierley S, Gray O, Harrow A W, Kutin S, Linden N, Shepherd D and Stather M 2013 *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences* **469** 20120686
[8] Cuomo D, Caleffi M and Cacciapuoti A S 2020 *IET Quantum Communication* **1** 3
[9] Avron J, Casper O and Rozen I 2021 *Phys. Rev. A* **104** 052404
[10] Fujii K, Yamamoto T, Koashi M and Imoto N 2012 arXiv:1202.6588 [quant-ph]
[11] Cacciapuoti A S, Caleffi M, Tafuri F, Cataliotti F S, Gherardini S and Bianchi G 2019 *IEEE Network* **34** 137
[12] Sundaram R G, Gupta H and Ramakrishnan C 2021 35*th International Symposium on Distributed Computing (DISC* 2021*)* (Schloss Dagstuhl-Leibniz-Zentrum für Informatik)
[13] Sundaram R G, Gupta H and Ramakrishnan C 2022 2022 *IEEE International Conference on Quantum Computing and Engineering (QCE)* (IEEE) pp. 415–425
[14] Davarzani Z, Zomorodi-Moghadam M, Houshmand M and Nouri-Baygi M 2020 *Quantum Inf. Process.* **19** 1
[15] Nikahd E, Mohammadzadeh N, Sedighi M and Zamani M S 2021 *Physica Scripta* **96** 035102
[16] Dadkhah D, Zomorodi M and Hosseini S E 2021 *International Journal of Theoretical Physics* **60** 3271
[17] Ghodsollahee I, Davarzani Z, Zomorodi M, Pławiak P, Houshmand M and Houshmand M 2021 *Quantum Inf. Process.* **20** 1
[18] Dadkhah D, Zomorodi M, Hosseini S E, Plawiak P and Zhou X 2022 *IEEE Access* **10** 70329
[19] Davarzani Z, Zomorodi M and Houshmand M 2022 *Scientific Reports* **12** 15421
[20] Luo T Y, Zheng Y Z, Fu X and Deng Y X 2024 *Chin. Phys. B* **33** 120302
[21] Mao Y, Liu Y and Yang Y 2023 *IEEE INFOCOM* 2023-*IEEE Conference on Computer Communications* (IEEE) pp. 1–10
[22] Jozsa R 2004 *IBM Journal of Research and Development* **48** 79
[23] Cheng X, Chen X, Cao K, Zhu P, Feng S and Guan Z 2023 *Quantum Inf. Process.* **22** 187
[24] Bennett C H, Brassard G, Crépeau C, Jozsa R, Peres A and Wootters W K 1993 *Phys. Rev. Lett.* **70** 1895
[25] Yimsiriwattana A and Lomonaco Jr S J 2004 arXiv:quant-ph/0402148
[26] Niu J, Zhang L, Liu Y, Qiu J, Huang W, Huang J, Jia H, Liu J, Tao Z, Wei W, *et al.* 2023 *Nat. Electron.* **235** 241
[27] Houshmand M, Mohammadi Z, Zomorodi-Moghadam M and Houshmand M 2020 *International Journal of Theoretical Physics* **59** 1315
[28] Daei O, Navi K and Zomorodi M 2021 *International Journal of Theoretical Physics* **60** 3494
[29] Cambiucci W, Silveira R M and Ruggiero W V 2023 *2023 IEEE International Conference on Quantum Computing and Engineering (QCE)* vol. 2 (IEEE) pp. 268–269
[30] Daei O, Navi K and Zomorodi-Moghadam M 2020 *International Journal of Theoretical Physics* **59** 3804
[31] Yan J T 2023 *IEEE Transactions on Quantum Engineering* **4** 1
[32] Andres-Martinez P and Heunen C 2019 *Phys. Rev. A* **100** 032308