Towards Flow Scheduling in A Quantum Data Center

Ruilin Zhou, Yuhang Gan, Yi Liu
University of California, Santa Cruz
Sanat Cruz, CA

ABSTRACT

We propose an architecture for future quantum data centers with the scheduling of quantum computing flows, which integrates quantum processors, quantum switches, and quantum networks. The versatility of these data centers is discussed, highlighting their role in distributed quantum computing and quantum multi-party protocols, which include quantum cryptography and multi-party computation. We address a key challenge within this proposed architecture, namely entanglement flow scheduling which resembles its counterpart in classical data centers. Classical flow scheduling algorithms are examined and benchmarked against our proposed metrics which evaluate fairness and efficiency. We believe the proposed architecture can help to develop quantum data centers that can address the limitation of near-future Noisy Intermediate-Scale Quantum (NISQ) devices. We expect our work can inspire more research on the design of future quantum data centers.

1 INTRODUCTION

Quantum computing system exhibits substantial potential for complex computational tasks due to their unique physical properties. However, practical quantum algorithms typically demand a vast amount of computational resources, specifically quantum bits (qubits). It is commonly assumed that millions of qubits are required according to the current qubits error rates to execute a practical quantum algorithm [15]. Nonetheless, there is a general consensus that we are in the era of Noisy Intermediate-Scale Quantum (NISQ) [18]. Existing quantum computing systems are affected by fabrication processes and physical hardware noise, with individual machines capable of implementing only a few hundred or even a few tens of qubits. Although some companies and research institutions have demonstrated plans to increase the qubit count in the coming years[11, 12], the gap between the current capacity and the required number of qubits is still immense.

To address the need for large-scale quantum algorithms and distributed quantum computing scenarios, the Quantum Data Center (QDC), a computing architecture composed of multiple quantum computers – has been proposed as a promising solution. This architecture provides benefits similar to traditional data centers. Compared to individual quantum computers, the combined distributed execution of multiple quantum computers substantially enhances the computational capacity. Specifically, like traditional distributed systems, the division of a quantum algorithm (typically a large quantum circuit) into different sub-tasks (sub-quantum circuits) can boost task parallelism[1, 13]. Moreover, in the foreseeable future, quantum computers are not seen as a replacement for traditional computers but as a tool for specific scenarios and complex computational tasks. Therefore, we anticipate that quantum computers will not be as readily available and widely used as traditional computers. As such, the "data center-multi tenancy" model is likely to be the most feasible use of quantum computing resources.

For distributed computing tasks and multi-tenant collaboration scenarios, quantum data centers, like traditional data centers, also involve stream scheduling and resource allocation issues. For a single distributed computing task, we need to allocate resources judiciously to maximize computational resource efficiency and optimal performance; for different computing tasks, we need to coordinate resource allocation among various tenants to ensure fair distribution and achieve optimal global resource allocation. Especially considering the issue of multi-party secure computation, we need to think about how to allocate sub-tasks and resources to minimize the collaborative communication costs between specific different tenants.

Therefore, in this paper, we first propose our vision for the architecture of a quantum data center. A quantum data center usually needs a central controller to handle task allocation and resource scheduling, several quantum computers or quantum processors to execute computing tasks, and quantum switches for communication and collaborative computing between different quantum computers. Secondly, we propose descriptions of different quantum computing task characteristics concerning the aforementioned resource scheduling issues and provide corresponding metrics to evaluate the performance of scheduling algorithms for various tasks.

QuNet '23, September 10–14, 2023, New York, NY, USA

ACM ISBN 979-8-4007-0306-5/23/09.
https://doi.org/10.1145/3610251.3610560

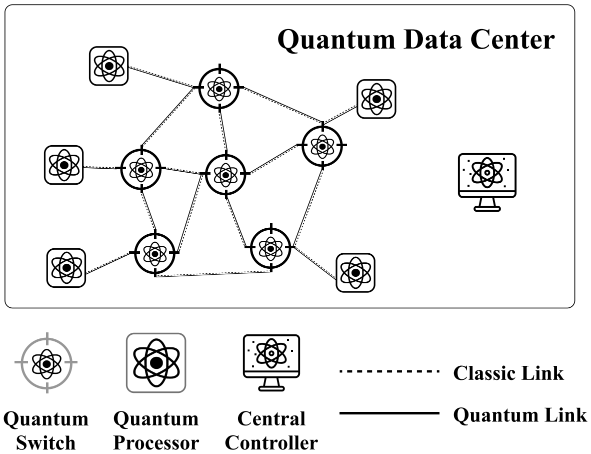

Figure 1: The Quantum Data Center.

Finally, we compare the effects of several common stream scheduling policies for different computing tasks in the proposed quantum data center architecture through extensive experimentation.

The rest of our paper is organized as follows. The definition and model for quantum data center (QDC) are given in Section 2. We introduce some related work about flow scheduling in traditional data centers and propose our observation on these algorithms in Section 5. Then, Section 3 demonstrates the workflow of the quantum data center and some corresponding metrics that need care. After that, we show our simulation results about the performance of different scheduling algorithms under the metric we proposed in Section 4. Finally, we conclude this work in Section 7.

2 DEFINATION AND MODEL

In this section, we introduce our model for quantum data centers including component and topology, quantum application, workflow, and our flow scheduling problem.

2.1 Component and Topology

As shown in 1, a quantum data center (QDC) consists of several key components. The core of the QDC consists of an array of quantum processors, each coupled with a dedicated classical computer that oversees the integration and management of both quantum and classical systems. A quantum network interconnects these processors, embedding quantum switches as nodes to guide entanglement among different quantum processors. Furthermore, a classical central controller plays a crucial role in the QDC, connecting to the quantum processors, and is primarily tasked with supervising operations and enforcing regulations.

Central Controller. The central controller plays an important role as the central management system of classical data centers. Its key responsibilities include efficient resource allocation, wherein it judiciously schedules tasks among quantum processors, optimizing the data center's overall performance. It oversees the intricate process of quantum program compilation, translating high-level quantum algorithms into executable quantum circuits tailored to the specific hardware at hand. Further, it governs job distribution, deciding the optimal quantum processor for each computational task based on current workloads and task requirements. Lastly, it delineates the best paths for establishing remote entanglement between quantum processors and managing quantum switches and networks to enable precise quantum state transfers. The central controller seamlessly weaves together these multifaceted components and functionalities to provide quantum application services efficiently.

Quantum Processor. Quantum processor serves as the basic computing unit in the quantum data center, each quantum processor is also equipped with a classical computer which is used to manipulate and manage the quantum processor, it's also responsible for transmitting classical information with the quantum switch or other quantum processor. Much experiment and theoretical progress have been made in recent years about the solution for interconnecting different quantum processors to address the scaling problem[5, 8, 14]. However, we're currently in the early stages of designing and interconnecting modular quantum processors. Precise manipulation of communication qubits presents significant challenges. As such, we make the assumption that each quantum processor has a limited number of outgoing links, preventing a direct connection between any two quantum processors. Thus the responsibility of connecting different quantum processors falls on quantum operations(entanglement swapping and purification) and quantum switch.

Quantum Switch. In the quantum data center, the quantum switch is responsible for creating end-to-end entanglement between quantum processors in the quantum data center. Compared to the quantum processor, it's equipped with fewer communication qubits and it can perform quantum operations such as entanglement swapping and entanglement purification. What's more, it has multiple interfaces with optical fiber and thus can provide various paths for each quantum processor pair.

Besides components, topology is another important topic in the design of quantum data center. A specific design of topology for the quantum data center is out of the scope of this paper, here we briefly introduce our design consideration: 1) Due to the nature of the quantum network and quantum computing, it will be hard to multiplex a single quantum link, e.g., given a path between two quantum processors, there should not be any other jobs trying to use any link along the path. Thus topology of the quantum data center should provide multipath given any two quantum processors. 2) Building entanglement between two quantum processors is fragile, the probability and the fidelity will both decay exponentially with the number of entanglement swapping. Thus we also need the length of the path given any two quantum processors to be short. 3) unlike classical data center adopting a tree-fashion topology such as fat-tree, the organization in the quantum data center doesn't hold a hierarchy structure, thus a random-fashion topology such as Jellyfish[20] may be more suitable.

2.2 Quantum Application

Distributed Quantum Computing The principal function of a quantum data center lies in executing large-scale quantum computing tasks. This is akin to how traditional data centers use computing clusters to tackle extensive computational loads. However, carrying out quantum algorithms on a solitary quantum processor presents substantial challenges. Quantum algorithms that confer quantum advantage often require a significant number of qubits. Scaling the number of qubits on a single processor, however, is hindered by a myriad of hardware-level constraints, such as fabrication difficulties and various forms of errors. Compounding this issue is the necessity for quantum error correction, which demands the allocation of additional physical qubits. This further reduces the count of actual logical qubits available on a single processor, amplifying the complexity of implementing large-scale quantum algorithms. However, another option is to integrate multiple quantum processors to solve the scalability problem of quantum computing. In IBM's quantum computing roadmap, it also envisions that the future quantum processor will combine multiple-chip and adopt quantum communication techniques. Besides physical experiment advances, compiling for distributed quantum programs has drawn more and more attention in recent years[2, 6, 7, 24].

Multi-Party Quantum Protocols A quantum data center is not solely purposed for quantum computing tasks; it can also serve as a robust platform to execute various quantum cryptographic protocols. These protocols leverage quantum mechanics to provide enhanced security and privacy in data communications, a distinct advantage over their classical counterparts. Two-party quantum protocols, such as quantum oblivious transfer[4] and quantum secret sharing[10], facilitate secure information transfer and sharing between entities. Additionally, quantum data centers can accommodate multi-party quantum protocols. These include quantum digital signatures[3], which provide non-forgeable authentication and quantum multi-party computation[22], enabling secure joint computation over private inputs from multiple parties. By offering these quantum services, quantum data centers significantly bolster the security and privacy aspects

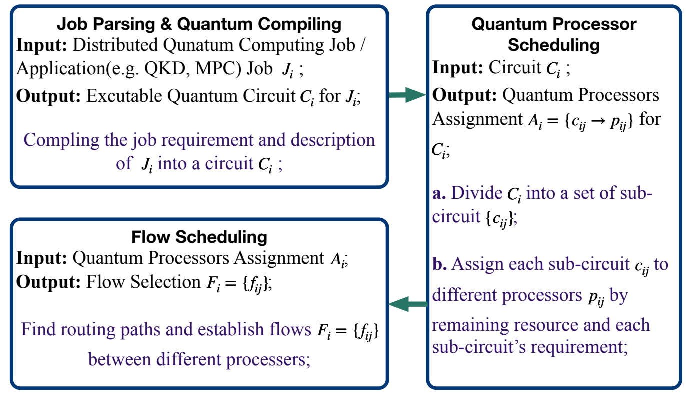

Figure 2: QDC Workflow

of data communication and processing. While these applications can perform classical tasks, their quantum-enhanced versions bring a new layer of security and may find potential application in cross-platform financial settings, among others.

3 METHODOLOGY

In this section, we briefly the workflow of the quantum data center, then we give a formal definition of the entanglement flow scheduling problem and our metrics for evaluating the system.

3.1 Workflow

In this section, we delineate the workflow within a quantum data center, emphasizing the synchronized and time-slotted operations of different quantum processors. We adopt this model due to two major considerations. First, executing distributed quantum computations necessitates nanosecond-precision synchronization for operations like remote entanglement and gate fidelity. Second, certain cryptographic protocols demand temporal coordination, where both parties must agree on the timelines for transmitting and receiving quantum bits. As shown in the figure, the job execution process encompasses the following phases:

Job Parsing and Quantum Compiling. At the beginning of each time slot, the jobs, furnished with their metadata, such as application type, number of 'shots', and fidelity requirements, are dispatched to the central controller. Here, each job J_i undergoes parsing to understand its details and requirements. Subsequently, high-level quantum programs are compiled into physically executable quantum circuits in a process called quantum compiling. This process involves partitioning the original quantum circuit and mapping inputs to physical qubits. Several specialized compilers have been proposed for handling distributed quantum computing tasks.

Quantum Processor Scheduling. The central controller assigns the compiled sub-circuit to the appropriate quantum processors, a step akin to classical data center scheduling. This decision is guided by the quantum circuit's specific requirements and possibly other factors such as priority.

Flow Scheduling. With the quantum circuit broken down into a sequence of operations, we now move to the flow scheduling phase. This includes handling requests for remote entanglement involving distant quantum processors. Each request is evaluated based on the number of remote entanglements, fidelity requirements, and deadline constraints. For each request, the central controller selects a path or multi paths, determines how many qubits will be allocated along the chosen path, and how many entanglements will be generated per time slot.

3.2 Flow Scheduling in Quantum Data Center

In this section, we first introduce our observation and motivation for the flow scheduling in quantum data centers and then give a definition of it.

Motivation. Building an end-to-end entanglement between quantum processors is a probabilistic process, involving entanglement swapping at intermediate quantum switches which also possess limited quantum resources in terms of quantum qubits. This limitation, combined with the resource-consuming nature of establishing entanglements, creates resource contention similar to network congestion in classical data centers. As one entanglement creation process takes place, it essentially preempts the available resources along its path, thereby restricting simultaneous entanglement establishment. This can be an analogy to a classical data center where each switch only has a very limited bandwidth and in a highly demanding setting, the contention among different jobs will exist and thus will cause the whole system to downgrade. Additionally, each job submitted to a quantum data center comes with unique requirements such as deadline or fidelity. The variability in these demands calls for a smart and efficient resource allocation mechanism. Hence, these considerations call for a mechanism to efficiently allocate resources, mitigate contention and at the same time improve overall performance such as the throughput and flow completion time.

Problem Formulation. Here, we introduced our definition of the entanglement flow scheduling problem. After step 2, the output is a list of tuples where each element denotes a flow and can be defined as f_i=(D_i,T_i,N_i,F_i), D_i is one source-destination pair (s_i, d_i) and $s_i, d_i \in V$, T_i is a positive number representing the minimum number of time-slot this request should be fulfilled, N_i is a positive number representing the number of entangled pair, for each flow f_i, the central controller will compute a path p_i along which entanglement swapping will be performed, and entanglement routing problem has been well studied in previous work[9, 16, 19]. Given the paths for each flow, the central controller will determine the order of resource allocation at each time slot, and for each time slot how many qubits it will deliver.

3.3 Classical Scheduling Policy

Here, we introduce the scheduling policy we used and evaluated in our work.

Shortest Flow First. Shortest Flow First(SFF) prioritizes jobs or flows based on their size, with smaller ones being given priority. The main goal of SFF is to minimize flow completion time (FCT), especially for short flows, which are often associated with latency-sensitive applications. SFF in a classical setting, is a provable optimized algorithm with a heavy-tailed distribution[23]. In our quantum data center setting, we define the shortest flow as the flow with a minimum size of entanglements, and if there is a tie we define a flow with the shortest path as the smallest flow.

Earliest Deadline First. The earliest deadline first(EDF), suggested by the name, in a classical setting, employs priority class based on per-pop packet deadline and allocates all resources such as bandwidth to a flow with the earliest deadline. And in our setting, we will always compute the path of the earliest deadline flow first, and then allocate all remaining qubits along a path to the flow.

First in First Out. FIFO schedules jobs based on their arrival times, with earlier-arriving jobs given priority. In classical data center research, FIFO outperforms other algorithms when the flows have a similar size[23]. Another characteristic is that it demonstrates randomness in average Flow Completion Time(FCT), and it can be considered a 'fair' scheduling policy than others.

Highest Expected Throughput First. The highest Expected Throughput First(HEFT) can be considered as the scheduling policy used in [19] where it works in a greedy fashion, and always schedules and allocates the flow that has the maximum expected throughput first.

4 EVALUATION

In this section, we evaluate the performance of the classical scheduling algorithm on the entanglement scheduling algorithm.

4.1 Evaluation Setting

In this work, we do not assume any specific topology and the network topologies are randomly generated, and we set both the probability of entanglement generation rate and swapping success rate to be 0.6. The number of communication qubits at each quantum switch is randomly picked from 10 to

14, and we let the number of quantum processors be 30. After the quantum compiling and processor scheduling step, each remote entanglement flow is denoted by a tuple (D, T, N, F), we generate users' requests as follows: src-dst pair is picked randomly from the topology. The latency requirement T is uniformly picked from 1 to 5. The throughput requirement N is uniformly picked from 1 to 10. The fidelity requirement is randomly picked from 0.7 to 0.999. At each time slot, we generate 10 flows, and let the system run 100 time slots. For the routing algorithm, we adopted a heuristic based on Q-CAST proposed in [19] where besides maximizing the expected throughput we also add check the fidelity constraint. For our default setting, we run 10 times where each run we let our simulation run 100 time slots, and for each time slot, we generate 20 flow requests and get the average of our metric.

4.2 Metrics

Here we introduce the metrics we use to evaluate the performance of our system.

- Completion Ratio: Completion Ratio (CR) is defined by the size of finished flow divided by total flow, in general, the completion ratio can reflect our scheduling algorithm's performance.
- Average Flow Completion Time (AFCT) is the mean time taken from the start to the finish of all data flows, which are the number of entangled pair built from a source to a destination.It can also be used to evaluate the efficiency of scheduling algorithms, as well as the overall network performance.
- Variance of Flow Completion Time (VFCT): Variance of Flow Completion Time (VFCT) is the variance in the times taken for different data flows in a system. This can provide insights into the fairness of resource allocation in the network's scheduling policy.
- Throughput: In our quantum data center setting, the throughput is defined by the number of entangled pair built from all flows, it can also reflect the performance of the system.

4.3 Evaluation Results

The performance results of various scheduling policies under the default setting are summarized in Fig **??**. Among the considered strategies, HEFT outperforms others in terms of completion rate (CR) and throughput, as it prioritizes flows with the most entanglements, thus maximizing the expected throughput as indicated in [19]. Conversely, EDF, despite its efficiency in reducing the average completion time (ACFT) by prioritizing urgent tasks, exhibits the lowest CR and throughput due to an increased probability of failure associated with the short-deadline-first approach. The SFT policy also shows good performance in CR by focusing on

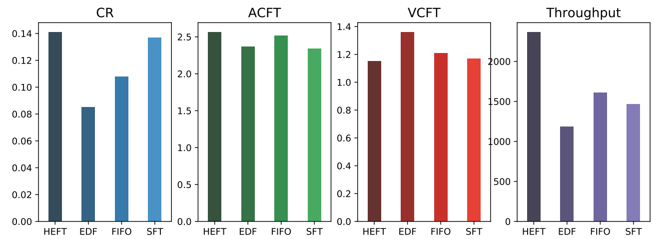

Figure 3: Evaluation of different scheduling methods

completing the 'easiest' tasks first, thus requiring minimal resource allocation. As for the variance in completion time (VCFT), both HEFT and SFT exhibit low values since they tend to complete most flows within 1-2 slots. On the other hand, EDF's VCFT is relatively high because it often handles the 'hardest' flows first, resulting in a random task completion pattern. These insights provide a comprehensive view of how different scheduling policies perform under the default setting.

5 RELATED WORK

Over the past decade, traffic scheduling in Data Center Networks (DCN) has been a key research field. Here we summarized some important works, for a more comprehensive review, we refer to [23] In 2009, Isard and others proposed Quincy, a new graph-based framework for cluster scheduling under a resource-sharing model. They mapped the fair-scheduling problem for cluster computing and the classical problem of min-cost flow in a directed graph, suggesting the fairness between jobs. Following this, Al-Fares and others proposed Hedera. Hedera aims to distribute network flows to non-conflicting paths. In particular, it uses a central controller to handle multiple flows in an uplink or downlink while satisfying the combined bandwidth requirements. Besides, Orchestra is another centralized scheduler that manages both intra- and inter-transmission activities. In addition to centralized scheduling algorithms, there are also some decentralized flow scheduling algorithms and protocols proposed. Hong *etal*. proposed Preemptive-Distributed Quick (PDQ), where flows with high priority can be operated first by pausing others. The high-level objective of a PDQ switch is to allow the most critical flow to complete as soon as possible. pFabric is also a decentralized scheduler. It demonstrates nearly optimal efficiency by decoupling flow scheduling from rate control. When designing a quantum data center, a natural question to ask is how to perform quantum computing in a distributed fashion or how to partition quantum algorithms. Distributed quantum compiler design with consideration on communications have been studied in [7, 24]. Other compilers design[17, 21] try to partition a large quantum circuit into smaller ones and run on different quantum processors.

These works can also be deployed in our quantum data center setting.

6 FUTURE WORK

There could be a large amount of possible future work on the topic of the design and flow scheduling of quantum data centers. Here we identify several possible future works: 1) Investigating the topology design of a quantum data center presents a fascinating area of exploration, particularly in understanding the desirable properties such a topology should exhibit. And a similar direction can be the organization and structure of quantum data center architecture 2) As shown in our evaluation, the completion rate is still low due to the limited path, so it will be natural to think about how to incorporate quantum purification and the use of quantum memory into our work. 3) In our design, our framework does not provide any guarantee of service, thus how to incorporate QoS functionality could also be interesting. 4) In our work, we provide a very high-level abstract of quantum operation and quantum hardware, a more precise model of the quantum data center can also be studied, such as a model of how fidelity decay with time.

7 CONCLUSION

In this work, we give an architecture of the quantum data center with the scheduling of quantum computing flows, which includes the components it needs and the quantum application it can provide. We also introduce design considerations such as the topology of it. We formulate a flow scheduling problem in a quantum data center and evaluate several flow scheduling policies in the classical data center on our metrics. We hope this work can inspire more novel designs and research on future quantum data centers.

ACKNOWLEDGMENT

The authors were partially supported by NSF Grants 1750704, 1932447, and 2114113, and DoE Grant DE-SC0022069. We thank the anonymous reviewers for their comments.

REFERENCES

[1] Pablo Andres-Martinez and Chris Heunen. 2019. Automated distribution of quantum circuits via hypergraph partitioning. *Physical Review A* 100, 3 (2019), 032308.

[2] Jonathan M Baker, Casey Duckering, Alexander Hoover, and Frederic T Chong. 2020. Time-sliced quantum circuit partitioning for modular architectures. In *Proceedings of the 17th ACM International Conference on Computing Frontiers*. 98–107.

[3] Howard Barnum, Claude Crépeau, Daniel Gottesman, Adam Smith, and Alain Tapp. 2002. Authentication of quantum messages. In *The 43rd Annual IEEE Symposium on Foundations of Computer Science, 2002. Proceedings.* IEEE, 449–458.

[4] Charles H Bennett, Gilles Brassard, Claude Crépeau, and Marie-Hélene Skubiszewska. 2001. Practical quantum oblivious transfer. In *Advances in Cryptology—CRYPTO’91: Proceedings*. Springer, 351–366.

[5] Jerry Chow, Oliver Dial, and Jay Gambetta. 2021. IBM Quantum breaks the 100-qubit processor barrier. *IBM Research Blog* (2021).

[6] Davood Dadkhah, Mariam Zomorodi, Seyed Ebrahim Hosseini, Pawel Plawiak, and Xujuan Zhou. 2022. Reordering and partitioning of distributed quantum circuits. *IEEE Access* 10 (2022), 70329–70341.

[7] Davide Ferrari, Angela Sara Cacciapuoti, Michele Amoretti, and Marcello Caleffi. 2021. Compiler design for distributed quantum computing. *IEEE Transactions on Quantum Engineering* 2 (2021), 1–20.

[8] MF Gonzalez-Zalba, S de Franceschi, E Charbon, T Meunier, M Vinet, and AS Dzurak. 2021. Scaling silicon-based quantum computing using CMOS technology. *Nature Electronics* 4, 12 (2021), 872–884.

[9] Laszlo Gyongyosi and Sandor Imre. 2017. Entanglement-gradient routing for quantum networks. *Scientific reports* 7, 1 (2017), 14255.

[10] Mark Hillery, Vladimír Bužek, and André Berthiaume. 1999. Quantum secret sharing. *Physical Review A* 59, 3 (1999), 1829.

[11] IBM. 2023. The IBM Quantum Development Roadmap. (2023). https://www.ibm.com/quantum/roadmap

[12] IonQ. 2020. Scaling IonQ’s Quantum Computers: The Roadmap. (2020). https://ionq.com/posts/december-09-2020-scaling-quantum-computer-roadmap

[13] Gushu Li, Yufei Ding, and Yuan Xie. 2019. Tackling the qubit mapping problem for NISQ-era quantum devices. In *Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems*. 1001–1014.

[14] Christopher Monroe and Jungsang Kim. 2013. Scaling the ion trap quantum processor. *Science* 339, 6124 (2013), 1164–1169.

[15] Don Monroe. 2022. Building a practical quantum computer. *Commun. ACM* 65, 7 (2022), 15–17.

[16] Mihir Pant, Hari Krovi, Don Towsley, Leandros Tassiulas, Liang Jiang, Prithwish Basu, Dirk Englund, and Saikat Guha. 2019. Routing entanglement in the quantum internet. *npj Quantum Information* 5, 1 (2019), 25.

[17] Tianyi Peng, Aram W Harrow, Maris Ozols, and Xiaodi Wu. 2020. Simulating large quantum circuits on a small quantum computer. *Physical review letters* 125, 15 (2020), 150504.

[18] John Preskill. 2018. Quantum computing in the NISQ era and beyond. *Quantum* 2 (2018), 79.

[19] Shouqian Shi and Chen Qian. 2020. Concurrent entanglement routing for quantum networks: Model and designs. In *Proceedings of the Annual conference of the ACM Special Interest Group on Data Communication on the applications, technologies, architectures, and protocols for computer communication*. 62–75.

[20] Ankit Singla, Chi-Yao Hong, Lucian Popa, and P Brighten Godfrey. 2012. Jellyfish: Networking data centers randomly. In *Presented as part of the 9th {USENIX} Symposium on Networked Systems Design and Implementation ({NSDI} 12)*. 225–238.

[21] Wei Tang, Teague Tomesh, Martin Suchara, Jeffrey Larson, and Margaret Martonosi. 2021. Cutqc: using small quantum computers for large quantum circuit evaluations. In *Proceedings of the 26th ACM International conference on architectural support for programming languages and operating systems*. 473–486.

[22] Dominique Unruh. 2010. Universally Composable Quantum Multiparty Computation.. In *EUROCRYPT*, Vol. 6110. Springer, 486–505.

[23] Kun Wang, Qihua Zhou, Song Guo, and Jiangtao Luo. 2018. Cluster frameworks for efficient scheduling and resource allocation in data center networks: A survey. *IEEE Communications Surveys & Tutorials* 20, 4 (2018), 3560–3580.

[24] Anbang Wu, Hezi Zhang, Gushu Li, Alireza Shabani, Yuan Xie, and Yufei Ding. 2022. AutoComm: A Framework for Enabling Efficient Communication in Distributed Quantum Programs. In *2022 55th IEEE/ACM International Symposium on Microarchitecture (MICRO)*. IEEE, 1027–1041.