Effective Scheduling for Quantum Data Centers

Zebo Yang[1], Chenliang Tian[2], Raj Jain[2], Ramana Kompella[3], Reza Nejabati[3], Mounir Hamdi[4], Aiman Erbad[5], Hassan Shapourian[6]

[1]Department of Electrical Engineering and Computer Science, Florida Atlantic University, Boca Raton, FL, USA
[2]Department of Computer Science and Engineering, Washington University in St. Louis, St. Louis, MO, USA
[3]Cisco Research, San Jose, CA, USA
[4]College of Science and Engineering, Hamad Bin Khalifa University, Doha, Qatar
[5]College of Engineering, Qatar University, Doha, Qatar
[6]Zyphra Technologies, Palo Alto, CA, USA

***Abstract*—Quantum computing continues to face challenges in scaling both qubit quality and quantity, motivating the shift toward Distributed Quantum Computing (DQC), where multiple interconnected Quantum Processing Units (QPUs) collaboratively solve problems too complex for monolithic systems. In a quantum data center, QPUs distributed across racks can perform remote gate operations by creating entanglement between them, enabling circuit execution through DQC. However, generating end-to-end entangled pairs in a quantum data center often causes congestion and resource contention. To mitigate this, we introduce a resource management framework with efficient schedulers that maximize fidelity-guaranteed throughput while preserving the dependency constraints of sub-circuits derived from the original quantum circuit. We benchmark the scheduling problem using a Mixed-Integer Linear Programming (MILP) model and then propose a set of lightweight schedulers—Least Resource First (LRF) and its extension, Group LRF (GLRF). These lightweight schedulers provide practical, though less optimal, solutions for handling the high volume of remote gates in DQC workloads. Our initial results on a DCell-based quantum data center show that such heuristics perform well under current DQC conditions, balancing runtime and performance while avoiding the heavy overhead of solvers or machine learning approaches, which often struggle to meet the strict timing demands of short-lived entanglement.**

I. Introduction

Recent progress in quantum computing has created a need to interconnect quantum systems, particularly in quantum data centers using optical switches [1]. Distributed Quantum Computing (DQC) enables multiple limited-capacity Quantum Processing Units (QPUs) to collaborate on tasks beyond the capability of a single device, offering exponential performance gains over classical linear scaling. However, DQC introduces challenges like decoherence and unreliable entanglement generation, rendering traditional resource management approaches ineffective. In DQC, quantum circuits are split into sub-circuits across QPUs. While local gates are straightforward, remote gates require inter-QPU end-to-end (E2E) entanglement, leading to congestion and resource contention. To manage this, we propose a resource management framework that respects gate dependencies and maximizes fidelity-guaranteed entangled pair throughput. We benchmark the problem using a Mixed-Integer Linear Programming (MILP) model and introduce lightweight heuristics—Least Resource First (LRF) and Group LRF (GLRF)—that prioritize low-resource requests. These heuristics have the potential to match MILP performance while running much faster, making them practical for large-scale DQC. Our initial simulation results show that such lightweight methods are well-suited for current DQC systems, avoiding the high overhead of solvers or machine-learning-based approaches.

This work was supported in part by Cisco University Research Grant 92627757; and in part by Qatar Research, Development, and Innovation (QRDI) Academic Research Grant ARG01-0501-230053.

II. Entanglement Distribution in DQC

In DQC, qubits within a QPU are categorized as data qubits, used for computation, and communication qubits, used to create inter-QPU entanglement. Fig. 1 shows two ways of executing a remote CNOT gate between QPUs: TeleData and TeleGate [2].

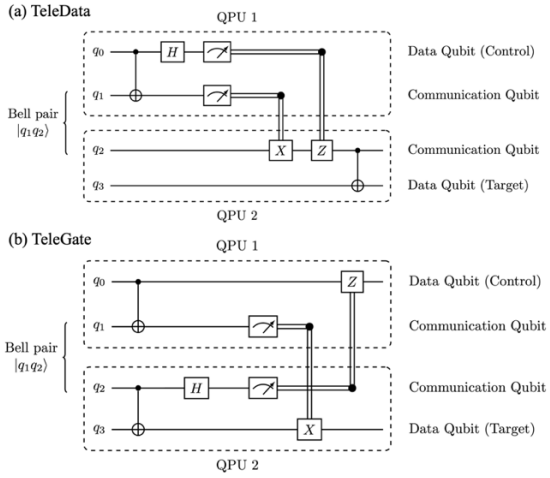


Fig. 1. (a) TeleData or (b) TeleGate for remote CNOT between QPUs 1 and 2.

In TeleData (Fig. 1a), a quantum state (e.g., in q_0 of QPU 1) is teleported to QPU 2 using an entangled pair (e.g., q_1 and q_2), enabling subsequent operations, such as a CNOT with q_3 as the target, to be performed locally on QPU 2. In contrast, TeleGate (Fig. 2b) performs a remote CNOT directly, with q_0 as the control and q_3 as the target, by applying local CNOTs between data and communication qubits on each QPU and using measurement outcomes from QPU 1 to drive operations on QPU 2. Each has its trade-offs: TeleData supports full state transfer but introduces latency, while TeleGate avoids this but requires tighter synchronization. The choice depends on the specific algorithm, latency requirements, and hardware capabilities.

In quantum data centers, both methods rely on an overlay network to distributed E2E entanglement. As the number of remote gates increases, simultaneous entanglement requests must be managed. Prior works often assumed a single request per pair of nodes [3], [4], but real workloads involve many gates, each with different order and QPU placement constraints. This creates dynamic link capacity demands and highlights the need for fair and efficient allocation of entangled resources.

III. Resource Management in DQC

In a quantum data center, at any given time, a batch of remote gate requests R requires E2E entanglement between QPUs, following specific execution orders. These requests may come

Eprint
2025 IEEE International Conference on Quantum Computing and Engineering (QCE), August 2025

from one or multiple sub-circuits, and each request r is defined by its two target QPUs, duration d_r, weight w_r, max hop count h_r^{max}, min fidelity f_r, and a dependency set D_r. Each request can be routed through multiple candidate paths P_r, each with a hop count h_r^{ρ}. All entangled pairs are assumed to share a uniform coherence time t^{co}, within which requests must be completed.

The request management works as follows: 1) Nodes generate entangled pairs with neighbors, forming a temporary topology with link capacities based on entanglement success probability p. 2) The network receives a batch of remote gate requests. 3) For each request r, a limited set of candidate paths P_r is identified (bounded by $|\mathrm{P}_r|^{max}$). 4) The best path and start time for each request are selected. 5) Repeat for the next batch. The challenge lies in step 4: selecting optimal paths and scheduling requests while respecting dependencies and resource limits.

IV. Optimal and Efficient Schedulers

A. Optimization by Mixed Integer Linear Programming

To maximize weighted throughput, the total weight of remote gate requests satisfied with fidelity-guaranteed E2E entangled pairs within one coherence time, we define the MILP problem:

$$\text{maximize} \sum_{\forall r \in R} \sum_{\forall \rho \in \mathrm{P}_r} x_r^{\rho} \cdot w_r \cdot \xi_r^{\rho} \quad (1)$$

$$s.t. \sum_{\forall r \in R} \sum_{\forall \rho \in \mathrm{P}_r : e \in \rho} x_r^{\rho} \cdot c_r^{\rho} \le c_e, \forall e \in E \quad (2)$$

$$s_i + d_i \le s_r, \forall r \in R, \forall i \in D_r \quad (3)$$

$$h_r^{\rho} \le h_r^{max}, \forall \rho \in \mathrm{P}_r, \forall r \in R \quad (4)$$

$$f_r^{\rho} \ge f_r^{min}, \forall \rho \in \mathrm{P}_r, \forall r \in R \quad (5)$$

$$\max_{\forall r \in R}(s_r + d_r) \le t^{co} \quad (6)$$

The objective of the optimization (1) is to maximize weighted throughput, the total weight of requests completed within one coherence time. Higher-weight requests are prioritized with shorter or multiple paths to improve success rates. The decision variables are $x_r^{\rho} \in \{0,1\}$, indicating if path ρ is selected for request r, and s_r, the start time of request r. The entanglement success rate $\xi_r^{\rho} = q^{h_r^{\rho}-1}$ depends on the path length.

Constraints are ensured by (2-6): (2) Link capacities are not exceeded, where c_r^{ρ} is the purification demand on each link, and c_e is the capacity of edge e in the physical topology $G(V, E)$ of the quantum data center. In this graph, each node $v \in V$ represents a QPU, and each edge$e \in E$ represents a link between two QPUs. (3) Request dependencies are respected ($s_i + d_i \le s_r$ if $i \in D_r$). (4) Path lengths do not exceed the maximum allowed. (5) Fidelity of path ρ, f_r^{ρ} meets the minimum threshold f_r^{min}. (6) All requests complete within the coherence time t^{co}.

B. Lightweight Schedulers

To overcome the short-lived nature of entanglement, we propose the LRF algorithm, which selects the lowest-resource path for each request while preserving dependency constraints. By minimizing resource usage per path, LRF reduces system load and avoids bottlenecks, offering fast execution due to its small solution space. To improve performance, we extend LRF to GLRF, which evaluates request pairs jointly (e.g., combinations of two) and selects paths that maximize remaining resources post-allocation. This group-based strategy explores a larger solution space, $C(k, 2)$ combinations for k requests, enabling better resource optimization. Requests are pre-ordered using topological sorting based on dependency graphs. We believe further expanding the heuristics' search space can bring their performance increasingly closer to that of the MILP solver.

V. Simulation Results

We simulate a quantum data center using a 5-cell DCell topology [1], applying the defined problem and schedulers, implemented with NetworkX [5]. We use First Come First Served (FCFS) as a lower-bound benchmark and the CPLEX MILP solver for problem (1) as the upper bound. For simplicity, the direct-link entanglement rate is fixed at 0.8, introducing variability in link capacities during simulation. Default settings are: link capacity = 20, minimum fidelity per request = random between 0.5~1, initial entangled pair fidelity = 0.9, up to 3 paths per request, max 10 hops per path, and 20 requests per batch.

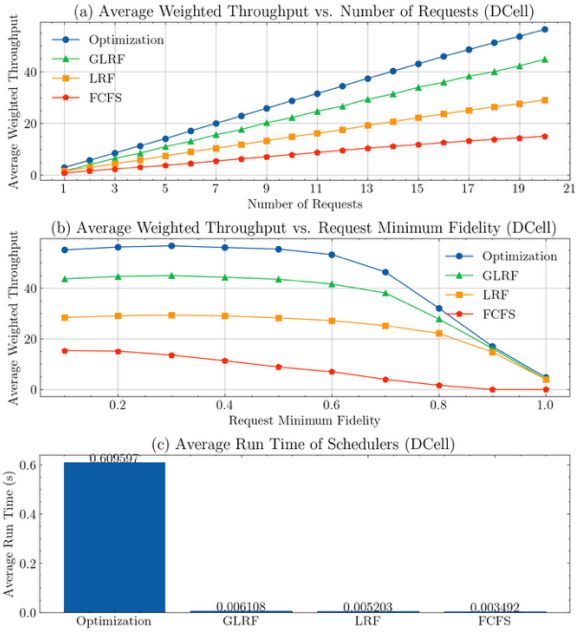


Fig. 2. Scheduler performance across various metrics in DCell topology.

As shown in Fig. 2(a) and (b), the optimization solver yields the best results, LRF and GLRF are less optimal but continue to approach the optimization solver. GLRF outperforms LRF, and both surpass FCFS. In particular, as shown in Fig. 2(c), GLRF, LRF, and FCFS run much faster than the solver, with only a modest increase in runtime from FCFS/LRF to GLRF. This provides a balance between execution time and solution quality.

References

[1] H. Shapourian, E. Kaur, J. Zhao, M. Kilzer, R. Kompella, and R. Nejabati, "Architectures for quantum data center networks," in *Quantum Computing, Communication, and Simulation V*, SPIE, Mar. 2025, pp. 144–146.

[2] M. Caleffi, M. Amoretti, D. Ferrari, J. Illiano, A. Manzalini, and A. S. Cacciapuoti, "Distributed quantum computing: A survey," *Comput. Netw.*, vol. 254, p. 110672, Dec. 2024.

[3] Z. Yang, A. Ghubaish, R. Jain, H. Shapourian, and A. Shabani, "Asynchronous entanglement routing for the quantum internet," *AVS Quantum Sci.*, vol. 6, no. 1, p. 013801, Jan. 2024.

[4] Z. Yang, A. Ghubaish, R. Jain, R. Kompella, and H. Shapourian, "Multi-Tree Quantum Routing in Realistic Topologies," *IEEE Commun. Mag.*, vol. 62, no. 10, pp. 78–85, Oct. 2024.

[5] "NetworkX — NetworkX documentation." Accessed: Jun. 2025. [Online]. Available: https://networkx.org/