


# Resource-aware distributed quantum circuits partitioning strategy and transmission cost optimization

Pengcheng Zhu(朱鹏程), Zongyuan Dai(戴宗原), Lihua Wei(卫丽华), Jin Qian(钱进), and Shi-Guang Feng(冯世光)
**Citation:**Chin. Phys. B, 2026, 35 (**7**): 070302. DOI: 10.1088/1674-1056/ae13ec
http://cpb.iphy.ac.cn; http://iopscience.iop.org/cpb

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

---

## Distributed quantum circuit partitioning and teleportation optimization based on a multi-dimensional evaluation strategy

Le Zhang(张乐), Zhijin Guan(管致锦), Shuo Qin(秦硕), Zheng Luo(罗政), Fei Ding(丁飞), and Xueyun Cheng(程学云)
Chin. Phys. B, 2026, 35 (**5**): 050305. DOI: 10.1088/1674-1056/ae1c30

## Hierarchical QAOA circuit design framework for distributed quantum computing

Ting-Yu Luo(骆挺宇) and Yu-Xin Deng(邓玉欣)
Chin. Phys. B, 2026, 35 (**4**): 040309. DOI: 10.1088/1674-1056/ae00ae

## Distributed Kuperberg's algorithm

Peng-Yu Yang(杨鹏宇), Xin Zhang(张新), and Song Lin(林崧)
Chin. Phys. B, 2026, 35 (**3**): 030303. DOI: 10.1088/1674-1056/adfb58

## 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)
Chin. Phys. B, 2025, 34 (**5**): 050305. DOI: 10.1088/1674-1056/adbada

## 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

---

# Resource-aware distributed quantum circuits partitioning strategy and transmission cost optimization

Pengcheng Zhu(朱鹏程)[1,2,†], Zongyuan Dai(戴宗原)[1], Lihua Wei(卫丽华)[1], Jin Qian(钱进)[1], and Shi-Guang Feng(冯世光)[3,‡]

[1] *College of Information Engineering, Taizhou University, Taizhou* 225300, *China*
[2] *College of Information Engineering, Suqian University, Suqian* 223800, *China*
[3] *School of Computer Science and Engineering, Sun Yat-sen University, Guangzhou* 510006, *China*

(Received 24 June 2025; revised manuscript received 14 October 2025; accepted manuscript online 16 October 2025)

To overcome the physical limitations of current quantum hardware in terms of available qubits and connectivity, distributed quantum computing (DQC) has emerged as a promising and scalable paradigm. However, in distributed settings, cross-node qubit interactions incur high communication overhead due to the use of costly quantum communication protocols. Efficient circuit partitioning and transmission cost optimization have thus become key challenges. This work addresses the often-overlooked issues of hardware heterogeneity and redundant transmission by proposing a resource-aware partitioning and transmission cost optimization method for distributed quantum circuits. First, we develop a partitioning framework constrained by qubit resources, which accommodates node capacity differences to enable flexible qubit allocation. Second, we model gate dependencies using a directed acyclic graph (DAG) representation and introduce formal criteria to detect "initial-state" and "final-state" redundancies. A measurement-reset strategy is then employed to replace part of the quantum communication, reducing inter-node data transmission. Experimental results on a variety of benchmark circuits and heterogeneous architectures demonstrate that our method significantly reduces transmission cost and improves overall resource utilization. These findings offer both theoretical insight and practical guidance for efficient distributed quantum computing.

**Keywords:** distributed quantum computing, quantum circuits, circuits partitioning, resource-aware

**PACS:** 03.67.–a, 03.67.Lx, 03.67.Ac

**DOI:** 10.1088/1674-1056/ae13ec **CSTR:** 32038.14.CPB.ae13ec

## 1. Introduction

Due to the limitations of current noisy intermediate-scale quantum (NISQ) hardware, quantum processors still suffer from physical constraints such as limited available qubits and restricted qubit connectivity.[1] Under these constraints, enhancing computational capability on existing quantum hardware has become a key direction in quantum computing research. Distributed quantum computing (DQC), as a scalable computational model, provides an effective approach to overcome current hardware bottlenecks by partitioning large quantum circuits across multiple physically separated quantum processing units (QPUs) for collaborative execution.[2,3] DQC not only breaks the scale limitation of single devices, but also offers a technical foundation for building modular and extensible quantum computing platforms.

A typical DQC architecture consists of multiple QPU nodes interconnected via a network. These nodes may be multi-core structures on the same chip or physically distributed quantum devices connected through a quantum network.[4,5] Inter-node quantum communication generally relies on protocols such as quantum teleportation,[6] telegate[7] or entanglement swapping,[8] enabling logical coupling of qubits across remote nodes. This collaboration allows global quantum gates involving qubits from different nodes to be executed jointly, enabling the solution of complex quantum tasks.

Although DQC exhibits good theoretical scalability, several critical challenges remain in practical deployment. From the hardware-infrastructure perspective, recent years have seen notable advances in quantum interconnects, enabling quantum state transfer and remote entanglement generation across different temperature regimes, distance scales, and link modalities.[9–12] Nevertheless, current interconnects face several persistent bottlenecks: low entanglement generation rates and limited throughput over long distances; loss, noise, and mode mismatch in photonic channels that degrade link fidelity and stability; and difficulty in integrating heterogeneous quantum modules. From the logic perspective, especially in circuit partitioning and communication efficiency, since inter-node quantum operations depend on quantum communication protocols, frequent cross-node interactions significantly increase communication load, latency, and error rates.[13,14] Therefore, one of the core problems in DQC is how to partition global circuits across QPU nodes while minimizing inter-node quantum state transmission without compromising computational

[†]Corresponding author. E-mail: zhupcnt@163.com
[‡]Corresponding author. E-mail: fengshg3@mail.sysu.edu.cn

© 2026 Chinese Physical Society and IOP Publishing Ltd. All rights, including for text and data mining, AI training, and similar technologies, are reserved.

correctness.

To address these challenges, various circuit partitioning and communication optimization frameworks have been proposed. Zomorodi *et al.*[15] modeled quantum circuits with a connectivity matrix and applied an NSGA-III genetic algorithm for balanced bipartitioning. Wu *et al.*[16] proposed the AutoComm framework based on burst communication optimization to reduce latency and resource usage. Dadkhah *et al.*[17] re-ordered circuits based on qubit locality and combined genetic algorithms and tabu search to partition the quantum circuit map model. Davarzani *et al.*[18] formulated partitioning as an integer linear programming problem to distribute circuits over $K$ partitions while minimizing teleportation cost. Mao *et al.*[19] applied a multilevel hybrid simulated annealing algorithm to optimize qubit allocation and reduce communication overhead. Cheng *et al.*[20] used transmission cost as a circuit partitioning metric directly and proposed a merged-transfer-based optimization strategy. Luo *et al.*[21] proposed AutoArch, an automated quantum chip network structure design tool, to reduce the number of remote CNOT gates in quantum algorithms. Chen *et al.*[22] introduced a spectral clustering-based circuit partitioner and a search-tree-based global gate merged transfer strategy to effectively reduce the transmission cost of distributed quantum circuits. Chen *et al.*[23] adopted a quantum annealing-based partitioner and a look-ahead transmission qubit selection strategy to reduce execution time.

However, most existing methods assume uniform qubit availability across QPUs, which leads to significant resource waste and overestimation of transmission costs in heterogeneous systems. To overcome these issues, this paper focuses on resource-aware circuit partitioning and transmission cost optimization in DQC. Our main work is as follows.

(i) We propose a partitioning framework that incorporates qubit resource constraints. Unlike traditional approaches enforcing balanced partitions regardless of hardware capabilities, our method improves resource utilization and reduces transmission cost without modifying the core partitioning algorithm, offering a more efficient solution for homogeneous and heterogeneous systems.

(ii) We further analyze transmission redundancy, including "initial-state" and "final-state" redundancies. A measurement-reset-based strategy is proposed to partially replace quantum teleportation. We formalize redundancy identification using DAG-based dependency analysis and design an optimization algorithm that reduces inter-node transmission while preserving circuit correctness.

This paper is organized as follows: Section 2 introduces the background of the DQC model and teleportation. Section 3 analyzes the limitations of existing partitioning methods and presents our qubit-resource-aware partitioning mechanism. Section 4 focuses on cross-node communication optimization, including redundancy detection and the proposed algorithm. Section 5 provides experimental evaluations. Section 6 concludes the paper.

## 2. Background

### 2.1. Distributed quantum computing model

DQC is a computational paradigm that decomposes large-scale quantum tasks and distributes them across multiple physical nodes for parallel execution. Each node possesses independent quantum computation and measurement capabilities and collaborates with other nodes via quantum and classical communication channels to complete the execution of the entire quantum circuit. Given the current limitations of quantum hardware in terms of qubit count, connectivity, and error rates, the distributed model offers a scalable architecture that breaks through the physical constraints of single quantum processors and enables the execution of more complex quantum algorithms. To illustrate the operational mechanism of this model, Fig. 1 presents a schematic overview of the DQC workflow. From left to right, the figure shows the partitioning of the quantum circuit, the mapping of logical qubits to physical hardware, and the collaborative execution across nodes in a distributed quantum architecture.

After circuit partitioning, each qubit is initially assigned to a specific node. During execution, qubit states may need to migrate across nodes, primarily due to the presence of global gates, *i.e.*, the quantum gates whose operands reside on different nodes. In contrast, local gates operate on qubits within the same node and incur no inter-node communication. Cross-node operations significantly increase the system's communication overhead.

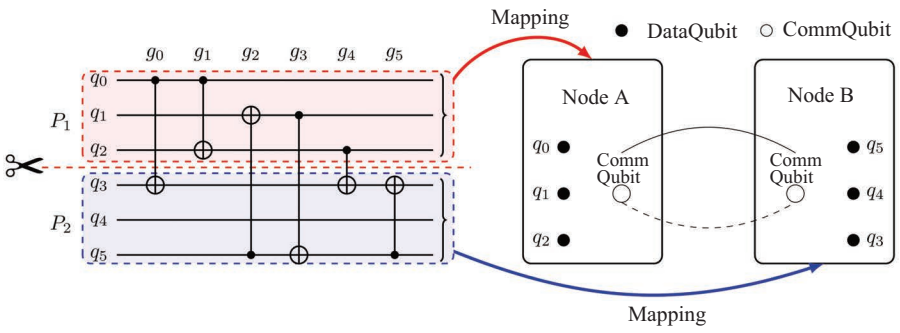



**Fig. 1.** Illustration of a distributed quantum computing model.

A DQC architecture typically consists of several interconnected QPU nodes linked by classical and quantum channels.[10,24] Qubits in the system are functionally categorized into data qubits and communication qubits. Data qubits carry the actual quantum information and are directly involved in circuit evolution, gate operations, and final measurements. Communication qubits are primarily used to support inter-node quantum communication. When a quantum gate operates on qubits located in different nodes, the system typically employs quantum teleportation to perform the operation. In this process, communication qubits act as a bridge to transmit the quantum state, at the cost of an entanglement pair and the transmission of measurement outcomes to reconstruct the quantum state on the target node. Therefore, during circuit partitioning and scheduling, a key optimization challenge is to effectively identify and eliminate unnecessary transmissions, and to allocate qubit resources efficiently. Such strategies are crucial for improving the performance and scalability of DQC systems.

### 2.2. Quantum teleportation

Quantum teleportation is a core technique for enabling quantum state transmission across nodes in DQC. It allows the transmission of an unknown quantum state from one node to another without physically moving the underlying quantum particle. This is achieved through the use of shared entanglement, classical communication, and local operations. The standard quantum teleportation protocol is the foundational framework for implementing inter-node quantum communication in quantum networks. As illustrated in Fig. 2, the protocol begins with the preparation of an entangled pair, which establishes quantum correlations between the source and target nodes. Upon performing a local Bell-state measurement on the source node and transmitting the outcome through a classical channel, a corresponding corrective operation is applied at the target node. This process effectively recreates the original quantum state at the destination.

Quantum teleportation underpins the execution of global quantum gates. When a gate involves data qubits located on different nodes, teleportation is used to inject the state of the source qubit into the target node. The required quantum operation can then be applied locally on the target node, ensuring correct cross-node gate execution. Each teleportation consumes shared entanglement and classical communication, and thus represents a unit of inter-node communication overhead. Therefore, the number of teleportations is usually considered as the transmission cost in DQC and is one of the most important optimization objectives.[25] Similarly, we define the transmission cost ($TC$) as the total number of quantum teleportations required to execute a given quantum circuit under a specific mapping and scheduling on a $k$-node architecture. This metric is hardware-agnostic and directly reflects the inter-node communication overhead in DQC.

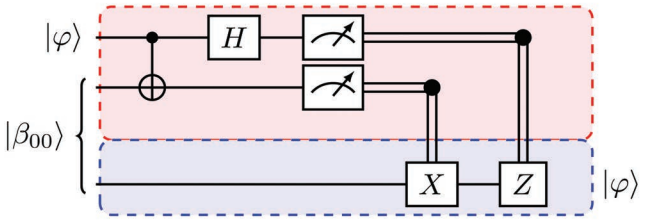



**Fig. 2.** Quantum teleportation protocol for inter-node state transmission.

## 3. Qubit-resource-aware partitioning of distributed quantum circuits

DQC enables collaboration across multiple nodes to overcome the qubit limitations of single-chip processors. However, conventional partitioning approaches often enforce strict load balancing without regard to underlying hardware constraints, resulting in poor resource utilization and high communication overhead. To address these issues, this section introduces a resource-constrained and flexible partitioning framework. The framework defines elastic partition size intervals to match the actual hardware capacities of heterogeneous nodes. This framework operates without altering the core logic of existing partitioning algorithms, allowing for effective reduction in the transmission cost. Overall, the proposed method offers a robust and efficient foundation for deploying distributed quantum circuits in heterogeneous quantum systems.

### 3.1. Problem definition and motivation

DQC mitigates the limitations of qubit count and connectivity on individual processors by partitioning large-scale quantum circuits into subcircuits and executing them across multiple QPUs. The quantum circuit partitioning problem can be formally defined as follows:

Given a quantum circuit $C = (Q, G)$, where $Q = \{q_1, q_2, \ldots, q_n\}$ is the set of logical qubits, and $G = \{g_1, g_2, \ldots, g_m\}$ is the set of quantum gates, the objective is to find a partition $P = \{P_1, P_2, \ldots, P_k\}$ over $k$ distributed QPU nodes $\{N_1, N_2, \ldots, N_k\}$, such that:

(i) $P_i \subseteq Q$ denotes the subset of qubits assigned to node $N_i$;

(ii) each node $N_i$ has a qubit capacity $C_i$ and includes dedicated communication qubits; the number of communication qubits $r_i$ is usually set to 1;

(iii) the constraint $|P_i| \leq C_i - r_i$ holds for all $i \in \{1, 2, \ldots, k\}$, which indicates that the number of logical qubits in the partition $P_i$ is not greater than the number of data qubits in the node $N_i$.

However, existing partitioning methods commonly rely on strong assumptions of load balancing, leading to significantly degraded performance on real quantum architectures.

These methods assume that logical qubits should be evenly distributed across QPU nodes to ensure balanced computational load. This balanced allocation strategy enforces a near-equal partitioning of $n$ logical qubits into $k$ nodes, *i.e.*, each node is assigned either $\lfloor n/k \rfloor$ or $\lceil n/k \rceil$ qubits. However, quantum circuits often contain groups of qubits with strong mutual coupling. A strict uniform partitioning may split such tightly coupled qubit groups across multiple QPUs, disrupting gate locality. As a result, operations that could be executed with no or few inter-node transmissions are fragmented into multiple unnecessary teleportations. This fragmentation not only directly increases transmission costs but also introduces additional delay and error accumulation due to the repeated consumption of entanglement resources and the overhead of classical coordination.

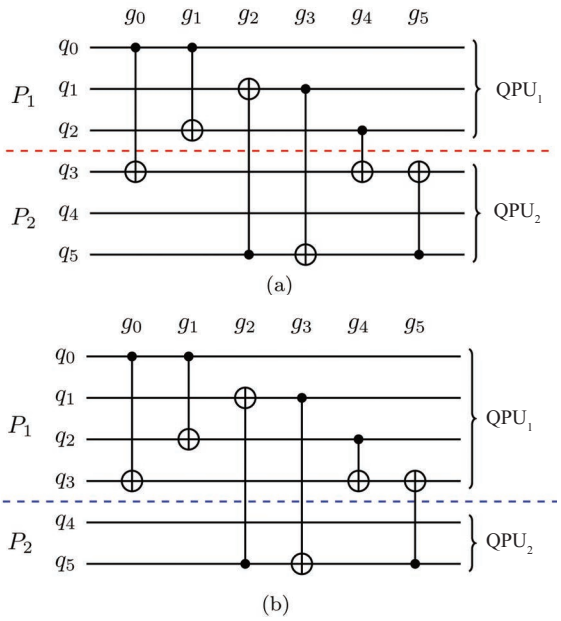



**Fig. 3.** Examples of two types of circuit partitioning: balanced and unbalanced.

For example, consider the 6-qubit circuit shown in Fig. 3(a). Under the red balanced partitioning scheme, the circuit incurs 4 global gates and requires 2 teleportations. In contrast, a flexible partitioning strategy assigns qubits $q_0$ through $q_3$ to one QPU and $q_4$ and $q_5$ to another, reducing the number of global gates to 3 and teleportations to 1, as shown in Fig. 3(b). This difference arises from the disruption of communication locality by the uniform strategy. It splits a highly interactive qubit group that should ideally be co-located, thereby introducing redundant state transmission. In contrast, the flexible strategy aligns partition sizes with both node capacity and circuit structure, thereby optimizing communication efficiency under resource constraints. This case study clearly illustrates that rigid equal-size partitioning can significantly amplify communication overhead by ignoring qubit coupling patterns. A resource-aware, flexible partitioning strategy mitigates this issue by intelligently allocating qubits to preserve locality and minimize inter-node transmission.

### 3.2. Partitioning method

Most existing distributed quantum circuit partitioning methods follow a "partition-then-adapt" workflow, in which the circuit is partitioned without considering hardware constraints, and only afterward are the subcircuits mapped to physical QPU nodes. This paradigm suffers from several key limitations: First, if the number of physical qubits in some nodes is insufficient, a forced uniform partitioning may generate infeasible solutions, rendering the task unexecutable. Second, in QPUs with higher qubit capacity, the uniform load-balancing strategy may leave a large portion of qubit resources underutilized. Third, to satisfy rigid partitioning constraints, algorithms may be forced to split strongly coupled qubit groups across nodes, which increases redundant teleportations due to broken gate locality.

Suppose a distributed architecture is built with IBMQ quantum processors that have a wide range of qubit capacities, from 5 to 133. Consider again the 6-qubit quantum circuit shown in Fig. 3. Under a homogeneous architecture with two QPUs, traditional load-balancing methods support only a single class of partitioning schemes, such as forced 3-3 equalization. This yields $\binom{6}{3} = 20$ possible partitioning options. Now suppose we select two real IBMQ processors, quito and belem, each with 5 physical qubits. If we reserve one communication qubit per node, each QPU can host up to 4 data qubits. In this heterogeneous setting, a wider range of feasible partitions becomes available, such as unbalanced 4-2 assignments. In this case, the total number of feasible partitions increases to $\binom{6}{3} + \binom{6}{2} = 20 + 15 = 35$, offering significantly more flexibility than the homogeneous model. This expansion in the feasible partition space provides a broader search landscape for downstream transmission cost optimization, and highlights the potential advantages of resource-aware and elastic partitioning strategies in practical DQC architectures.

To address the limitations of traditional strategies, this work proposes a novel "qubit-resource-aware partitioning" workflow, which adopts an "architecture–first, partition–later" process, as shown in Fig. 4. First, an architecture selection step chooses two or more QPUs from the available pool, each with its own qubit capacity, based on the circuit size. Next, rather than ignoring hardware during initial graph cuts, we feed both the circuit and the selected QPU capacities into an architecture-aware partitioning stage. This stage constrains the partitioner so that no subcircuit exceeds its host QPU's qubit budget. Depending on whether the chosen QPUs are identical (homogeneous) or differ in size (heterogeneous), the engine can produce balanced splits, *e.g.* 3–3, or unbalanced splits, *e.g.* 4–2, respectively. This method directly incorpo-

rates QPU hardware constraints into the partitioning process, without modifying the core of existing graph partitioning algorithms. This framework applies to both homogeneous and heterogeneous distributed quantum architectures.

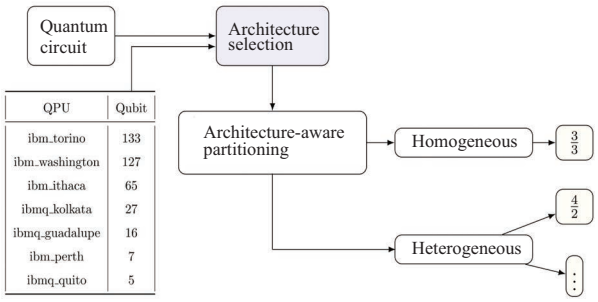



**Fig. 4.** The workflow of qubit-resource-aware partitioning.

In the architecture selection stage (this step corresponds to line 1 in Algorithm 2), our goal is to select a minimal subset $\mathcal{S}$ of candidate QPUs such that the total available data qubits (each QPU's physical capacity $C_i$ minus its reserved communication qubits $r_i$) meet or exceed the circuit's logical qubit demand $|Q|$. This constraint can be expressed as follows:

$$\sum_{N_i \in \mathcal{S}} (C_i - r_i) \geq |Q|. \tag{1}$$

Under this feasibility constraint, we pursue two optimization objectives: on the one hand, we minimize the number of selected QPUs, *i.e.*, $\min|\mathcal{S}|$, to save QPU resources; on the other hand, we maximize the average qubit utilization $U(\mathcal{S})$,

$$U(\mathcal{S}) = \frac{|Q| + \sum_{N_i \in \mathcal{S}} r_i}{\sum_{N_i \in \mathcal{S}} C_i}. \tag{2}$$

Here, the numerator $|Q| + \sum_{N_i \in \mathcal{S}} r_i$ represents the data qubits and communication qubits actually used by the QPUs in $\mathcal{S}$, and the denominator $\sum_{N_i \in \mathcal{S}} C_i$ represents the total qubit capacity of $\mathcal{S}$. The architecture selection process is implemented using a simulated annealing algorithm, which effectively explores the combinatorial search space to identify near-optimal QPU subsets under the given resource constraints. The pseudocode for the detailed process of architecture selection based on simulated annealing in line 1 of Algorithm 2 is presented in Algorithm 1.

**Algorithm 1:** SimulatedAnnealingSelect

**Input:** Candidate QPU set $\mathcal{H}$, logical qubit demand $|Q|$

**Output:** Selected QPU subset $\mathcal{S}$

$\mathcal{S} \leftarrow$ random subset of $\mathcal{H}$ satisfying constraint (1)
$T \leftarrow T_{\text{initial}}$
**while** $T > T_{min}$ **do**
  $\mathcal{S}' \leftarrow \text{perturb}(\mathcal{S})$; // Generate neighbor by adding/removing a QPU
  **if** $\sum_{N_i \in \mathcal{S}'}(C_i - r_i) \geq |Q|$ **then** // Check feasibility constraint
    $\Delta \leftarrow \alpha(|\mathcal{S}'| - |\mathcal{S}|) - \beta(U(\mathcal{S}') - U(\mathcal{S}))$; // Combined cost function
    **if** $\Delta < 0$ ***or*** $\exp(-\Delta/T) > random(0,1)$ **then**
      $\mathcal{S} \leftarrow \mathcal{S}'$
    **end**
  **end**
  $T \leftarrow T \times \text{cooling_rate}$
**end**
**return** $\mathcal{S}$

In the stage of architecture-aware partitioning (this step corresponds to lines 2–6 in Algorithm 2), we define an elastic interval for partition sizes and convert each QPU's physical qubit capacity into a tunable input constraint for the partitioning algorithm. For each QPU node $N_i$ with physical capacity $C_i$, assuming one dedicated communication qubit is reserved, the maximum number of data qubits is $C_i^{\max} = C_i - 1$. The permissible partition size for $N_i$ is defined as:

$$|P_i| \in [\alpha \cdot C_i^{\max}, C_i^{\max}], \tag{3}$$

where $\alpha \in (0,1]$ is a minimum utilization factor, and we further enforce $\alpha \cdot C_i^{\max} \geq 2$ to ensure each partition contains at least two qubits. In addition, $P_i$ needs to strictly satisfy $\sum_{N_i \in \mathcal{S}} |P_i| = |Q|$, to ensure that all logical qubits are divided into a partition.

Based on the partition size and the number of qubits in the circuit, a recursive approach can be used to generate all feasible vectors for circuit partition sizes. This step corresponds to the function GenerateSizeVectors in line 6 of Algorithm 2. To illustrate this function, consider an example where a 6-qubit circuit is to be partitioned across two QPUs, each with a physical capacity of 5. After reserving one com-

munication qubit per QPU, the maximum data qubit capacity becomes $C_i^{\max} = 4$ for each. Assuming a minimum utilization factor $\alpha = 0.5$, the permissible size range for each partition is $[\max(2, \lceil 0.5 \times 4 \rceil), 4] = [2, 4]$. The GenerateSizeVectors function will generate the set of all vectors $\boldsymbol{\ell} = (\ell_1, \ell_2)$ that satisfy $\ell_1 + \ell_2 = 6$ and $\ell_1, \ell_2 \in [2, 4]$, which are $(2, 4)$, $(3, 3)$, and $(4, 2)$. These vectors represent the different ways to distribute the 6 logical qubits across the two QPUs while adhering to their elastic capacity constraints.

Finally, the qubit resource constraints are passed as partition size requirements to standard graph partitioning algorithms such as spectral clustering or other heuristic methods (this step corresponds to lines 7–14 in Algorithm 2). For each valid partitioning scheme $\{P_i\}$, the partitioning algorithm is called with the minimum global gate count as the goal to find the optimal partitioning scheme among all partitioning schemes. This method solves the problem of low resource utilization and high cost of state transmission caused by the disconnection between load balancing assumptions and hardware resources in traditional methods. The pseudocode for all the processes mentioned above is shown in Algorithm 2.

**Algorithm 2:** Resource-AwarePartitioning (Architecture-First, Partition-Later)

**Input:** Circuit $C = (Q, G)$; candidate QPUs $\mathcal{H} = \{(N_i, C_i, r_i)\}$; min utilization $\alpha \in (0, 1]$.

**Output:** Selected QPU subset $\mathcal{S} \subseteq \mathcal{H}$; partition $\mathcal{P} = \{P_i\}_{N_i \in \mathcal{S}}$.

// Architecture selection

1 $\mathcal{S} \leftarrow$ SimulatedAnnealingSelect$(\mathcal{H}, |Q|)$; // $\sum_{N_i \in \mathcal{S}} (C_i - r_i) \geq |Q|$, $\min |\mathcal{S}|$, $\max U(\mathcal{S})$

// Capacity → elastic size constraints

2 **foreach** $N_i \in \mathcal{S}$ **do**

3 $\quad C_i^{\max} \leftarrow C_i - 1$; // reserve one communication qubit

4 $\quad L_i \leftarrow \max\{2, \lceil \alpha C_i^{\max} \rceil\}$, $U_i \leftarrow C_i^{\max}$;

5 **end**

6 $\mathcal{L} \leftarrow$ GenerateSizeVectors$(\{[L_i, U_i]\}_{N_i \in \mathcal{S}}, |Q|)$; // all (or sampled) $\boldsymbol{\ell}$ s.t. $\sum \ell_i = |Q|$

// Size-constrained partitioning; objective: minimize #global gates

7 $best_cut \leftarrow +\infty$, $best_\mathcal{P} \leftarrow \perp$;

8 **foreach** $\boldsymbol{\ell} = (\ell_i) \in \mathcal{L}$ **do**

9 $\quad \mathcal{P} \leftarrow$ SizeConstrainedPartition$(C, \mathcal{S}, \boldsymbol{\ell})$; // call existing graph partitioning methods

10 $\quad cut \leftarrow$ GlobalGateCount$(\mathcal{P}, G)$; // # two-qubit gates crossing nodes

11 $\quad$ **if** $cut < best_cut$ **then**

12 $\quad\quad best_cut \leftarrow cut$; $best_\mathcal{P} \leftarrow \mathcal{P}$;

13 $\quad$ **end**

14 **end**

15 **return** $\mathcal{S}$, $best_\mathcal{P}$

The overall runtime is dominated by the foreach loop in line 8 over the size of the set $\mathcal{L}$ and the partitioning call within each iteration in line 9. Assume that spectral clustering is used for partitioning, which has a complexity of $\mathcal{O}(n^3)$ and a total complexity of $\mathcal{O}\big(|\mathcal{L}| n^3\big)$, where $n$ is the number of qubits in the circuit.

## 4. Measurement-reset-based transmission cost optimization

This section focuses on optimizing transmission cost in DQC by leveraging qubit measurement and reset techniques to reduce unnecessary quantum state transmissions. We propose two optimization scenarios: initial-state transmission redundancy and final-state transmission redundancy. By analyzing the dependency structure of quantum circuits using a directed acyclic graph (DAG) representation, we formalize the conditions under which transmission can be deemed redundant. Based on these criteria, we design a transmission cost optimization algorithm. The proposed method enables a significant reduction in inter-node transmission costs.

### 4.1. Optimization scenarios

Qubit measurement and reset are fundamental operations in quantum computing and DQC. The feasibility of fast measurement and reset operations has been demonstrated on multiple superconducting and trapped-ion platforms.[26–28] Moreover, commercial cloud quantum platforms such as IBM Quantum (IBMQ) have provided native measure and reset instructions in Qiskit, allowing users to perform fast mid-circuit measurement and reset directly on superconducting backends. In the quantum teleportation protocol, measurement is performed on certain qubits to obtain classical outcomes, which are then used at the target node to reconstruct the quantum

state. Afterward, the communication qubit must be reset for reuse. Using local techniques such as qubit measurement and reset, it is possible to further optimize the transmission cost of distributed quantum circuits. In particular, we identify two types of optimizable scenarios: initial-state transmission redundancy and final-state transmission redundancy. For this purpose, we introduce the following definitions.

**Definition 1** In a DQC setting, the initial state refers to the state of a qubit before it is transmitted, typically prepared by the source node and intended for use at the target node in subsequent operations. The final state refers to the state of a qubit after it has completed its required operations on the target node. The qubit would conventionally be sent back to the source node for continued computation.

#### 4.1.1. Initial-state transmission redundancy

In cases where a qubit $q$ must be transmitted from a source node to a target node, and $q$ is initialized in a standard quantum state such as a basis state like $|0\rangle$ or $|1\rangle$ or a superposition state, e.g., $|+\rangle$, and so on, a series of states that can be prepared by one or more single-qubit gates. The standard teleportation protocol would incur a full quantum state transmission, resulting in initial-state transmission redundancy. However, such states can be locally prepared on the target node using basic single-qubit gates such as $X$ or $H$, without consuming teleportation resources.

This type of redundancy is prevalent in DQC. For instance, as illustrated in Fig. 5(a), if the source partition intends to teleport qubit $q_0$ in state $|0\rangle$ to a communication qubit $q_{\text{com}}$ in the target partition, the standard teleportation protocol incurs unnecessary resource consumption. In contrast, as shown in Fig. 5(b), the same outcome can be achieved by resetting $q_{\text{com}}$ and preparing it in $|0\rangle$ locally, thereby eliminating the need for teleportation. This optimization is especially relevant in quantum algorithms such as Grover's or Shor's algorithm, where initial preparation of uniform superposition states is required. If each qubit's superposition state were individually teleported from a source node, it would result in substantial redundant communication overhead. Instead, by resetting the communication qubit and applying a Hadamard gate $H$ directly on the target node, the $|+\rangle$ state can be prepared in place, avoiding the need for teleportation altogether.

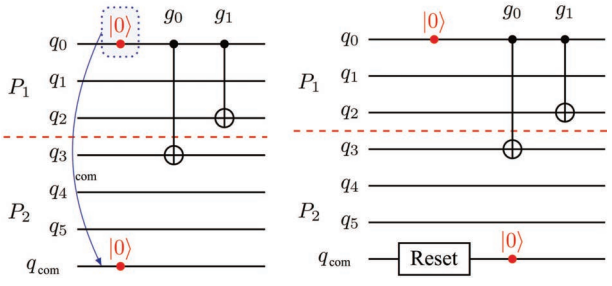



**Fig. 5.** Initial-state transmission redundancy.

In this scenario, the optimization strategy involves the following steps: first, detect the state of qubit $q$ to determine whether it is locally preparable; second, reset the communication qubit on the target node; finally, apply appropriate single-qubit gates to prepare the desired initial state. This approach is physically feasible, because the standard basis states can be realized through hardware-level reset operations. The states of the qubits after reset are all changed to $|0\rangle$. For some initial states, functional equivalence with teleportation can be established through the following transformations:

$$\text{Teleport}(|0\rangle_{\text{source}}) \equiv \text{Reset}(|\psi\rangle_{\text{target}}), \tag{4}$$

$$\text{Teleport}(|1\rangle_{\text{source}}) \equiv \text{Reset}(|\psi\rangle_{\text{target}}) + X(|0\rangle_{\text{target}}), \tag{5}$$

$$\text{Teleport}(|+\rangle_{\text{source}}) \equiv \text{Reset}(|\psi\rangle_{\text{target}}) + H(|0\rangle_{\text{target}}). \tag{6}$$

#### 4.1.2. Final-state transmission redundancy

Due to the no-cloning theorem, once a quantum state is transmitted across nodes via teleportation, it cannot be retained on the source partition. As a result, the quantum state must often be transmitted back from the target node to the source node after computation is complete. This leads to final-state transmission redundancy. However, in certain situations, this return transmission of $q$ can be omitted. If there is no subsequent quantum gate operation on $q$, or there is a quantum gate but it does not act on $q$, neither situation will affect the calculation result on $q$. In this case, the communication qubit is directly measured at the target node to obtain the classical result, and the communication qubit is further reset for subsequent state transmission.

Specifically, as illustrated in Fig. 6(a), when the state $|\psi\rangle$ of qubit $q_1$ is teleported from the source node to the target node, and its final state $|\psi'\rangle$ is no longer used in any subsequent operations (*i.e.*, no quantum gates act on $q$ in the remaining circuit), it is redundant to send $|\psi'\rangle$ back to the source node for measurement. Instead, as illustrated in Fig. 6(b), a local measurement on the target node suffices, eliminating the need for an additional teleportation.

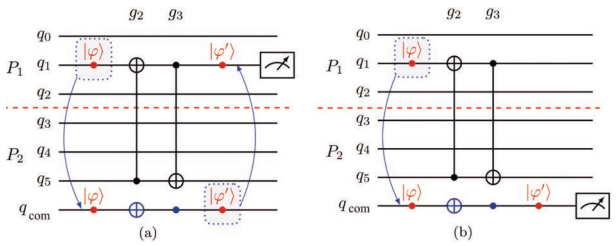



**Fig. 6.** Final-state transmission redundancy.

The optimization strategy for this case is as follows: first, determine whether $q$ has any gate dependencies in the subsequent circuit; second, if no future gates act on $q$, or only act on qubits other than $q$, then perform measurement directly on the communication qubit at the target node; finally, reset the communication qubit for reuse. Since quantum teleportation

maintains the consistency of the quantum state before and after transmission, the measurement result is independent of the physical location of the qubit. Thus, the equivalence can be expressed as follows:

$$\text{Measure}(\text{Teleport}(|\psi\rangle)) \equiv \text{Measure}(|\psi\rangle). \tag{7}$$

The above optimizations of initial-state transmission redundancy and final-state transmission redundancy apply in scenarios where teleportation is not strictly required. Initial-state redundancy typically arises in the early stages of a quantum circuit, particularly during initial state preparation. Final-state redundancy tends to occur near the end of the circuit, often just before measurement. Leveraging measurement and reset operations on communication qubits can significantly reduce unnecessary quantum teleportations and improve overall communication efficiency in DQC systems.

Initial-state and final-state transmission redundancy is widely present in distributed quantum circuits corresponding to quantum algorithms. Figure 7 demonstrates a 4-qubit QFT circuit mapped to two QPUs with partition $\{x_0, x_1\} \to N_A$ and $\{x_2, x_3\} \to N_B$. Let the order of the gates be $g_1, \ldots, g_{10}$, and $g_3$,$g_4$ ,$g_6$, and $g_7$ be global gates. If $|x_1\rangle$ is selected as the transmission qubit of $g_3$, then $g_3$, and $g_4$ conform to the final-state transmission redundancy shown in Fig. 6. Similarly, $g_6$ and $g_7$ also conform to the final-state transmission redundancy. Therefore, the transmission cost can be reduced from 4 to 2. Moreover, if $|x_1\rangle$ is chosen as the transmission qubit of $g_3$, and $|x_1\rangle$ is a state that can be prepared by some single quantum gate, then $g_3$ conforms to the initial-state transmission redundancy shown in Fig. 5.

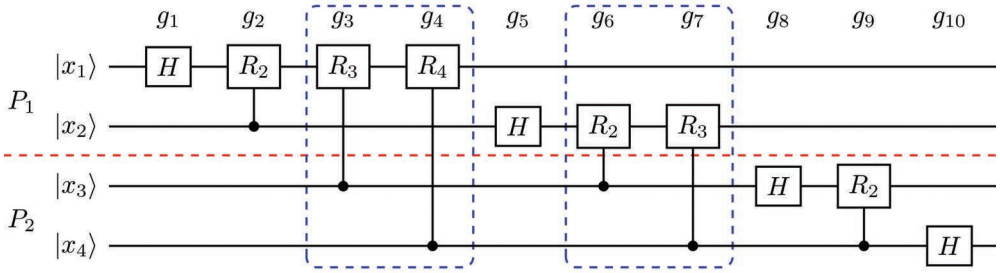



**Fig. 7.** Redundant transmission in QFT circuit.

### 4.2. Transmission cost optimization method

To optimize transmission cost in distributed quantum circuits, it is essential to determine whether teleportation operations are truly necessary. This requires examining whether any quantum gates before or after a global gate influence the feasibility of omitting teleportation. If there exist dependent or interfering operations, the associated teleportation cannot be optimized away. Hence, the key to identifying redundant communication lies in detecting the dependency relationships between global and non-global quantum gates. To achieve this, we introduce the quantum gate dependency graph, which explicitly models data dependencies and temporal constraints among gates in the circuit. The quantum gate dependency graph is a directed acyclic graph (DAG). This graph provides a structured and visual framework for assessing the necessity of quantum communication operations.

A quantum gate dependency graph abstracts the quantum circuit as a DAG $G = (V, E)$, where each vertex $v_i \in V$ represents a quantum gate operation $g_i$, and a directed edge $e_{ij} \in E$ indicates a dependency from gate $g_i$ to $g_j$. The label on edge $e_{ij}$ denotes the qubit(s) involved in this dependency. If no path exists between two gate nodes, they are logically independent. Figure 8(a) illustrates the DAG for the circuit in Fig. 3(a), where each node represents a gate and edges reflect input–output qubit dependencies of gates. To model distributed quantum circuits more precisely, we extend the DAG representation by annotating each gate node with an attribute indicating whether it is a local gate or a global gate, and visually distinguish them using solid lines (local) and dashed lines (global), as shown in Fig. 8(b).

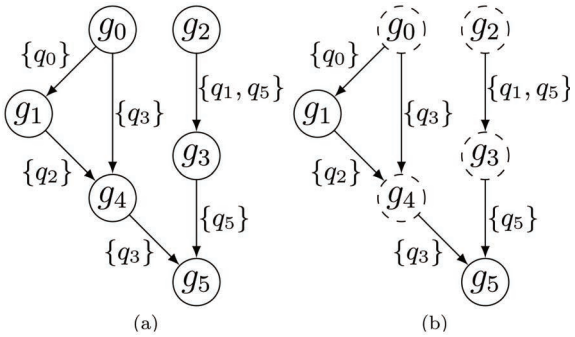



**Fig. 8.** Dependency graph for quantum gates in quantum circuits and distributed quantum circuits.

Based on this dependency graph model, identifying communication redundancy reduces to analyzing the topological connections of global gate nodes. Specifically, initial-state transmission redundancy can be detected if a global gate node satisfies one of the following three conditions. (i) The current global gate node has no predecessor nodes in the DAG; (ii) The ancestor nodes of the current global gate node are all single-qubit gates; (iii) Among the ancestor nodes of the

current global gate node there exist two-qubit gates, and those two-qubit gates do not act on the transmission qubit.

Figure 9 illustrates these three cases. In all examples, $g_0$ denotes the current global gate, and $q_0$ is the transmission qubit: In Fig. 9(a), condition 1 applies: $g_0$ has no predecessors in the DAG, implying that there is no quantum gate before the global gate $g_0$. In Fig. 9(b), condition 2 applies: $g_0$ is only preceded by single-qubit gates acting on $q_0$. That is, in the DAG, node $g_0$ has only single-qubit gates as ancestors. In Fig. 9(c), condition 3 applies: the only ancestor is a two-qubit gate that does not involve $q_0$, hence it has no effect on $q_0$'s initial state. In all these scenarios, the transmission of the initial state of $q_0$ can be eliminated. Instead, the target partition can reset its communication qubit and reconstruct the required state locally, thereby avoiding one instance of unnecessary initial-state teleportation.

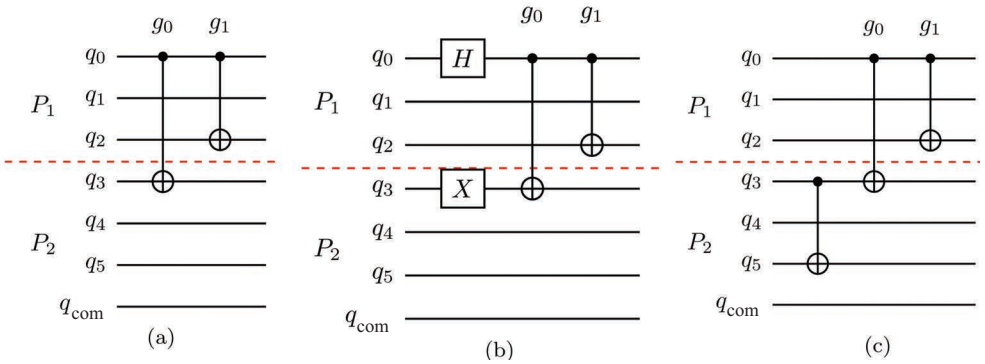



**Fig. 9.** Three cases of initial-state redundancy.

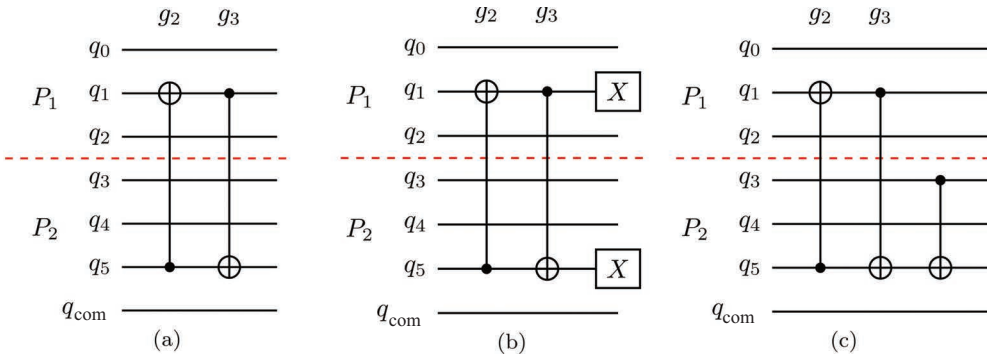



**Fig. 10.** Three cases of final-state redundancy.

Similarly, identifying final-state transmission redundancy in the DAG representation requires that a global gate node satisfies one of the following three conditions. (I) The current global gate node has no successor nodes in the DAG; (II) The descendant nodes of the current global gate node are all single-qubit gates; (III) Among the descendant nodes of the current global gate node there exist two-qubit gates, and those two-qubit gates do not act on the transmission qubit.

Figure 10 illustrates these three scenarios in a distributed quantum circuit. Let $g_3$ denote the global gate under consideration, and $q_1$ be the transmission qubit. Assume that the state of $q_1$ is transmitted to the communication qubit $q_{\text{com}}$ in the target partition. Final-state redundancy occurs if, after the operation of $g_3$, the state does not need to be sent back to the source partition $P_1$. In Fig. 10(a), condition 1 applies: the gate $g_3$ has no successors in the DAG, implying that the resulting state on $q_{\text{com}}$ is never used again. In Fig. 10(b), condition 2 applies: the only descendant nodes are single-qubit gates, whose outputs can be measured locally. In Fig. 10(c), condition 3 applies: the only descendant node is a two-qubit gate that does not involve the transmission qubit $q_1$, hence the resulting state has no further impact on the computation. In all these cases, the final state of the transmission qubit can be measured directly at the target partition. The corresponding communication qubit $q_{\text{com}}$ can then be reset and reused in subsequent operations, thereby eliminating the need for teleporting the final state back to the source partition. This leads to the reduction of unnecessary final-state transmissions.

By constructing a quantum gate dependency graph in the form of a DAG for a given distributed quantum circuit, we systematically characterize both the data dependencies and temporal orderings among gates. The redundancy detection for initial and final state transmission is thus transformed into a graph analysis problem involving predecessor and successor relationships of global gate nodes. This approach not only enables a formal method for identifying transmission redundancy but also provides an intuitive, structural basis for visualizing optimization opportunities. It is particularly useful dur-

ing circuit partitioning and communication scheduling stages, allowing early identification of regions that can avoid redundant transmission.

### 4.3. Algorithm and example

To translate the aforementioned redundancy detection method into an executable optimization procedure, this work proposes a DAG-based optimization framework to identify and eliminate unnecessary state transmissions in distributed quantum circuits. Specifically, we utilize the quantum gate dependency graph to detect redundant initial-state and final-state transmissions associated with global gates. In prior works,[20,23] the merged transmission strategy groups consecutive cross-partition global gates into a transmission queue, performing one initial-state transmission at the start of the queue and one final-state transmission at the end. While this method effectively reduces the overall number of teleportations, it overlooks the necessity of transmissions at the boundary gates of the queue; specifically, whether the first and last global gates truly require teleportation. To further optimize the merged transmission approach by addressing these residual redundancies, we propose a redundant transmission optimization algorithm based on the DAG model. This algorithm serves as a complementary mechanism to merged transmission, focusing on analyzing the dependency structure of the first and last global gates in each transmission queue.

Given a constructed DAG as input, the algorithm iterates through each transmission queue and inspects the head and tail global gate nodes. It identifies optimizable transmission operations by applying a breadth-first search (BFS) algorithm on the DAG to find ancestor or descendant nodes and verifying whether the head gate and tail gate in a transmission queue satisfy any of the three structural patterns indicating initial-state or final-state redundancy, respectively. If such patterns are found, the corresponding transmission operations are marked as optimizable. The pseudocode for the DAG-based redundant transmission detection and optimization is presented in Algorithm 3.

For each transmission queue $T_i$ $(i = 1,\ldots,n)$ in the merged transmission list $\mathcal{T}$, Algorithm 3 inspects the head and tail global gate nodes. To identify initial-state or final-state redundancy, it performs up to two BFSs on the DAG $= (V,E)$ to verify ancestor or descendant relationships. Each BFS traversal has a worst-case time complexity of $O(|V|+|E|)$, where $|V|$ is the number of gate nodes and $|E|$ is the number of dependency edges. Since the algorithm processes $n$ such queues, the overall time complexity is $O(n(|V|+|E|))$.

**Algorithm 3:** RedundantTransmissionDetection

**Input:** Quantum gate dependency DAG $DAG = (V,E)$; merged transmission list $\mathcal{T} = \{T_1, T_2, \ldots, T_n\}$, where each $T_i = \{g_1, g_2, \ldots, g_k\}$ is a queue of global gates and each $T_i$ corresponds to a transmission qubit $q_t$.

**Output:** Optimized transmission plan $\mathcal{T}'$ with redundancy flags for each $T_i$.

Initialize $\mathcal{T}' \leftarrow \emptyset$;
**foreach** *transmission queue* $T_i \in \mathcal{T}$ **do**
  $g_{\text{first}} \leftarrow T_i[1]$;
  $g_{\text{last}} \leftarrow T_i[k]$ ;
  Set `init_redundant` $\leftarrow$ **False**;
  Set `final_redundant` $\leftarrow$ **False**;
  **if** $g_{first}$ *satisfies initial-state redundancy conditions* **then**
    `init_redundant` $\leftarrow$ **True**;
  **end**
  **if** $g_{last}$ *satisfies final-state redundancy conditions* **then**
    `final_redundant` $\leftarrow$ **True**;
  **end**
  Update $T_i$ with redundancy flags;
    $T_i$.`initial_redundant` $\leftarrow$ `init_redundant`;
    $T_i$.`final_redundant` $\leftarrow$ `final_redundant`;
  Append updated $T_i$ to $\mathcal{T}'$;
**end**
**return** $\mathcal{T}'$;

To demonstrate the workflow and feasibility of the proposed algorithm, we adopt the 6-qubit distributed quantum circuit shown in Fig. 3(a) as an example. In this circuit, qubits $q_0$, $q_1$, and $q_2$ are located in partition $P_0$, while qubits $q_3$, $q_4$, and $q_5$ belong to partition $P_1$. The circuit contains four cross-partition global gates: $g_0$, $g_2$, $g_3$, and $g_4$. Under the merged transmission strategy, these gates are grouped into two transmission queues: $[g_0, g_4]$ and $[g_2, g_3]$, corresponding to the transmission of qubits $q_3$ and $q_1$, respectively. The total transmission cost is 4.

The optimization result using the proposed redundant transmission detection algorithm is illustrated in Fig. 11. For each transmission queue, Algorithm 3 is applied to examine whether the initial-state or final-state transmission is redundant. For the queue $[g_0, g_4]$, the head gate $g_0$ satisfies condition 1 for initial-state redundancy, i.e., no predecessors in the DAG, and hence the corresponding teleportation can be removed, reducing the cost by one. For the tail gate $g_4$, we examine whether it satisfies any of the final-state redundancy conditions. However, since gate $g_5$ operates on the communication qubit after $g_4$, the condition for redundancy is not satisfied, and this final teleportation cannot be optimized. Similarly, for the queue $[g_2, g_3]$, gate $g_2$ satisfies Condition 1 for initial-state redundancy, and gate $g_3$ satisfies condition 3 for final-state redundancy, i.e., only one successor gate that does not act on the transmitted qubit, allowing both teleportations to be eliminated. Consequently, the total transmission cost is reduced from 4 to 1.

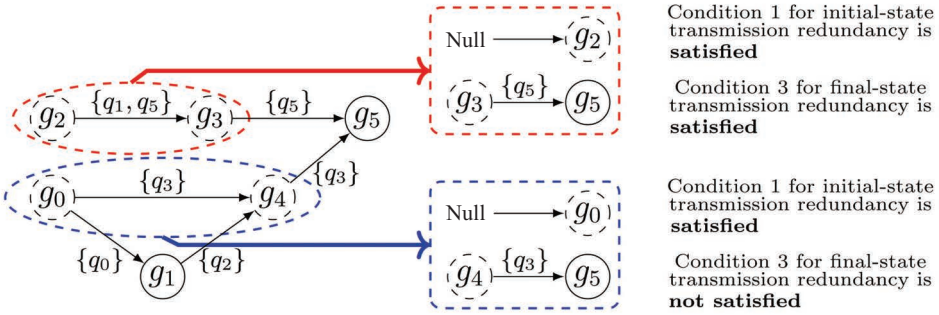



**Fig. 11.** Example of redundant transmission optimization based on DAG.

The proposed DAG-based redundant transmission detection algorithm serves as an effective enhancement to the merged transmission strategy. By identifying structural patterns of the head and tail global gates in each transmission queue, the algorithm further prunes unnecessary initial and final teleportations without altering the logic or structure of the queue. This optimization substantially reduces the transmission cost in distributed quantum circuits. Moreover, the method is highly generalizable and can be seamlessly integrated into mainstream distributed quantum compilation workflows.

## 5. Experiments and results analysis

To evaluate the effectiveness of our proposed methods, we selected three representative studies as comparative baselines. The authors in Ref. [15] adopted a genetic algorithm-based load-balanced partitioning strategy that evenly distributed qubits across subsystems. However, it did not consider physical qubit constraints or transmission redundancy optimization. This method represents the majority of existing load-balanced partitioning approaches. The authors in Ref. [22] proposed a spectral clustering-based circuit partitioning strategy, which was inherently unbalanced and similarly disregarded qubit capacity constraints and redundancy optimization for state transmissions. Both approaches utilized a merged transmission model for handling global gates. The authors in Ref. [29] used hypergraph partitioning combined with a Steiner tree to reduce the entanglement resource (ebit) consumption required for distributed quantum computing on heterogeneous modular quantum networks. It is important to note that their framework adopts a telegate protocol for nonlocal gate implementation, in contrast to our teleportation (teledata) approach. Furthermore, the authors in Ref. [29] assumed a specific compilation gate set (*e.g.*, {H,RZ,CRZ}) while we target the widely used CNOT and other single-qubit gates.

To validate our resource-aware qubit allocation strategy, we implement two versions given below.

(i) Method 1 (Base): Load-balanced partitioning and randomizing the 100 types of partitioning.

(ii) Method 2 (RA): Resource-aware partitioning without enforcing load balancing.

To ensure the influence of the measurement-reset-based transmission cost optimization (MRBTCO) is isolated, both method 1 and method 2 are initially evaluated without MRBTCO. The benchmark circuits are selected from Ref. [15], and the results are presented in Table 1.

Table 1 summarizes the transmission costs, including the transmission cost (TC) of Refs. [15,22,29], Base and RA. Whether via telegate or teledata, the TC metric in Refs. [15,22] and our method is fundamentally equivalent to the EPR pair count reported by Ref. [29], ensuring a fair and direct compar-

ison of the required entanglement resources.

**Table 1.** Comparisons with Refs. [15,22,29] (without MRBTCO) in TC.

| Circuit | Qubits | TC[15] | TC[22] | TC[29] | TC (ours) | |
|---|---|---|---|---|---|---|
| | | | | | Base | RA |
| min_alu_305 | 10 | 16 | 6 | 16 | 8 | 6 |
| sym9_147 | 12 | 8 | 10 | 12 | 14 | 14 |
| sym6_316 | 14 | 14 | 6 | 44 | 6 | 14 |
| parity_247 | 17 | 2 | 2 | 1 | 2 | 2 |
| ham7_299 | 21 | 34 | 14 | 14 | 12 | 10 |

The results indicate that resource-aware partitioning does not consistently outperform balanced partitioning across all circuits. In the RA, we select QPU combinations to maximize overall qubit utilization. For example, for the 10-qubit circuit min_alu_305, we select 5-qubit and 7-qubit quantum architectures to form a distributed architecture. At this time, the utilization rate of the QPU is 100%, which is higher than the utilization rates of the QPUs in two 7-qubit quantum architectures. However, this drive for maximum utilization can increase transmission cost, as observed in sym6_316, where RA's 7 + 5 + 5 heterogeneous partition yields higher transmission cost than the balanced 7 + 7 partition. Table 2 details the qubit resource utilization comparisons. In the table, "partitions" indicates the partitioning method of logical qubits in the circuit, and "QPUs" indicates the number of physical qubits of QPU in the distributed quantum architecture.

**Table 2.** Qubit resource utilization comparisons.

| Circuit | Qubits | Partitions[15] | QPUs[15] | Partitions (ours) | QPUs (ours) | Utilization (%) | |
|---|---|---|---|---|---|---|---|
| | | | | | | Ref. [15] | Ours |
| min_alu_305 | 10 | 5 + 5 | 7 + 7 | 6 + 4 | 7 + 5 | 85.71 | 100 |
| sym9_147 | 12 | 6 + 6 | 7 + 7 | 6 + 6 | 7 + 7 | 100 | 100 |
| sym6_316 | 14 | 7 + 7 | 16 + 16 | 6 + 4 + 4 | 7 + 5 + 5 | 50 | 100 |
| parity_247 | 17 | 9 + 8 | 16 + 16 | 13 + 4 | 16 + 5 | 59.38 | 90.48 |
| ham7_299 | 21 | 11 + 10 | 16 + 16 | 15 + 6 | 16 + 7 | 71.88 | 100 |

The proposed method achieves 100% qubit utilization on most circuits by flexibly selecting homogeneous or heterogeneous QPU combinations. It verifies that the architecture selection targeting QPU utilization can significantly improve the efficiency of hardware resource utilization. Although this strategy may increase the transmission cost due to partition fragmentation in certain circuits such as sym6_316, it can achieve both resource utilization and communication performance improvements in most scenarios.

To further demonstrate the effectiveness of MRBTCO in optimizing transmission cost by reducing redundant initial and final state transmissions, we apply MRBTCO to the Base and RA methods. The results are summarized in Table 3. We denote "Base + MRBTCO" as the load-balanced scheme with MRBTCO only, and "RA + MRBTCO" as the full method combining resource-aware partitioning and MRBTCO.

**Table 3.** Comparisons with Refs. [15,22,29] (with MRBTCO) in TC.

| Circuit | Qubits | TC[15] | TC[22] | TC[29] | Ours | | Imp (%) | |
|---|---|---|---|---|---|---|---|---|
| | | | | | Base + MRBTCO | RA + MRBTCO | Base + MRBTCO | RA + MRBTCO |
| min_alu_305 | 10 | 16 | 6 | 16 | 4 | 4 | 33.33 | 33.33 |
| sym9_147 | 12 | 8 | 10 | 12 | 9 | 9 | –12.50 | -12.50 |
| sym6_316 | 14 | 14 | 6 | 44 | 5 | 12 | 16.67 | –100.00 |
| parity_247 | 17 | 2 | 2 | 1 | 1 | 1 | 0.00 | 0.00 |
| ham7_299 | 21 | 34 | 14 | 14 | 11 | 8 | 21.43 | 42.86 |

As shown, MRBTCO generally leads to a significant reduction in transmission cost. The average optimization rates are 11.8% for Base, indicating that MRBTCO is highly adaptive and beneficial for circuits with significant initial/final-state redundancy. However, some circuits like sym6_316 and sym9_147 exhibit limited or negative optimization effects. The reasons are as follows. To maximize QPU utilization, RA employs uneven splits, *e.g.*, one 7-qubit and two 5-qubit QPUs for sym6_316. Although this achieves 100% physical-qubit utilization, it induces unbalanced logical-qubit communication, increasing the number of global gates and fragmenting potential merged transmissions. This is also the reason why the transmission cost in Ref. [29] is significantly higher than that of other methods. Moreover, the authors in Refs. [15,22] introduced gate-movement rules within their merged-transmission models, aggregating operations both temporally and spatially. This inherently merges more gates into a single transmission, lowering transmission cost.

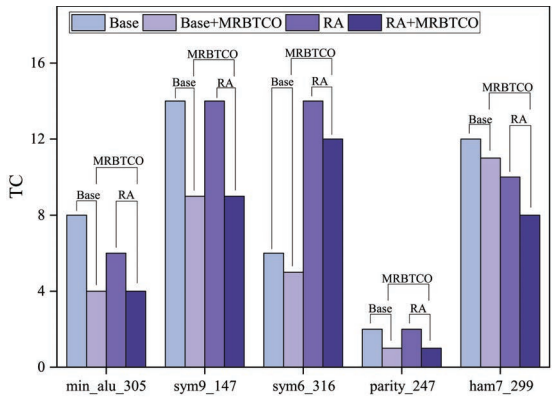



**Fig. 12.** Bar graph comparison of data from the four methods.

Figure 12 presents a bar chart of the four methods (Base, RA, Base + MRBTCO, RA + MRBTCO) across all benchmark circuits from Tables 1 and 3. It is evident that RA + MRBTCO consistently achieves the lowest transmission costs, and that integrating MRBTCO universally lowers the bars compared to their non-MRBTCO counterparts. This validates the effectiveness of MRBTCO.

Beyond the reversible-circuit baseline, we include four representative algorithm families from QASMBench[30] and MQTBench:[31] Deutsch–Jozsa (DJ), Ising, QFT, and QAOA, to assess the generality of our approach. All circuits are evaluated based on two consistent metrics: global gate count and TC, with lower being better. As summarized in Fig. 13, across a range of circuit sizes our method RA + MRBTCO achieves lower values on both metrics than those of Ref. [29] on most circuits. Even in QFT circuits, although RA + MRBTCO has more global gates, the TC is lower. However, on the DJ circuits, owing to the differing structure sensitivity and different gate sets of the methods, our approach slightly underperforms.[29]

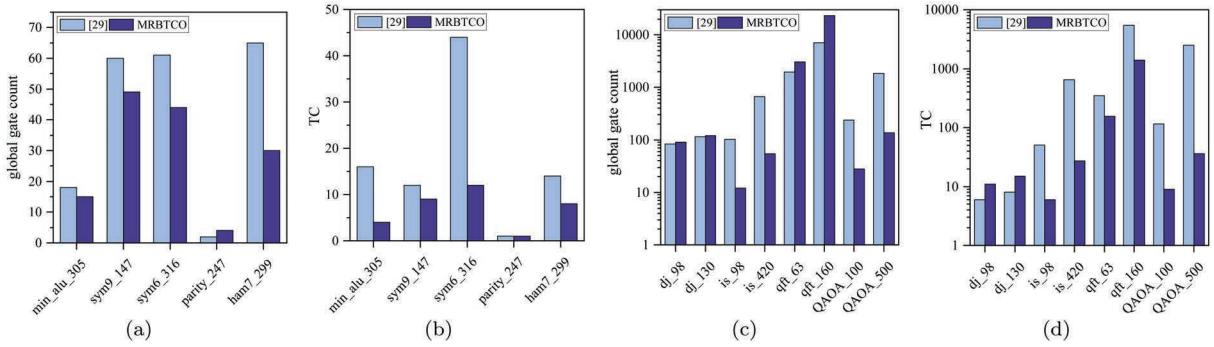



**Fig. 13.** Global gate count and transmission cost across several reversible circuits and quantum algorithms: MRBTCO *versus* Ref. [29].

To evaluate the robustness of the proposed method in reducing transmission cost for representative quantum algorithms in distributed settings, we further assess QFT and QAOA and compare against[22] under the resource-aware partitioning of Algorithm 2. For QFT, the number of qubits ranges from 32 to 256. For QAOA, we instantiate Max–Cut problems with layers $l = 10$ and set the qubit count to range from 100 to 1000. Both algorithm families are simulated under $k = 2$ to $k = 48$ partitions, and the resulting transmission costs are compared in Fig. 14. In the QFT, the transmission cost exhibits a regular increase with the qubit count $n$. This trend is consistent with the long-range controlled-rotation structure of QFT: partition cuts induce cross-chip interactions that accumulate with qubit count, thereby driving the transmission cost upward. In the QAOA, the transmission cost remains essentially stable over the explored range and is largely independent of $n$. Although the mixing and phase-separator layers contain more gates as $n$ increases, the number of global gates does not change, and therefore the transmission cost remains unchanged. These experiments further substantiate the effectiveness and robustness of the proposed method at larger scales.

It is also worth noting that the partitioning methods used in Refs. [15,22] do not explicitly account for practical QPU resource configurations during the architecture matching phase. In certain cases, this may result in a failure to find a suitable QPU combination or necessitate the use of QPUs with unnecessarily large physical qubit capacities, thereby leading to resource waste and reduced QPU utilization. In the current NISQ era, the quantum qubit resources are extremely scarce, and the number of logical qubits that a QPU can accommodate is strictly limited. Consequently, such resource inefficiencies are unacceptable in practical deployments and significantly constrain the scale of solvable problems. In contrast, the method proposed in the present study explicitly incorporates QPU capacity constraints during the partitioning process. By adopting a resource-utilization-first strategy, it flexibly selects either homogeneous or heterogeneous QPU combinations. This approach achieves 100% qubit utilization across multiple representative circuits, ensuring maximal hardware efficiency. Moreover, the method is designed with tunability, laying the groundwork for future research, specifically in exploring the optimal trade-off between the QPU resource utilization and inter-QPU communication overhead. This supports the development of adaptive distributed quantum circuit mapping strategies tailored to the specific architectural conditions.

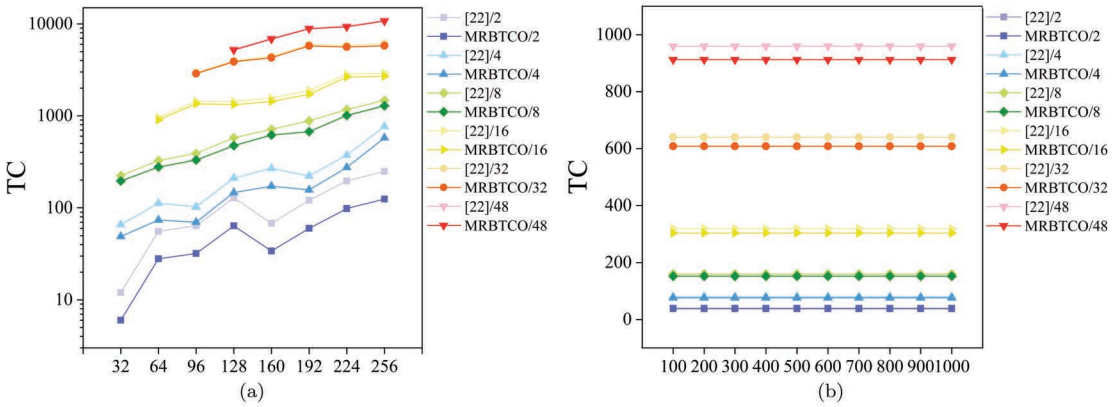



**Fig. 14.** Transmission cost across QFT (a) and QAOA (b) on 2–48 partitions: MRBTCO *versus* Ref. [22].

## 6. Related work

Recent research aims to reduce communication overhead, circuit depth, and error in quantum networks while adopting models and solvers that better reflect the physical constraints of heterogeneous architectures. Building on Ref. [32], the authors in Ref. [29] integrated hypergraph partitioning with the Steiner tree to accommodate heterogeneity and introduced an embedding technique[33] that uses a single EPR pair to implement as many controlled gates as possible locally, thereby lowering entanglement consumption.[34] The authors in Ref. [35] empirically compared multiple hypergraph partitioners and reported that the Fiduccia–Mattheyses (FM) algorithm achieves lower average EPR usage on benchmark circuits. For scheduling and routing, the authors in Ref. [19] formulated the communication cost of global gates under the Telegate communication model as an integer linear programming (ILP) objective and searched qubit assignments with a multi-stage hybrid simulated annealing method; the authors in Ref. [36] further modeled link stochasticity via EPR generation rates and entanglement-swapping success probabilities, adopted multipath routing to increase entanglement throughput, and retained the annealing framework to minimize expected cost. To address more complex networks, the authors in Ref. [37] coordinated Telegate and Teledata: subcircuits are first partitioned with Telegate and then stitched using Teledata, which significantly reduces the total number of EPR pairs. At the compiler level, the authors in Ref. [38] cast the problem as a Markov decision process (MDP) that jointly considers EPR generation and routing, global-gate scheduling, and local SWAP gate insertion, and leveraged deep Q-network (DQN) and double deep Q-network (DDQN) agents to learn policies that reduce circuit depth and error. Finally, the authors in Ref. [39] compared an entanglement-swapping–driven strategy (Telegate plus swapping) with a data-qubit–migration strategy (Teledata) on one-dimensional nearest-neighbor and hypercube topologies; relative to Ref. [32], both often yield shallower circuits, with the Teledata-based strategy typically performing better.

Research on distributed quantum circuit compilation has progressed from homogeneous to heterogeneous network models and from single to hybrid communication protocols. Going forward, the field should emphasize hardware- and resource-aware design that exploits concrete architectural characteristics (native gate sets, coupling graphs, coherence windows, link latencies) to deliver scalable, end-to-end compilers that jointly optimize qubit placement, global-gate scheduling, entanglement routing, and protocol selection (Telegate/Teledata). At the same time, we should actively explore ways to integrate distributed quantum circuit compilation with other quantum algorithms to promote the practical application of quantum computing.

## 7. Conclusion

In this paper, we address the communication overhead introduced by quantum state transmissions across QPUs in DQC. We propose a quantum circuit partitioning and optimization method that integrates resource constraints with redundancy detection. By incorporating hardware resource limits, we design an elastic partitioning strategy. We further construct a DAG-based gate dependency graph to identify redundant transmissions and eliminate them using measurement-reset operations. This ensures correct circuit execution while reducing the number of inter-node communications and improving resource utilization efficiency. Experimental results demonstrate that the proposed method delivers significant optimization effects across multiple quantum circuits and heterogeneous architectures. This validates its potential to inform the design and scheduling of DQC systems.

## Acknowledgments

Project supported by the National Natural Science Foundation of China (Grant No. 62072259), in part by the Natural Science Foundation of Jiangsu Province, China (Grant

No. BK20221411), and in part by the Quantum Science Strategic Initiative Project of Guangdong Province, China (Grant No. GDZX2303007).

## References


[1] Preskill J 2018 *Quantum* **2** 79

[2] Cuomo D, Caleffi M and Cacciapuoti A S 2020 *IET Quantum Communication* **1** 3

[3] Caleffi M, Amoretti M, Ferrari D, Illiano J, Manzalini A and Cacciapuoti A S 2024 *Computer Networks* **254** 110672

[4] Rodrigo S, Abadal S, Alarcon E, Bandic M, Van Someren H and Almud'ever C G 2021 *IEEE Micro* **41** 48

[5] Barral D, Cardama F J, Díaz-Camacho G, Faílde D, Llovo I F, Mussa-Juane M, Vázquez-Pérez J, Villasuso J, Piñeiro C, Costas N, *et al.* 2025 *Computer Science Review* **57** 100747

[6] Bennett C H, Brassard G, Crépeau C, Jozsa R, Peres A and Wootters W K 1993 *Phys. Rev. Lett.* **70** 1895

[7] Yimsiriwattana A and Lomonaco S J Jr 2004 arXiv preprint quant-ph/0402148

[8] Zukowski M, Zeilinger A, Horne M and Ekert A 1993 *Phys. Rev. Lett.* **71** 4287

[9] Magnard P, Storz S, Kurpiers P, Schär J, Marxer F, Lütolf J, Walter T, Besse J C, Gabureac M, Reuer K, *et al.* 2020 *Phys. Rev. Lett.* **125** 260502

[10] Niu J, Zhang L, Liu Y, Qiu J, Huang W, Huang J, Jia H, Liu J, Tao Z, Wei W, *et al.* 2023 *Nature Electronics* **6** 235

[11] Hu Z G, Xu K, Zhang Y X and Li B B 2023 *Chin. Phys. Lett.* **41** 014203

[12] Main D, Drmota P, Nadlinger D, Ainley E, Agrawal A, Nichol B, Srinivas R, Araneda G and Lucas D 2025 *Nature* **638** 383

[13] Chou K S, Blumoff J Z, Wang C S, Reinhold P C, Axline C J, Gao Y Y, Frunzio L, Devoret M, Jiang L and Schoelkopf R 2018 *Nature* **561** 368

[14] Qiu J, Liu Y, Niu J, Hu L, Wu Y, Zhang L, Huang W, Chen Y, Li J, Liu S, *et al.* 2023 arXiv preprint arXiv: 2302.08756

[15] Ghodsollahee I, Davarzani Z, Zomorodi M, P?lawiak P, Houshmand M and Houshmand M 2021 *Quantum Information Processing* **20** 1

[16] Wu A, Zhang H, Li G, Shabani A, Xie Y and Ding Y 2022 "Autocomm: A framework for enabling efficient communication in distributed quantum programs", 2022 55*th IEEE/ACM International Symposium on Microarchitecture* (MICRO) (IEEE) pp. 1027–1041

[17] Dadkhah D, Zomorodi M, Hosseini S E, Plawiak P and Zhou X 2022 *IEEE Access* **10** 70329

[18] Davarzani Z, Zomorodi M and Houshmand M 2022 *Sci. Rep.* **12** 15421

[19] Mao Y, Liu Y and Yang Y 2023 "Qubit allocation for distributed quantum computing", *IEEE INFOCOM* 2023-*IEEE Conference on Computer Communications* (IEEE) pp. 1–10

[20] Cheng X, Chen X, Cao K, Zhu P, Feng S and Guan Z 2023 *Quantum Information Processing* **22** 187

[21] Luo T Y, Zheng Y Z, Fu X and Deng Y X 2024 *Chin. Phys. B* **33** 120302

[22] Chen Z, Guan Z, Zhao S and Cheng X 2025 *Chin. Phys. B* **34** 050305

[23] Chen X, Chen Z, Zhu P, Cheng X and Guan Z 2025 *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* **44** 3350–3362

[24] Grebel J, Yan H, Chou M H, Andersson G, Conner C R, Joshi Y J, Miller J M, Povey R G, Qiao H, Wu X, *et al.* 2024 *Phys. Rev. Lett.* **132** 047001

[25] Van Meter R and Devitt S J 2016 *Computer* **49** 31

[26] Yu Y, Yan K, Biswas D, Zhang V N, Harraz B, Noel C, Monroe C and Kozhanov A 2025 *Phys. Rev. Research* **7** 043355

[27] Ding J, Li Y, Wang H, Xue G, Su T, Wang C, Sun W, Li F, Zhang Y, Gao Y, *et al.* 2025 *Phys. Rev. Appl.* **23** 014012

[28] Diniz C M, de Assis R J, de Almeida N G and Villas-Boas C J 2023 *Phys. Rev. A* **108** 052605

[29] Andres-Martinez P, Forrer T, Mills D, Wu J Y, Henaut L, Yamamoto K, Murao M and Duncan R 2024 *Quantum Science and Technology* **9** 045021

[30] Li A, Stein S, Krishnamoorthy S and Ang J 2023 *ACM Transactions on Quantum Computing* **4** 1–26

[31] Quetschlich N, Burgholzer L and Wille R 2023 *Quantum* **7** 1062

[32] Andres-Martinez P and Heunen C 2019 *Phys. Rev. A* **100** 032308

[33] Wu J Y, Matsui K, Forrer T, Soeda A, Andrés-Martínez P, Mills D, Henaut L and Murao M 2023 *Quantum* **7** 1196

[34] Cambiucci W, Silveira R M and Ruggiero W V 2023 "Hypergraphic partitioning of quantum circuits for distributed quantum computing", 2023 *IEEE International Conference on Quantum Computing and Engineering* (QCE), Vol. 2 (IEEE) pp. 268–269

[35] Fiduccia C M and Mattheyses R M 1988 "A linear-time heuristic for improving network partitions", *Papers on Twenty-five Years of Electronic Design Automation* pp. 241–247

[36] Mao Y, Liu Y and Yang Y 2023 "Probability-aware qubit-to-processor mapping in distributed quantum computing", *Proceedings of the* 1*st Workshop on Quantum Networks and Distributed Quantum Computing* pp. 51–56

[37] Sundaram R G, Gupta H and Ramakrishnan C 2022 "Distribution of quantum circuits over general quantum networks", 2022 *IEEE International Conference on Quantum Computing and Engineering* (QCE) (IEEE) pp. 415–425

[38] Promponas P, Mudvari A, Della Chiesa L, Polakos P, Samuel L and Tassiulas L 2024 arXiv preprint arXiv: 2404.17077

[39] Promponas P, Mudvari A, Della Chiesa L, *et al.* 2025 "Compiler for distributed quantum computing: a reinforcement learning approach", 2025-*IEEE International Conference on Communications*, IEEE, 2025, pp. 4615–4621