Efficient quantum circuit execution with continuous distribution of entanglement

Hao Fu,[1] Fangzheng Chen,[1] Mingzheng Zhu,[1] Chi Zhang,[2] Jun Wu,[1] Wei Xie,[1,*] and Xiang-Yang Li[1,3,†]
[1]*School of Computer Science and Technology, University of Science and Technology of China, Hefei 230027, China*
[2]*School of Computer Science and Information Engineering, Hefei University of Technology, Hefei 230009, China*
[3]*Hefei National Laboratory, University of Science and Technology of China, Hefei 230088, China*

(Received 4 March 2025; accepted 18 June 2025; published 8 July 2025)

In recent years, the rapid development of quantum networks and distributed quantum clusters has created new opportunities to implement large-scale quantum algorithms. However, this advancement also presents a significant challenge: efficiently executing quantum programs based on entanglement distribution protocols within quantum clusters. This paper analyzes existing entanglement distribution protocols in the context of distributed quantum circuit execution, focusing on the on-demand distribution of entanglement (on-demand protocols) and the continuous distribution of entanglement (CD protocols). We define the min-depth qubit mapping and scheduling problem within the CD protocols, and we propose a systematic solution that encompasses protocol analysis, circuit partitioning and mapping, and remote operation scheduling. Furthermore, we introduce a connectivity-first swapping (CFS) scheme that outperforms the single random swap (SRS) scheme in the execution of distributed quantum circuits in most cases. Extensive evaluations demonstrate the effectiveness of our algorithms, achieving significant reductions in circuit execution times of 52.4%, 49.9%, and 53.1% in clusters with average bandwidths of 1, 3, and 5, respectively. The CFS protocol further enhances performance, reducing execution time by 24.5%, 19.7%, and 17.8% compared to the SRS protocol. Our method also reduces entanglement costs, decreasing consumption by an average of 20.3% compared to the baseline method at an average bandwidth of 3, while the CFS protocol further reduces costs by 11.5% compared to the SRS protocol.

DOI: 10.1103/3xc7-j3mn

I. INTRODUCTION

In recent years, quantum computing has rapidly evolved, and numerous applications, which promise advantages over classical computing, have emerged. Recent advances in hardware technology, such as Sycamore [1], *Zuchongzhi* [2], and trap-ion quantum computers [3], have further validated the feasibility of quantum computing. However, as the scale of the systems increases, precise control becomes increasingly challenging. With the development of quantum networks, including link layer protocols [4] and entanglement swapping [5], as well as the hardware implementation of distributed quantum computing [6,7], there is a foreseeable future where quantum algorithms requiring a large number of qubits, such as Shor's algorithm [8], could be realized in distributed computing scenarios.

Executing large-scale quantum algorithms in distributed settings typically involves three steps: preprocessing for circuits and cluster characterization, circuit partitioning and mapping, and remote operations scheduling. First, during the preprocessing stage, quantum algorithms are converted into quantum circuits. We also acquire a hardware model of the entire distributed cluster, including the capabilities of quantum computers within the cluster and the network performance. Second, the quantum circuit is partitioned and allocated. Given that a large-scale quantum circuit may not be executed independently on a single machine, it is necessary to divide the circuit into subcircuits with fewer qubits. Each subcircuit is then mapped to a machine for execution. Inevitably, the involved qubits of some multiqubit gates may be distributed across different subcircuits, hence executed on different machines. Therefore, the third step involves managing these multiqubit quantum gates executed across different machines, known as remote operations. Typically, this requires entanglement between the involved qubits through quantum teleportation [9], with entanglement generation and entanglement swapping [5]. Then, we implement the remote operations with telegate or teledata [10]. This process consumes predistributed entangled pairs, such as EPR pairs, between the qubits of target quantum devices (also called nodes in the cluster). Given the significant variability in network conditions [e.g., the bandwidth (maximal number of distributed entangled pairs) between nodes may be different and entangled states may be consumed by other applications], execution strategies are essential to enhance circuit execution performance.

To efficiently manage tasks such as entanglement generation, swapping, and verification, researchers have developed two types of entanglement distribution protocols: on-demand distribution of entanglement (on-demand protocols) and continuous distribution of entanglement (CD protocol) [11,12]. These two protocols have been designed to address different application scenarios and requirements. On-demand protocols generate entanglement in response to requests from

*Contact author: xxieww@ustc.edu.cn
†Contact author: xiangyangli@ustc.edu.cn

specific nodes, ensuring immediate utilization of the generated entanglement. Conversely, CD protocols continuously generate, swap, and verify entangled states among nodes. However, existing quantum devices and distributed quantum clusters encounter several challenges: the limited lifetime of qubits, which constrains the persistence of generated entangled states; errors in quantum operations that may introduce uncertainty into calculation outcomes; and restricted connectivity within the cluster, where each machine can physically link to only a finite number of others. To execute large-scale quantum algorithms in a distributed manner, it is often necessary to consume substantial entanglement resources, specifically in the form of EPR pairs.

When executing a quantum circuit on a distributed system, on-demand protocols necessitate real-time requests for entanglement generation and swapping, a requirement that increases waiting times and, consequently, diminishes the overall performance of circuit execution to some extent, i.e., requiring more execution time for requests and responses. In contrast, CD protocols can allocate resources more efficiently and mitigate traffic congestion in large quantum networks, particularly when executing quantum circuits on distributed quantum clusters. For instance, CD protocols are capable of dynamically adjusting resource allocation to ensure optimal efficiency during high-demand or complex tasks. Therefore, in scenarios where efficiency and resource utilization are critical, the continuous entanglement distribution method should be prioritized: entanglement is generated continuously between nodes according to a specific continuous entanglement distribution protocol, and entanglement swapping occurs between any two nodes with a predefined scheme. An entanglement cutoff is implemented to discard entangled states beyond a certain limit [12–17]. In summary, the potential of continuous entanglement distribution protocols to enhance resource allocation and execution efficiency represents a promising avenue for the advancement of quantum computing.

Previous studies have investigated executing quantum circuits in quantum networks under certain assumptions, such as the ability to generate entanglement between any pair of nodes arbitrarily, the predistribution of entangled states, and the unlimited lifetime of entangled states. Based on these assumptions, various algorithms have been proposed to optimize communication costs and circuit execution times. For instance, Pablo *et al.* introduced a circuit distribution method based on hypergraph partitioning solvers [18], while Jonathan *et al.* employed a time-sliced partitioning technique to reduce the width and depth of quantum circuits [19]. Wu *et al.* [20,21] enhanced internode communication by analyzing communication patterns, whereas Mao *et al.* [22] concentrated on minimizing the total cost of remote operations among different processors. Cuomo *et al.* [10] aimed to reduce circuit depth. However, these works do not address the entanglement distribution protocols necessary for circuit execution in quantum distributed computing, and consequently they fail to optimize resource consumption and execution efficiency within the relevant protocols. Therefore, the development of entanglement distribution protocols remains a critical challenge for enhancing the performance of quantum circuit execution. In this study, we investigate the distributed execution of quantum circuits under a continuous entanglement distribution protocol, assuming that entanglement resources are abundant. As a result, our primary goal is to optimize the execution time of quantum circuits.

In this paper, we first analyze existing entanglement distribution protocols with the requirements of distributed quantum circuit execution, specifically the on-demand distribution protocol and the CD protocol of entanglement. We define the min-depth qubit mapping and scheduling problem in the continuous entanglement distribution protocol, and we propose a systematic solution, including protocol analysis, circuit partitioning and mapping, and remote operation scheduling. The protocol analysis gathers statistical information about the entanglement topology (virtual topology) among distributed cluster nodes during the operation of the network protocol. Based on this analysis, we adopt appropriate circuit partitioning schemes and design algorithms for efficiently mapping the partitioned circuits onto the nodes. Finally, we develop a remote operation execution scheme based on gate priority to minimize circuit execution time cost. Building on this foundation, we further propose a connectivity-first swapping (CFS) scheme, which is more suitable for the distributed execution of quantum circuits compared to the single random swap (SRS) scheme [12].

Extensive evaluations demonstrate the effectiveness of our proposed algorithms. In clusters with average bandwidths of 1, 3, and 5, our qubit mapping and scheduling method achieves significant reductions in circuit execution time compared to baseline methods, specifically 52.4%, 49.9%, and 53.1%, respectively. The CD protocol with the CFS scheme (CFS protocol) also provides performance improvements, reducing execution time by 24.5%, 19.7%, and 17.8%, respectively, compared to the SRS protocol. Additionally, our mapping and scheduling method significantly decreases entanglement costs, reducing entanglement consumption by an average of 20.3% compared to the baseline method on average bandwidth 3, while the CFS protocol further decreases entanglement costs by 11.5% compared to the SRS protocol.

The rest of this paper is organized as follows. We present preliminaries in Sec. II. In Secs. III and IV, we formulate a min-depth qubit mapping and scheduling problem in continuous distribution of entanglement and provide our mapping and scheduling algorithm, which consists of protocol analysis, circuit partitioning and mapping, remote operation scheduling, and connectivity-first entanglement swapping scheme. We evaluate our algorithms and analyze the performance in Sec. V. Related works are presented in Sec. VI. We conclude this work with future directions in Sec. VII. Table I presents a list of the notation symbols used in the paper.

II. PRELIMINARIES

A. Quantum teleportation and remote operation

1. Quantum teleportation

Quantum communication between nodes can be achieved through distant Einstein-Podolsky-Rosen (EPR) entanglement. An EPR pair maintains one of the Bell states, which represent four maximally entangled two-qubit states [23]. Preestablished entanglement can facilitate the transfer of

TABLE I. Notation symbols used in this paper.

Notations	Definition
D, d, n	Distributed quantum cluster, device and the number of nodes
G_D, E, e_{ij}	Physical connectivity graph of D, all physical links and physical link of d_i and d_j (entanglement can be distributed to d_i and d_j)
$\mathcal{G}_D, \mathcal{E}, \mathbf{e}_{ij}$	Entanglement link graph (virtual graph) of D, all virtual links and virtual link between d_i and d_j (d_i and d_j share an entangled pair)
$\mathcal{L}(\mathbf{e}_{ij}, t)$, T_{cutoff}	Remaining life time of $\mathbf{e}_{ij}$ at time t, cutoff time of all entanglement links
$\mathcal{A}(G, t)$	Execution results of CD protocol $\mathcal{A}$ at time t on cluster with physical link graph G, $\mathcal{A}(G, t) = \mathcal{G}_D^t$
$\mathcal{P}, G_{\mathcal{P}}, \mathcal{P}_{\text{cuts}}, \mathcal{P}'$	Quantum circuit, DAG representation of $\mathcal{P}$, sub-circuits of $\mathcal{P}$ and circuits with only remote operations
$\mathcal{M}, l(i)$	Mapping of sub-circuits to nodes and location of qubit i
$\mathcal{P}^s$, $\Delta_{\mathcal{P}^s}$	Scheduling result of $\mathcal{P}_{\text{cuts}}$ and its execution time cost
A_p, A_v	Communication matrix of sub-circuits $\mathcal{P}_{\text{cuts}}$, sampling virtual entanglement matrix
$\mathcal{C}_s, \mathcal{B}_d$	Communication demands of each sub-circuit, communication capabilities of nodes
$p_{\text{gen}}, p_{\text{swap}}, p_{\text{cons}}, q_{\text{swap}}$	Parameters in CD protocol: the probability of successful entanglement generation, the probability of successful entanglement swap, the probability that two nodes consume shared links p_{cons} in each time slot by other applications (not for target circuit execution), and the probability of performing swaps (only for the SRS protocol)

quantum states, remote operations, and the establishment of longer-distance quantum entanglement by entanglement swapping, as we will discuss further.

2. Remote operation

In traditional distributed computing scenarios, when two nodes need to share data, they can either replicate the data on both nodes or execute a task on one node and then transmit the data to another for further processing. However, in quantum distributed settings, neither of these approaches is applicable due to the no-cloning theorem for quantum states and the inherently error-prone nature of quantum operations. The interface between nodes can be established by directly transferring quantum information; however, losses in the interconnecting quantum channels may lead to irretrievable loss of quantum information [7]. Therefore, this work focuses on two methods of remote quantum operations based on EPR entanglement without data transfer among nodes. Assuming that nodes A and B share a pair of entangled qubits, we can utilize the circuit shown in Fig. 1(a) to perform remote operations between A and B, consuming one unit of entanglement, referred to as direct execution. When nodes A and B do not share entangled qubits, but nodes A and C and C and B do, we can implement remote operations between A and B using the circuit illustrated in Fig. 1(b), which requires two units of entanglement and is termed indirect execution. The consumption of entanglement depends on the distance between the nodes necessary for implementing remote operations. We refer to these two implementations as telegate [10], which do not require the movement of quantum data among nodes.

B. Entanglement distribution and quantum cluster

1. Entanglement distribution and swapping

Entanglement distribution and entanglement swapping serve as crucial foundations for quantum state transmission and remote quantum operations. These processes can generally be achieved through optical fibers [24,25] or free space [26,27]. When it is necessary to share an entangled state between nodes A and B, and a physical link exists between them, an EPR pair can be generated in which each node holds one entangled qubit. It is important to note that entangled states are constrained by their cutoff time, which is limited by storage duration, to minimize decoherence effects; exceeding this limit renders the entangled states unavailable [12–17]. Since quantum memory decoherence, which varies with hardware parameters, primarily determines the cutoff time, we adopt a simplified model that applies a uniform cutoff time across all entangled states.

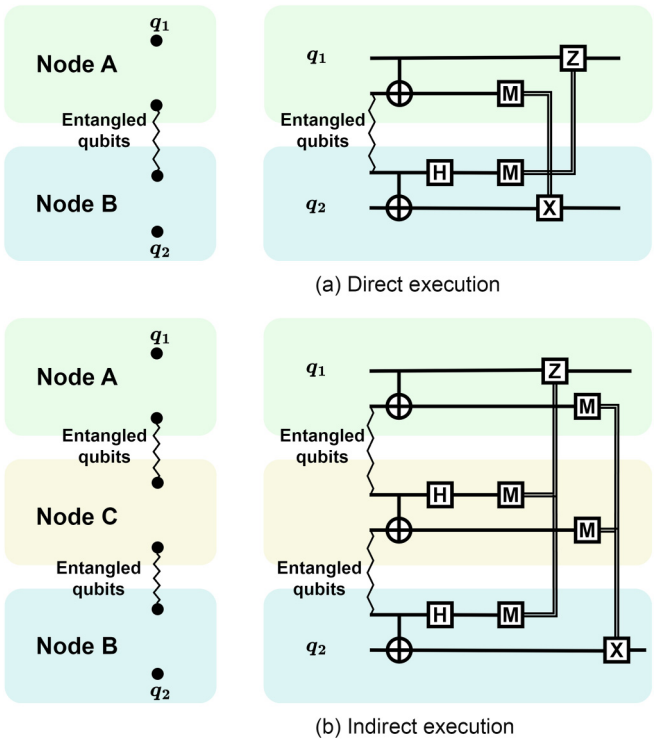


FIG. 1. The execution of remote operations with direct/indirect entangled qubit pairs.

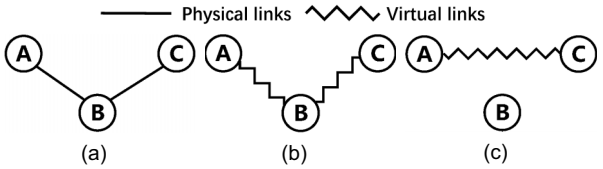


FIG. 2. Physical and virtual topologies of a small cluster. (a) A physical topology, (b) a virtual topology by generating entanglement between A and B, B and C, and (c) another virtual topology by entanglement swapping from (b).

Entanglement swapping allows the establishment of entanglement between two nodes that lack a direct physical link. For instance, if there are physical links between nodes A and B, and B and C, and we want nodes A and C to share an EPR pair, this can be achieved by performing a Bell state measurement (BSM) circuit on two qubits at node B, as the resulting virtual topology after this operation is illustrated in Fig. 2(c), i.e., nodes A and C share an entangled state. This method necessitates that entanglement be pregenerated between A and B, and between B and C, as shown in Fig. 2(b). Furthermore, entanglement swapping must be completed before the entanglement expires. This process consumes the entanglement between A and B, and between B and C, to produce entanglement between A and C. In general, operations can be executed concurrently at each intermediate node when establishing entanglement between a source node and a destination node through entanglement swapping performed at multiple intermediate nodes. Thus, the long-distance entanglement established via entanglement swapping is independent of the number of intermediate nodes between the two nodes.

When nodes A and B, A and C, or B and C share an entangled state, they are also considered virtual neighbors, and the links between them are referred to as virtual links. It is important to note that A and B can be both physical and virtual neighbors. We can represent a distributed quantum cluster's physical links and nodes as a physical connectivity graph to depict its topology, and the weights in the graph indicate the bandwidth between nodes, which means the number of physical links of two adjacent nodes. The physical links connect nodes by a physical channel with which nodes can send quantum states to other nodes. Similarly, the shared entangled state at a given time, represented as virtual links and nodes, can be depicted in a virtual connectivity graph to illustrate the current communication structure, in which the bandwidth means the number of shared entangled states currently. The physical and possible virtual connectivity graphs for a distributed quantum cluster composed of nodes A, B, and C are shown in Fig. 2.

2. Entanglement distribution protocols

There are two primary approaches to distributing entanglement among nodes: on-demand [11,28–31] and continuous protocols [12]. The operations mentioned earlier—entanglement generation, swapping, utilization, and application of cutoffs—are performed according to a specific protocol. In on-demand distribution of entanglement (on-demand protocols), entangled states are distributed only after nodes explicitly request them. Previous works on on-demand entanglement distribution typically consider solving a routing problem [28,30,31], which aims to distribute end-to-end entangled states efficiently within different network settings, and scheduling a set of operations on a subset of nodes. When employing on-demand protocols, it is necessary to adjust scheduling strategies to inform the relevant nodes to execute the required operations based on real-time demands. As the number of nodes involved in the entanglement generation process increases, the scheduling strategy can become significantly more complex, potentially resulting in substantial delays and complicating the ability to meet the requirements of real-time applications.

Conversely, continuous distribution of entanglement (CD protocols) means that entangled states are continuously distributed among the nodes. The entangled states that have been generated can be utilized by background applications (such as entanglement swapping and quantum circuit execution) before reaching their cutoff time. Unlike on-demand protocols, CD protocols do not necessitate a specific application-dependent schedule. Therefore, in the context of quantum circuit execution in distributed systems, CD protocols do not require frequent requests for entanglement distribution compared to on-demand protocols. The applications that execute the circuit only need to use entanglement and request entanglement swapping based on the current network entanglement topology, thereby potentially achieving higher efficiency. As illustrated in Fig. 3, the circuit shown in Fig. 3(b) needs to be executed on the distributed cluster depicted in Fig. 3(a). We partition the circuit into two subcircuits, allocating qubits q_0 and q_1 to node d_0, and qubits q_2 and q_3 to node d_1. Figure 3(c) demonstrates a possible execution process for implementing the circuit with on-demand distribution of entanglement. The application (which executes the quantum circuit) requests an entangled link between d_0 and d_1 for executing gate g_1. The protocol then implements the entanglement processing procedure, which includes parsing the request, identifying an entanglement path, distributing the entangled states, and then sending a response to the application for subsequent execution of g_1. These procedures—request, parsing, entanglement processing, and response—are iterated until circuit execution is complete. In Fig. 3(d), circuit execution constitutes part of the entanglement consumption phase in the CD protocol (the details of the CD protocol are presented in Sec. IV A). Compared to the on-demand scenario, executing a circuit in the CD network only requires utilizing preestablished entangled links rather than requesting entanglement distribution, which will become more complex when the number of nodes involved in this process is large.

III. PROBLEM FORMULATION

A. Distributed quantum cluster

We consider a distributed quantum cluster $D = \{d_1, d_2, \ldots, d_n\}$, where n denotes the number of nodes in D, and $|d_i|$ represents the number of qubits in d_i. The physical links of D are described by physical connectivity graph $G_D = \{D, E\}$, $E \subseteq D \times D$. The edge e_{ij} between d_i and d_j, $e_{ij} \in E$, exists if and only if there is a physical link. In CD protocols, time is divided into nonoverlapping time slots, and each operation is allocated within a time slot, as a common assumption in quantum networking [28]. We use T_{cutoff} to

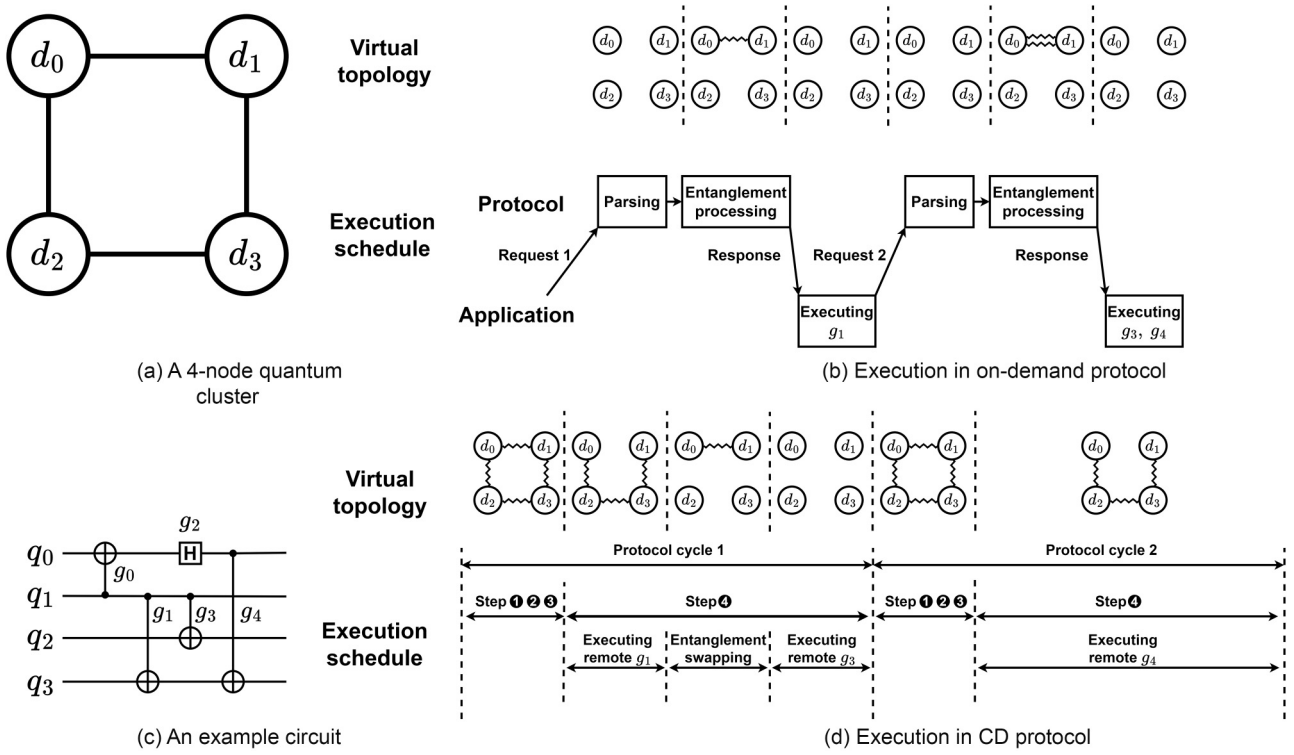


FIG. 3. An example of executing a quantum circuit in a cluster with on-demand/continuous distribution of entanglement. (a) The topology of a four-node quantum cluster, (b) a quantum circuit that parted into two segments (q_0 and q_1 in one node, q_2 and q_3 in another). We allocate q_0 and q_1 to node d_0, and q_2 and q_3 to node d_1. The symbols ❶, ❷, ❸, and ❹ represent the four steps of a CD protocol cycle: verification and generation of entangled states, entanglement swapping by protocol, updating the connectivity graph, and consumption of entangled states, respectively.

denote the cutoff time of all EPR pairs. When we share an EPR pair between d_i and d_j at time t and the lifetime of this link $\mathcal{L}(\mathbf{e}_{ij}, t) \leqslant T_{\text{cutoff}}$, then we consider $\mathbf{e}_{ij}$ as an available virtual link, and d_i and d_j are virtual neighbors. Thus the available virtual links of D at time t are described by virtual connectivity graph $\mathcal{G}_D^t = \{D, \mathcal{E}\}, \forall \mathbf{e}ij \in \mathcal{E}, \mathcal{L}(\mathbf{e}_{ij}, t) \leqslant T_{\text{cutoff}}$.

To share an EPR pair between node d_i and d_j, if e_{ij} does not exist, we must apply entanglement swapping operations in the intermediate nodes in the physical path from d_i to d_j, consuming E_c pregenerated entangled states, where E_c equals the number of intermediate nodes in this path.

B. CD protocols

A CD protocol $\mathcal{A}$ is designed to give a specific protocol for entanglement generation, swapping, entanglement consumption, and application of cutoffs. Entanglement generation and swapping determine when to generate entanglement between two physical neighbors and choose some nodes to run entanglement-swapping operations. Entanglement consumption for each pair of nodes that shares an entangled pair depends on the background application, like entanglement swapping and executing remote operations. For the application of cutoffs, the protocol will check whether the established entanglement has expired the cutoff time T_{cutoff} and will handle the expired entanglement accordingly, such as discarding or entanglement purification [32,33]. CD protocol execution results $\mathcal{A}(G, t)$ on D indicate all virtual connectivity links $\mathcal{E}$, which shows all the available virtual links of D at each time t, i.e., $\mathcal{A}(G, t) = \mathcal{G}_D^t$. Moreover, circuit execution will consume entangled states, which may change $\mathcal{G}_D^t$.

C. Quantum circuit

We consider an input quantum circuit $\mathcal{P} = \{U_1(i), U_2(i, j)\}$ with m qubits, where U_1, U_2 denote single-qubit gates and two-qubit gates (CNOT gates in this work), respectively, and i, j are the involved qubits of the gates. We can also transform this circuit $\mathcal{P}$ into a directed acyclic graph (DAG) $G_{\mathcal{P}}$. In $G_{\mathcal{P}}$, gates and the dependencies between gates are represented as nodes and edges in a DAG. The execution time of the circuit is equal to the depth of the longest path (critical path), denoted as $\Delta_{\mathcal{P}}$. When the input circuit cannot be executed on any device in the distributed quantum cluster D, i.e., $\max\{|d_i|\} < m$, the circuit must be partitioned into subcircuits $\mathcal{P}_{\text{cuts}} = \{P_0, P_1, \ldots, P_k\}$ with fewer qubits, and the number of qubits in subcircuit P_i is denoted as $|P_i|$. Intuitively, let $\sum_{i=1}^{k} |P_i| = m$.

To execute the subcircuits on D, we first map each subcircuit to a target device d (node in the cluster D). We have a mapping $\mathcal{M} = f : \mathcal{P}_{\text{cuts}} \to D$ during the execution process, and the location of each qubit i in $\mathcal{P}$ is denoted by $l(i)$. This means the circuit qubit i is mapped to a physical qubit of $l(i) \in D$. With a specific mapping, the operations of each subcircuit thus can be implemented by the target quantum device $l(i)$. The execution of each gate $U_2(i, j) \in U_2$ falls into two scenarios: either the involved qubits i and j are allocated on the same node, i.e., $l(i) = l(j)$, which we refer to as a local

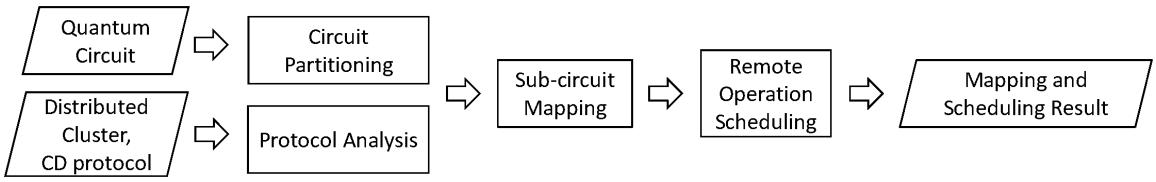


FIG. 4. Overall mapping and scheduling procedure.

operation, or i and j are on different nodes, i.e., $l(i) \neq l(j)$, requiring the use of telegate for execution, which we refer to as a remote operation $U_2^r(i, j)$. To simplify the model, we assume that all local operations can be performed directly, without being constrained by the internal topology of quantum nodes, and only telegate is used to execute remote operations, according to the irretrievable loss by directly transferring quantum information [7].

D. Time cost

Each time cycle consists of the following procedures: entanglement generation and swapping according to the protocol, along with unlimited local and currently executable remote operations when resources are sufficient. The overall scheduling result is denoted by $\mathcal{P}^s = \{\mathcal{P}_{\text{cuts}}, \mathcal{U}_2\}$. We use $\{\mathcal{M}, \mathcal{P}^s, D, \mathcal{A}\}$ to represent the mapping and scheduling results of $\mathcal{P}$ on D under the CD protocol $\mathcal{A}$. The execution time cost $\Delta_{\mathcal{P}^s}$ of the circuit $\mathcal{P}$ is defined as the minimum number of cycles required for all operations to be executed.

Based on the system model defined above, we then define the min-depth qubit mapping and scheduling problem in continuous distribution of entanglement:

Problem 1. Min-depth qubit mapping and scheduling problem in continuous distribution of entanglement.

Input: A quantum circuit $\mathcal{P}$ and a distributed quantum cluster D with topology G_D, a CD protocol $\mathcal{A}$.

Output: A scheduling result $\{\mathcal{M}, \mathcal{P}^s, D, \mathcal{A}\}$ where $\Delta_{\mathcal{P}^s}$ is minimized.

IV. SYSTEM DESIGN

In this section, we propose a qubit mapping and scheduling method for the distributed execution of quantum circuits within continuous entanglement distribution protocols to improve the execution efficiency of quantum circuits effectively. We divide the execution process into three steps: virtual network topology analysis, circuit partitioning and mapping, and remote operation scheduling, as illustrated in Fig. 4. Moreover, considering that the entanglement swapping strategy can significantly affect the distributed execution performance of quantum circuits, we propose the connectivity-first swap (CFS) scheme to achieve more efficiency for the CD protocol in circuit distributed execution.

A. Protocol analysis

As previously mentioned, the continuous distribution of entanglement (CD protocol) is more appropriately suited for the distributed execution of large-scale quantum circuits. However, constraints arising from hardware technologies continue to limit this protocol. The single random swap (SRS) protocol serves as a pertinent example, and the procedure is outlined as follows:

❶ Verification and Generation of Entangled States: All entangled states are assessed, with those that have timed out being discarded. Entanglement generation is executed on each physical link, provided the corresponding physical links are in an idle state.

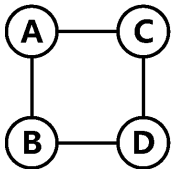
❷ Random Selection of Nodes for Entanglement Swapping: For node i, randomly select the qubit q_i^j (where q_i represents node i, and j denotes the corresponding entangled qubit on the communicated node j) that is entangled with node j (the qubit at node j being q_j^i), as well as the qubit q_i^k entangled with node k (with the qubit at node k being q_k^i). This entanglement swapping occurs in the absence of a physical link between nodes j and k with a probability of q_{swap}. If successful, entanglement is established between nodes j and k, such that q_j^i and q_k^i share the entangled state, while q_i^j and q_i^k become idle.

❸ Updating the Connectivity Graph: The virtual connectivity graph of the entire network and the state of all entangled states are updated, discarding those entangled links that exceed the maximum number of allowable entanglement swaps.

❹ Consumption of Entangled States: Entangled states are utilized in executing remote operations and for background applications.

During the distributed execution of large-scale quantum circuits, the network state significantly impacts circuit execution, encompassing the following considerations: (1) Even physically adjacent nodes cannot guarantee the continuous presence of entanglement, as this is dependent upon the number of communicable qubits (communication buffer size) and the success rates of both entanglement generation and swapping. (2) During the lifetime of successfully generated entangled states, corresponding communication qubits will be occupied, which limits the entanglement generation at the next time cycle.

Consequently, when analyzing the SRS protocol, it is crucial to investigate the relationship between physical and virtual links, as well as the impact of background applications on entanglement. Specifically, for a given physical topology G, CD protocol $\mathcal{A}$, and related parameters such as the probabilities of generating and swapping entangled states, a new virtual topology $\mathcal{G}$ should be established. The virtual topology of the entire cluster D at time t can be expressed as $\mathcal{A}(G, t) = \mathcal{G}_D^t$. During a continuous execution period, particularly during remote quantum operations across various nodes, analyzing the cluster's virtual topology is essential. As the selection and occurrence of entanglement swapping nodes are randomized in the protocol, and the consumption of entangled states for background applications is also variable, virtual topology



(a)

	A	B	C	D
A	0	0.43	0.43	0.39
B	0.43	0	0.38	0.41
C	0.43	0.38	0	0.42
D	0.39	0.41	0.42	0

(b)

FIG. 5. A 2×2 quantum cluster and its possible virtual adjacency matrix with sampling from CD protocol execution: (a) physical topology, (b) adjacency matrix of virtual topology.

information can be derived through random-walk algorithms or sampling methods. As illustrated in Fig. 5, the virtual connectivity of the cluster under one configuration of the SRS protocol can be represented by the adjacency matrix in Fig. 5(b). While the connectivity of the original graph [Fig. 5(a)] is limited, the virtual topology may be represented as a weighted complete graph, with weights indicating the normalized frequency of virtual topology existence between corresponding nodes throughout the sampling period. This enhancement is crucial for executing quantum circuits in distributed clusters. Additionally, we apply the adjacency matrix to the subsequent circuit mapping process to access the benefit of mapping a subcircuit to a node, achieving a better mapping result.

B. Circuit partitioning and mapping

In distributed quantum circuit partitioning, it is imperative to first consider the additional costs introduced by partitioning, such as the number of edges between subcircuits. Quantum circuits can be depicted as interaction graphs, where nodes represent qubits, edges signify couplings between qubits (specifically the CNOT gates referenced in this study), and edge weights denote the number of gates. Consequently, the process of quantum circuit partitioning can be treated as dividing an undirected weighted graph into n subgraphs, where the size of each subgraph does not exceed the number of qubits available on each node, a problem recognized as NP-hard [34]. Previous research has explored transforming this challenge into graph partitioning problems. For example, Andres *et al.* [18] approach the issue as hypergraph partitioning, utilizing third-party solvers for resolution. Similarly, Baker *et al.* [19] regard each circuit layer as an independent graph based on interaction graphs, applying a foresight strategy to adjust edge weights and ultimately employing an overall extreme exchange (OEE) method, which encompasses both static and relaxed strategies. Wu *et al.* [20,21] implement the static OEE method introduced by Baker *et al.* These methods partition circuits based on the preliminary characteristics of hardware devices, such as the number of nodes and qubits; however, they largely overlook the mapping process and do not adequately address the features of the entanglement distribution protocol during circuit execution. We adopt the Metis [35] method, an iterative graph partitioner to generate subcircuits. To effectively map the subcircuits to nodes with communication dictated by the CD protocol, we leverage the findings from Sec. IV A and reformulate the mapping problem as a bipartite matching problem solvable via the

ALGORITHM 1. Circuit partitioning and mapping.

Input: A quantum circuit $\mathcal{P}$, virtual entanglement matrix A_{v} for a distributed cluster D, the number of nodes $|D| = n$.
Output: A mapping $\mathcal{M} = f : \mathcal{P}_{\mathrm{cuts}} \to D$.
$\mathcal{P}_{\mathrm{cuts}}, A_{\mathrm{p}} = \mathrm{CircuitPartitioning}(\mathcal{P}, n)$;
$\mathrm{max\_value} = \mathrm{AssessCommRequirement}(\mathcal{P}_{\mathrm{cuts}}, A_{\mathrm{v}})$;
$\mathcal{C}_{\mathrm{s}}, \mathcal{B}_d = [\,]$;
$\mathcal{C} = \mathrm{matrix}(n, n)$;
for $i = 1$ **to** n **do**
  $\mathcal{C}_{\mathrm{s}}[i] = \sum_{j=0}^{n} A_{\mathrm{p}}[i][j]$;
  $\mathcal{B}_d[i] = \sum_{j=0}^{n} A_{\mathrm{v}}[i][j]$;
$\mathcal{C}_{\mathrm{s}} = \mathcal{C}_{\mathrm{s}}/\mathrm{max\_value}$;
for $i = 1$ **to** n **do**
  for $j = 1$ **to** n **do**
    $\mathcal{C}[i][j] = \max\{0, \mathcal{C}_{\mathrm{s}}[i] - \mathcal{B}_d[j]\}$;
$\mathcal{M} = \mathrm{Jonker\_Volgenant\_assignment}(\mathcal{C})$;
return $\mathcal{M}$.

Jonker-Volgenant algorithm [36]. We utilize the sum of the weights of edges between node i and its neighbors to determine both the communication demands ($\mathcal{C}_{\mathrm{s}}[i] = \sum_{j=0}^{n} A_{\mathrm{p}}[i][j]$) and the capabilities of the nodes in the cluster ($\mathcal{B}_d[i] = \sum_{j=0}^{n} A_{\mathrm{v}}[i][j]$), subsequently designing a cost matrix $\mathcal{C}$ to estimate the cost of mapping a subcircuit to a node.

The virtual topology among distributed nodes reflects the likelihood of entanglement between nodes at any moment, which is closely associated with the bandwidth between nodes in the cluster and relevant protocol parameters. To assess the communication demands between subcircuits at each moment, we estimate the total execution time of the original circuit on the cluster by calculating the total communication demand multiplied by the average distance between nodes in the graph, divided by the bandwidth. Thus, the cost of mapping subcircuit i to node j is calculated as $\max\{0, \mathcal{C}_{\mathrm{s}}[i] - \mathcal{B}_d[j]\}$. This methodology enables us to ascertain the communication demand for each subcircuit at any given moment. Hence, by employing a modified Jonker-Volgenant algorithm without initialization from [37], we achieve minimized cost matching for subcircuits to the nodes in the cluster. The specific algorithm is delineated in Algorithm 1, with the following steps:

(1) Partition the circuit uniformly to the nodes, utilizing the Metis [35] method in our approach.

(2) Calculate the connectivity requirements for each subcircuit alongside the virtual link resources of each distributed node.

(3) Define the cost of mapping subcircuit i to node j using the function $\mathcal{C}[i][j] = \max\{0, \mathcal{C}_{\mathrm{s}}[i] - \mathcal{B}_d[j]\}$.

(4) Compute the mapping $\mathcal{M}$ that minimizes the total cost by the Jonker-Volgenant algorithm.

Additionally, we employ straightforward methods, such as matching the entanglement requirements $\mathcal{C}_{\mathrm{s}}[i]$ of the previously calculated quantum circuit with the communication capabilities provided by the node $\mathcal{B}_d[j]$ in a sequential manner. We prioritize placing subcircuits with high entanglement requirements on nodes with robust communication capabilities, a strategy we refer to as the greedy method. This approach

ALGORITHM 2. Priority-based gate scheduling.

Input: The sub-circuits $\mathcal{P}_{\text{cuts}}$ of original quantum circuit $\mathcal{P}$, the mapping from sub-circuits to nodes $\mathcal{M}$, the parameters of entanglement distribution protocol $\mathcal{A}$, the topology graph of the distributed cluster G_D.
Output: The execution schedule of input quantum circuit on D.
$t = 0$;
Initialize $\mathcal{G}_D^t$, $\mathcal{P}'$;
$\mathcal{P}^{\text{s}} = \emptyset$;
while $\mathcal{P}'$ is not $\emptyset$ **do**
  F = get_front_layer($\mathcal{P}'$);
  if check_executed(F) **then**
    for gate g in F **do**
      priority[g] = get_priority($\mathcal{G}_D^t, F, \mathcal{P}'$);
    Execute $g_{\max}$ with maximum priority;
    $\mathcal{P}^{\text{s}}$.append($g_{\max}$, consumed_links);
    $\mathcal{G}_D^t$.update(consumed_links);
    $\mathcal{P}'$.remove($g_{\max}$);
  else
    $t = t + 1$;
    $\mathcal{G}_D^t = \mathcal{A}(\mathcal{G}_D^{t-1}, G_D, \text{parameters})$;
return $\mathcal{P}^{\text{s}}$

also demonstrates strong performance in experiments, leading us to combine these two methods in further testing.

C. Remote operation scheduling

In distributed quantum computing scenarios, where the execution time of local operations is significantly shorter than that of remote operations, the key to efficient quantum circuit execution lies in optimizing the scheduling sequence and methodology of remote operations while utilizing limited network entanglement resources at each temporal instance. Existing studies [10,20–22] demonstrate that, when disregarding entanglement distribution protocols, remote operations can be categorized based on the physical link availability between nodes. For physically connected nodes i and j, entanglement establishment requests can be directly initiated to execute remote operations, with the execution sequence exhibiting no impact on total entanglement resource consumption and vice versa.

Under the CD protocol framework, however, entanglement generation and exchange are strictly protocol-regulated, permitting only entanglement usage requests (distinct from entanglement establishment requests) to be submitted to nodes. Crucially, the scheduling strategy for limited entanglement resources substantially influences the efficiency of distributed quantum circuit execution. Considering the inherent information loss risks in quantum state transmission, this study adopts a constrained approach where computational qubits and associated information remain fixed at designated nodes, explicitly prohibiting qubit transfer via quantum teleportation. We propose a Priority-Based Distributed Gate Scheduling algorithm that determines operation priorities through a comprehensive evaluation of execution costs and their temporal impacts on overall circuit execution. The pseudocode (see Algorithm 2) implements the following workflow:

(1) *Initialization*: Set the timestamp $t = 0$, construct the cluster quantum network $\mathcal{G}_D^t$, and derive the remote-operation-only circuit $\mathcal{P}'$ by removing all local operations from $\mathcal{P}$.

(2) *Candidate Set Generation*: While $\mathcal{P}' \neq \emptyset$, collect executable remote operations F with all predecessor gates completed.

(3) *Priority Scheduling*: If there exist gates in F that can be executed with current network $\mathcal{G}_D^t$: Compute priority[g] for each $g \in F$, execute $g_{\max}$ with largest priority, and record its execution scheme in $\mathcal{P}^{\text{s}}$. Then remove $g_{\max}$ from $\mathcal{P}'$ and consuming corresponding entangled states. If no gate can be executed with $\mathcal{G}_D^t$, increment timestamp $t = t + 1$ and update network.

(4) *Iteration*: Repeat steps 2 and 3 until $\mathcal{P}' = \emptyset$, then return the circuit execution schedule $\mathcal{P}^{\text{s}}$.

The priority computation model addresses dual optimization objectives: minimizing total execution time and entanglement resource consumption. Given the NP-hard nature of precise impact assessment, for gate g whose two involved qubits are located at node d_i and d_j, we formulate three parametric approximations:

(1) *Direct Gain* (for directly entangled nodes):

$$h_{\text{direct}} = \frac{\mathcal{G}_D^t[d_i][d_j]}{\max \mathcal{G}_D^t}. \tag{1}$$

(2) *Indirect Gain* (for indirectly entangled nodes):

$$h_{\text{indirect}} = \frac{\min_{\text{bw}} \text{path}(d_i, d_j)}{\max \mathcal{G}_D^t} - \frac{\min_{\text{length}} \text{path}(d_i, d_j)}{\max \min \text{path}(F)}. \tag{2}$$

(3) *Global Impact*:

$$h_{\text{global}} = \frac{\sum_{f \in \mathcal{F}, f != g} \text{exe}(f)}{|F|} + \frac{\text{num\_successors}(g)}{|\text{rest\_gate}|}. \tag{3}$$

Equation (1) describes the influence of directly executing gate g, $\max \mathcal{G}_D^t$ represents the maximum value of the bandwidth of shared entangled pairs between nodes in the virtual topology of the graph, and $\mathcal{G}_D^t[d_i][d_j]]$ represents the entanglement bandwidth between d_i and d_j. Similarly, Eq. (2) describes the impact of execution when d_i and d_j do not directly share an entangled pair: the first term $\min_{\text{bw}} \text{path}(d_i, d_j)$ represents the ratio of the minimum bandwidth on the entanglement path between d_i and d_j to the maximum virtual bandwidth in the network, and the second term represents the loss caused by the length of the entanglement path, where $\min_{\text{length}} \text{path}(d_i, d_j)$ represents the minimum length of the entanglement path between d_i and d_j, $\max \min \text{path}(F)$ represents the maximum value of the minimum length of the entanglement path in the current gate F to be executed, and h_{indirect} is the difference between these two terms. The larger the value, the lower the loss of executing gate g. Equation (3) represents the importance of gate g in the entire execution process: the first item is the impact of executing gate g on other gates in F, $\sum_{f \in \mathcal{F}, f != g} \text{exe}(f)$ represents how many other gates in F can continue to execute after executing g, the second item is the importance of gate g in the circuit, num_successors(g) represents the number of subsequent gates of gate g, |rest_gate| represents the number

of gates left in the circuit, and the larger h_{global} is, the more important the execution of gate g is. We use the parametrized linear combination of these three parameters in Eq. (4) to represent the priority of executing gate g. The higher $h(g)$ means the greater the time benefit brought by executing $h(g)$ to the execution of the entire circuit, and the lower the cost required,

$$h(g) = w_1 h_{\text{direct}} + w_2 h_{\text{indirect}} + w_3 h_{\text{global}}. \tag{4}$$

D. Connectivity-first swapping protocol

The continuous entanglement distribution protocol encompasses a crucial component known as the entanglement SWAP scheme, which fundamentally dictates the evolution of the network's virtual topology $\mathcal{G}_D^t$ and its stabilization A_{v} (as characterized by the previously sampled virtual topology adjacency matrix). Key factors influencing the efficiency of quantum circuit execution—such as total execution time, resource consumption, and overall success rate—are directly related to the network's virtual topology. Consider a scenario in which each remote quantum operation executed in the circuit involves two quantum nodes that directly share entangled states. In such cases, the time required to execute the circuit within the network is significantly reduced; assuming local operation times are negligible, the entire quantum circuit could be completed with a single entanglement generation period. The previously mentioned single random swap (SRS) protocol [12] assesses the entanglement status of each node and performs entanglement swapping in each cycle. For instance, if node d_i shares entanglement with nodes d_j, d_k, and d_l at time t, according to the SRS protocol, the node randomly selects two of these nodes to execute the entanglement swap, provided that no physical link exists between them. The primary advantage of this single random swapping scheme lies in its ability to maintain high execution efficiency without necessitating updates to the overall virtual topology of the cluster. However, the protocol may perform suboptimally during circuit execution due to the lack of a clear objective in entanglement swapping; random entanglement swaps may increase the costs of remote operations with longer entanglement paths or even render execution unfeasible if the selected nodes are unconnected. This limitation hinders the protocol's adaptability to more complex situations in distributed circuit execution. Consequently, we propose the connectivity-first swapping (CFS) protocol for entanglement distribution and swapping, which prioritizes the connectivity of the cluster's virtual topology, as described in Algorithm 3. This protocol operates as follows:

(1) *Cutoff Verification*: The protocol begins with a cutoff check to determine if any entangled links should be removed based on the threshold T_{cutoff}.

(2) *Entanglement Generation*: After the cutoff process, entanglement generation is attempted on all physical links in the network with a success probability p_{gen}.

(3) *Swapping Process*: For each node i in the quantum network, (a) identify all possible swap actions SWAPS that can be performed by the node, (b) assess the potential benefits of each swap action concerning promoting the virtual adjacency S, and (c) perform the optimal swap $s_{\max}$, where $s_{\max}$ represents the most beneficial identified swap.

ALGORITHM 3. Connectivity-first entanglement swapping.

Input:
(1) Quantum network with an configuration of entangled links $\mathcal{G}_D^t$ at time t and physical topology G_D;
(2) probability of successful entanglement generation p_{gen};
(3) probability of successful swap p_{swap};
(4) probability of link consumption p_{cons}.

Output: Quantum network $\mathcal{G}_D^{t+1}$ with updated configuration of links.

```
S = G_D^t;
check_cutoffs(S, T_cutoff);
generate_all_physical_links(G_D, p_gen);
for node i in all nodes do
    SWAPS = possible_swaps(i);
    for swap s in SWAPS do
        benifit[s] = assess_benifits(S, s);
    S = SWAP(S, s_max);
G_D^{t+1} = entanglement_consumption(S, p_cons);
return G_D^{t+1}.
```

(4) *Entanglement Consumption*: Finally, each node engages in entanglement consumption, wherein entangled links are consumed with probability p_{cons}.

Here, in order to implement the connectivity-first entanglement swapping strategy, we use the maximum distance estimation scheme to calculate the benefit of performing entanglement swapping, i.e., the difference between the sum of the distances between nodes before and after the entanglement swapping. Let the cluster virtual topology before the entanglement exchange be S, and after the exchange let it be S'. The benefit is expressed as

$$\sum_{j>i}^{i,j} \text{dist}(S, i, j) - \sum_{j>i}^{i,j} \text{dist}(S', i, j),$$

where $\text{dist}(S, i, j)$ denotes the length of shortest entanglement path between d_i and d_j under the configuration of entangled links S. This protocol enhances the efficiency of entanglement generation and link utilization while ensuring a structured approach to swapping operations within the quantum network.

V. EVALUATION

We evaluate the performance of our methods on benchmark circuits [38], a widely used benchmark for the NISQ-era, and circuits generated by IBM Qiskit [39]. We compare our methods with baseline approaches in terms of circuit execution time and entanglement cost across different topologies and bandwidth clusters. The details of the experiment results are shown in Sec. V B. We highlight our key findings as follows:

(i) Our mapping and scheduling scheme delivers outstanding performance under the single random swap (SRS) entanglement distribution protocol, achieving execution time reductions of 52.4%, 49.9%, and 53.1% on a 3×3 grid cluster

TABLE II. Benchmark circuits information with different partitioning methods. n, d, g denote the number of qubits, the circuit depth, and the number of gates. N_{Trivial}, N_{OEE}, and N_{Metis} means the number of remote operations after partitioning the circuits using trivial partitioning (based on the order of qubits), OEE [40], and Metis [35], respectively.

Benchmark name	n	d	g	N_{Trivial}	N_{OEE}	N_{Metis}
knn_n129	129	518	1218	484	240	224
knn_n67	67	270	629	250	120	128
multiplier_n45	45	2397	5981	1118	1033	1019
multiplier_n75	75	6771	17077	2340	2167	2029
qaoa_n100	100	206	2381	1294	1040	1032
qaoa_n200	200	419	8926	5068	4308	4264
qaoa_n300	300	689	20838	12086	10548	10552
qft_n100	100	199	5099	4464	4416	4440
qft_n200	200	399	20199	17848	17760	17756
qft_n300	300	599	45299	40120	39984	39976
qft_n29	29	221	2059	728	728	734
qft_n63	63	493	9828	3528	3528	3524
qugan_n111	111	508	2235	448	280	256
qugan_n71	71	328	1415	298	298	192
rca_n100	100	1178	1667	515	88	88
rca_n200	200	2378	3367	1015	198	88
rca_n300	300	3578	5067	1515	286	88
swap_test_n115	115	462	1085	432	216	204
swap_test_n83	83	334	781	310	148	152
vqe_n100	100	199	5150	4416	4416	4436
vqe_n200	200	399	20300	17756	17756	17755
vqe_n300	300	599	45450	39984	39984	39975

with average bandwidths of 1, 3, and 5, respectively, compared to the baseline method.

(ii) The connectivity-first swap (CFS) protocol further enhances efficiency, cutting execution times by 24.5%, 19.7%, and 17.8% compared with the SRS protocol for average bandwidths of 1, 3, and 5, respectively.

(iii) Our approach significantly lowers entanglement consumption. The CFS protocol can further decrease consumption, thereby increasing the entanglement resources for other applications.

(iv) Our strategy demonstrates both stability and scalability, resulting in notable performance improvements as network connectivity and entanglement resources increased.

A. Experimental setup

1. Benchmark circuits

We utilize quantum circuits from QASMbench [38], as well as QAOA, VQE, RCA, and QFT circuits generated with 100, 200, and 300 qubits by IBM Qiskit [39]. These circuits are transformed into only single-qubit and CNOT gates. The details of the circuits are presented in Table II, which includes the number of qubits, circuit depth, number of gates, and the number of remote operations after partitioning the circuits using trivial partitioning (based on the order of qubits), OEE [40], and Metis [35]. To achieve load balancing in distributed quantum computing, we do not impose a maximum qubit limit for each node in the cluster. Consequently, we uniformly partition the circuits into subcircuits (see Table III).

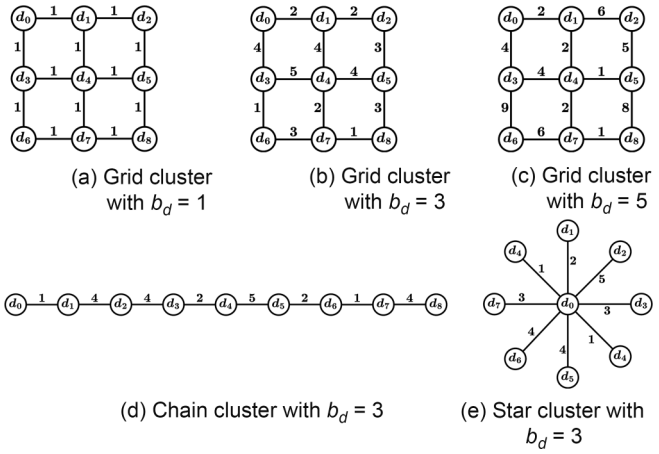


FIG. 6. The distributed clusters used in evaluation.

2. Cluster topology and CD protocol configuration

We evaluate our mapping and scheduling algorithm on three 3×3 grid distributed clusters, with average physical entanglement bandwidths between nodes b_d of 1, 3, and 5, respectively (randomly sampled from a normal distribution). The bandwidth means the average number of physical links between two adjacent nodes, that is, the maximum number of entangled pairs that can be distributed at once when the corresponding channel resources are completely idle with CD protocols. The physical topology graphs of these three clusters are shown in Figs. 6(a), 6(b), and 6(c). In our evaluation, the entanglement distribution protocols employed are SRS (Sec. IV A) and CFS (Sec. IV D). We define the main parameters in the protocol as follows: the probability of successful entanglement generation p_{gen}, the probability of successful entanglement swap p_{swap}, and the probability that two nodes consume shared links p_{cons} in each time slot are set to 1, 0.95, and 0.05, respectively. Additionally, the probability of performing swaps q_{swap} according to the SRS protocol is set to 0.12. For sensitivity analysis of our algorithm on clusters with different physical topologies, we further deploy clusters with chain and star topologies with $b_d = 3$, as shown in Figs. 6(d) and 6(e).

Evaluation Platform. Our experiments are performed on Intel Xeon Platinum 8360Y CPU@2.40GHz, with 1024GB DDR4 memory and Ubuntu 22.04.

Baseline. Given the limited prior research on distributed quantum circuit execution with CD protocol, we propose a baseline method that integrates OEE circuit partitioning [40], our subcircuit mapping algorithm, and topological order gate scheduling (where all remote gates are scheduled according to the topological order of the original quantum circuits). All entanglement swapping paths required for internode gate execution between nodes (d_i and d_j) are optimized using the shortest path. For all experiments, we utilize the greedy method and the Jonker-Volgenant algorithm (introduced in Sec. IV B) with the clustering coefficient of subcircuit communication matrix A_{p} (a measure of the degree to which nodes

TABLE III. Comparison of total distributed circuit execution methods.

	Baseline			Ours with SRS[a]			Ours with CFS[a]		
Benchmark name	$b_d = 1$[b]	$b_d = 3$	$b_d = 5$	$b_d = 1$	$b_d = 3$	$b_d = 5$	$b_d = 1$	$b_d = 3$	$b_d = 5$
knn_n129	208.3	77	70.35	7.3%	15.2%	25.1%	44.8%	34.4%	31.2%
knn_n67	103.8	37.1	35.25	–7.2%	0.5%	16.0%	38.9%	21.3%	24.3%
multiplier_n45	1001.7	425.05	376.5	59.7%	59.0%	57.6%	67.8%	64.6%	67.7%
multiplier_n75	2086.85	829.35	732.5	50.2%	45.2%	56.4%	62.5%	55.8%	63.0%
qaoa_n100	1066	369.05	331.95	64.1%	55.7%	53.7%	72.9%	63.8%	64.7%
qaoa_n200	4696.3	1611.5	1459.8	70.3%	59.2%	56.4%	75.4%	67.8%	69.1%
qaoa_n300	11585.65	4113.2	3703.05	71.6%	60.2%	58.4%	75.8%	67.8%	69.4%
qft_n100	4827.45	1844.25	1675.2	67.6%	60.1%	57.9%	72.2%	67.2%	66.0%
qft_n200	19815.05	8035.2	7165.95	67.7%	58.9%	62.0%	72.7%	68.7%	69.6%
qft_n300	45161.3	18806.65	16688.8	68.2%	62.3%	62.6%	74.5%	70.6%	73.6%
qft_n29	746.65	263.95	241.5	53.8%	45.9%	50.8%	67.3%	55.6%	60.1%
qft_n63	4000.9	1547.55	1346.2	63.9%	55.4%	53.8%	74.3%	65.0%	69.5%
qugan_n111	263.4	111.05	98.15	35.3%	37.6%	44.2%	55.9%	49.7%	50.8%
qugan_n71	276.6	109.65	94.15	58.2%	56.9%	57.2%	70.9%	64.7%	63.8%
rca_n100	88.4	39	36.8	67.4%	56.8%	58.0%	67.3%	62.7%	57.3%
rca_n200	221.1	96.6	86.05	82.5%	88.5%	82.6%	86.7%	89.2%	86.8%
rca_n300	316.35	138.2	124.25	89.8%	85.5%	90.7%	90.2%	91.3%	90.9%
swap_test_n115	178.55	66.55	62.05	–1.6%	20.5%	24.3%	42.8%	34.3%	28.2%
swap_test_n83	130.7	44.4	42.75	–3.2%	4.2%	20.5%	41.0%	22.1%	26.0%
vqe_n100	4808.95	1801.9	1579.4	58.3%	50.8%	55.5%	69.8%	62.2%	64.2%
vqe_n200	19886.45	8050	6871.2	63.7%	56.8%	60.9%	71.6%	64.0%	70.9%
vqe_n300	45180.4	18880.05	16286.9	66.6%	63.2%	63.4%	74.0%	71.8%	72.7%

[a]Our results are presented as an optimization ratio, which is calculated by Eq. (5); a larger value means more optimizations.
[b]b_d means average bandwidth of the distributed cluster; the topologies are shown in Figs. 6(a), 6(b), and 6(c), respectively.

in a graph tend to cluster together): using the greedy method with a clustering coefficient more than 0.9, otherwise the Jonker-Volgenant algorithm. Empirically, we set the weighted parameters of direct gain w_1, indirect gain w_2, and global impact w_3 as 0.3, 0.2, and 0.5. All results are obtained by the average simulation results of 20 repeated samplings.

Metrics. As discussed in Sec. III, we focus on the execution depth of the quantum circuit, i.e., total execution time cost (cycles) of all remote operations for our main evaluation. We set the duration of a single CD protocol operation to 1, meaning that the time for local operations, entanglement swapping, and remote operations is omitted. To better present the results, we define the optimization ratio as follows:

$$\gamma = \frac{\text{baseline result} - \text{current method result}}{\text{baseline result}}, \tag{5}$$

where a positive value indicates a better result than the baseline, and vice versa.

B. Experimental results

Following the configurations outlined above, we first present the overall result of our evaluations. Then we analyze the performance of our mapping and scheduling algorithms concerning entanglement consumption, along with the differences in entanglement consumption between the SRS and CFS protocols. Then, we examine the impact of various circuit partitioning schemes on the circuit execution time cost, specifically focusing on the effects of the trivial, OEE [40], and Metis [35] methods. Furthermore, we analyze the impact of key parameters and cluster topology on execution time cost, including p_{swap} (the probability of successful entanglement swap) and p_{cons} (the probability that nodes consume entangled states) across different average bandwidths, along with the chain, star, and grid physical topology of the cluster.

1. Overall result

Our mapping and scheduling scheme demonstrates exceptional performance under the single random swap (SRS) entanglement distribution protocol, achieving average reductions in execution time cost of 52.4%, 49.9%, and 53.1% on a 3×3 grid cluster with average bandwidths of 1, 3, and 5, respectively, when compared to the baseline method, as shown in Table III. Our protocol, connectivity-first swap (CFS), offers additional performance improvements, reducing execution time by 24.5%, 19.7%, and 17.8% for average bandwidths of $b_d = 1, 3,$ and5, respectively, compared with the SRS protocol. Our mapping and scheduling method also significantly decreases entanglement cost. For instance, with an average bandwidth $b_d = 3$, our method reduces entanglement consumption by an average of 20.3%. The CFS protocol further decreases entanglement cost E_c by 11.5% compared to the SRS protocol, thereby providing more abundant entanglement resources for other applications within the network. Our method demonstrates both stability and scalability. As network connectivity improves and entanglement resources increase, both our method and protocol yield significant performance enhancements.

2. The consumption of entanglement

In Table IV, we illustrate the impact of our method under the SRS and CFS protocol on entanglement resource

TABLE IV. Comparison of entanglement cost results.

Benchmark name	Baseline		Ours with SRS		Ours with CFS	
	E_c	E_d	γ_{E_c}	γ_{E_d}	γ_{E_c}	γ_{E_d}
knn_n129	499.35	30.4	11.7%	–41.0%	13.2%	17.8%
knn_n67	249.55	8.6	1.1%	–169.8%	1.3%	–47.7%
multiplier_n45	2421.05	261.7	19.1%	84.0%	25.2%	80.1%
multiplier_n75	5055.2	530.95	10.7%	66.4%	22.1%	69.1%
qaoa_n100	2429.7	102.15	9.9%	87.2%	22.2%	57.1%
qaoa_n200	10287.45	595.6	13.6%	88.4%	26.3%	70.2%
qaoa_n300	25368.05	1844.3	12.6%	90.7%	25.8%	74.6%
qft_n100	11146.1	1085.75	14.7%	89.2%	27.2%	79.3%
qft_n200	45922.2	6329.15	15.7%	84.0%	29.6%	83.9%
qft_n300	103781.6	17130.3	16.6%	88.9%	30.8%	86.9%
qft_n29	1888.7	67.15	15.1%	41.0%	26.7%	42.7%
qft_n63	9346.95	973.1	17.0%	76.6%	29.5%	76.9%
qugan_n111	630.25	74.1	19.5%	17.1%	24.4%	48.8%
qugan_n71	681.85	46.2	43.2%	34.2%	46.8%	61.4%
rca_n100	245.5	12.4	10.2%	83.1%	31.5%	62.5%
rca_n200	563.2	70.55	76.1%	98.2%	76.3%	95.6%
rca_n300	817.55	109.05	75.0%	95.6%	81.9%	96.5%
swap_test_n115	451.6	22.35	10.4%	11.0%	12.6%	31.8%
swap_test_n83	310.55	12.55	4.8%	–115.9%	5.7%	–36.3%
vqe_n100	11504.8	875.8	15.6%	63.5%	27.4%	64.5%
vqe_n200	46378.2	6736.55	18.2%	80.3%	29.2%	80.5%
vqe_n300	104393.65	18417.7	17.1%	91.5%	31.4%	88.6%

utilization—specifically with a grid topology cluster whose average bandwidth $b_d = 3$. We focus on the requesting of entanglement swaps and the use of entanglement for remote operations (we add these two as E_c)—as well as the number of entangled states discarded E_d due to timeouts in the network (entangled states whose lifetime are more than T_{cutoff}). The experimental results demonstrate that our method significantly reduces entanglement resource utilization compared to the baseline method, achieving reductions of up to 76.1% and an average reduction of 20.3%. Furthermore, the CFS protocol offers a network entanglement topology that is better suited for circuit execution, resulting in an additional average reduction of 11.6% in entanglement resource utilization compared to the SRS protocol. Additionally, in most cases, the number of entangled states discarded due to timeouts is significantly reduced, with an average decrease of 47.5%, thereby allowing more resources to be available for other applications within the network.

3. Impact of partitioning methods

We present the results of three circuit partitioning methods in Fig. 7: Trivial (partitioned by the order of qubits), OEE [40], and Metis [35]. These methods are evaluated alongside the same mapping and scheduling approach proposed in this paper under average bandwidths of b_d of 1, 3, and 5. As shown in Table II, the Metis partitioning method outperforms the other two methods in most cases and demonstrates strong performance during actual execution. For the multiplier_n75 circuit, the differences in performance arise from the frequent operations in the original circuit, where the partitioning methods yield similar numbers of remote operations across varying bandwidth conditions. This observation, in conjunction with Table III, further underscores the effectiveness of our proposed mapping and scheduling scheme.

4. Impact of CD protocol parameters

The impact of $\boldsymbol{p}_{\text{swap}}$ *and* $\boldsymbol{p}_{\text{cons}}$. In Fig. 8, we illustrate the impact of the entanglement swap success rate p_{swap} and the entanglement consumption rate p_{cons} on circuit execution time cost within the cluster across different average bandwidths shown in Figs. 6(a), 6(b), and 6(c). Overall, as both p_{swap} and the average bandwidth of the cluster b_d increase, the time cost of circuit execution under both the SRS and CFS protocols exhibits a decreasing trend. However, as p_{cons} increases, circuits require more time to complete execution. Notably, when the average bandwidth of the cluster $b_d = 3$ and the entanglement swap success rate $p_{\text{swap}} = 0.5$, our method under the CFS protocol does not perform better than under the SRS protocol, as shown in Fig. 8(b). This occurs because the CFS protocol aims to achieve high connectivity in the network's virtual topology, which reduces the available entanglement when the entanglement swap success rate p_{swap} is low. Similarly, a high entanglement consumption rate p_{cons} results in a comparable situation.

The impact of cluster topology. In Fig. 9, we present the performance of circuit execution across three topologies with an average bandwidth $b_d = 3$: chain, star, and grid, as shown in Figs. 6(d), 6(e), and 6(b). It is evident that physical connectivity is critical for circuit execution; as connectivity improves, the time required for execution significantly decreases. This observation underscores the importance of the CFS protocol: when enhancing physical topology is challenging,

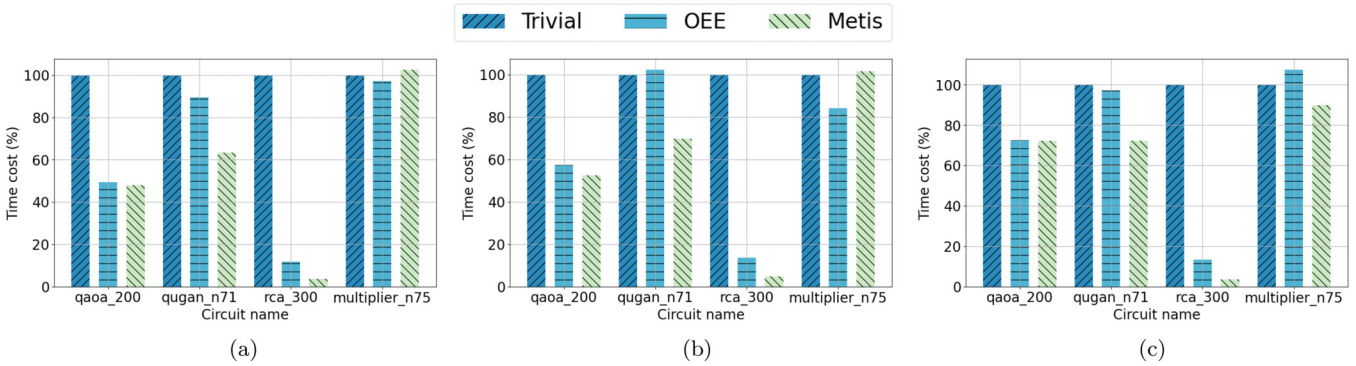


FIG. 7. Comparison of the circuit partitioning method of trivial (partitioned by the order of qubits), OEE, and ours on the time cost of circuit execution, with the same mapping and scheduling method on the cluster with average bandwidth $b_d = 1, 3,$ and 5. We take the result of the trivial partitioning method as 100%. (a) On cluster with average bandwidth 1 [shown in Fig. 6(a)]. (b) On cluster with average bandwidth 3 [shown in Fig. 6(b)]. (c) On cluster with average bandwidth 5 [shown in Fig. 6(c)].

entanglement distribution protocols that depend on cluster virtual connectivity as a metric become essential.

VI. RELATED WORK

The software and hardware of quantum network infrastructure have experienced rapid development in recent years. In entanglement distribution, numerous studies have focused on optimizing entanglement routing paths [11,28–31], implementing hardware-based entanglement distribution protocols using repeaters or quantum memories [41–48]. These advances have led to foreseeable performance improvements for running quantum programs in both on-demand distribution of entanglement (on-demand protocols) and continuous distribution of entanglement (CD protocols). Delle *et al.* further proposed a quantum network operating system to maximize the utilization of network hardware [49].

Executing quantum algorithms on actual quantum machines requires transforming and optimizing quantum circuits generated by algorithms, which involves qubit mapping and routing (with a distributed scenario in this paper, we call this qubit mapping and scheduling). Quite a lot of work [50–59] has been focused on qubit mapping and routing within single quantum machine scenarios, also known as quantum circuit compilation or qubit allocation. These efforts target hardware architectures like superconducting quantum computers, aim-

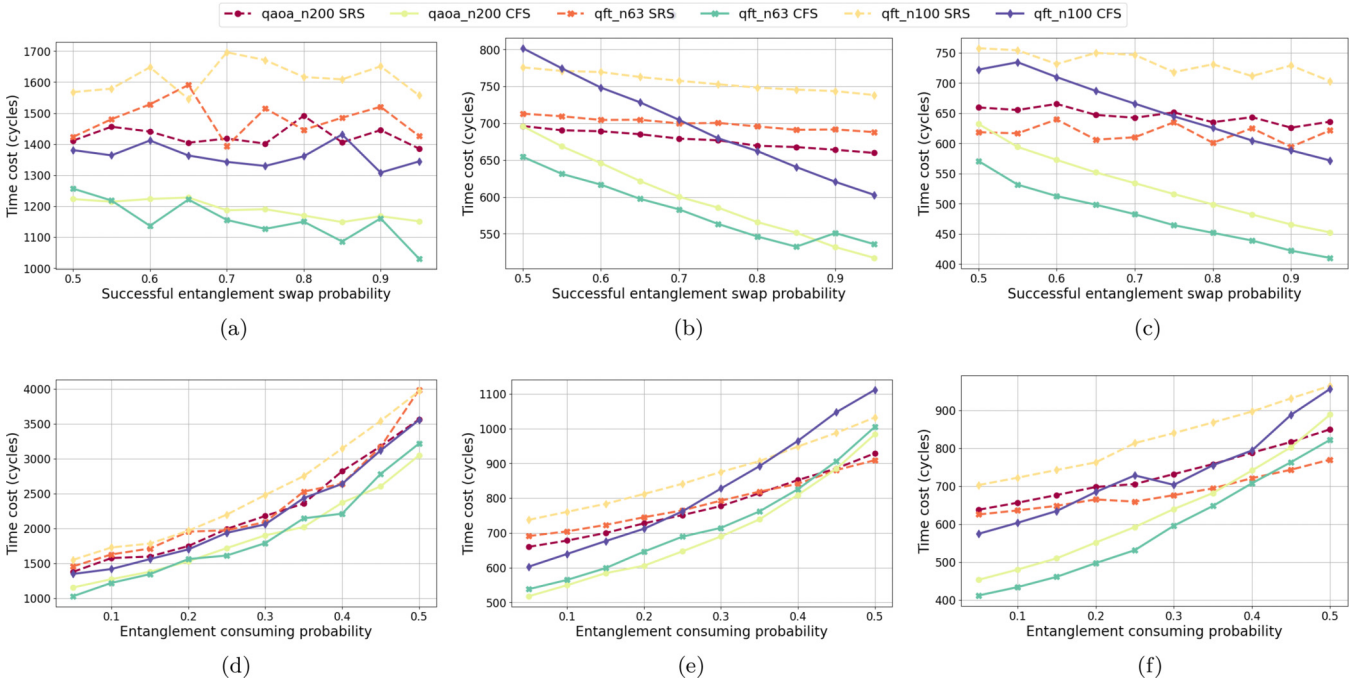


FIG. 8. Impact of p_{swap} (the probability of successful entanglement swap, from 0.5 to 0.95, step by 0.05), p_{cons} (the probability that nodes consuming entangled states, from 0.05 to 0.5, step by 0.05) on circuit execution time cost with different average bandwidth on two CD protocols. (a), (b), and (c) With SRS protocol; (d), (e), and (f) with CFS protocol. (a) Impact of p_{swap} on cluster in Fig. 6(a). (b) Impact of p_{swap} on cluster in Fig. 6(b). (c) Impact of p_{swap} on cluster in Fig. 6(c). (d) Impact of p_{cons} on cluster in Fig. 6(a). (e) Impact of p_{cons} on cluster in Fig. 6(b). (f) Impact of p_{cons} on cluster in Fig. 6(c).

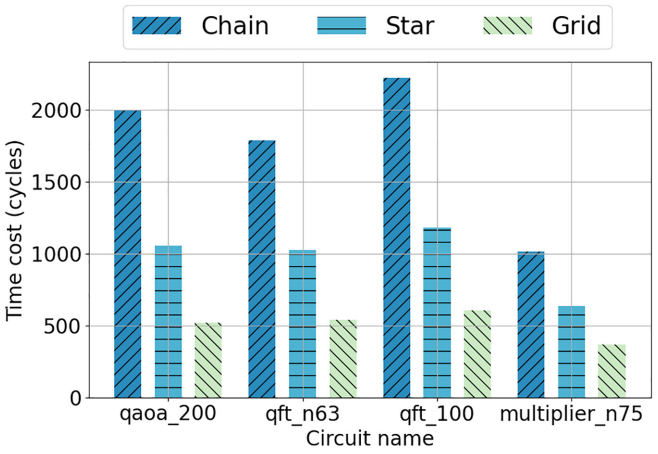


FIG. 9. Impact of topology of cluster, chain [Fig. 6(d)], star [Fig. 6(e)], and grid [Fig. 6(b)], on circuit execution time cost with average bandwidth $bd = 3$.

ing to reduce the number of additional quantum gates, the total execution time of quantum circuits, or the effects of noise, among others. Furthermore, some works [60,61] are dedicated to mapping and scheduling error correction code circuits. Under the premise of on-demand distribution of entanglement (on-demand protocols), the qubit mapping and routing methods for single quantum machines can facilitate the execution of remote quantum gates by converting some SWAP gates into teledata.

Previous studies [10,18–22] often assume arbitrary communication between distributed quantum machines in the quantum network without any entanglement distribution protocols. However, for qubit mapping and scheduling in multi-quantum-machine scenarios, on-demand protocols tend to produce excessive latency (classical communication and processing). In contrast, continuous distribution of entanglement (CD protocols) [12] is more suited for the distributed execution of quantum circuits.

Previous works on distributed circuit mapping and scheduling have mainly focused on reducing communication costs, execution time, or the total cost of circuit execution during the process. Early work [62] by Yimsiriwattana and Lomonaco proposed an execution scheme for the distributed Shor algorithm, introducing primitive distributed computing operators, namely cat-entangler and cat-disentangler. Baker [19] *et al.* considered a time-sliced circuit partitioning method to reduce communication costs in quantum circuit execution. Andres *et al.* [18] designed execution schemes to reduce communication costs in distributed scenarios with hypercube connectivity using hypergraph partitioning. Wu *et al.* [20,21] analyzed the patterns of burst communication and collective communication in quantum circuits and designed scheduling algorithms for these communication patterns. Mao *et al.* [22] demonstrated the difficulty of the qubit allocation problem in distributed quantum computing and proposed a multistage hybrid simulated annealing algorithm to reduce the total cost of remote gates during circuit execution. Cuomo *et al.* [10] formulated the problem as a dynamic network flow and constructed an optimization problem model, combining runtime minimization with entangled states.

In this work, we introduce the continuous distribution of entanglement (CD protocol) for distributed quantum circuit execution and then propose a general method for partitioning circuits, mapping subcircuits to distributed clusters with heterogeneous bandwidth (where the links between nodes are not uniform), and scheduling remote operations. Our method takes into account both circuit execution time and entanglement costs. We summarize the relevant results in Table V.

VII. CONCLUSION AND DISCUSSION

This work presents a qubit mapping and scheduling algorithm designed to efficiently execute quantum circuits within continuous distribution of entanglement (CD protocols). We analyze existing entanglement distribution protocols for distributed quantum circuit execution. We define the min-depth qubit mapping and scheduling problem within the CD protocol and propose a systematic solution that includes protocol analysis, circuit partitioning, and remote operation scheduling. Additionally, we introduce a new entanglement swapping scheme, connectivity-first swapping, within the CD protocol to enhance the execution of large quantum circuits in distributed clusters. Our extensive evaluations demonstrate that our algorithms achieve significant reductions in circuit execution time and entanglement costs, with reductions of up to 53.1% in execution time and 20.3% in entanglement consumption compared to baseline methods.

Limitation and future work. As the critical procedure of quantum circuit execution in a distributed setting, circuit partitioning, mapping, and remote operations scheduling require

TABLE V. Results of quantum circuit execution in a distributed system. Note that most works focused on reducing the entanglement cost and do not consider the bandwidth between nodes in the cluster. RCX: remote gate execution shown in Fig. 1. Data Transfer: using methods like quantum teleportation to transmit the information of a qubit to another node to execution remote operations.

Algorithms	Remote operation type	Network protocol	Bandwidth	E-cost optimization	Time optimization
Cuomo *et al.* [10]	RCX	on-demand	Homogeneity	✗	√
MHSA [22]	RCX	on-demand	Homogeneity	√	✗
Hyper-partitioning [18]	RCX	on-demand	Homogeneity	√	✗
Autocomm [20]	RCX	on-demand	Homogeneity	√	✗
Qucomm [21]	Data transfer	on-demand	Homogeneity	√	✗
Baker *et al.* [19]	Data transfer	N/A[a]	N/A[a]	√	✗
This work	RCX	continuous	Heterogeneity	√	√

[a]Baker *et al.* [19] do not consider the entanglement protocol and cluster bandwidth.

more effective optimizations. However, solving for an optimal k-way partition is known as NP-hard [34], while maximizing the parallel remote operations remains an open problem. We must explore effectively combining cluster topology and quantum circuit characterization. Furthermore, incorporating detailed models of entanglement distribution protocols and asynchronous quantum network behavior could provide additional optimization opportunities for distributed circuit execution.

ACKNOWLEDGMENTS

This work was partially supported by Innovation Program for Quantum Science and Technology (Grant No. 2021ZD0302901), Anhui Initiative in Quantum Information Technologies (Grant No. AHY150100), National Natural Science Foundation of China (Grant No. 62102388), National Key R&D Program of China under Grant No. 2021ZD0110400, and China National Natural Science Foundation with No. 62132018, 62231015, "Pioneer" and "Leading Goose" R&D Program of Zhejiang, 2023C01029, and 2023C01143.

DATA AVAILABILITY

The data that support the findings of this article are openly available [63].

[1] F. Arute, K. Arya, R. Babbush, D. Bacon, J. C. Bardin, R. Barends, R. Biswas, S. Boixo, F. G. Brandao, D. A. Buell *et al.*, Nature (London) **574**, 505 (2019).
[2] Y. Wu, W.-S. Bao, S. Cao, F. Chen, M.-C. Chen, X. Chen, T.-H. Chung, H. Deng, Y. Du, D. Fan *et al.*, Phys. Rev. Lett. **127**, 180501 (2021).
[3] M. Malinowski, D. T. C. Allcock, and C. J. Ballance, PRX Quantum **4**, 040313 (2023).
[4] A. Dahlberg, M. Skrzypczyk, T. Coopmans, L. Wubben, F. Rozpedek, M. Pompili, A. Stolk, P. Pawelczak, R. Knegjens, J. de Oliveira Filho, R. Hanson, and S. Wehner, in *Proceedings of the ACM Special Interest Group on Data Communication, SIGCOMM 2019*, edited by J. Wu and W. Hall (Association for Computing Machinery, New York, 2019), pp. 159–173.
[5] J.-W. Pan, D. Bouwmeester, H. Weinfurter, and A. Zeilinger, Phys. Rev. Lett. **80**, 3891 (1998).
[6] J. Niu, L. Zhang, Y. Liu, J. Qiu, W. Huang, J. Huang, H. Jia, J. Liu, Z. Tao, W. Wei *et al.*, Nat. Electron. **6**, 235 (2023).
[7] D. Main, P. Drmota, D. P. Nadlinger, E. M. Ainley, A. Agrawal, B. C. Nichol, R. Srinivas, G. Araneda, and D. M. Lucas, Nature (London) **638**, 383 (2025).
[8] P. W. Shor, SIAM Rev. **41**, 303 (1999).
[9] D. Bouwmeester, J.-W. Pan, K. Mattle, M. Eibl, H. Weinfurter, and A. Zeilinger, Nature (London) **390**, 575 (1997).
[10] D. Cuomo, M. Caleffi, K. Krsulich, F. Tramonto, G. Agliardi, E. Prati, and A. S. Cacciapuoti, ACM Trans. Quantum Comput. **4**, 1 (2023).
[11] K. Chakraborty, F. Rozpedek, A. Dahlberg, and S. Wehner, arXiv:1907.11630.
[12] Á. G. Iñesta and S. Wehner, Phys. Rev. A **108**, 052615 (2023).
[13] O. A. Collins, S. D. Jenkins, A. Kuzmich, and T. A. B. Kennedy, Phys. Rev. Lett. **98**, 060502 (2007).
[14] F. Rozpędek, K. Goodenough, J. Ribeiro, N. Kalb, V. C. Vivoli, A. Reiserer, R. Hanson, S. Wehner, and D. Elkouss, Quantum Sci. Technol. **3**, 034002 (2018).
[15] S. Khatri, C. T. Matyas, A. U. Siddiqui, and J. P. Dowling, Phys. Rev. Res. **1**, 023032 (2019).
[16] F. Rozpędek, R. Yehia, K. Goodenough, M. Ruf, P. C. Humphreys, R. Hanson, S. Wehner, and D. Elkouss, Phys. Rev. A **99**, 052330 (2019).
[17] B. Li, T. Coopmans, and D. Elkouss, IEEE Trans. Quantum Eng. **2**, 4103015 (2021).
[18] P. Andrés-Martínez and C. Heunen, Phys. Rev. A **100**, 032308 (2019).
[19] J. M. Baker, C. Duckering, A. Hoover, and F. T. Chong, in *Proceedings of the 17th ACM International Conference on Computing Frontiers* (Association for Computing Machinery, New York, 2020), pp. 98–107.
[20] A. Wu, H. Zhang, G. Li, A. Shabani, Y. Xie, and Y. Ding, in *Proceedings of the 55th IEEE/ACM International Symposium on Microarchitecture (MICRO)* (IEEE, Piscataway, NJ, 2022), pp. 1027–1041.
[21] A. Wu, Y. Ding, and A. Li, in *Proceedings of the 56th Annual IEEE/ACM International Symposium on Microarchitecture* (IEEE, Piscataway, NJ, 2023), pp. 479–493.
[22] Y. Mao, Y. Liu, and Y. Yang, in *Proceedings of the IEEE INFOCOM 2023-IEEE Conference on Computer Communications* (IEEE, Piscataway, NJ, 2023), pp. 1–10.
[23] A. Einstein, B. Podolsky, and N. Rosen, Phys. Rev. **47**, 777 (1935).
[24] K.-i. Yoshino, T. Ochi, M. Fujiwara, M. Sasaki, and A. Tajima, Opt. Express **21**, 31395 (2013).
[25] L. J. Stephenson, D. P. Nadlinger, B. C. Nichol, S. An, P. Drmota, T. G. Ballance, K. Thirumalai, J. F. Goodwin, D. M. Lucas, and C. J. Ballance, Phys. Rev. Lett. **124**, 110501 (2020).
[26] R. Ursin, F. Tiefenbacher, T. Schmitt-Manderbach, H. Weier, T. Scheidl, M. Lindenthal, B. Blauensteiner, T. Jennewein, J. Perdigues, P. Trojek *et al.*, Nat. Phys. **3**, 481 (2007).
[27] J. S. Sidhu, S. K. Joshi, M. Gündoğan, T. Brougham, D. Lowndes, L. Mazzarella, M. Krutzik, S. Mohapatra, D. Dequal, G. Vallone *et al.*, IET Quantum Commun. **2**, 182 (2021).
[28] Á. G. Iñesta, G. Vardoyan, L. Scavuzzo, and S. Wehner, npj Quantum Inf. **9**, 46 (2023).
[29] H.-J. Briegel, W. Dür, J. I. Cirac, and P. Zoller, Phys. Rev. Lett. **81**, 5932 (1998).
[30] M. Victora, S. Tserkis, S. Krastanov, A. S. de la Cerda, S. Willis, and P. Narang, Phys. Rev. Res. **5**, 033171 (2023).
[31] M. Azari, P. Polakos, and K. P. Seshadreesan, IEEE Trans. Quantum Eng. **5**, 4101115 (2024).
[32] L. Hartmann, B. Kraus, H.-J. Briegel, and W. Dür, Phys. Rev. A **75**, 032310 (2007).
[33] W. Dür and H. J. Briegel, Rep. Prog. Phys. **70**, 1381 (2007).
[34] T. N. Bui and C. Jones, Inf. Proc. Lett. **42**, 153 (1992).

[35] G. Karypis, Technical report, Department of Computer Science, University of Minnesota, 1997.

[36] R. Jonker and T. Volgenant, in *DGOR/NSOR: Papers of the 16th Annual Meeting of DGOR in Cooperation with NSOR/Vorträge der 16. Jahrestagung der DGOR zusammen mit der NSOR* (Springer, 1988), pp. 622–622.

[37] D. F. Crouse, IEEE Trans. Aerospace Electron. Syst. **52**, 1679 (2016).

[38] A. Li, S. Stein, S. Krishnamoorthy, and J. Ang, ACM Trans. Quantum Comput. **4**, 1 (2023).

[39] Qiskit contributors, Qiskit: An open-source framework for quantum computing (2023), https://zenodo.org/records/8190968.

[40] T. Park and C. Y. Lee, Comput. Indust. Eng. **28**, 899 (1995).

[41] N. Sangouard, C. Simon, H. de Riedmatten, and N. Gisin, Rev. Mod. Phys. **83**, 33 (2011).

[42] C.-L. Li, H.-L. Yin, and Z.-B. Chen, Rep. Prog. Phys. **87**, 127901 (2024).

[43] K. Azuma, S. E. Economou, D. Elkouss, P. Hilaire, L. Jiang, H.-K. Lo, and I. Tzitrin, Rev. Mod. Phys. **95**, 045006 (2023).

[44] C.-L. Li, Y. Fu, W.-B. Liu, Y.-M. Xie, B.-H. Li, M.-G. Zhou, H.-L. Yin, and Z.-B. Chen, Opt. Lett. **48**, 1244 (2023).

[45] X.-Y. Chang, P.-Y. Hou, W.-G. Zhang, X.-Q. Meng, Y.-F. Yu, Y.-N. Lu, Y.-Q. Liu, B.-X. Qi, D.-L. Deng, and L.-M. Duan, Nat. Phys. **21**, 583 (2025).

[46] P. Chen, D.-W. Luo, and T. Yu, Phys. Rev. Res. **7**, 013161 (2025).

[47] Y. Pang, J. E. Castro, T. J. Steiner, L. Duan, N. Tagliavacche, M. Borghi, L. Thiel, N. Lewis, J. E. Bowers, M. Liscidini *et al.*, PRX Quantum **6**, 010338 (2025).

[48] W. J. Munro, A. M. Stephens, S. J. Devitt, K. A. Harrison, and K. Nemoto, Nat. Photon. **6**, 777 (2012).

[49] C. Delle Donne, M. Iuliano, B. van der Vecht, G. Ferreira, H. Jirovská, T. van der Steenhoven, A. Dahlberg, M. Skrzypczyk, D. Fioretto, M. Teller *et al.*, Nature (London) **639**, 321 (2025).

[50] A. Zulehner, A. Paler, and R. Wille, Efficient mapping of quantum circuits to the IBM QX architectures, in *Design, Automation Test in Europe Conference Exhibition* (Dresden, Germany, 2018), pp. 1135–1138.

[51] M. Y. Siraichi, V. F. dos Santos, C. Collange, and F. M. Q. Pereira, in *Proceedings of the 2018 International Symposium on Code Generation and Optimization, CGO'18* (Association for Computing Machinery, New York, NY, 2018), pp. 113–125.

[52] M. Y. Siraichi, V. F. d. Santos, C. Collange, and F. M. Q. Pereira, Proc. ACM Program. Lang. **3**, 1 (2019).

[53] A. Shafaei, M. Saeedi, and M. Pedram, in *Proceedings of the 19th Asia and South Pacific Design Automation Conference (ASP-DAC)* (IEEE, Piscataway, NJ, 2014), pp. 495–500.

[54] B. Tan and J. Cong, in *Proceedings of the IEEE/ACM International Conference On Computer Aided Design (ICCAD)* (IEEE, Piscataway, NJ, 2020), pp. 1–9.

[55] S. S. Tannu and M. K. Qureshi, in *Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems, ASPLOS'19* (Association for Computing Machinery, New York, NY, 2019), pp. 987–999.

[56] G. Li, Y. Ding, and Y. Xie, in *Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems, ASPLOS'19* (Association for Computing Machinery, New York, NY, 2019), pp. 1001–1014.

[57] M. G. Pozzi, S. J. Herbert, A. Sengupta, and R. D. Mullins, ACM Trans. Quantum Comput. **3**, 1 (2022).

[58] C. Zhang, A. B. Hayes, L. Qiu, Y. Jin, Y. Chen, and E. Z. Zhang, in *Proceedings of the 26th ACM International Conference on Architectural Support for Programming Languages and Operating Systems* (Association for Computing Machinery, New York, NY, 2021), pp. 360–374.

[59] H. Fu, M. Zhu, J. Wu, W. Xie, Z. Su, and X.-Y. Li, in *Proceedings of the IEEE/ACM International Conference on Computer Aided Design (ICCAD)* (IEEE, Piscataway, NJ, 2023), pp. 1–9.

[60] A. Javadi-Abhari, P. Gokhale, A. Holmes, D. Franklin, K. R. Brown, M. Martonosi, and F. T. Chong, in *Proceedings of the 50th Annual IEEE/ACM International Symposium on Microarchitecture* (IEEE, Piscataway, NJ, 2017), pp. 692–705.

[61] M. Zhu, H. Fu, J. Wu, C. Zhang, W. Xie, and X.-Y. Li, in *Proceedings of the IEEE/ACM International Symposium on Code Generation and Optimization (CGO)* (IEEE, Edinburgh, United Kingdom, 2024), pp. 158–169.

[62] A. Yimsiriwattana and S. J. Lomonaco, Jr., in *Quantum Information and Computation II* (SPIE, 2004), Vol. 5436, pp. 360–372.

[63] H. Fu, Efficient quantum circuit execution with continuous distribution of entanglement, https://github.com/hflash/qnet/tree/master.