Remote Gate Scheduling in Distributed Quantum Computing

Xu Xu*, Yu Liu†, Yingling Mao*, Yuanyuan Yang*
*Department of Electrical and Computer Engineering, Stony Brook University, Stony Brook, USA
†Department of Computing, Hong Kong Polytechnic University, Hong Kong SAR
{xu.xu, yingling.mao, yuanyuan.yang}@stonybrook.edu, yu-y.liu@polyu.edu.hk

***Abstract*—Quantum computing has the potential to outperform classical computing in solving specific problems. However, the limited qubit capacity of existing Quantum Processing Units (QPUs) poses significant barriers to the practical implementation of quantum computing. Distributed quantum computing (DQC) offers a promising approach to scaling the qubit capacity of quantum systems by interconnecting multiple QPUs and enabling collaborative computation. Nevertheless, DQC necessitates implementing remote quantum gate operations that consume entangled qubit pairs, which poses a significant challenge for DQC. In this work, we formulate and investigate the remote gate scheduling (RGS) problem, considering two approaches for remote gate operations: Telegate and Teledata. We propose a hybrid heuristic algorithm that dynamically schedules quantum gate operations within a circuit, executed on distributed QPUs, while minimizing entanglement consumption. We conducted extensive simulations using real-world quantum circuits and processors to evaluate the proposed approach. The results show that our approach reduces entanglement consumption by up to 90% and 25% compared to the two baselines, Telegate-SA and Teledata-ZS, respectively. Furthermore, the execution time of our approach is significantly shorter than that of the baselines.**

I. Introduction

Quantum computing [1] is an emerging computational paradigm. Using principles of quantum mechanics, such as superposition, entanglement, and quantum interference, it surpasses classical computing in solving certain problems. For instance, Shor's algorithm [2], a groundbreaking quantum algorithm, achieves polynomial-time integer factorization, providing an exponential speedup over classical algorithms. This capability presents a significant challenge to cryptographic systems such as RSA [3], a classical encryption algorithm that relies on the difficulty of integer factorization for its security. With the rapid development of quantum hardware, the capacity of Quantum Processing Units (QPUs) has now reached the scale of hundreds, and in some cases, even thousands of qubits. For example, IBM's Eagle QPU is equipped with 127 physical qubits, and its newest Condor QPU has $1,121$ physical qubits [4].

However, even with such advances, executing practical quantum circuits on a single QPU remains impractical. So far, the largest number which has been reliably factored by Shor's algorithm with a physical system is 21 [5], which remains far from practical applications. For example, factoring a large number approximately equal to 2^{1024} using Shor's algorithm requires over $5,000$ perfect logical qubits [6]. This challenge is further aggravated by the inherent noise in current QPUs, which renders computing results far away from the truth. To achieve reliable quantum computing, Quantum Error Correction (QEC) [7] which encodes a single logical qubit using multiple physical qubits, is usually used to fight against noise. Although quantum circuits gain more robustness against noise with QEC, this encoding significantly increases the overall circuit size, making quantum circuit execution even harder. Therefore, existing QPUs with limited qubits are short for real-world applications.

To implement practical quantum circuits, Distributed Quantum Computing (DQC) [8], [9] provides a scalable solution by interconnecting multiple QPUs. This approach enables resource integration across QPUs, potentially overcoming the limitations of existing hardware. By doing so, it provides a promising path forward for overcoming the capacity bottleneck and laying the groundwork for large-scale quantum computing.

The quantum gates in which qubits involved are mapped to the same QPU are known to be local quantum gates. In terms of local quantum gates, there have been sufficient researches on the implementation of local gates inside a single QPU, and they can now be executed easily and precisely. In contrast, due to the physical isolation of QPUs, the remote quantum gate, (i.e. the quantum gates in which qubits involved are located in different QPUs), cannot be easily executed by directly applying the same techniques for local gates. Therefore, although DQC has the potential to scale up to larger circuits, implementing remote gates in QDC presents a significant challenge.

To address the challenge of implementing remote quantum gates, quantum entanglement plays a crucial role. As a unique phenomenon in quantum mechanics, quantum entanglement enables spatially separated qubits to interact and influence each other, forming the foundation of quantum communication. Building on this principle, methods such as Telegate, Teledata, and Cat-entanglement provide distinct approaches for realizing remote quantum gates, paving the way for more sophisticated distributed quantum computing systems.

A remote gate can be directly implemented using Telegate, while Teledata facilitates qubit teleportation between QPUs, and Cat-entanglement enables qubits to be temporarily copied and shared across multiple QPUs with certain constraints.

This work was supported in part by the National Science Foundation under grant numbers CNS-2231040 and CNS-1191278.

However, given that Cat-entanglement is sensitive to local gates [10] and gate decomposition leads to frequent local operations, it becomes inefficient for practical use. Therefore, Cat-entanglement is not considered in this work.

While Telegate and Teledata expand the capabilities of distributed quantum systems, their execution introduces additional quantum gates and is heavily dependent on entanglement. The use of excessive remote quantum gates in distributed quantum computing poses significant challenges.

The additional quantum operations, particularly the creation and maintenance of entangled states, grow rapidly with the scale of the quantum circuit, resulting in substantial resource overhead as the complexity of the quantum system increases. Besides, quantum systems are inherently susceptible to noise, and entanglement is not perfect, excessive entanglement will decrease the circuit fidelity, making output unreliable. What's more, qubits have a finite coherence time. The complex remote gate operations will extend the execution time of quantum circuits. When the execution time exceeds the maximum coherence time of the qubits, the qubits will lose coherence, rendering subsequent operations meaningless.

These challenges highlight the critical issues posed by remote gate implementations in distributed quantum computing. Excessive reliance on entanglements not only results in significant resource consumption but also degrades the fidelity of the quantum circuit and risks qubit decoherence, eventually leading to circuit failure. Therefore, efficiently scheduling remote quantum gates to minimize the use of entanglements in DQC is a critical challenge in the field.

In this paper, we address the Remote Gate Scheduling problem for Distributed Quantum Computing (RGS-DQC). Since local quantum gates within the same QPU have been extensively studied and are relatively straightforward to implement, our research focuses on remote gates across QPUs. The objective of RGS-DQC is to effectively schedule the remote quantum gates to minimize the use of Entanglement.

Unlike telegate, which performs remote gates while maintaining the existing qubit mapping, teledata allows qubits to be dynamically remapped to other QPUs during the execution of a quantum circuit. This capability enables dynamic adjustments to the qubit mapping, thereby reducing remote two-qubit operations. However, changing the qubit mapping through teledata incurs its own cost. Therefore, a wise and proper use of telegate and teledata is critical to addressing the challenges of RGS-QDC.

The remainder of this paper is organized as follows. Section II provides an overview of related works. Section III discusses preliminary concepts in quantum computing. Section IV introduces the system model and formulates the gate scheduling problem. Section V presents our proposed algorithm. Section VI presents the simulation results. Finally, Section VII concludes the paper.

II. Related Works

DQC has emerged as a promising approach for enabling large-scale quantum circuits [8]. One of the primary challenges in DQC is the establishment of entanglement between QPUs [11], [12]. Remote quantum gates between qubits residing in different QPUs require the sharing of Bell states. Numerous studies have focused on link-layer entanglement generation using a variety of quantum technologies [13]–[15]. For example, Stephenson *et al.* demonstrated the generation of Bell states for trapped-ion qubits at an average rate of 182 Hz [13].

Due to the challenges of implementing remote gates, determining a good mapping from the quantum circuit to the real QDC system so that remote gates across different QPUs can be minimized has become one of the most important questions in distributed quantum computing. In [16], Mao *et al.* solve the qubit-mapping problem under different network topologies when using only the Telegate protocol. This paper proposes a multistage hybrid algorithm that combines local search and simulated annealing to find a near-optimal qubit mapping.

In [17], Davarzani *et al.* convert the input quantum circuit into a bipartite graph and use dynamic programming to partition the bipartite graph into K subgraphs, minimizing the communication cost across partitions. In [18], by selecting the CZ gate as the two-qubit gate in the universal gate set, the authors make use of the fact that qubits can be shared across different QPUs by cat-entanglements, proposing a two-step heuristic algorithm, which performs well on random quantum circuits. Although in this work qubits are "shared" across QPUs, the qubit mapping doesn't really change through the execution of a quantum circuit. Unlike [16] and [18], the authors of [10] consider changing the qubit mapping during the circuit execution by leveraging Teledata. They design two different algorithms to execute distributed quantum computing with fewer entanglements. One of them, called "Local-Best", chooses the target QPU by computing the benefits it will bring to the related quantum gates in the near future. The other one, named "Zero-Stitching", considers a set of zero-cost subcircuits and "stitch" them together by dynamic programming. In this paper, "Local-Best" is shown to perform better in random circuits while "Zero-Stitching" performs better in the Quantum Fourier Transform(QFT) circuits, which are fundamental building blocks in some famous quantum algorithms.

III. Quantum Computing Preliminaries

A. Logical and Physical Qubit

In quantum computing, just as the bit is the fundamental unit in classical computing, the qubit (quantum bit) serves as the foundational unit. Logical and physical qubits are essential concepts in theoretical computation and practical implementation, respectively. Logical qubits are idealized qubits used for theoretical analysis in quantum computing. They are assumed to be free from noise and physical constraints, such as decoherence, providing a perfect high-level abstraction of qubits for computation. In contrast, physical qubits are the actual hardware implementations of logical qubits, realized through various approaches such as trapped ions [19], superconducting circuits [20], NV centers [21], and

neutral atoms [22]. Unlike their logical counterparts, physical qubits are subject to environmental noise, decoherence, and other real-world imperfections [23]. Since physical qubits are imperfect and prone to errors [24], error correction codes are used to encode a logical qubit using multiple physical qubits [25], [26], ensuring reliable quantum computation.

Depending on their intrinsic characteristics, physical qubits may have different funtional roles and can be further categorized into data qubits and communication qubits. Data qubits offer longer coherence times, making them suitable for storing logical qubit. In contrast, communication qubits are less suited for storage, but they are well-suited for establishing entanglement across QPUs [27]. In this work, the QPU capacity specifically refers to the number of data qubits allocated for storing logical qubits, excluding communication qubits used for establishing remote entanglement across QPUs. We denote logical qubits by Q_l, data qubits by Q_d and communication qubits byQ_c.

B. Quantum Gate and Quantum Circuit

Quantum gates are quantum operations applied on qubits and are fundamental building blocks of quantum circuits. Based on the number of qubits involved, quantum gates are categorized into single-qubit gates, two-qubit gates, and multi-qubit gates.

A quantum circuit is a concrete representation of a quantum algorithm consisting of a set of qubits and a sequence of quantum gates. A quantum computation is completed by executing a quantum circuit and measuring the resulting quantum states, which provide the final output. Quantum gates that are executed simultaneously form a layer of a quantum circuit, which represents a time-slot in the overall circuit execution where operations are applied on different qubits in parallel. Fig. 1a is an example of quantum circuit with 16 layers and the quantum gates in the same layer are implemented in parallel.

For a quantum circuit, it has three key parameters as follows:

- **Width**: The width represents the total number of qubits used in the quantum circuit.
- **Depth**: The depth indicates the number of time slots in the quantum circuit, i.e., the total number of circuit layers.
- **Size**: The size refers to the total number of quantum gates used in the circuit. Since we focus on two-qubit gates in this work, the size of a quantum circuit is otherwise referred to as the number of two-qubit gates in this work, unless otherwise specified.

For the circuit in Fig. 1a, it has a width of 3, depth of 16 and size of 9. Quantum circuits provide a structured framework for implementing quantum algorithms and are essential for realizing quantum computation in practice.

According to the Quantum Gate Universality Theorem [28], any quantum gate can be decomposed into single-qubit gates and two-qubit Controlled-NOT (CNOT) gates. This decomposition is of significant importance for the practical realization of quantum computation, as single-qubit gates and CNOT gates are relatively easy to implement in multi-qubit quantum systems. Therefore, in this work, we focus on quantum circuits that have already been fully decomposed into single-qubit and CNOT gates, facilitating practical implementation and aligning with the constraints of current quantum computing systems,

C. Quantum Entanglement

Quantum entanglement is a unique and fundamental phenomenon in quantum mechanics, where the states of entangled qubits become inseparably linked. In an entangled state, the state of each qubit cannot be described independently of the other's, regardless of their spatial separation.

For two qubits, their max-entangled states are referred to as **Bell states** or **EPR pairs** [1]. Bell states, expressed in the Dirac notation, are as follows:

$$|\phi^{\pm}\rangle = \frac{1}{\sqrt{2}}(|00\rangle \pm |11\rangle), \quad |\psi^{\pm}\rangle = \frac{1}{\sqrt{2}}(|01\rangle \pm |10\rangle).$$

These entangled quantum states are central to quantum communication protocols and distributed quantum computing. They enable the connection of two spatially separated physical systems, forming the foundation for implementing CNOT gates across QPUs.

D. Quantum Processing Unit

QPU is the core computational component of a quantum computer, responsible for executing quantum circuits. Within a single QPU, the implementation and optimization of quantum gates have been extensively studied, leading to significant advancements in gate fidelity and system stability.

Since intra-QPU quantum operations are well understood, this paper focuses on the more challenging domain of quantum gates across distinctive QPUs, often referred to as **remote quantum gates**. Remote gates are essential for enabling DQC, where quantum circuits span across physically isolated QPUs. These gates leverage quantum communication techniques, such as quantum entanglement, to enable interactions between qubits located on different QPUs, overcoming the limitations of single-QPU systems and paving the way for scalable quantum computation.

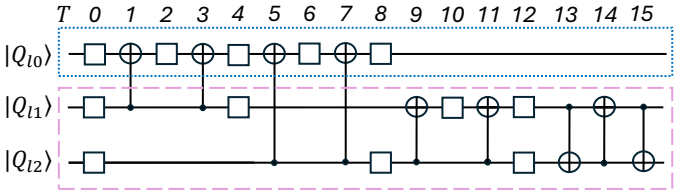
IV. System Model and Problem Formulation

In this section, we introduce the system model for DQC and formulate the remote gate scheduling problem.

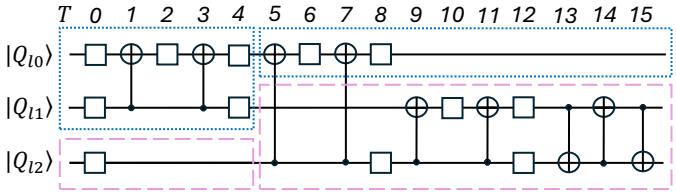
A. System Model for DQC

To address the limitations imposed by the finite number of data qubits in a single QPU, DQC offers a feasible solution. By interconnecting multiple QPUs, DQC enables the execution of large-scale quantum circuits across multiple QPUs, effectively increasing the upper limit on qubit capacity. In this paper, we consider a fully connected QDC system, where the cost of generating entanglement between any pair of QPUs is assumed to be uniform, as proposed in [29], [30] .

In DQC, logical qubits within a quantum circuit are distributed across different QPUs. While this significantly enhances overall qubit capacity, it introduces a critical challenge: qubits participating in the same CNOT gate may reside in

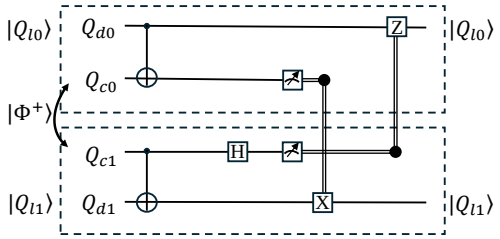


(a) 3-qubit QFT circuit using Telegate protocol of cost 4.

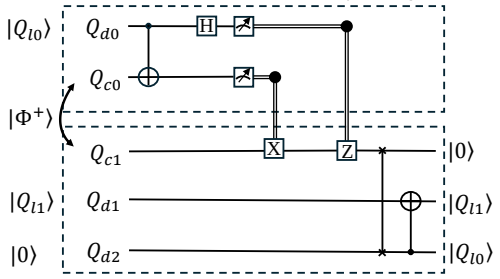


(b) 3-qubit QFT circuit using Hybrid protocol of cost 3.

Fig. 1: Qubit mappings for 3-qubit QFT circuit. Blue and red boxes indicate QPU0 and QPU1 respectively.



(a) Circuit of remote CNOT by Telegate.



(b) Circuit of remote CNOT by Teledata.

Fig. 2: Circuits of two different protocols for remote gates, dashed boxes represent QPUs.

different QPUs, and this leads to remote CNOT gates. In contrast to a local CNOT gate that acts between qubits located within the same QPU, remote CNOT gates cannot be directly implemented through internal QPU operations, posing significant challenges to DQC.

Although remote gates are more challenging to implement than local gates due to the physical isolation between QPUs, quantum entanglement provides an indirect mechanism for realizing remote CNOT operations. Leveraging entangled quantum states, two distinct protocols have been developed to enable remote quantum gate. Fig. 2 is the diagram for the two remote gate protocols. In this diagram, two logical qubit Q_{l0} and Q_{l1} are mapped onto data qubits Q_{d0} and Q_{d1}, which lie on different QPU respectively. A Bell state is shared between communication qubits Q_{c0} and Q_{c1}, with which a remote gate can be implemented through the following routine:

- **Telegate:** Fig. 2a illustrates the realization of a CNOT gate between logical qubits Q_{l0} and Q_{l1} using the Telegate protocol. First, Each QPU performs a local CNOT gate, and a Hadamard gate is applied to Q_{c1}. Measurements are then performed on Q_{c0} and Q_{c1}. The outcome of Q_{c0} is transmitted to Q_{d1} via a classical channel, represented by a solid double line between Q_{c0} and Q_{d1} in the figure. This classical signal determines whether an X gate should be applied to Q_{d1}. Similarly, the measurement result of Q_{c1} dictates whether a Z gate should be applied to Q_{d0}. Finally, the CNOT operation between logical qubits Q_{l0} and Q_{l1} is completed, and their logical mappings remain the same as the beginning.
- **Teledata:** Unlike Telegate, Teledata enables logical qubits to be transferred between QPUs, thereby converting remote gates into local gates by bringing the logical qubits together. Fig. 2b shows how Teledata works and thus how a remote CNOT gate is accomplished, a free data qubit Q_{d2} is initialized to $|0\rangle$. First, a CNOT gate followed by a Hadamard gate is applied to entangle Q_{p0} and Q_{c0}. Subsequent measurements dictate the application of X and Z gates on Q_{d1}. At this point, the logical qubit Q_{l0} has been successfully remapped to Q_{c1}. A SWAP gate (which can be decomposed into three alternating CNOT gates), denoted by a solid line with crosses at each end, then swaps the states of Q_{c1} and Q_{d2} , Q_{l0} is finally remapped to Q_{d2}. Finally, the remote CNOT gate between Q_{d0} and Q_{d1} is transformed into a local CNOT gate between Q_{d2} and Q_{d1}.

Although both protocols enable remote CNOT gate, they come with significant overhead due to the preparation and imperfections of entanglement. The entanglement generation, as estimated in [31], is approximately 300 times more costly than local gate operations. Therefore, minimizing entanglement consumption becomes a critical objective in DQC, as emphasized in previous studies [16], [17].

The Telegate protocol offers a straightforward method for implementing remote gates, without changing the qubit mapping. However, when QPUs have limited qubit capacity and the quantum circuit scales up, the number of remote CNOT gates increases. Since the entanglement cost grows proportionally with the number of remote gates, this leads to significant overhead in executing the circuit within a DQC system. In contrast to Telegate, Teledata allows logical qubits to be dynamically reallocated across QPUs during circuit execution. This not only enables remote CNOT gate operations, but also reduces the number of remote CNOT gates required in the remaining circuit layers.

This work focuses on minimizing the entanglement overhead caused by remote gates in DQC. We propose a hybrid

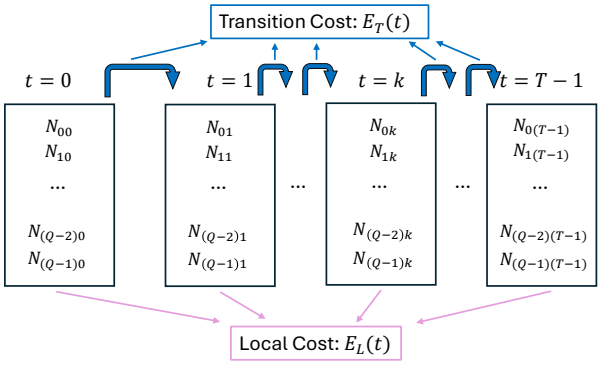


Fig. 3: Mapping matrix for DQC.

protocol combining Telegate and Teledata, taking advantage of the strengths of both approaches. By dynamically selecting between the two protocols during circuit execution, our approach reduces entanglement consumption, which will further shorten circuit execution time, alleviate decoherence effects, and improve overall circuit fidelity.

Some readers familiar with DQC might argue that, in addition to Telegate and Teledata, there is another protocol for implementing remote gates called cat-entanglement [32]. Cat-Entanglement achieves remote gates by sharing an entangled "copy" of the controlling qubit across QPUs, and remote gates can be realized through the "copies" . However, cat-entanglement has some drawbacks. As discussed in [10], any single-qubit gate applied to the controlling qubit will cause the Cat-Entanglement to disentangle, making it hard to preserve. When decomposing a quantum circuit with single gates and CNOT gates, a CNOT gate is often followed by a single-qubit gate. These frequent local gates repeatedly disrupt the cat-entanglement, making it difficult to fully leverage its benefits. Moreover, cat-entanglement across QPUs is susceptible to decoherence, making it difficult to maintain over time and further limiting its effectiveness. For these reasons, the Cat-Entanglement is not considered in this work.

B. Problem Formulation

In this subsection, we will dive into details about the problem formulation of RGS-DQC and will provide a rigorous mathematical model.

Fig.1 illustrates how appropriate gate scheduling using Telegate and Teledata can reduce entanglement usage in a distributed 3-qubit QFT circuit. Fig. 1a is a 3-qubit QFT circuit executed using only Telegate. In this figure, the three logical qubits are mapped onto two QPUs and remain stationary throughout the circuit execution. Since all remote gates are implemented using Telegate, the circuit requires 4 entanglements to realize the 4 remote CNOT gates. At the end of the circuit, the qubit mapping remains the same as at the beginning.

In contrast, the qubit mapping changes during execution and only 3 entanglements are required in Fig. 1b. Between layers 4 and 5 in this diagram, Q_{l1} is remapped to QPU1 by Teledata. This operation significantly benefits the circuit because it converts the CNOT gates in layers 9, 11, 13, 14, and 15 to local gates. Consequently, with the addition of two Telegates for the CNOT gates in layers 5 and 7, the circuit is completed using only 3 entanglements, which is 25% fewer than in Fig. 1a. At the right end of the figure is the final qubit mapping, it is no longer consistent with the initial mapping.

Since CNOT gates are the primary foucs in DQC, we abstract all CNOT gates in a quantum circuit using a structured representation. For any quantum circuit that has been decomposed into a universal gate set consisting of single gates and CNOT gate, we define a circuit matrix $G \in \mathbb{Z}^{Q\times T}$, where Q denotes the circuit width and T denotes the circuit depth. Each entry G_{qt} encodes the gate information for logical qubit q in layer t. Specifically:

- If logical qubit q is involved in a CNOT gate in layer t, and the other logical qubit is $\bar{q}$, then we set $G_{qt} = \bar{q}$ and $G_{\bar{q}t} = q$ to reflect the pairwise nature of CNOT gates.
- If logical qubit q is not involved in a CNOT gate at layer t(i.e., it undergoes a single-qubit operation or no operation), we set $G_{qt} = -1$ to indicate the absence of a CNOT gate.

The circuit matrix G compactly captures all CNOT gate relationships layer by layer. As an example, for layers 0 to 7 of the circuit in Fig. 2, G is given as follows:

$$G = \begin{bmatrix} -1 & 1 & -1 & 1 & -1 & 2 & -1 & 2 \\ -1 & 0 & -1 & 0 & -1 & -1 & -1 & -1 \\ -1 & -1 & -1 & -1 & -1 & 0 & -1 & 0 \end{bmatrix}.$$

To capture the mapping of logical qubits throughout the circuit execution, we define a mapping matrix $N \in \mathbb{Z}^{Q\times T}$, where each entry N_{qt} denotes the QPU to which qubit q is assigned in layer t. The mapping matrix must satisfy the following constraints:

$$N_{qt} \in \{0, 1, \ldots, M-1\}, \forall q \in \mathcal{Q}, \forall t \in \mathcal{T} \tag{1}$$

$$\sum_{m=0}^{M-1} \delta(N_{qt}, m) = 1, \forall q \in \mathcal{Q}, \forall t \in \mathcal{T} \tag{2}$$

$$\sum_{q=0}^{Q-1} \delta(N_{qt}, m) \leq L_m, \forall m \in \mathcal{M}, \forall t \in \mathcal{T} \tag{3}$$

Constraints (1) and (2) ensure that every logical qubit is assigned to and only to one valid QPU at every layer. Constraint (3) ensures that no QPU exceeds its capacity limit L_m at any layer .

As an example, the mapping matrix from layer 0 through 7 in Fig. 2 is given below:

$$N = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}.$$

Before proceeding to the objective function, we first introduce a reversed Kronecker delta function to indicate whether

TABLE I: Notations

Definition	Notation
Number of logical qubits in quantum circuit	Q
Set of logical qubits in quantum circuit, $\mathcal{Q} = \{0, 1, \ldots, Q-1\}$	$\mathcal{Q}$
Number of QPUs used for running quantum circuit	M
Set of QPUs used for running quantum circuit, $\mathcal{M} = \{0, 1, \ldots, M-1\}$	$\mathcal{M}$
Number of layers in quantum circuit	T
Set of layers in quantum circuit, $\mathcal{T} = \{0, 1, \ldots, T-1\}$	$\mathcal{T}$
Mapping of qubit q in layer t	N_{qt}
Capacity of QPU m	L_m
Qubit involved in CNOT gate with qubit q in layer t. Specifically, $G_{qt} = -1$ if qubit q not in CNOT	G_{qt}
Local cost for implementing remote gate by Telegate in layer t	$E_L(t)$
Transition cost for changing mapping from $t-1$ to t	$E_T(t)$

two qubits reside on the same QPU. We define the reversed Kronecker delta function as follows:

$$\delta'(a,b) = \begin{cases} 1, & \text{if } a \neq b, \\ 0, & \text{if } a = b. \end{cases}$$

Next, we construct the objective function with reference to Fig. 3, which provides a visual representation of the mapping matrix. Each column in the figure corresponds to a specific layer, and the entries within the same column represent the qubit mappings at that layer. Once the mapping for a particular layer is determined, any remote CNOT gates within that layer must be executed using Telegate. For any given layer t, we define the number of entanglements used for Telegate as $E_L(t)$, which represents the intra-layer (local) cost. Since each CNOT gate involves a pair of qubits, the entanglement cost per qubit is $\frac{1}{2}$. Therefore, $E_L(t)$ is defined as:

$$E_L(t) = \frac{1}{2} \sum_{q=0, G_{qt} \neq -1}^{Q-1} \delta' \left(N_{qt}, N_{(G_{qt})t} \right). \tag{4}$$

Here $G_{qt} = -1$ is filtered out because it makes $N_{(G_{qt})t}$ meaningless, and the reversed Kronecker delta function is used to indicate whether qubit q in layer t is involved in a remote CNOT gate.

When the mapping of a logical qubit differs across consecutive layers, the transition is realized through the use of Teledata. For example, if $N_{00} = 2$ and $N_{01} = 4$, it means that qubit 0 must be transferred from QPU 2 to QPU 4 between layer 0 and layer 1. This transition is achieved using the Teledata protocol, with the cost of one entanglement each time. Accordingly, for any layer t, the transition cost quantifies the number of entanglements required to transfer qubits from their previous mappings to their current ones. It is defined as:

$$E_T(t) = \sum_{q=0}^{Q-1} \delta' \left(N_{qt}, N_{q(t-1)} \right), \tag{5}$$

where $\delta' \left(N_{qt}, N_{q(t-1)} \right)$ indicates whether a qubit's mapping changes between layers $t-1$ and t.

The objective function of RGS-DQC problem should include the local cost and transition cost at every possible layer, and it can be expressed as:

$$\mathbb{E} = \sum_{t=0}^{T-1} E_L(t) + \sum_{t=1}^{T-1} E_T(t). \tag{6}$$

Now the RGS-DQC becomes an optimization problem aiming to minimize the objective function (6), given circuit matrix G, and subject to the constraints in (1), (2) and (3).

This problem is a nonconvex binary quadratic programming problem, which is generally NP-hard [33] [34]. Also, a similar but simpler problem QA-DQC has been proved to be np-hard in [16]. Since the RGS-DQC cannot be easier than the QA-DQC, the RGS-DQC problem should also be np-hard. Due to space limitations, detailed proofs are omitted.

V. Algorithm Design

Since the RGS-DQC problem is NP-hard, which cannot be solved in polynomial time as believed, we adopt a heuristic algorithm in this section to achieve suboptimal performance.

We first present a theorem that significantly narrows the action space, forming the theoretical basis of our approach. Building on this, we introduce the Pruned Greedy algorithm and the Gate Localization method. To enhance performance, we incorporate a customized initialization strategy for the mapping matrix. Finally, we integrate these components into a hybrid heuristic algorithm, Pruned Greedy–Gate Localization (PG-GL).

As shown in Fig. 3, a mapping matrix has a size of $Q \times T$, and for each element in the matrix, there are at most M choices to consider. Considering there are at most $M^{Q \times T}$ choices, finding the optimal matrix which minimizes the entanglement cost is a difficult task. A naive thought is to employ a greedy strategy to move qubits to the most suitable QPU, aiming to minimize the total cost at the current moment. However, in a DQC system, particularly when QPU capacity is highly limited, a large number of QPUs may be required to ensure adequate computing resources. The increase in the number of QPUs introduces a multitude of choices for the algorithm, significantly increasing its complexity. To address this problem, we propose a theorem demonstrating that only three candidates need to be considered in each step, regardless of the size of the QDC system.

Theorem V.1. *Given a specific qubit q at circuit layer t, and assuming all other mappings are fixed, there are at most three candidate positions to consider for updating the mapping of q at layer t in order to reduce entanglement cost. These candidates are:*

- *The position of q at layer $t{+}1$, if it exists;*
- *The position of q at layer $t{-}1$, if it exists;*
- *The position of the qubit that interacts with q through a CNOT gate at layer t, if applicable.*

The proof of the theorem is omitted due to space constraints.

Algorithm 1: PRUNED_GREEDY(PG)

Input: N, G, $Depth$, $Width$
Output: N
for $t \leftarrow 0$ **to** $Depth - 1$ **do**
  for $q \leftarrow 0$ **to** $Width - 1$ **do**
    $S \leftarrow \emptyset$;
    if $t < Depth - 1$ **then**
      $S \leftarrow S \cup \{N_{q(t+1)}\}$;
    if $t > 0$ **then**
      $S \leftarrow S \cup \{N_{q(t-1)}\}$;
    if $G_{qt} \neq -1$ **then**
      $S \leftarrow S \cup \{N_{G_{qt}t}\}$;
    $Candidate \leftarrow$ MAJORITYVOTE(S);
    if *QPU_Candidate is available in layer* t **then**
      $N_{qt} \leftarrow Candidate$;
return N;

Algorithm 2: GATE_LOCALIZATION(GL)

Input: N, G, $Depth$, $Width$
Output: N
for $t \leftarrow 0$ **to** *Depth* $- 1$ **do**
  for $q \leftarrow 0$ **to** *Width* $- 1$ **do**
    if $G_{qt} = -1$ **then**
      continue;
    $\bar{q} \leftarrow G_{qt}$;
    if $N_{qt} = N_{\bar{q}t}$ **then**
      continue;
    else
      if *QPU_*$N_{\bar{q}t}$ *is available in layer* t **then**
        $N_{qt} \leftarrow N_{\bar{q}t}$;
      else if *QPU_*N_{qt} *is available in layer* t **then**
        $N_{\bar{q}t} \leftarrow N_{qt}$;
return N;

Algorithm 3: PG-GL

Input: G, $Depth$, $Width$, $MaxIter$, $Interval$
Output: N
Initialize N;
for $i \leftarrow 1$ **to** $MaxIter$ **do**
  if $i \bmod Interval \neq 0$ **then**
    $N \leftarrow \text{PG}(N, G, Depth, Width)$;
  else
    $N \leftarrow \text{GL}(N, G, Depth, Width)$;
return N;

A. Pruned Greedy

In the following, we outline a Pruned Greedy(PG) algorithm based on theorem V.1.

Algorithm 1 is the pseudocode for the PG algorithm. We traverse the mapping of every logical qubit in each layer. For each qubit, we do the majority voting for the three candidates and the returned value is the target mapping.

For any q and t, we extract from $\mathbb{E}$ the terms related to them, and denote it as $\mathbb{E}'(q,t)$. In the ideal case where all three candidates are applicable, $\mathbb{E}'(q,t) = \delta'(N_{qt}, N_{q(t-1)}) + \delta'(N_{qt}, N_{q(t+1)}) + \delta'(N_{qt}, N_{G_{qt}t})$. If the candidates are identical, assigning N_{qt} to that value minimizes $\mathbb{E}'(q,t)$ to 0. When two of the candidates are equal and the third is different, setting N_{qt} to the common value reduces $\mathbb{E}'(q,t)$ to 1. In the worst case, where all three candidates differ, assigning N_{qt} to any of them results in $\mathbb{E}'(q,t) = 3$. The case is similar when only one or two candidates are available, the majority vote still selects the mapping that most effectively reduces $\mathbb{E}'$ and thus reduce the total entanglement cost.

Achieving a satisfying mapping in a single round of the PG algorithm may be unrealistic. Therefore, we continue to apply the Pruned Greedy algorithm iteratively. Due to its greedy nature, the objective function, which represents the entanglement cost, will eventually converge to a stable value.

The notable feature of the PG algorithm is its tendency to map each logical qubit to the majority candidate, which guarantees the local optimality of $\mathbb{E}$. However, this feature may pose two potential challenges:

a) The greedy nature of the algorithm makes it prone to being trapped in local optima.
b) If a qubit's mapping remains unchanged across consecutive layers, particularly when the initial mapping is identical across all layers, the algorithm becomes ineffective.

The first challenge prevents us from starting with an approximate optimal solution of the DQA problem which only uses Telegate, because the qubit mapping in the DQA problem remains unchanged during circuit execution. Since the initial mapping of a heuristic algorithm significantly impacts its performance, we must design a tailored initial mapping based on the characteristics of the PG algorithm. The second challenge causes the PG algorithm to quickly converge to a local minimum, leaving most of the solution space unexplored. A common approach to address this is to introduce perturbations. However, since RGS-DQC involves T-layer mappinRGS, the size of the solution space grows exponentially with T. Random perturbations not only explore the solution space inefficiently, but may also cause the solution to deviate significantly from the current optimal solution. Therefore, it is essential to design appropriate perturbations based on the characteristics of the algorithm.

B. Gate Localization

As discussed above, the greedy nature of the algorithm tends to result in convergence to a local optimum, which can be far away from the global optimum. To address this challenge, we introduce a perturbation called Gate Localization(GL) to help

the algorithm jump out of local optima and explore potentially better solutions.

In heuristic algorithms, introducing perturbations is a common approach when dealing with complex problems. However, local optima obtained after iterations, though not good enough, often exhibit structured properties. This means that introducing random perturbations to every qubit in every layer would generate excessive noise, disrupting the structured local optimum and possibly resetting the mapping. This kind of random perturbation negates the previous optimization efforts and is highly inefficient.

In contrast, introducing minimal random perturbations to only a few qubits and layers results in a mapping too close to the prior local optimum, making it likely to get back to the same local optimum.

GL is a well-structured perturbation strategy which balances these two challenges. Algorithm 2 presents the pseudocode for GL. By relocating qubits involved in CNOT gates to the same QPU, GL intentionally disrupts the temporal consistency of the mapping. This introduces moderate perturbations while preserving the benefits of prior optimization. As a result, new local gates are constructed, enabling the PG algorithm to resume exploration from a modified yet structurally related solution.

C. Initial Mapping

Due to the strong correlation between mappings of adjacent layers, the mapping of a logical qubit may propagate across layers. As a result, the PG algorithm tends to preserve temporal consistency in the mapping, making it less inclined to localize remote qubit gates. To solve this problem, for each layer we begin with an initial mapping in which qubits involved in the same CNOT gates are assigned to the same QPU. This approach ensures that the total local cost for the initial mapping satisfies $\sum_{t=0}^{T-1} E_L(t) = 0$, indicating that there are no remote gates in the QDC system. However, the total transition cost, $\sum_{t=1}^{T-1} E_T(t)$, becomes considerable due to frequent mapping changes throughout layers. Also, to make efficient use of limited resources, logical qubits should be preferentially initialized onto already-utilized QPUs, since each QPU in a QDC system represents a valuable asset.

In addition to facilitating the construction of local gates, this initialization introduces variations among the qubit mappings across different layers, thereby preventing the algorithm from becoming trapped in identical layer mappings.

Regarding the initial mapping, a subtle but effective trick is to deliberately reserve an additional data qubit for each QPU. In other words, during the initialization phase, we treat the QPU capacity as $L-1$ instead of L ,leaving at lease one data qubit unassigned. This is crucial because when a QPU in the DQC system is fully occupied, the teledata method fails because there is no data qubit available to store the transferred logical qubits. This strategy leads to notable performance improvements, as further discussed in Section VI-B.

TABLE II: Summary of Tested QFT and RD Circuits

	QFT		RD	
Width	**Depth**	**Size**	**Depth**	**Size**
30	232	915	108	696
40	312	1620	152	1231
50	392	2435	183	1913
60	472	3250	235	2847
70	552	4065	273	3826
80	632	4880	325	5056
90	712	5695	351	6349
100	792	6510	400	7902

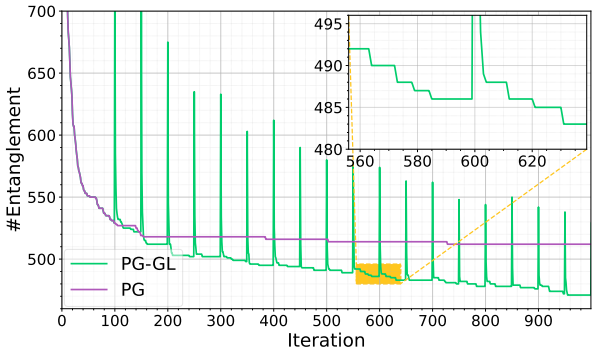


Fig. 4: Convergence progress on QFT100.

D. PG-GL

After initializing the mapping matrix N, we iteratively optimize it using the PG algorithm, interleaved with the GL algorithm. Based on this procedure, we propose the PG-GL algorithm tailored for RGS-QDC. Algorithm 3 is the pseudocode for PG-GL, where $MaxIter$ and $Interval$ denote the total number of iterations and the interval at which the GL algorithm is applied, respectively.

VI. PERFORMANCE EVALUATION

In this section, we provide a comprehensive evaluation of the proposed algorithm's performance.

A. Simulation Settings

We conduct simulations using two types of quantum circuits: QFT Circuits(QFT) and Random circuits (RD). The QFT circuits are fundamental quantum algorithms widely employed in previous studies related to DQC [10], [35], [36]. RD circuits, by contrast, consist of sequences of random quantum operations and are commonly used to evaluate the performance of quantum computers. In particular, Google employed RD circuits to demonstrate Quantum Supremacy [37].

In our simulations, the circuits are decomposed into single-qubit and CNOT gates using Qiskit [38]. To differentiate circuits of varying sizes, we use a naming convention that appends the circuit width to the circuit type (e.g., QFT100 denotes a 100-qubit QFT circuit). Table II provides a detailed summary of the circuits used for testing.

We compare our algorithm with two widely used benchmarks from the literature. The first benchmark, **Telegate by**

Simulated Annealing (Telegate-SA) [16], uses Simulated Annealing to determine the initial qubit mapping, with remote gates implemented exclusively through Telegate. The second benchmark, **Teledata by Zero-Stitching (Teledata-ZS)**, is proposed in [10], which employs the Teledata protocol to implement remote gates.

B. Performance Evaluation

Convergence: Fig. 4 illustrates the convergence behavior of PG and PG-GL when executing QFT100, with the capacity of each QPU set to 25. The performance of PG converges after approximately 150 iterations, indicating that PG can converge quickly. With the incorporation of GL, PG-GL experiences a sudden performance degradation during the GL phase, followed by a rapid recovery to a lower-cost solution. By alternating between the PG and GL processes multiple times, PG-GL is able to find a better solution. In particular, PG-GL achieves approximately a 10% performance improvement compared to PG.

Performance Comparison: Fig. 5 presents the performance of PG-GL and benchmarks across different quantum circuits (RD and QFT) and varying QPU capacities. Fig. 5a–5c demonstrate the performance of the three algorithms under RD circuits. Across different circuit widths and QPU capacities, the performance of PG-GL and Telegate-SA is comparable, with both significantly outperforming Teledata-ZS in terms of entanglement consumption. Notably, PG-GL consistently achieves slightly better performance than Telegate-SA across configurations. Fig. 5d–5f evaluate the three algorithms under the QFT circuits. Unlike the RD case, Telegate-SA now consistently performs the worst. PG-GL shows significantly better performance compared to Teledata-ZS, highlighting its efficiency. Overall, PG-GL demonstrates robust performance across both RD and QFT circuits. In particular, in real-world QFT circuits, PG-GL significantly outperforms the other two algorithms, with up to 90% and 25% improvements over Telegate-SA and Teledata-ZS.

The performance differences observed between RD and QFT circuits can be attributed to their distinct structural properties. RD circuits are randomly generated and lack inherent structure or gate correlations. In such unstructured settings, the Teledata protocol—which relies on circuit regularity to transform remote gates into local interactions by relocating logical qubits—becomes less effective. As a result, the Teledata-ZS method, which stitches together zero-remote-gate subcircuits via Teledata, struggles to yield high-quality solutions.

In contrast, Telegate-SA performs better on RD circuits by applying simulated annealing to optimize initial qubit mappings. Our PG-GL algorithm, which integrates both Teledata and Telegate protocols, consistently outperforms Telegate-SA in this setting by leveraging their complementary strengths.

On the other hand, the structured nature of QFT circuits favors the Teledata-based approach. The regular gate layout in QFT circuits allows Teledata-ZS to construct efficient solutions, while Telegate-SA fails to take advantage of such structural patterns. Notably, PG-GL delivers the best performance across nearly all QFT configurations, demonstrating its adaptability to structured circuits. This observation is particularly important for practical QDC, where real-world quantum algorithms are typically designed with structural coherence in mind.

In summary, the results across both RD and QFT circuits underscore the robustness of PG-GL. In particular, its superior performance on structured circuits like QFT highlights its potential for optimizing real-world quantum applications.

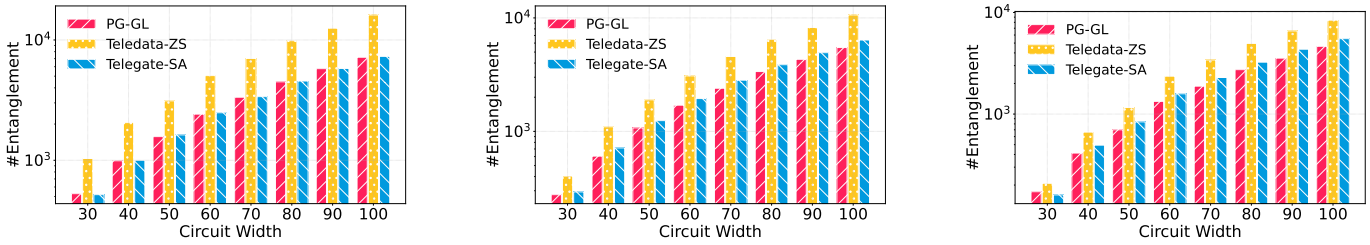
Comparison of Initialization Methods: In Section V-C, we claim that reserving an extra data qubit would improve the performance of PG-GL. To see how this trick benefits the algorithm, we evaluate PG-GL using various QFT circuits with $L = 25$, as shown in Fig. 6. The figure shows the number of entanglements consumed, as the size of QFT circuits increases, more entanglements are saved by leaving extra space at the initialization phase .

Protocol Composition in PG-GL: Thanks to the hybrid protocol adopted by PG-GL, it performs well on both RD and QFT circuits. To better understand the internal composition of the hybrid strategy, we define the Tele-ratio as the proportion of entanglement operations assigned to Telegate relative to the total used by both Telegate and Teledata. This metric reflects how the hybrid protocol adapts under different circuits and QPU configurations. Fig. 7 shows the Tele-ratio across various settings. For RD circuits, the Tele-ratio consistently exceeds 0.3 across different QPU capacities, indicating substantial involvement of Telegate. In contrast, for QFT circuits, Telegate usage remains consistently near zero. These observations further support our earlier conclusion: Telegate is more effective for unstructured circuits, while Teledata is better suited for structured quantum circuits.

Running Time Comparison: By combining PG and GL, PG-GL allows it to efficiently find a good solution for the RGS-DQC problem within a specified number of iterations, achieving this in linear time. Fig. 8 presents the running-time evaluation curves of PG-GL and Teledata-ZS. With a QPU capacity of 25, PG-GL consistently completes execution in a short period, whereas the runtime of Teledata-ZS increases significantly with the scale of the circuit. Fig. 8 shows that the PG-GL algorithm enables fast remote gate scheduling, which is critical for large-scale DQC.

Approximation Ratio: To assess the effectiveness of our algorithm and its proximity to the optimal solutions of the Hybrid and Telegate protocols, we employed the Gurobi solver [39] to compute exact solutions. Since the underlying problem is a 0-1 quadratic program that is NP-hard and non-convex, the solver becomes prohibitively slow as the problem size increases. Therefore, we limited our evaluation to a small-scale instance, QFT20, as shown in Table III.

Despite the relatively small size, the solver still required several days to reach optimality under certain parameter settings. To ensure practical termination, we imposed a precision threshold based on the real-time gap between the incumbent solution and the lower bound. Once the gap fell below



(a) RD, L=5 (b) RD, L=15 (c) RD, L=25

(d) QFT, L=5 (e) QFT, L=15 (f) QFT, L=25

Fig. 5: Performace evaluation of PG-GL and the benchmarks for different quantum circuits under different QPU Capacity.

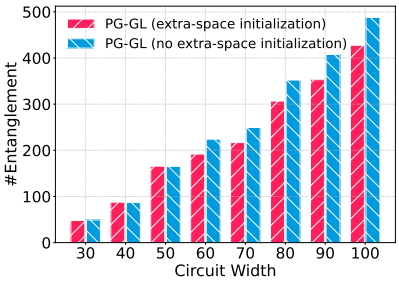


Fig. 6: Performance comparison of different initialization.

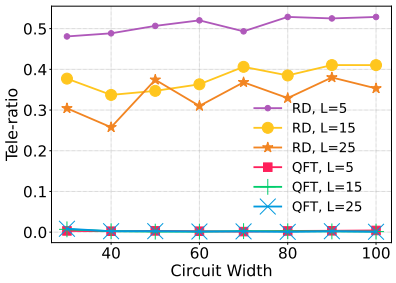


Fig. 7: Tele-ratio of PG-GL under different circuits.

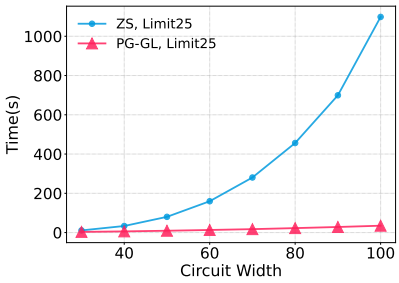


Fig. 8: Running time evaluation.

the threshold, the solver returned the incumbent solution. Although a gap remains, it is worth noting that much of the solver's runtime is spent tightening the lower bound, suggesting the returned solution may be closer to optimal than the gap implies.

Table III shows that PG-GL achieves significantly lower DQC costs than the Teledata-only protocol. In most cases, the solutions obtained by PG-GL are within a factor of two of the optimal Hybrid solutions returned by Gurobi under the specified gaps. These results indicate that PG-GL effectively leverages the strengths of the Hybrid protocol to produce solutions that are reasonably close to optimal.

TABLE III: Comparison of PG-GL and Optimal solution under different QPU capacity limits.

QPU Limit	PG-GL	Optimal-Hybrid	Optimal-Telegate
6	106	67 (80%)	288
8	71	43 (80%)	256
10	60	50 (50%)	200
12	44	24 (0%)	192
14	38	18 (0%)	168

VII. Conclusion

In this paper, we addressed the problem of remote gate scheduling in DQC, aiming to minimize overall communication overhead. Specifically, both Telegate and Teledata protocols were considered for performing remote gate operations. We formulated this problem as a binary quadratic problem and proposed an algorithm named PG-GL to solve it. By greedily exploring the reduced action space and employing Gate Localization as a small perturbation, PG-GL efficiently finds a suboptimal solution for the RGS-DQA problem. Extensive simulations were conducted to evaluate the performance of our proposed algorithm. The results from the distributed execution of QFT circuits show that the entanglement cost of our approach is approximately 90% and 25% lower than those of two popular baselines, Telegate-SA and Teledata-ZS, respectively.

REFERENCES

[1] M. A. Nielsen and I. L. Chuang, *Quantum computation and quantum information*. Cambridge university press, 2010.

[2] P. W. Shor, "Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer," *SIAM review*, vol. 41, no. 2, pp. 303–332, 1999.

[3] R. L. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," *Communications of the ACM*, vol. 21, no. 2, pp. 120–126, 1978.

[4] J. Chow, O. Dial, and J. Gambetta, "Ibm quantum breaks the 100-qubit processor barrier," *IBM Research Blog*, 2021.

[5] E. Martin-Lopez, A. Laing, T. Lawson, R. Alvarez, X.-Q. Zhou, and J. L. O'brien, "Experimental realization of shor's quantum factoring algorithm using qubit recycling," *Nature photonics*, vol. 6, no. 11, pp. 773–776, 2012.

[6] D. Beckman, A. N. Chari, S. Devabhaktuni, and J. Preskill, "Efficient networks for quantum factoring," *Physical Review A*, vol. 54, no. 2, p. 1034, 1996.

[7] B. M. Terhal, "Quantum error correction for quantum memories," *Reviews of Modern Physics*, vol. 87, no. 2, pp. 307–346, 2015.

[8] M. Caleffi, M. Amoretti, D. Ferrari, J. Illiano, A. Manzalini, and A. S. Cacciapuoti, "Distributed quantum computing: a survey," *Computer Networks*, vol. 254, p. 110672, 2024.

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

[10] R. G. Sundaram and H. Gupta, "Distributing quantum circuits using teleportations," in *2023 IEEE International Conference on Quantum Software (QSW)*. IEEE, 2023, pp. 186–192.

[11] A. S. Cacciapuoti, M. Caleffi, F. Tafuri, F. S. Cataliotti, S. Gherardini, and G. Bianchi, "Quantum internet: Networking challenges in distributed quantum computing," *IEEE Network*, vol. 34, no. 1, pp. 137–143, 2020.

[12] D. Cuomo, M. Caleffi, K. Krsulich, F. Tramonto, G. Agliardi, E. Prati, and A. S. Cacciapuoti, "Optimized compiler for distributed quantum computing," *ACM Transactions on Quantum Computing*, vol. 4, no. 2, pp. 1–29, 2023.

[13] L. Stephenson, D. Nadlinger, B. Nichol, S. An, P. Drmota, T. Ballance, K. Thirumalai, J. Goodwin, D. Lucas, and C. Ballance, "High-rate, high-fidelity entanglement of qubits across an elementary quantum network," *Physical review letters*, vol. 124, no. 11, p. 110501, 2020.

[14] P. C. Humphreys, N. Kalb, J. P. Morits, R. N. Schouten, R. F. Vermeulen, D. J. Twitchen, M. Markham, and R. Hanson, "Deterministic delivery of remote entanglement on a quantum network," *Nature*, vol. 558, no. 7709, pp. 268–273, 2018.

[15] S. Krastanov, H. Raniwala, J. Holzgrafe, K. Jacobs, M. Lončar, M. J. Reagor, and D. R. Englund, "Optically heralded entanglement of superconducting systems in quantum networks," *Physical Review Letters*, vol. 127, no. 4, p. 040503, 2021.

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

[17] Z. Davarzani, M. Zomorodi-Moghadam, M. Houshmand, and M. Nouri-Baygi, "A dynamic programming approach for distributing quantum circuits by bipartite graphs," *Quantum Information Processing*, vol. 19, pp. 1–18, 2020.

[18] R. G Sundaram, H. Gupta, and C. Ramakrishnan, "Efficient distribution of quantum circuits," in *35th International Symposium on Distributed Computing (DISC 2021)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2021.

[19] C. D. Bruzewicz, J. Chiaverini, R. McConnell, and J. M. Sage, "Trapped-ion quantum computing: Progress and challenges," *Applied Physics Reviews*, vol. 6, no. 2, 2019.

[20] M. Kjaergaard, M. E. Schwartz, J. Braumüller, P. Krantz, J. I.-J. Wang, S. Gustavsson, and W. D. Oliver, "Superconducting qubits: Current state of play," *Annual Review of Condensed Matter Physics*, vol. 11, no. 1, pp. 369–395, 2020.

[21] L. Childress and R. Hanson, "Diamond nv centers for quantum computing and quantum networks," *MRS bulletin*, vol. 38, no. 2, pp. 134–138, 2013.

[22] T. Graham, Y. Song, J. Scott, C. Poole, L. Phuttitarn, K. Jooya, P. Eichler, X. Jiang, A. Marra, B. Grinkemeyer *et al.*, "Multi-qubit entanglement and algorithms on a neutral-atom quantum computer," *Nature*, vol. 604, no. 7906, pp. 457–462, 2022.

[23] A. Shnirman, Y. Makhlin, and G. Schön, "Noise and decoherence in quantum two-level systems," *Physica Scripta*, vol. 2002, no. T102, p. 147, 2002.

[24] J. M. Martinis, "Qubit metrology for building a fault-tolerant quantum computer," *npj Quantum Information*, vol. 1, no. 1, pp. 1–3, 2015.

[25] A. R. Calderbank and P. W. Shor, "Good quantum error-correcting codes exist," *Physical Review A*, vol. 54, no. 2, p. 1098, 1996.

[26] A. G. Fowler, M. Mariantoni, J. M. Martinis, and A. N. Cleland, "Surface codes: Towards practical large-scale quantum computation," *Physical Review A—Atomic, Molecular, and Optical Physics*, vol. 86, no. 3, p. 032324, 2012.

[27] C. Knaut, A. Suleymanzade, Y.-C. Wei, D. Assumpcao, P.-J. Stas, Y. Huan, B. Machielse, E. Knall, M. Sutula, G. Baranes *et al.*, "Entanglement of nanophotonic quantum memory nodes in a telecom network," *Nature*, vol. 629, no. 8012, pp. 573–578, 2024.

[28] A. Barenco, C. H. Bennett, R. Cleve, D. P. DiVincenzo, N. Margolus, P. Shor, T. Sleator, J. A. Smolin, and H. Weinfurter, "Elementary gates for quantum computation," *Physical review A*, vol. 52, no. 5, p. 3457, 1995.

[29] Y. Liu, Y. Mao, X. Xu, X. Shang, F. Ye, and Y. Yang, "A nonblocking multistage switching network for distributed quantum computing," *IEEE Transactions on Networking*, 2025.

[30] Y. Mao, Y. Liu, X. Xu, X. Shang, and Y. Yang, "Performance analysis of interconnection networks for distributed quantum computing," in *2024 IEEE Global Communications Conference (GLOBECOM)*. IEEE, 2024, pp. 2785–2790.

[31] C. Monroe, R. Raussendorf, A. Ruthven, K. R. Brown, P. Maunz, L.-M. Duan, and J. Kim, "Large-scale modular quantum-computer architecture with atomic memory and photonic interconnects," *Physical Review A*, vol. 89, no. 2, p. 022317, 2014.

[32] A. Yimsiriwattana and S. J. Lomonaco Jr, "Generalized ghz states and distributed quantum computing," *arXiv preprint quant-ph/0402148*, 2004.

[33] S. Sahni, "Computationally related problems," *SIAM Journal on computing*, vol. 3, no. 4, pp. 262–279, 1974.

[34] P. M. Pardalos and S. A. Vavasis, "Quadratic programming with one negative eigenvalue is np-hard," *Journal of Global optimization*, vol. 1, no. 1, pp. 15–22, 1991.

[35] O. Daei, K. Navi, and M. Zomorodi-Moghadam, "Optimized quantum circuit partitioning," *International Journal of Theoretical Physics*, vol. 59, no. 12, pp. 3804–3820, 2020.

[36] P. Andres-Martinez and C. Heunen, "Automated distribution of quantum circuits via hypergraph partitioning," *Physical Review A*, vol. 100, no. 3, p. 032308, 2019.

[37] 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.*, "Quantum supremacy using a programmable superconducting processor," *Nature*, vol. 574, no. 7779, pp. 505–510, 2019.

[38] A. Javadi-Abhari, M. Treinish, K. Krsulich, C. J. Wood, J. Lishman, J. Gacon, S. Martiel, P. D. Nation, L. S. Bishop, A. W. Cross, B. R. Johnson, and J. M. Gambetta, "Quantum computing with qiskit," 2024. [Online]. Available: https://arxiv.org/abs/2405.08810

[39] B. Bixby, "The gurobi optimizer," *Transp. Re-search Part B*, vol. 41, no. 2, pp. 159–178, 2007.