

FQAS: A Fidelity-Aware Qubit Allocation System for Efficient Distributed Quantum Computing

Hongding Zhang¹, Wei Liang¹, Senior Member, IEEE, Yuxiang Chen¹, Neal N. Xiong¹, Senior Member, IEEE, Sheng Huo, Zijian Zhang², Senior Member, IEEE, and Jie Wu³, Fellow, IEEE

Abstract—Distributed quantum computing (DQC) integrates the computational resources of multiple small-to-medium-scale quantum processors to form a larger composite system capable of solving problems beyond the capacity of a single device. However, remote quantum communication between processors incurs significant overhead and is highly sensitive to entanglement degradation and link noise. Existing compiler-level qubit allocation strategies typically overlook these fidelity variations, leading to inefficient or unreliable mappings. In this work, we propose a Fidelity-aware Qubit Allocation System (FQAS). This system explicitly models how purification requirements and resource consumption affect the cost of remote communication in DQC, and it is the first to integrate a fidelity model into the distributed qubit allocation. FQAS further incorporates a Fidelity-aware Path Planning (FPP) algorithm that identifies communication paths with minimal resource consumption. The resulting Fidelity-aware Qubit Allocation Problem (FQAP) is formulated as a generalized quadratic assignment problem and solved using a customized hybrid genetic algorithm (HGA). We evaluate FQAS across various quantum circuits and network topologies. Experimental results demonstrate that the proposed system effectively reduces remote communication overhead, achieving an average improvement of approximately 12.5% on real circuits and 5.3% on random circuits compared with existing methods.

Index Terms—Distributed quantum computing (DQC), entanglement purification, quantum communication, qubit allocation.

I. INTRODUCTION

IN RECENT years, quantum computing has attracted significant attention for its potential to accelerate specific

computational tasks and to support advanced services in distributed systems such as the Internet of Things (IoT) [1], [2]. Quantum processors exploit phenomena such as superposition and entanglement to encode and manipulate information in fundamentally different ways from classical processors [3]. For certain problem classes, these properties promise asymptotic or practical speedups. Additionally, quantum-enabled techniques have the potential to improve aspects of IoT systems such as resource allocation and scheduling. However, realizing these benefits in practical IoT deployments requires overcoming constraints of current quantum hardware and network architectures.

Although the advantages of quantum computing have become apparent, there are still many obstacles to its practical application. A crucial problem is the limited number of qubits in quantum processors. Fortunately, distributed quantum computing (DQC) offers a solution to increase the scale of quantum computing. By relying on the interconnection of quantum networks, small-scale quantum processors can cooperate through the DQC paradigm. By sharing or transferring data between different quantum processors via entanglement-based quantum communication, remote gate (also known as nonlocal gate) operations can be performed [4], creating a quantum computing cluster with a larger number of qubits. However, the execution of remote gates between nodes necessitates the establishment of quantum entanglement among them, which inherently requires additional consumption for computational tasks [5]. In distributed clusters, logical qubits involved in two-qubit gates may reside on separate nodes, causing such operations to become remote gates that require inter-node entanglement operations. Consequently, the compiler should strive to execute gates locally whenever possible to minimize communication overhead.

A quantum program is typically described as a quantum circuit in which logical qubits represent abstract program variables, while physical qubits denote actual hardware qubits on processors. A DQC compiler must therefore realize a mapping from logical to physical qubits (qubit allocation) such that the logical circuit's operations can be executed on the distributed hardware [6]. In a distributed setting, the cost of implementing a remote two-qubit gate depends not only on the network topology, link quality, and required entanglement resources, but also on the relative positions of the nodes in the network. Longer communication paths require additional entanglement operations, consuming more resource and time, while being more susceptible to decoherence and noise. As a result, allocation choices that ignore these factors may

Received 23 October 2025; revised 23 December 2025; accepted 19 January 2026. Date of publication 23 January 2026; date of current version 9 April 2026. This work was supported in part by the Joint Key Project of National Natural Science Foundation of China under Grant U2468205, in part by the National Natural Science Foundation of China under Grant 62472168, in part by the Hunan Provincial Key Research and Development Program under Grant 2024AQ2028, and in part by the Postgraduate Scientific Research Innovation Project of Hunan Province under Grant CX20251516. (Corresponding authors: Neal N. Xiong; Wei Liang.)

Hongding Zhang, Wei Liang, Yuxiang Chen, and Neal N. Xiong are with the School of Computer Science and Engineering, Hunan University of Science and Technology, Xiangtan 411201, China, also with the Sanya Institute, Hunan University of Science and Technology, Sanya 572024, China, and also with the Hunan Key Laboratory for Service Computing and Novel Software Technology, Xiangtan 411201, China (e-mail: hdzhang@mail.hnust.edu.cn; wliang@hnust.edu.cn; chen-yuxiang@hnust.edu.cn; xiongnai@hnust.edu.cn).

Sheng Huo is with Zhuzhou CRRC Times Electric Company Ltd., Zhuzhou 412001, China (e-mail: huosheng@csrzc.com).

Zijian Zhang is with the School of Cyberspace Science and Technology, Beijing Institute of Technology, Beijing 100811, China (e-mail: zhangzijian@bit.edu.cn).

Jie Wu is with the Department of Computer Science and Information Sciences, Temple University, Philadelphia, PA 19122 USA (e-mail: jiewu@temple.edu).

Digital Object Identifier 10.1109/IJOT.2026.3657616

2327-4662 © 2026 IEEE. All rights reserved, including rights for text and data mining, and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission.

See <https://www.ieee.org/publications/rights/index.html> for more information.

Authorized licensed use limited to: Goa University. Downloaded on October 08, 2026 at 09:01:51 UTC from IEEE Xplore. Restrictions apply.

substantially increase communication overhead and reduce fidelity.

Although several studies have attempted to address qubit allocation in DQC systems, they have primarily focused on optimizing the layout of qubit allocation to reduce the number of remote gates [7], [8], [9], [10], [11] or to shorten the distance between each remote gate in the network topology [12]. These approaches generally assume that entanglement links are ideal or sufficiently reliable. In practice, however, quantum links suffer from noise and decoherence, which limit the fidelity of generated entangled pairs; achieving higher fidelity typically requires entanglement purification, which introduces additional resource consumption and time overhead. As a result, neglecting link fidelity and purification costs systematically underestimates communication overhead, leading to suboptimal mapping solutions that increase total resource usage or compromise computational correctness.

To overcome this limitation, we extend existing mapping frameworks in this work. First, we identify link-level entanglement reliability as a key factor shaping communication cost and construct an end-to-end (E2E) communication cost framework that explicitly incorporates purification overhead. This framework is the first to unify path-level purification requirements and communication resource consumption within a single cost function, enabling more realistic and physically grounded estimations of remote communication cost. Second, we incorporate a Fidelity-aware Path Planning (FPP) algorithm into Fidelity-aware Qubit Allocation System (FQAS), which evaluates and selects candidate communication paths based on link fidelity and purification cost, thereby choosing lower-cost routes and reducing the overall overhead of remote quantum communication. Finally, based on these components, we formalize the Fidelity-aware Qubit Allocation Problem (FQAP) and design a hybrid genetic algorithm (HGA) to efficiently solve it.

The main contributions of this article are as follows.

- 1) We propose a fidelity-aware qubit allocation framework that incorporates and analyzes in detail the impact of entanglement fidelity on inter-node communication overhead in DQC systems, thereby providing a more realistic and reliable estimation of communication costs for DQC.
- 2) We develop FPP, a path planning algorithm that integrates link fidelity and purification overhead into path selection, enabling low-cost entanglement provisioning for remote gates.
- 3) We formulate the FQAP as a generalized quadratic assignment problem and design a HGA to solve it efficiently.
- 4) We validate our approach through extensive simulations, showing that FQAS substantially reduces the communication overheads incurred between nodes in DQC models.

II. RELATED WORK

The parallelism inherent in quantum computing allows it to address complex problems more efficiently than classical computers [13]. However, technological constraints in design limit the number of qubits in a quantum computer to a relatively

low level, which may result in insufficient physical qubits for solving complex problems. Consequently, the construction of a DQC architecture has emerged as a consensus for continuously scaling computation, as it can aggregate different quantum processors to overcome the capacity limitations of individual computers.

In a DQC architecture, different quantum processors can exchange data through remote quantum communication and collaboration [14]. However, the exchange process incurs additional overheads for computational tasks. Consequently, how to compile quantum circuits that represent computational tasks into the physical cluster with minimal overhead has become an important problem in the design of distributed architectures. Andrés-Martínez and Heunen [7] constructed a hypergraph based on the interactions within the logical circuit, with each hyperedge representing a remote gate between two logical qubits. The qubit allocation was then achieved by finding the minimum cut that partitions the hypergraph into a specified number of sections. Building on this work, Sundaram et al. [8] further decomposed the problem and proposed a two-stage heuristic approach. In the first stage, a simpler minimum cut problem was solved for graph partitioning, followed by a second stage to find the minimum migration set for all remote gates. Dadkhah et al. [9] introduced a modeling method that converts the logical circuit into a weighted undirected graph and reordered the placement of qubits before graph partitioning to enhance the performance of qubit allocation. Similar to these works are those by [10] and [11], which primarily focus on qubit allocation by solving graph representation and partitioning problems of quantum circuits. However, these works concentrate on minimizing the number of remote communications, neglecting that the interactions between different remote operations are not equivalent, and the overhead of achieving these interactions is influenced by the network topology and network state. In contrast, Mao et al. [12] calculate the overhead of remote gate interactions between processors based on the number of hops in the network structure and propose a multistage hybrid simulated annealing algorithm to reduce the total communication overhead of all remote gates. Nevertheless, this work does not account for the fidelity differences among different links in the network.

In DQC, inter-node communication relies on establishing remote entanglement between two qubits. However, the inevitable presence of quantum noise and decoherence can cause the entangled pairs to degrade as the propagation distance increases [15]. To minimize errors in remote communication, purifying the entangled pairs used for communication is necessary. This purification process enhances the probability of the entangled pairs being in the correct state, thereby providing the highest possible reliability for the fragile communication process. Nevertheless, entanglement purification unavoidably leads to an increase in communication overheads. Currently, numerous works have been conducted to design protocols for DQC models based on entanglement fidelity [16], [17], [18], [19]. Victoria et al. [17] employed a weighted Dijkstra algorithm to select appropriate entanglement paths based on link fidelity and entanglement generation rates in the quantum network. Li et al. [16] developed entanglement routing and purification methods that ensure fidelity

for entanglement connections within quantum networks and provide routing solutions tailored to different computational resource requirements. Zhao et al. [19] quantified the impact of entanglement purification on the E2E fidelity of entanglement connections and maximized network throughput by selecting paths with higher E2E fidelity to reduce the number of purifications. These works thoroughly consider the variability in the reliability of entanglement connections. However, no previous studies have linked this variability to the overheads incurred during the compilation stage of remote gates. To the best of our knowledge, this is the first work to integrate a purification-aware E2E fidelity model into the distributed qubit allocation.

III. DQC SYSTEMS

A. Quantum Terminologies

1) *Quantum Gates and Quantum Circuit*: Quantum logic gates, also known as quantum operators, are a set of basic quantum operators used to control quantum data. According to the number of qubits upon which the quantum gate acts, we categorize those that act on a single qubit as single-qubit gates and those that act on multiple qubits as multiqubit gates. Multiqubit gates are typically employed to facilitate interactions between qubits. Among the most representative multiqubit gates is the Control-NOT (CNOT) gate, which acts on two qubits. Specifically, the $\text{CNOT}(q_1, q_2)$ gate determines the state of the target qubit q_2 based on the state of the control qubit q_1 . If q_1 is $|1\rangle$, the CNOT gate flips the state of q_2 . Otherwise, the gate does not operate.

The quantum computation process is typically represented as a quantum circuit composed of a set of logical qubits and a collection of gate instructions that operate on these qubits. In a quantum circuit, each qubit carrying data is represented by a horizontal line, and its state is controlled by a sequence of gates executed from left to right. According to the designer's intent, a quantum circuit can systematically connect different quantum gates to realize a specific information processing process.

2) *Quantum Processor*: As the physical carriers of qubits and quantum computing systems, the number of qubits a quantum computer can accommodate is a critical factor in determining the future development of quantum computing. Qubits within a quantum processor have corresponding names based on their functional roles. Specifically, a part of qubits in each quantum processor is reserved for communication between processors, referred to as communication qubits. To distinguish them from the other qubits specifically dedicated to computational or storage tasks, known as data qubits [20]. Supported by a quantum network, quantum processors distributed across different regions (serving as network nodes) can be integrated to form a DQC system [21].

3) *Quantum Entanglement and Purification*: Quantum characteristics endow quantum technology with a unique mode of communication. When a set of qubits cannot be independently described in terms of their quantum state without reference to the other qubits within the set, we refer to this set as being in a unique correlated state known as quantum entanglement. In this article, we employ a type of entanglement known as Bell state, which represents a pair of qubits

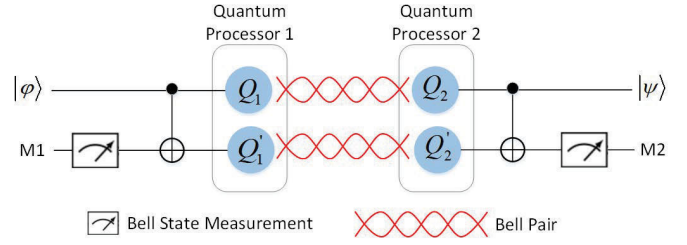


Fig. 1. Illustration of entanglement purification. The Bell pair (Q'_1, Q'_2) is used as a consumed resource to enhance the fidelity of the Bell pair (Q_1, Q_2) .

(referred to as a Bell pair) where each qubit maintains a state of $(|00\rangle + |11\rangle)/\sqrt{2}$. With the advancement of quantum communication, many physical technologies can generate remote Bell pairs between quantum nodes at the ends of a link, which can be considered quantum communication channels between the nodes [22]. However, when Bell pairs have low fidelity, the probability of bit-flip errors increases, disrupting the expected entangled state and rendering the resulting entanglement link ineffective for communication [19].

Entanglement purification can enhance the fidelity of a Bell pair (or its corresponding entanglement link) by consuming quantum resources. As shown in Fig. 1, there is a Bell pairs (Q_1, Q_2) between Quantum Processor 1 and Quantum Processor 2, which establishes an entanglement link between them. At this point, we do not know whether the entanglement link is in the expected entangled state. To this end, we use another Bell pairs (Q'_1, Q'_2) to test the entanglement link. The two-qubit pairs will be connected by a CNOT gate, with Q_1 and Q_2 (acting as control qubits) controlling the states of Q'_1 and Q'_2 (acting as target qubits). After passing through the CNOT gate, Quantum Processor 1 and Quantum Processor 2 measure the quantum states of Q'_1 and Q'_2 , respectively, and send their measurement results to each other. If the measurement results of the two qubits (i.e., $M1$ and $M2$) are consistent, it means that the Bell pair (Q_1, Q_2) has a high probability of being in the expected entangled state and the entanglement link is retained. Otherwise, the Bell pair (Q_1, Q_2) is considered to have made an error and is no longer used. It is important to note that, regardless of whether the entanglement link is retained, the Bell pair (Q'_1, Q'_2) has already been consumed because it will no longer maintain entanglement after the measurement.

In this article, we assume that all Bell pairs generated on the same quantum link have the same initial fidelity. For two Bell pairs both initialized with fidelity F , the fidelity of the retained entangled link after purification is

$$F' = \frac{F^2}{F^2 + (1 - F)^2} > F. \quad (1)$$

If resources are sufficient, we can repeatedly perform purification operations on the same entanglement link until its fidelity meets the desired transmission requirements. Given that entanglement fidelity decreases rapidly over time after creation, FQAS opts to generate and consume multiple Bell pairs simultaneously for link purification. Although this approach may place greater demands on entanglement preparation, it allows us to disregard changes in Bell pair fidelity due to entanglement decoherence. It is worth noting that although

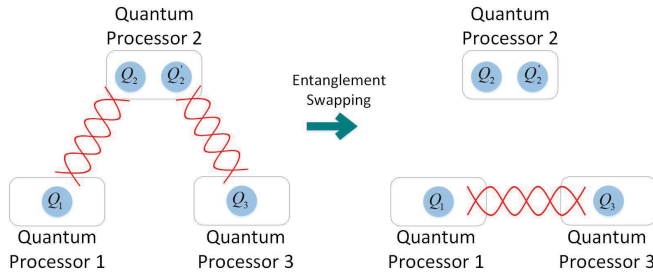


Fig. 2. When two processors are not directly adjacent, the remote gates are achieved through entanglement swapping.

the fidelity of a Bell pair can take any value between 0 and 1, Bell pairs with fidelity no higher than 0.5 are discarded rather than purified due to their insufficient reliability [23].

4) *Entanglement Swapping*: Entanglement swapping can assist two nonadjacent quantum nodes in establishing long-distance entanglement connections. As illustrated in Fig. 2, two communication qubits on Quantum Processor 2, Q_2 and Q_2' , are maximally entangled with Q_1 on Quantum Processor 1 and Q_3 on Quantum Processor 3, respectively. Subsequently, a Bell state measurement [24] is performed on the communication qubits Q_2 and Q_2' at the relay node Quantum Processor 2. This results in the transformation of the two entanglement links into a direct entanglement connecting Q_1 and Q_3 , and the two communication qubits on Quantum Processor 2 are also released. For more general scenarios, if there are multiple repeater nodes along the path, by performing entanglement swapping operations at all repeater nodes along the path, we can “suture” several entanglement links into a direct entanglement between the source node and the destination node.

- Step 1: Determine the set of paths P that includes all the links between the two target nodes.
- Step 2: Generate Bell pairs on each link.
- Step 3: Perform entanglement swapping at each repeater node.

B. Remote Gate Establishment

In a distributed quantum framework, logical qubits assigned to different physical processor nodes may need to execute remote CNOT gates. The key to enabling this communication is to establish an E2E entanglement connection between their respective nodes. These quantum nodes, which aim to establish such entanglement connections, are referred to as the source–destination (S-D) pair.

Mao et al. [12] formulated the distributed qubit allocation problem and analyzed the implementation process and incurred overhead of remote CNOT gates. When there is an entanglement link between the S-D pair, a maximally entangled pair (i.e., a Bell pair) can be directly generated on the link using communication qubits. This Bell pair can then execute a remote CNOT gate on any data qubits in the quantum processor. As illustrated in Fig. 3, to establish a remote CNOT gate between Q_1 and Q_2 , it suffices to perform local CNOT operations on Q_1' and Q_2' corresponding to their respective processors, along with some single-qubit gates and

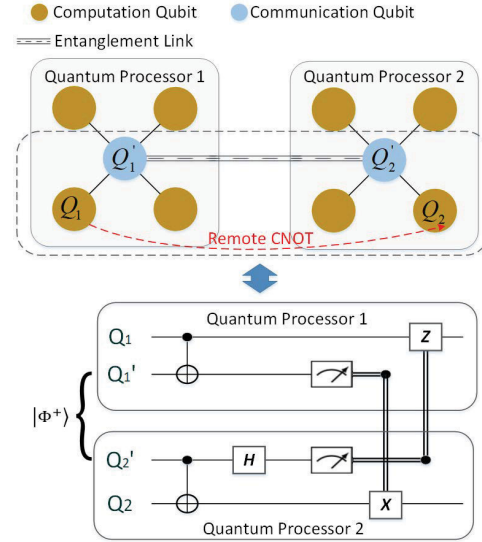


Fig. 3. Process of realizing the entangled link based remote CNOT operation.

measurement operations. In this case, the overhead of remote communication includes the overhead required to generate a pair of communication qubits C_{ep} and the definite overhead of a remote CNOT circuit C_{rcn} . When the S-D pair lacks a direct connection, establishing E2E entanglement requires relay nodes to perform entanglement swapping on adjacent links along the path. The overhead of an entanglement swapping operation is denoted as C_{esp} . For an S-D pair connected by a communication path passing through N repeater nodes, N entanglement swapping operations are required to connect $N + 1$ Bell pairs on the links. In summary, the remote CNOT communication overhead for achieving remote communication between any pair of different quantum nodes is represented as

$$c_{\text{comm.}}(S, D) = (N + 1) \cdot C_{\text{ep}} + N \cdot C_{\text{esp}} + C_{\text{rcn}}, N \geq 0. \quad (2)$$

However, this work only accounts for the basic overheads associated with establishing remote gates during the analysis, overlooking the additional overheads necessary to ensure reliable communication for the implementation of these remote gates.

IV. OUR PROPOSED FQAS

In this section, we focus on detailing the components of the FQAS. First, in Section IV-A, we present a formalized description of the FQAP within distributed computing systems. Then, in Section IV-B, we evaluate the overhead required to establish reliable remote gates and provide detailed calculation formulas. Subsequently, in Section IV-C, we discuss the routing problem for a single S-D pair and propose a fidelity-aware routing algorithm to determine the optimal path and purification decisions with minimal communication overhead. Finally, in Section IV-D, we present a HGA to optimize the mapping layout, aiming to achieve optimal performance for the mapping problem. The overall framework of the model is depicted in Fig. 4.

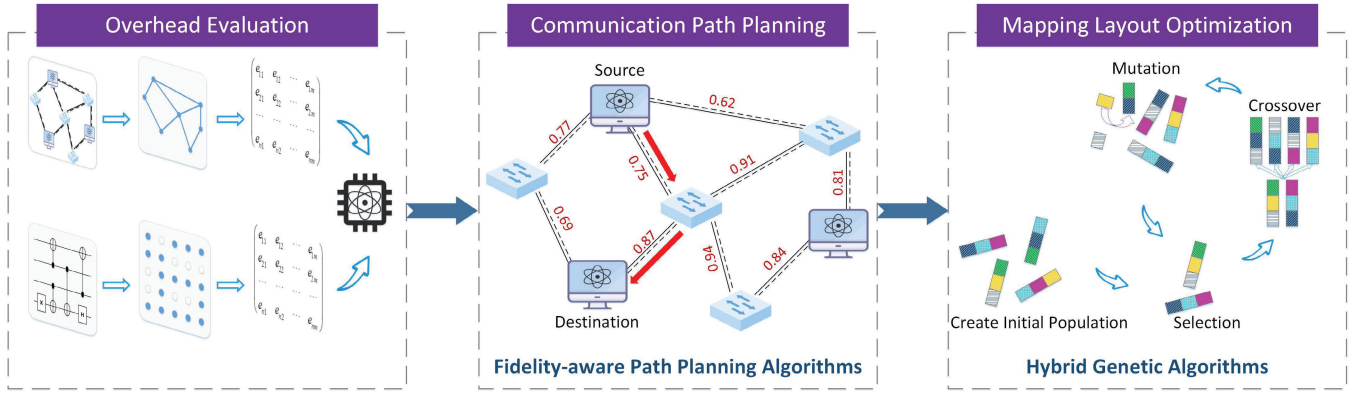


Fig. 4. Specific framework diagram of the FQAS proposed in this article.

A. Problem Formulation

Achieving an efficient qubit mapping to minimize remote interactions between processors and reduce remote communication costs is essential. This approach not only alleviates the system load but also enhances the reliability of the computation process. In this section, we first establish the formal representation of the FQAP. Specifically, the input for FQAP consists of the following three components.

- 1) A quantum computing cluster, denoted as $H = (V, E)$ in the network topology graph, where each node $v \in V$ represents a quantum processor and each edge $e = (v_i, v_j) \in E$ represents a quantum link connecting processors v_i and v_j . The initial fidelity of the Bell pairs generated on these links is denoted as F_e .
- 2) A logical circuit composed of a set of logical qubits Q and a set of CNOT gates G , where each CNOT gate $g = \text{CNOT}(q_i, q_j) \in G$ represents whether the value of qubit q_j is to be inverted based on the value of qubit q_i . It is noteworthy that $\text{CNOT}(q_i, q_j) \neq \text{CNOT}(q_j, q_i)$.
- 3) The generation overhead c_{epf} for producing a pair of Bell states on the quantum link, the overhead c_{esp} for performing an entanglement swapping operation, and the fixed overhead c_{rcn} for establishing a remote CNOT circuit.

For any logical qubit $q \in Q$ and node $v \in V$ hosting data qubits, their mapping relationship is denoted as $\mathcal{M}(\cdot)$, where $\mathcal{M}(q, v) = 1$ if and only if q is precisely mapped to node v for execution. It is important to emphasize that each logical or data qubit must have a unique mapping, meaning that a qubit already mapped cannot be remapped to another qubit. To ensure that each logical qubit is mapped to exactly one physical node, we impose the following constraint:

$$\sum_{v \in V} \mathcal{M}(q, v) = 1 \quad \forall q \in Q. \quad (3)$$

Additionally, each quantum node has an upper limit $n_{\text{cap.}}(v)$ on the number of physical qubits available for computation. Therefore, all quantum nodes must satisfy the capacity constraint

$$\sum_{q \in Q} \mathcal{M}(q, v) \leq n_{\text{cap.}}(v) \quad \forall v \in V. \quad (4)$$

For any two quantum nodes v_1 and v_2 , let $c(v_1, v_2)$ denote the cost of establishing a reliable remote CNOT gate between

them. For any pair of logical qubits $q_i, q_j \in Q$, let $n(q_i, q_j)$ represent the number of CNOT gates in the logical circuit. Consequently, when logical qubits q_i and q_j are mapped to quantum nodes v_1 and v_2 , the resulting cost for the remote CNOT gate is given by $n(q_i, q_j) \cdot c(v_1, v_2)$.

After accounting for all mapping relationships, the total cost of all remote CNOT gates is

$$\begin{aligned} \mathbb{O} = & \sum_{q_i, q_j \in Q} \sum_{v_1, v_2 \in V} n(q_i, q_j) \cdot c(v_1, v_2) \\ & \cdot \mathcal{M}(q_i, v_1) \cdot \mathcal{M}(q_j, v_2). \end{aligned} \quad (5)$$

Ultimately, the FQAP in the distributed computing system can be formulated as follows:

$$\begin{aligned} \min & \sum_{q_i, q_j \in Q} \sum_{v_1, v_2 \in V} n(q_i, q_j) \cdot c(v_1, v_2) \\ & \cdot \mathcal{M}(q_i, v_1) \cdot \mathcal{M}(q_j, v_2) \\ \text{s.t.} & \sum_{v \in V} \mathcal{M}(q, v) = 1 \quad \forall q \in Q \\ & \sum_{q \in Q} \mathcal{M}(q, v) \leq n_{\text{cap.}}(v) \quad \forall v \in V. \end{aligned} \quad (6)$$

B. Remote CNOT Gate Overhead Evaluation

As noted in Section III, establishing a basic E2E entanglement connection is insufficient for DQC tasks. Under conditions of lower E2E fidelity, entanglement is unlikely to provide reliable guarantees for long-distance quantum information transmission between the source and destination. Therefore, in this section, we analyze how to use entanglement purification to ensure that the E2E entanglement connections required for implementing remote CNOT gates achieve sufficient fidelity, as well as determine the associated costs for establishing these reliable connections. It is worth noting that although increasing E2E entanglement fidelity can effectively reduce the failure probability of remote quantum gates, performing multiple rounds of purification incurs substantial resource and latency overhead, thereby reducing the link resources available for other tasks in the network. Therefore, it is necessary to flexibly set the minimum required path-fidelity threshold according to the quality-of-service requirements of different quantum tasks [25], so as to strike a reasonable

balance between the reliability of remote gate operations and the overall network resource consumption.

First, we draw on [19] as a means to quantify the reliability of remote entanglement, which provides a formula for calculating the E2E fidelity of entanglement connections established from a given set of entangled links with specified fidelity. Specifically, for a path with N repeater nodes, the E2E fidelity F^{E2E} can be represented as

$$\begin{aligned} F^{\text{E2E}} &= 1 + UV - U - V \\ V &\approx 2p_N(1 - p_N)\Phi + p_N^2 \sum_{i=0}^{N-1} (1 - p_i)\Phi_i \\ U &\approx \sum_{i=0}^{N-1} (1 - p_i)\Phi_i \end{aligned} \quad (7)$$

which $\Phi_i = \prod_{k=0, k \neq i}^{N-1} p_k$, $\Phi = \prod_{k=0}^{N-1} p_k$, and p_i represents the probability that a qubit on link $l_i \in L$ [where $0 \leq i \leq N$] does not flip during the measurement process. This probability is related to the link fidelity F_i as follows:

$$p_i^2 + (1 - p_i)^2 = F_i. \quad (8)$$

According to (7), we compare the fidelity F^{E2E} with the path fidelity threshold F^* to determine whether purification is necessary before establishing the path.

We provide a brief overview of the critical purification scheme process used in FQAS. In Step 1, we verify whether the fidelity of each entanglement link F_i in the path meets the $F_i > F_{\min}$ constraint, and purify the links that do not satisfy the fidelity requirement. This approach prevents the purification trap discussed in [19], where purifying lower-fidelity links can potentially reduce the overall fidelity of the E2E entanglement connection. The $F_{\min}$ denotes the link fidelity threshold and calculated as

$$F_{\min} = \frac{(3N - 1)^2}{9N^2}, N \geq 1. \quad (9)$$

Subsequently, in Step 2, we iteratively execute the judgment and purification operations until $F^{\text{E2E}} \geq F^*$. Please note that in each round, we perform the purification operation only once on the critical link with the maximum partial derivative rather than multiple times. This is because each purification will alter the partial derivative of the link.

Now, we begin to analyze the purification overhead required to establish an E2E entanglement connection that satisfies the fidelity constraint F^* . Initially, we analyze the purification overhead consumed by Step 1. We denote the original fidelity of entanglement link F_i as F . According to (1), we can derive a direct relationship between the number of consumed purification Bell pairs t and the fidelity of the purified link $F_{(t)}$

$$F_{(t)} = \frac{F_{(0)}^{2^t}}{F_{(0)}^{2^t} + (1 - F_{(0)})^{2^t}} \quad (10)$$

where $F_{(0)} = F$. Therefore, the formula for calculating the minimum number of purification Bell pairs t needed for the link l_i to satisfy the fidelity constraint $F_{\min}$ is

$$t_i = \begin{cases} 0, & F \geq F_{\min} \\ \left\lceil \log_2 \left(\frac{\ln \left(\frac{F_{\min}}{1 - F_{\min}} \right)}{\ln \left(\frac{F}{1 - F} \right)} \right) \right\rceil, & 0.5 < F < F_{\min}. \end{cases} \quad (11)$$

Furthermore, for all links in the entire path, the total number of purified Bell pairs required (i.e., the resources consumed in Step 1) is

$$T_{\text{purify_link}}(S, D) = \sum_{i=0}^N t_i. \quad (12)$$

According to (7), the fidelity of the E2E path can be obtained and is denoted as F_0^{E2E} . When $F_0^{\text{E2E}} \geq F^*$, the path is already sufficiently reliable and does not consume additional resources. However, in other cases, further purification processes are still necessary. We denote the improvement in E2E fidelity brought about by the K th purification operation in Step 2 as $\Delta F_k = F_k^{\text{E2E}} - F_{k-1}^{\text{E2E}}$ ($k \geq 1$). Therefore, the number of purification Bell pairs required to continuously purify the critical links until the E2E fidelity satisfy the requirements is

$$\begin{aligned} T_{\text{purify_E2E}}(S, D) &= \min K \\ \text{s.t. } \sum_{k=1}^K \Delta F_k &\geq F^* - F_0^{\text{E2E}}. \end{aligned} \quad (13)$$

Given the known entanglement path and fidelity, the minimum number of Bell pairs required for purification to achieve reliable communication between different processors, denoted as $T_{\text{purify}}(S, D) = T_{\text{purify_link}}(S, D) + T_{\text{purify_E2E}}(S, D)$. Therefore, the entanglement purification overhead is

$$c_{\text{pur.}}(S, D) = C_{\text{epr}} \cdot T_{\text{purify}}(S, D). \quad (14)$$

Finally, combining (2) and (14), the total overhead for achieving one Remote CNOT operation with sufficient reliability is

$$c(S, D) = c_{\text{comm.}}(S, D) + c_{\text{pur.}}(S, D). \quad (15)$$

C. Communication Path Planning

Although the overhead calculation method for remote CNOT has been established, an unresolved issue remains: how to determine the path for communication between different nodes. As can be inferred from (15), the fidelity along the path is a crucial factor in determining the final overhead of remote CNOT operations. The fidelity determines the overhead of consuming valuable quantum resources for path purification, which may be more expensive than the basic overhead of establishing remote CNOT operations. Therefore, we argue that a simple choice of the shortest path is not sufficient. Instead, the purification decision-making process must be integrated with the path search process to plan the optimal path for communication between processors.

For a given S-D pair, we aim to find the routing path that satisfies fidelity constraints with minimal overhead. To achieve this goal, we employ depth-first search (DFS) to expand the search space of solutions as much as possible. However, as the network scale increases, the computational overhead and execution time of attempting to find all feasible complete paths become prohibitive. Therefore, it is necessary to set reasonable limits on the path search process to terminate the exploration of potential paths that are costly. Based on the above consideration, we propose a FPP, presented in Algorithm 1.

Algorithm 1 FPP Algorithm

Input: Quantum network topology graph H , Bell pair fidelity over each link F_l , Minimum fidelity requirement F^* , Source S , Destination D .

Output: Minimum overhead path P_{min} .

- 1 Use the Dijkstra algorithm to find the shortest path P_{min} and select it as the *candidate_path*;
- 2 Calculate the number of Bell pairs required to build the *candidate_path* as the *cost_threshold*;
- 3 Calculate the overhead of *candidate_path* according to Eq.(15) as *min_overhead*;
- 4 *stack.APPEND(S)*;
- 5 **while** *stack* is not empty **do**
- 6 *subpath* $\leftarrow$ *stack.POP()*;
- 7 **for** exist unvisited neighboring node R **do**
- 8 *new_subpath* = *subpath* + $[R]$;
- 9 Calculate the number of Bell pairs required to build the *new_subpath* as the *cost*;
- 10 **if** *cost* $\leq$ *cost_threshold* **then**
- 11 **if** $R = D$ **then**
- 12 *path* = *new_subpath*;
- 13 Calculate *overhead* of *path* according to Eq.(15);
- 14 **if** *overhead* < *min_overhead* **then**
- 15 *min_overhead* = *overhead*;
- 16 *candidate_path* = *path*;
- 17 **end**
- 18 **else**
- 19 *stack.APPEND(new_subpath)*;
- 20 **end**
- 21 **end**
- 22 **end**
- 23 **end**

1) *Path Threshold Determination*: Effective path overhead constraints can undoubtedly reduce unnecessary iterations and quickly eliminate overly costly path options. To ensure that the constraints sufficiently assist the algorithm in filtering potential paths, we designed an entanglement cost threshold. This provides a comparative metric within the range of near-optimal solutions for the path search process. Subsequently, the Dijkstra algorithm is used to obtain the path between the S-D pair with the minimum hops. This path is denoted as the initial candidate path, and the number of Bell pairs consumed to achieve a reliable remote CNOT operation on this path is set as the cost threshold.

2) *Cost-Constrained DFS Search*: After obtaining the cost threshold, DFS initiates path exploration between S-D pairs in the network topology starting from the source node. Beginning with the first iteration, the algorithm records corresponding exploration details for each sub-path, including visited nodes and their sequence, the quantum links connecting these nodes, and the fidelity of these links. During each iteration (Lines 5–22 of Algorithm 1), the algorithm sequentially extracts a sub-path (S, R) from the stack and calculates the quantity of Bell pairs required for the sub-path based on the exploration information to make a decision, and it can be

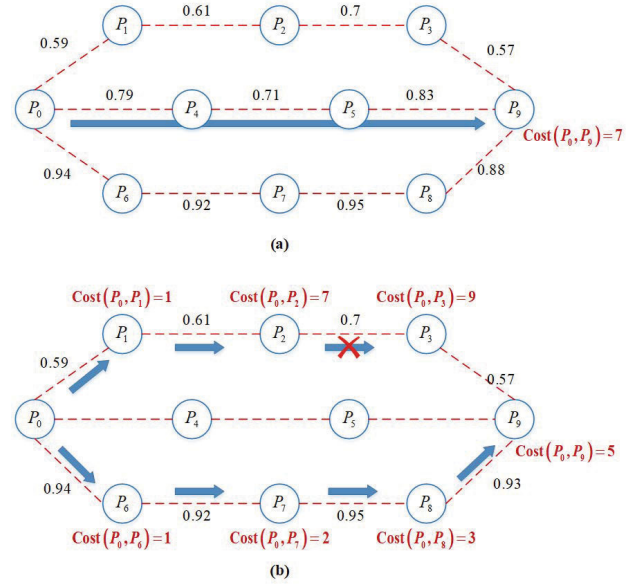


Fig. 5. Example of a path search process. The E2E fidelity threshold F^{E2E} is set to 0.8. (a) Shortest path selection and threshold calculation. (b) Search process under the threshold constraint.

calculated as

$$\text{Cost}(S, R) = \begin{cases} T_{\text{purify}}(S, R) + N + 1, & \text{if } R = D; \\ T_{\text{purify_link}}(S, R) + N + 1, & \text{otherwise.} \end{cases} \quad (16)$$

When R is the destination node, we calculate the total number of Bell pairs required to purify the path, (i.e., $T_{\text{purify}}(S, R)$). Otherwise, since the path length remains undetermined, FPP only considers the number of Bell pairs required to purify each entangled link between the source node S and the last visited node R on the path. At this stage, the degree of purification for the links is determined by the number of relay nodes along the sub-path. Additionally, the $N + 1$ Bell pairs necessary for establishing the link are also taken into account. The $\text{Cost}(S, R)$ will be compared with the path threshold, and if it exceeds the threshold, we will stop exploring this sub-path. If it exceeds this threshold, FPP terminates exploration of the sub-path. Otherwise, if the sub-path overhead is reasonable and the destination node has not been reached, all unexplored neighboring nodes of R become new endpoints for sub-paths, and exploration records are updated accordingly. All new sub-paths are added to the stack for further exploration (Line 18 of Algorithm 1). For each complete path identified, FPP calculates its total overhead according to (15) and updates the candidate path accordingly (Lines 11–16 of Algorithm 1). Finally, when the stack containing paths to be explored is empty, the minimum overhead serves as the expected overhead for remote interactions between S-D pairs, and the path with the minimum overhead is regarded as the optimal communication path between S-D pairs.

We use a nonspecialized routing example to clearly demonstrate the FPP search process and its advantages, as shown in Fig. 5. In this example, the fidelity of the links follow the range criterion of $(0.5, 1)$ [23]. Initially, FPP identifies the shortest-hop path $P_0 \rightarrow P_4 \rightarrow P_5 \rightarrow P_9$ and calculates the expected Bell pair consumption for establishing this path as $\text{Cost}(P_0, P_9) = 7$, setting it as the cost threshold. Then, FPP

begins large-scale exploration starting from P_0 . The iterative search along relay P_1 is terminated early to reduce complexity, as the expected entanglement cost of the sub-path $P_0 \rightarrow P_1 \rightarrow P_2 \rightarrow P_3$ already exceeds the cost threshold, making further exploration unlikely to yield a better path. Meanwhile, FPP discovers a new path $P_0 \rightarrow P_6 \rightarrow P_7 \rightarrow P_8 \rightarrow P_9$. Although this path is longer, its higher average fidelity results in minimal additional purification costs. Ultimately, FPP selects the path $P_0 \rightarrow P_6 \rightarrow P_7 \rightarrow P_8 \rightarrow P_9$ as the final expected S-D path, as it requires fewer Bell pairs and achieves lower overhead. In contrast, the shortest-hop path $P_0 \rightarrow P_4 \rightarrow P_5 \rightarrow P_9$ requires more Bell pairs for necessary purification, contradicting the expectation that the shortest path is always optimal.

It is important to note that the above analysis and examples rely on deterministic fidelity parameters and a deterministic purification model to preplan routing paths and estimate E2E communication costs, thereby determining the optimal allocation configuration. However, during real-time communication in a quantum network, link fidelity may fluctuate due to decoherence, environmental noise, and cross-layer interference. Such variations can cause the actually generated entangled pairs to fall short of the target fidelity assumed during the planning stage. Therefore, when entanglement distribution is executed online, the system continuously monitors the fidelity of each generated entangled pair through the underlying network protocol stack. If the observed fidelity is lower than the target threshold specified by the FPP plan, the system automatically triggers additional purification rounds and reinforces the critical links to restore or improve the entanglement quality.

D. Hybrid Genetic Algorithm

After the analysis in Sections IV-B and IV-C, the communication overhead for generating a remote CNOT gate between any pair of network nodes has been determined. Furthermore, the number of remote CNOT operations required between all pairs of logical qubits is known. Therefore, we reformulate the FQAP as a generalization of the quadratic assignment problem and proceed to solve it. Genetic algorithm (GA) are widely utilized to find optimal solutions to complex optimization problems [26]. This algorithm simulates the Darwinian evolution process by iteratively applying selection, crossover, and mutation to generate the optimal solution. Due to its ability to explore a broad solution space, the GA is well-suited to adapt to the significant expansion of the potential solution set as the number of qubits to be mapped increases.

Firstly, we introduce the encoding designed for the FQAP to describe how mapping relationships are represented within the framework of GA. Each mapping layouts (i.e., a solution to the FQAP) is considered an “individual” in the GA, characterized by its unique features represented in chromosome structure. As shown in Fig. 6, we represent all available data qubits in the quantum computing cluster as a placement sequence of length N_{phy} . (total number of data qubits), assigning a unique, nonrepeating number from 1 to N_{log} . (total number of logical qubits) to each position to indicate which logical qubit is mapped to that position. It is noteworthy that when the number of available data qubits exceeds the number of qubits in the logical circuit, ancilla qubits are used to occupy all remaining data qubits, ensuring that N_{phy} equals N_{log} . Ancilla qubits do

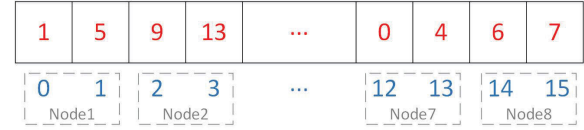


Fig. 6. Representation of mapping layouts in chromosomes. Red numbers denote logical qubit indices, while data qubits are represented by array indices (blue numbers).

not perform any quantum operations and thus do not affect optimization results.

The quality of chromosome structures can be described by the fitness of individuals to their environment, represented by the total overhead generated from mapping relationships, calculated by fitness function formula (5). GA iteratively replace or inherit features of chromosome structures in an attempt to improve the gene until obtaining the solution with the best fitness. In addition to mapping layouts, the fitness function also depends on two $N_{\text{phy}} \times N_{\text{phy}}$ matrices: the communication overhead matrix M_c , which records the communication overheads for remote CNOT operations between each pair of physical qubits, and the remote CNOT gate matrix M_g , which records the number of remote CNOT operations required among all logical qubits.

Algorithm 2 Hybrid Genetic Algorithm

Input: Communication overhead matrix M_c , Remote CNOT gate matrix M_g , Individual numbers N_{indiv} , Iteration numbers N_{iter} , Mutation threshold N_{mutate} .

Output: Mapping layout with minimum overhead.

```

1 pop = GENERATEINITIALPOPULATION();
2 for iter = 0 to  $N_{\text{iter}}$  do
3   while  $\text{childs\_pop.num} < N_{\text{indiv}}/2$  do
4     parents = TOURNAMENTSELECTION(pop);
5     child_indiv =
        UNIFORMLIKECROSSOVER(parents);
6     child_indiv =
        LIMITLOCALSEARCH(child_indiv);
7     childs_pop.APPEND(child_indiv);
8   end
9   pop = UPDATE(pop, childs_pop);
10  if  $\text{pop}[0] < \text{best\_indiv}$  and
     $\text{last\_mutated\_iter} > N_{\text{mutate}}$  then
11    for every pop_indiv in pop do
12      pop_indiv =
        SHIFTMUTATION(pop_indiv);
13    end
14    last_mutated_iter += 1;
15  else
16    best_indiv = pop[0];
17  end
18 end

```

Next, we define the HGA structure used to find the optimal solution, presented in Algorithm 2. HGA employs a tournament strategy to select parent individuals from the current population and performs crossover on parent chromosomes to generate new offspring (Lines 4 and 5 of Algorithm 2).

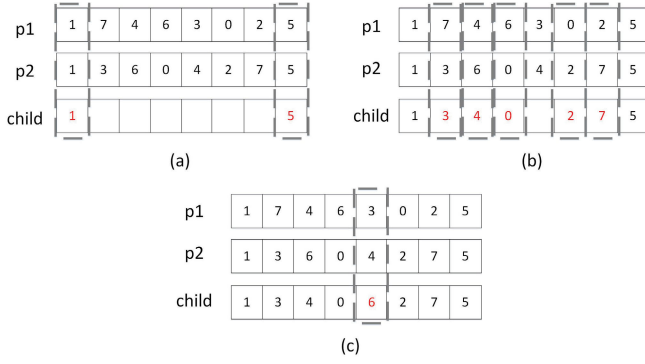


Fig. 7. Example of the ULX process for generating offspring chromosomes. (a) Inherit identical genes from both parents. (b) Randomly inherit a gene from one of the parents. (c) Randomly select and allocate the remaining unassigned genes.

The crossover operator plays a crucial role in optimizing the final performance of HGA by controlling the genes of the new individuals. To effectively balance the challenge of preserving favorable parental characteristics while maintaining diversity among offsprings, HGA uses uniform-like crossover (ULX) [27] to determine the newly generated chromosomes, as illustrated in Fig. 7. ULX follows the priority rules for determining offspring genes: firstly, offspring inherit genes from both parents at positions where they are consistent and identical (i.e., $p1_i = p2_i$ where $p1_i$ and $p2_i$ are values of both parents at position i). These genes are placed in their corresponding positions. Subsequently, in cases where there is no conflict with existing genes, offspring randomly inherit genes from either parent at the same position with equal probability. Finally, any remaining genes are randomly placed in empty positions to form complete chromosomes. ULX crossover operator enhances individual freedom while retaining superior parental characteristics, enabling HGA to effectively maintain population diversity.

Furthermore, we aim to fully exploit the potential of each offspring individual. To achieve this, HGA incorporates limited local search in line 6 of Algorithm 2, which attempts to enhance the offspring's fitness through localized gene exchanges. Initially, limited local search employs the 2-opt operator to iteratively swap genes at adjacent positions within the chromosome, comparing the fitness values before and after the swap. If the swap improves the individual's fitness, limited local search retains the change. Otherwise, limited local search has a certain probability of reversing the swap. Following this, the algorithm applies a predetermined number of perturbations to the chromosome to prevent entrapment in local optima and prepares for another round of local exchanges using the 2-opt operator. This process is repeated for several rounds, ultimately returning the chromosome with the highest fitness found. Through the detailed local search provided by limited local search, HGA can escape potential local optima traps and continuously optimize individuals, thereby improving the exploration of the solution space.

Finally, GAs may stagnate if no new solution spaces are explored over an extended period. To counter this, mutation operators introduce random variations into individuals, enhancing population diversity. In HGA, random

mutation is implemented through segment swapping within the chromosome (Lines 10–14 of Algorithm 2). Specifically, the operator cuts the chromosome at a designated position, reorders the two gene sequences, and then reassembles them to form a new chromosome representation. The initial cut point is chosen at the second gene position of the chromosome, and with each execution of the mutation operator, this cut point is shifted one position further along the chromosome.

E. Complexity and Scalability Analysis

In this work, we propose a HGA for mapping layout optimization, which integrates the standard GA framework with a 2-Opt local search procedure. The mapping problem takes as input an $N_{\text{phy}} \times N_{\text{phy}}$ matrix and optimizes a mapping sequence of length N_{phy} . In the proposed HGA, the selection, crossover, and mutation operators all have controllable linear complexity. Specifically, the tournament selection strategy is executed $|N_{\text{indiv}}|$ times in each generation to form the population, resulting in a complexity of $O(|N_{\text{indiv}}|)$. Both crossover and mutation operate on every chromosome of length N_{phy} in the population, each requiring $O(N_{\text{phy}} \cdot |N_{\text{indiv}}|)$ time. Therefore, the overall complexity of the genetic operators within a single generation is $O(N_{\text{phy}} \cdot |N_{\text{indiv}}|)$. In addition, HGA applies 2-Opt local search to all newly generated individuals. The complexity of 2-Opt on a single chromosome is typically $O(N_{\text{phy}}^2)$, and thus $O(|N_{\text{indiv}}| \cdot N_{\text{phy}}^2)$ for the entire population. Combining the genetic operators and local search, the per-generation complexity of HGA is $O(|N_{\text{indiv}}| \cdot N_{\text{phy}}^2)$. Since HGA performs N_{iter} iterations in total, the overall computational complexity is $O(N_{\text{iter}} \cdot |N_{\text{indiv}}| \cdot N_{\text{phy}}^2)$.

In the proposed HGA, all genetic operators maintain linear complexity, making their computational cost relatively insensitive to the increase in problem size. As the number of physical qubits N_{phy} grows, the 2-Opt local search progressively becomes the dominant computational bottleneck. However, recent engineering advances have significantly improved the practical efficiency of 2-Opt on large-scale instances [28], [29]. With incremental cost evaluation techniques and multi-core or GPU-parallel implementations, 2-Opt can still achieve strong runtime performance and convergence on instances with hundreds of elements. Therefore, the proposed HGA can efficiently handle mapping problems at the scale of hundreds of qubits and has the potential to scale further to even larger systems.

Next, we introduce the FPP algorithm to precompute optimal communication paths between processors and estimate their E2E overhead. FPP first performs a Dijkstra search to obtain the shortest path and its cost threshold, with complexity denoted as T_{Dijkstra} . Based on this, FPP enters a DFS-based constrained path search that enumerates all feasible S-D subpaths from the source node. At each step, the algorithm evaluates whether a path should continue expanding according to the cost formula. Under the cost threshold, DFS prunes overly expensive paths and compresses the effective search depth to d_{eff} . During this process, a node may be revisited multiple times, with corresponding cost updates and state checks, resulting in a total cost of $O(V \cdot d_{\text{eff}} \cdot T_{\text{calc}})$, where T_{calc} is the time required to compute the cost increment of a subpath [determined by (16)]. On the other hand, the expansion

TABLE I
LIST OF SIX DIFFERENT TESTED QUANTUM CIRCUITS

Name	Number of qubits	Number of CNOTs
<i>clip</i>	14	14772
<i>sao2</i>	14	16864
<i>co14</i>	15	7840
<i>dc2</i>	15	4131
<i>ham15</i>	15	3858
<i>mixex1</i>	15	2100

of each edge during the search is governed by the pruning strategy, and an edge is expanded only when the path cost has not yet reached the threshold. We use k to denote the effective path-expansion factor after pruning. The total edge-expansion cost is therefore $O(E \cdot k \cdot T_{\text{calc}})$. Combining node visits and edge expansions, the overall complexity of the constrained DFS search becomes $\tilde{O}(V \cdot d_{\text{eff}} \cdot T_{\text{calc}} + E \cdot k \cdot T_{\text{calc}})$. Thus, the total time complexity of FPP is $T_{\text{Dijkstra}} + \tilde{O}(V \cdot d_{\text{eff}} \cdot T_{\text{calc}} + E \cdot k \cdot T_{\text{calc}})$.

The scalability of FPP depends on how quickly the path cost increases with hop count and the number of feasible paths in the topology. In quantum networks, path fidelity typically deteriorates rapidly as the number of hops increases, which substantially raises purification cost. Leveraging this property, FPP sets the purification cost of the shortest path as the pruning threshold, eliminating a large number of high-cost paths at early stages. This effectively reduces the DFS depth and prevents exponential path explosion. Therefore, the threshold-based pruning mechanism enables FPP to maintain strong scalability in quantum networks.

V. PERFORMANCE ANALYSIS

In this section, we analysis the performance of the proposed method through extensive qubit allocation simulations.

A. Simulation Setting

1) *Tested Quantum Circuits*: In our simulations, the circuits used for testing are divided into two parts: real circuits and random circuits. The real circuits include six different logical quantum circuits listed in Table. I, and their.qasm files can be found in [30]. The random circuits were initialized with a number of logical qubits equal to the computational qubits in the topology, and CNOT gates were executed between randomly chosen pairs of qubits.

2) *Quantum Processor and Network Topology*: In our simulations, each quantum processor on all quantum nodes can accommodate up to two computational qubits and a maximum of four communication qubits. As shown in Fig. 8, we generated nine different network topologies and implemented simulations on these topologies. These include three topologies with eight nodes [22] and six topologies with 16 nodes, with varying numbers of quantum links in each topology.

3) *Parameters Setting*: We set the initial fidelity of Bell pairs generated on different links to follow a Gaussian distribution $\mathcal{N}(0.75, 0.2)$, and each pair of nodes can provide sufficient Bell pairs for entanglement purification [16]. In the quantum network, the overheads are defined as follows: Bell pair generation overhead $C_{\text{ep}} = 7$, entanglement swapping overhead $C_{\text{esp}} = 3$, and fixed overhead for remote CNOT circuits $C_{\text{rcn}} = 5$ [12].

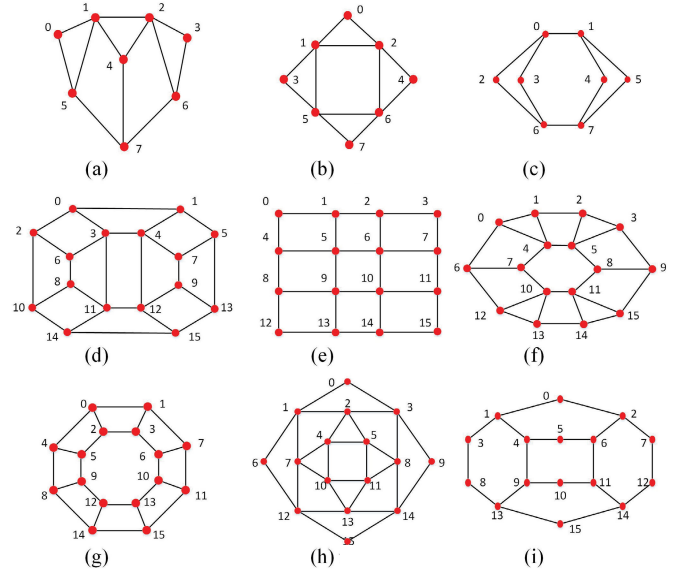


Fig. 8. Different quantum network topologies used for simulations. (a) topo1. (b) topo2. (c) topo3. (d) topo4. (e) topo5. (f) topo6. (g) topo7. (h) topo8. (i) topo9.

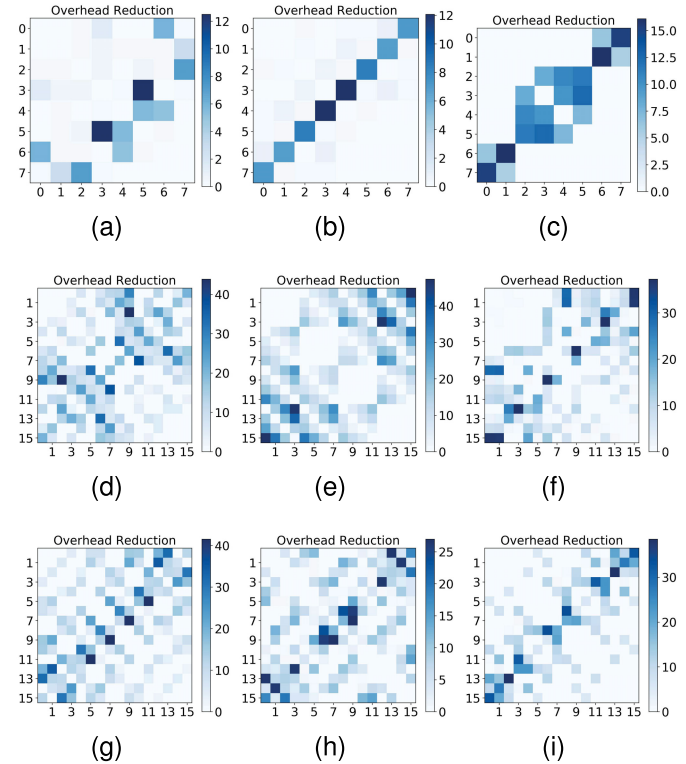


Fig. 9. Performance comparison between the proposed FPP algorithm and the shortest-path algorithm under different network topologies. Each subfigure illustrates the cost difference between the paths selected by the two algorithms. (a) topo1. (b) topo2. (c) topo3. (d) topo4. (e) topo5. (f) topo6. (g) topo7. (h) topo8. (i) topo9.

B. Performance Analysis of FPP

According to the remote CNOT overhead evaluation method described in Section IV-B, we conducted 100 simulation experiments to compare the path-planning performance of FPP and the shortest-path method. For each network topology, both methods were used to plan communication paths for

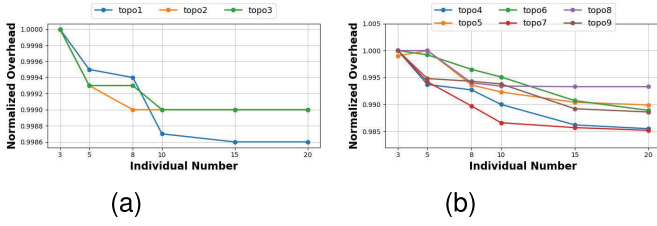


Fig. 10. Performance of HGA with varying population sizes. (a) Eight nodes topology. (b) 16 nodes topology.

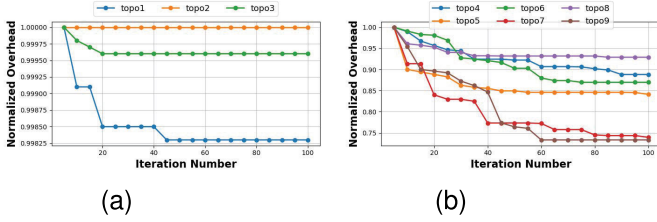


Fig. 11. Performance of HGA with varying iteration numbers. (a) Eight nodes topology. (b) 16 nodes topology.

all node pairs, and their expected communication overhead was compared. Fig. 9 presents the average overhead differences between the two algorithms across various topologies. The results show that the proposed FPP algorithm not only outperforms the shortest-path method in controlling communication overhead but also adapts more effectively to diverse network structures. Moreover, by comparing the remote-communication overhead on 8-node and 16-node computing clusters, we observe that the overhead gap between the two routing schemes becomes increasingly pronounced as the topology scales up, particularly when the inter-node distances are larger.

C. Hyper-Parameters Analysis of HGA

In this section, we conduct a detailed evaluation of the key hyper-parameters in the HGA to analyze how different parameter settings influence its final performance. All comparisons are performed under the same circuit and topology configurations, and the results are reported as the average over 20 independent runs. For ease of observing performance trends, all communication overhead values obtained on different topologies are normalized by the maximum overhead

$$\text{Overhead}_{\text{norm}} = \frac{\text{Overhead}}{\text{Overhead}_{\text{max}}}. \quad (17)$$

We first evaluate how the number of individuals affects HGA performance across all quantum network topologies. On each topology, HGA is executed with [3, 5, 8, 10, 15, 20] individuals, and the results are presented in Fig. 10. The results show that the sensitivity of HGA to population size is closely related to the problem scale. On the small-scale topology with 8 nodes [Fig. 10(a)], increasing the number of individuals leads to only marginal performance improvements, suggesting that a small population is sufficient to explore near-optimal solutions for such problems. In contrast, for the large-scale topology with 16 nodes [Fig. 10(b)], the normalized overhead exhibits a more significant reduction as the

population size increases, indicating that a larger number of individuals improves the algorithm's ability to explore a larger search space and thereby reduces communication overhead. It is worth noting that the performance gain gradually diminishes when the population size reaches 20, implying diminishing returns. Therefore, we set the default population size to $N_{\text{indiv}} = 20$ in all subsequent experiments.

Next, we analyze the impact of the number of iterations on HGA performance, with the results shown in Fig. 11. On the small-scale topology [Fig. 11(a)], HGA requires only a relatively small number of iterations to cover the main solution region. In contrast, on the large-scale topology [Fig. 11(b)], HGA requires a significantly larger number of iterations to reach convergence. This indicates that increasing the network scale raises the problem complexity, thereby necessitating more iterations to effectively explore the larger search space and obtain better solutions. When the number of iterations reaches $N_{\text{iter}} = 100$, the performance improvement becomes marginal. Thus, this value is adopted as the default for all further experiments.

After determining the settings for the population size and number of iterations, we further evaluate the effect of the mutation threshold in HGA. The results show that, given the sufficiently large population and iteration count, HGA maintains stable performance across different mutation thresholds. This indicates that the algorithm preserves a strong exploration capability during the search process and does not suffer from premature convergence. Therefore, the mutation threshold is fixed to 20 in all experiments presented in this article.

D. Performance Analysis of FQAS

In the following experiments, the proposed mapping scheme FQAS is compared against four baseline mapping schemes.

- 1) Graph Partitioning with ShortestPath (GP+Shortest-Path). In this method, communication overhead between processors is estimated using the shortest-path algorithm, after which the graph partitioning model is applied to determine the mapping layout. Graph partitioning is a widely used mapping approach, and many studies [7], [8], [9] have built their models based on graph partition techniques.
- 2) Graph Partitioning with FPP (GP+FPP).
- 3) Multistage Hybrid Simulated Annealing with Shortest-Path (MHSA+ShortestPath). MHSA [12] is one of the state-of-the-art mapping algorithms. In this variant, the shortest-path method is used for path selection.
- 4) Multistage Hybrid Simulated Annealing with FPP (MHSA+FPP).

First, we compare the proposed method with four baseline mapping schemes across six real quantum circuits and three quantum-cluster topologies. The overall overhead produced by different mapping schemes on each topology is shown in Fig. 12. Overall, FQAS achieves the best performance in all tested scenarios, with its E2E communication overhead significantly lower than that of all baseline methods. In comparison, MHSA+FPP performs slightly worse than FQAS but still maintains high solution quality. A limitation of MHSA is that its search is restricted to refining

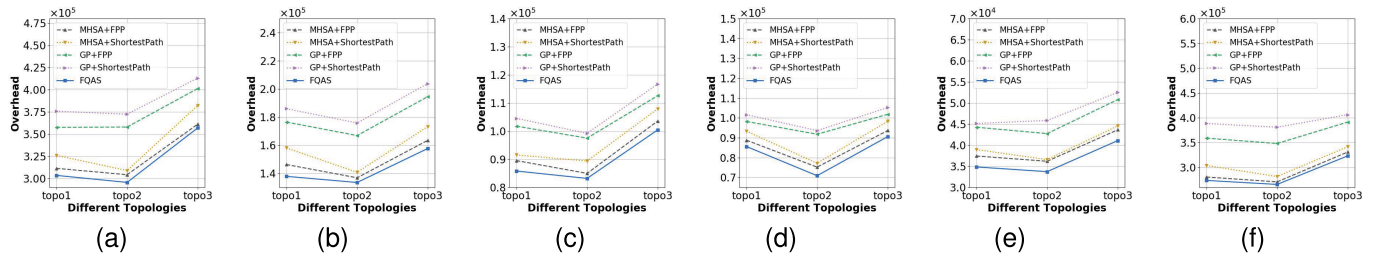


Fig. 12. Performance comparison of the proposed mapping scheme and four baseline schemes on six real quantum circuits. (a) clip. (b) col14. (c) dc2. (d) ham15. (e) misex1. (f) sao2.

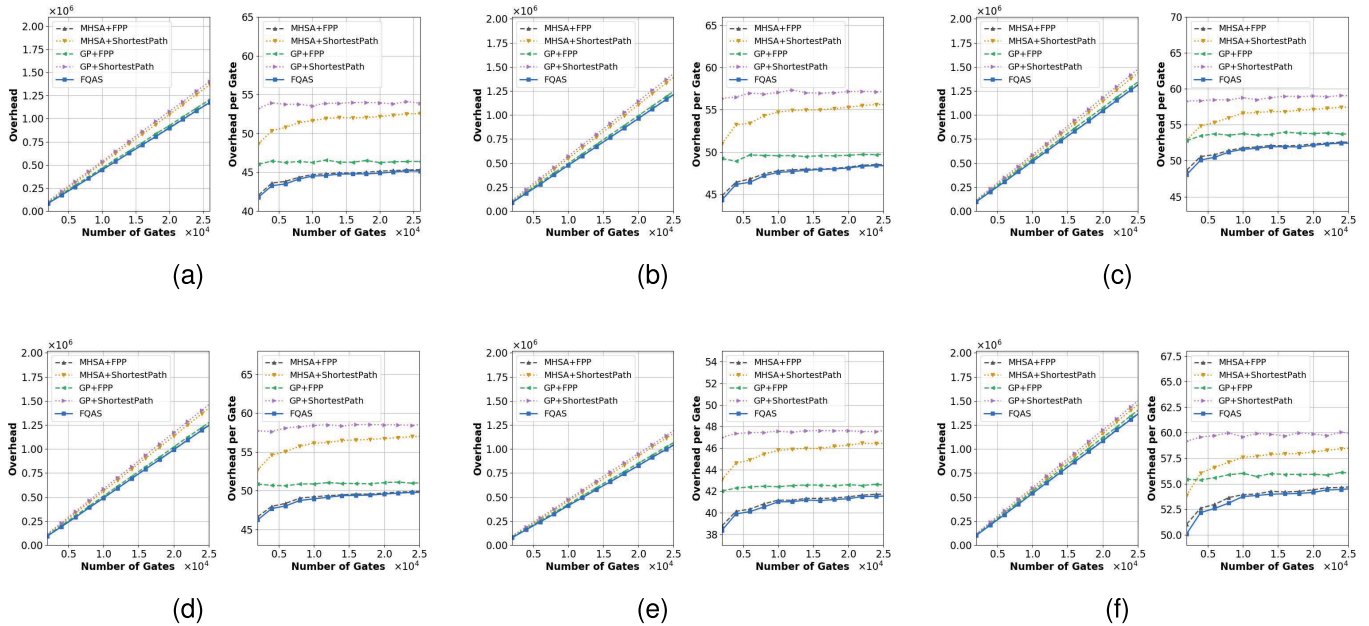


Fig. 13. Performance comparison of the proposed mapping scheme and four baseline schemes with varying numbers of CNOT gates. In each subplot, the left graph shows the total overhead incurred after the allocation process, while the right graph depicts the average overhead per CNOT gate within the circuit. (a) topo4. (b) topo5. (c) topo6. (d) topo7. (e) topo8. (f) topo9.

a single solution, which narrows its exploration space and prevents it from achieving the level of global search capability enabled by HGA's large-scale population-based strategy. As a result, its performance is slightly weaker on complex mapping tasks. Meanwhile, MHSA+ShortestPath performs noticeably worse than MHSA+FPP, revealing that the path-planning module has a substantial impact on the overall mapping quality. Integrating MHSA with the more accurate FPP planner effectively improves mapping performance. In addition, both graph-partitioning methods (GP+ShortestPath and GP+FPP) clearly underperform FQAS and the two MHSA-based baselines. This is mainly because graph-partitioning methods construct and cut graphs based solely on remote-gate interactions, ignoring the inherent overhead heterogeneity across network topologies. Compared with all baseline methods, FQAS achieves an average performance improvement of approximately 12.5%.

After completing the experiments on real quantum circuits of various sizes, we further evaluated the performance of different mapping schemes on large-scale random circuits. These circuits were initialized with the same number of logical qubits as the physical qubits in each topology, and CNOT gates were applied between randomly selected qubit pairs. In our

simulations, the number of CNOT gates ranged from 2000 to 26000. Fig. 13 presents the mapping performance of different schemes across six 16-node topologies. Overall, all methods exhibit approximately linear growth in communication cost as circuit size increases, yet the growth rates differ notably across algorithms. In particular, FQAS consistently achieves the lowest communication overhead across all circuit sizes and topologies, and its curve maintains the smallest slope, indicating superior scalability when handling large-scale circuits. Compared with all other baseline algorithms, FQAS achieves an average reduction of 5.3% in communication overhead.

Finally, to further assess how different E2E fidelity requirements influence mapping performance, we conducted experiments on quantum circuits containing 10000 CNOT gates across six network topologies. Fig. 14 illustrates how the communication overhead of each mapping scheme varies under the same fidelity-threshold constraints. The results show that as the E2E fidelity threshold increases, the communication overhead of all mapping methods rises monotonically. This behavior occurs because higher fidelity requirements force each remote binary gate to undergo more rounds of entanglement purification, which introduces additional communication overhead. This trend remains consistent across all tested

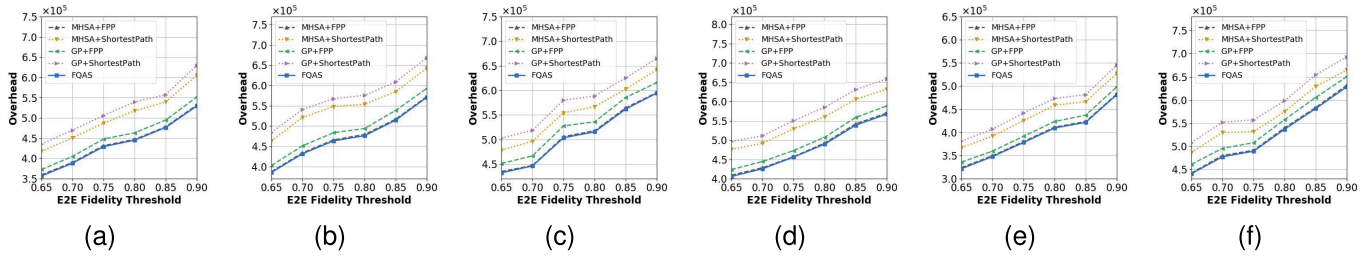


Fig. 14. Performance comparison of the proposed mapping scheme and four baseline mapping schemes with different E2E fidelity requirements. (a) topo4. (b) topo5. (c) topo6. (d) topo7. (e) topo8. (f) topo9.

topologies, indicating that fidelity constraints exert a broadly applicable impact on DQC overhead. Across all topologies and threshold settings, FQAS exhibits the most stable behavior and consistently delivers the best performance. This demonstrates that FQAS is able to provide higher-quality mappings under a wide range of system conditions.

VI. CONCLUSION AND DISCUSSION

In this article, we discuss compilation strategies for DQC. We identify that existing DQC compilers insufficiently account for the impact of varying link fidelities in quantum networks on the execution of remote gates during qubit-to-processor mapping. This factor is essential to the overall performance of distributed quantum programs. To address this problem, we propose a fidelity-aware qubit mapping model and introduce a FPP algorithm. Extensive simulation results show that the proposed approach effectively reduces inter-node communication overhead in DQC while ensuring adequate fidelity for remote communication processes, achieving significant improvements over baseline methods.

However, several challenges still remain for the proposed method. In practical quantum networks, factors such as noise, decoherence, temporal fluctuations in link quality, and cross-node operation errors can affect the entanglement generation process and the final fidelity. These effects may cause the actual execution performance to deviate from the theoretical estimations obtained during compilation. To address this problem, we plan to incorporate key hardware parameters and dynamic link characteristics into the cost modeling process in future work. Our goal is to develop a dynamic cost evaluation framework that reflects time-dependent behaviors and stochastic fluctuations, enabling compilation strategies that better align with realistic quantum network environments. We also aim to conduct experiments on network simulators that employ noise models closer to real hardware in order to further assess the feasibility and effectiveness of the proposed method in practical settings. Another challenge of interest is that, in scenarios involving very large scale quantum processors, engineering optimizations alone may not be sufficient to meet the increasing computational demands. A hierarchical mapping strategy that partitions the global mapping task into local subtasks and integrates them step by step may serve as a promising direction for improving scalability.

REFERENCES

- [1] S. E. Crawford et al., "Quantum sensing for energy applications: Review and perspective," *Adv. Quantum Technol.*, vol. 4, no. 8, Aug. 2021, Art. no. 2100049.
- [2] W. Liang, Y. Li, J. Xu, Z. Qin, D. Zhang, and K.-C. Li, "QoS prediction and adversarial attack protection for distributed services under DLaaS," *IEEE Trans. Comput.*, vol. 73, no. 3, pp. 669–682, Mar. 2024.
- [3] A. Politi, J. C. F. Matthews, and J. L. O'Brien, "Shor's quantum factoring algorithm on a photonic chip," *Science*, vol. 325, no. 5945, p. 1221, Sep. 2009.
- [4] H. J. Kimble, "The quantum Internet," *Nature*, vol. 453, no. 7198, pp. 1023–1030, 2008.
- [5] M. M. Wilde, "Entanglement cost and quantum channel simulation," *Phys. Rev. A, Gen. Phys.*, vol. 98, no. 4, Oct. 2018, Art. no. 042338.
- [6] M. Y. Siraichi, V. F. D. Santos, S. Collange, and F. M. Q. Pereira, "Qubit allocation," in *Proc. Int. Symp. Code Gener. Optim.*, 2018, pp. 113–125.
- [7] P. Andrés-Martínez and C. Heunen, "Automated distribution of quantum circuits via hypergraph partitioning," *Phys. Rev. A, Gen. Phys.*, vol. 100, no. 3, Sep. 2019, Art. no. 032308.
- [8] R. G. Sundaram, H. Gupta, and C. R. Ramakrishnan, "Efficient distribution of quantum circuits," in *Proc. 35th Int. Symp. Distrib. Comput. (DISC)*, 2021, pp. 1–10.
- [9] D. Dadkhah, M. Zomorodi, S. E. Hosseini, P. Plawiak, and X. Zhou, "Reordering and partitioning of distributed quantum circuits," *IEEE Access*, vol. 10, pp. 70329–70341, 2022.
- [10] O. Daei, K. Navi, and M. Zomorodi-Moghadam, "Optimized quantum circuit partitioning," *Int. J. Theor. Phys.*, vol. 59, no. 12, pp. 3804–3820, Dec. 2020.
- [11] Z. Davarzani, M. Zomorodi-Moghadam, M. Houshmand, and M. Nouri-Baygi, "A dynamic programming approach for distributing quantum circuits by bipartite graphs," *Quantum Inf. Process.*, vol. 19, no. 10, pp. 1–18, Oct. 2020.
- [12] Y. Mao, Y. Liu, and Y. Yang, "Qubit allocation for distributed quantum computing," in *Proc. IEEE Conf. Comput. Commun.*, May 2023, pp. 1–10.
- [13] W. Liang, Y. Liu, C. Yang, S. Xie, K. Li, and W. Susilo, "On identity, transaction, and smart contract privacy on permissioned and permissionless blockchain: A comprehensive survey," *ACM Comput. Surv.*, vol. 56, no. 12, pp. 1–35, Dec. 2024.
- [14] D. Ferrari, A. S. Cacciapuoti, M. Amoretti, and M. Caleffi, "Compiler design for distributed quantum computing," *IEEE Trans. Quantum Eng.*, vol. 2, 2021, Art. no. 4100720.
- [15] J.-W. Pan, C. Simon, Č. Brukner, and A. Zeilinger, "Entanglement purification for quantum communication," *Nature*, vol. 410, no. 6832, pp. 1067–1070, Apr. 2001.
- [16] J. Li et al., "Fidelity-guaranteed entanglement routing in quantum networks," *IEEE Trans. Commun.*, vol. 70, no. 10, pp. 6748–6763, Oct. 2022.
- [17] M. Victora, S. Krastanov, A. S. de la Cerda, S. Willis, and P. Narang, "Purification and entanglement routing on quantum networks," 2020, *arXiv:2011.11644*.
- [18] C. Li, T. Li, Y.-X. Liu, and P. Cappellaro, "Effective routing design for remote entanglement generation on quantum networks," *Npj Quantum Inf.*, vol. 7, no. 1, p. 10, Jan. 2021.
- [19] Y. Zhao, G. Zhao, and C. Qiao, "E2E fidelity aware routing and purification for throughput maximization in quantum networks," in *Proc. IEEE Conf. Comput. Commun.*, May 2022, pp. 480–489.
- [20] M. Caleffi et al., "Distributed quantum computing: A survey," 2022, *arXiv:2212.10609*.

- [21] C. Qiao, Y. Zhao, G. Zhao, and H. Xu, "Quantum data networking for distributed quantum computing: Opportunities and challenges," in *Proc. IEEE Conf. Comput. Commun. Workshops*, May 2022, pp. 1–6.
- [22] Y. Mao, Y. Liu, and Y. Yang, "Probability-aware qubit-to-processor mapping in distributed quantum computing," in *Proc. 1st Workshop Quantum Netw. Distrib. Quantum Comput.*, Sep. 2023, pp. 51–56.
- [23] Z. Wang et al., "An efficient scheduling scheme of swapping and purification operations for end-to-end entanglement distribution in quantum networks," *IEEE Trans. Netw. Sci. Eng.*, vol. 11, no. 1, pp. 380–391, Jan. 2024.
- [24] M. Żukowski, A. Zeilinger, M. A. Horne, and A. K. Ekert, "'Event-ready-detectors' bell experiment via entanglement swapping," *Phys. Rev. Lett.*, vol. 71, no. 26, pp. 4287–4290, Dec. 1993.
- [25] R. Zhou, Y. Gan, Y. Liu, K. Obraczka, and C. Qian, "Towards QoS-aware quantum networks," in *Proc. Int. Conf. Quantum Commun., Netw., Comput. (QCNC)*, Jul. 2024, pp. 288–296.
- [26] J. H. Holland, *Adaptation in Natural and Artificial Systems: An Introductory Analysis With Applications to Biology, Control, and Artificial Intelligence*. Cambridge, MA, USA: MIT Press, 1992.
- [27] A. Misevicius, "A fast hybrid genetic algorithm for the quadratic assignment problem," in *Proc. 8th Annu. Conf. Genetic Evol. Comput.*, Jul. 2006, pp. 1257–1264.
- [28] W.-B. Qiao and J.-C. Créput, "Multiple k -opt evaluation multiple k -opt moves with GPU high performance local search to large-scale traveling salesman problems," *Ann. Math. Artif. Intell.*, vol. 88, no. 4, pp. 347–365, Apr. 2020.
- [29] M. A. O'Neil, D. E. Tamir, and M. Burtcher, "A parallel GPU version of the traveling salesman problem," in *Proc. Int. Conf. Parallel Distrib. Process. Techn. Appl. (PDPTA)*, 2011, p. 1.
- [30] A. Zulehner, A. Paller, and R. Wille, "An efficient methodology for mapping quantum circuits to the IBM QX architectures," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 38, no. 7, pp. 1226–1236, Jul. 2019.



Hongding Zhang received the M.S. degree from Guilin University of Electronic Technology, Guilin, China, in 2023. He is currently pursuing the Ph.D. degree with Hunan University of Science and Technology, Xiangtan, China.

His current research interests include quantum computing and quantum networks.



Wei Liang (Senior Member, IEEE) received the Ph.D. degree in computer science and technology from Hunan University, Changsha, China, in 2013.

He was a Post-Doctoral Scholar with Lehigh University, Bethlehem, PA, USA, from 2014 to 2016. He is currently a Professor with the School of Computer Science and Engineering, Hunan University of Science and Technology, Xiangtan, China. He has authored or co-authored more than 220 journal/conference papers, such as IEEE TRANSACTIONS ON COMPUTERS, IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY, IEEE TRANSACTIONS ON SERVICES COMPUTING, IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS, IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS, IEEE TRANSACTIONS ON EMERGING TOPICS IN COMPUTING, IEEE TRANSACTIONS ON COMPUTATIONAL BIOLOGY AND BIOINFORMATICS, and IEEE INTERNET OF THINGS JOURNAL. His research interests include intelligent transportation, security of IoV, blockchain, and embedded systems.

ACTIONS ON COMPUTERS, IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY, IEEE TRANSACTIONS ON SERVICES COMPUTING, IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS, IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS, IEEE TRANSACTIONS ON EMERGING TOPICS IN COMPUTING, IEEE TRANSACTIONS ON COMPUTATIONAL BIOLOGY AND BIOINFORMATICS, and IEEE INTERNET OF THINGS JOURNAL. His research interests include intelligent transportation, security of IoV, blockchain, and embedded systems.



Yuxiang Chen received the Ph.D. degree in computer application from Hunan University, Changsha, China, in 2021.

He was a Post-Doctoral Fellow with the Department of Computing, Hunan University, from July 2021 to March 2023. He is currently an Assistant Professor with Hunan University of Science and Technology, Xiangtan, China. His research interests include network monitoring and federated learning.



Neal N. Xiong (Senior Member, IEEE) received the first Ph.D. degree in software system engineering from Wuhan University, Wuhan, China, in 2007, and the second Ph.D. degree in dependable communication networks from Japan Advanced Institute of Science and Technology, Nomi, Japan, in 2008.

He is currently a Professor with the School of Computer Science and Engineering, Hunan University of Science and Technology, Xiangtan, China. He has published about 300 IEEE journal articles and over 100 international conference papers. His research interests include cloud computing, security and dependability, parallel and distributed computing, networks, and optimization theory.



Sheng Huo received the Ph.D. degree from Hunan University, Changsha, China, in 2019.

His current research interests include network architecture and transportation systems.



Zijian Zhang (Senior Member, IEEE) is currently a Professor with the School of Cyberspace Science and Technology, Beijing Institute of Technology, Beijing, China. He has published more than 100 research papers in international peer-reviewed conferences, transactions, and magazines, including ICML, WWW, AAAI, INFOCOM, IEEE TRANSACTIONS ON DEPENDABLE AND SECURE COMPUTING, IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY, IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, and IEEE TRANSACTIONS ON MOBILE COMPUTING. His current research interests include anonymous communication, blockchain traffic analysis, consensus, and vulnerability detection of smart contracts.

Dr. Zhang won the Best Paper Award on IWQoS'17, SmartCom'22, and ICC'24.

Dr. Zhang won the Best Paper Award on IWQoS'17, SmartCom'22, and ICC'24.



Jie Wu (Fellow, IEEE) received the Ph.D. degree in computer engineering from Florida Atlantic University, Boca Raton, FL, USA, in 1989.

He was the Chair of the Department of Computer and Information Sciences from Summer 2009 to Summer 2016 and an Associate Vice Provost of International Affairs from Fall 2015 to Summer 2017. He is currently the Director of the Center for Networked Computing and a Laura H. Carnell Professor with Temple University, Philadelphia, PA, USA, where he is the Director of International Affairs with the College of Science and Technology. Prior to joining Temple University, he was the Program Director of the National Science Foundation and a Distinguished Professor with Florida Atlantic University. He regularly publishes in scholarly journals, conference proceedings, and books. His current research interests include mobile computing and wireless networks, routing protocols, cloud and green computing, network trust and security, and social network applications.

Affairs with the College of Science and Technology. Prior to joining Temple University, he was the Program Director of the National Science Foundation and a Distinguished Professor with Florida Atlantic University. He regularly publishes in scholarly journals, conference proceedings, and books. His current research interests include mobile computing and wireless networks, routing protocols, cloud and green computing, network trust and security, and social network applications.