Distributed Qubit Allocation that Minimizes the Total Execution Time for Heterogeneous Quantum Computing

Makoto Nakai[1,2,a)] Rodney Van Meter[1,2,b)]

Abstract: Task allocation is one of the essential procedures in distributed computing and this will be also true for distributed quantum computing, which is a promising approach to achieve large-scale quantum computing. Total execution time and the stability in the entire system are the two main optimization criteria for this problem, but existing works for distributed qubit allocation only deal with the latter. This work proposes the heuristic algorithm to decide how each qubit on the given quantum circuit should be allocated on quantum processors with different execution time in order to minimize the total execution time. The performance of the proposed scheme is also evaluated by a simple numerical simulation.

Keywords: Distributed quantum computing, Task allocation problem

1. Introduction

Quantum computers can theoretically solve several problems, such as factoring [1] and unstructured search [2] faster than classical computers, and it is already known that they need thousands of qubits in order to solve problems that are even intractable for their classical counterparts [3].

Two main approaches have been proposed to achieve large-scale quantum computing, which is building a single large quantum processor [3] and perform quantum computing in a distributed manner [4]. Distributed quantum computing is considered more realistic because each quantum processor requires less number of qubits [5].

Recently, several works have been focused on a software called distributed quantum compiler [6], which maps qubits on the program onto physical qubits, optimizes the whole quantum circuit, and reduces the number of communication.

Task allocation problem itself is an NP-problem [7] and classical task allocation algorithms for distributed system are classified into those which aim to minimize the total execution time [8] and those aim to maximize the total reliability of the entire system [9]. However, all the previous works about distributed quantum compilers only care about the error rates in quantum processors and quantum links [10–16]. Execution time might be a new measure of efficiency after the fault-tolerant quantum computing becomes real, especially processors with various physical architectures are connected.

In this work, we propose a heuristic algorithm that provides optimized distributed qubit allocation scheme which minimizes the total execution time.

[1] Faculty of Environment and Information Studies, Keio University SFC, Fujisawa, Kanagawa 252-0882 Japan
[2] Keio University Quantum Computing Center, Yokohama 223-8522 Japan
[a)] dave@sfc.wide.ad.jp
[b)] rdv@sfc.wide.ad.jp

2. Background

2.1 Distributed Quantum Computing

This subchapter explains each technological layer of the distributed quantum computing system, that are discussed in [17].

2.1.1 Distributed Quantum Algorithms

Some quantum algorithms, such as Shor's algorithm and Grover's algorithm distributed quantum computing setting [18, 19] were proposed. These algorithms take quantum circuit for inter-processor communications into account.

2.1.2 Distributed Quantum Compiler

Distributed quantum compiler converts the quantum circuit in the user's program to executable form on each physical processor. For example, it plays the three main roles as follows.

- Finding the optimal mapping of between qubits in the given quantum circuit and those on physical quantum processors, which leads to more efficient quantum computing
- Decomposing the given quantum circuit into available gate-set on each quantum processor
- Optimizing the given quantum circuit in order to reduce the number of both local and inter-processor operations

2.1.3 Virtual Quantum Processor

Virtual quantum processor is the multiple physical quantum processors connected by communication links, which will be explained later.

2.1.4 Remote Operations

Remote operations are quantum gates applied between two physical quantum processors, and three main approaches have been proposed, which are

- Swapping-gate-based approach [6]
- Non-local-CNOT-gate-based approach [20]
- Quantum-teleportation-based approach [21]

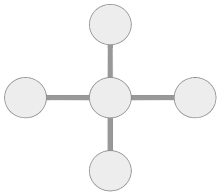

Fig. 1 An example of qubit layout in a physical processor white circles represent physical qubits, gray links represent physical connection between two qubits

2.1.5 Local Operations

Local operations mean quantum gates that can be executed on a single physical quantum processor.

2.1.6 Physical Quantum Processor

The connection of qubits in a single quantum processor is restricted to physical limitation. The link between two qubits indicates the fact that a physical CNOT gate can be executed between these qubits. However, an additional sequence of SWAP gates is required in order to apply a physical CNOT gate between non-neighboring qubits. An example of physical qubit connectivity is shown in Fig.1

2.1.7 Communication Link

Distributed quantum computing requires two types of communication links, which are classical links and quantum links. For instance, quantum entanglement has to be established between qubits on two different quantum processors, and the measurement outcomes on these qubits are transmit through classical communication links.

2.2 Task Allocation Problem

Here are some definitions of the task allocation problem in distributed system [25]. Given a distributed system $G =< V, E >$, where V is the set of nodes and E is the set of communication link between two different nodes, i.e. $\forall v_i.v_j \in V, \exists < v_i, v_j > \in E$. The set of the resources in a_i is R_{a_i} and that required by the task t is R_t.

(1) $R_t \subseteq \bigcup_{\forall a_i \in A_t} R_{a_i}$.

(2) The objective should be either minimizing the execution time or the maximizing reliability of the whole system.

(3) The nodes in A_t can execute the allocated task under the constraint of the network structure $\forall a_i, a_j \in A_t \Rightarrow P_{ij} \subseteq E$ where P_{ij} denotes the path between a_i and a_j.

2.3 Classical Distributed Task Allocation Algorithms

This subsection discusses the task allocation algorithm that minimizes the total execution time. In other words, the author has to come up with the optimal allocation A, which $A(i) = p$ indicates that the task i is allocated to the processor p and $tasks_p$ is a group of tasks that are allocated to a processor p. The total execution in the heterogeneous computing cluster is same as the execution time depends on the one in the most heavily loaded processor. Two types of costs should be considered. One is the execution cost. The execution load in the processor p is the cost of processing all the tasks that are assigned to p for the allocation A.

Suppose C_{ip} is the cost of processing the task i on the processor p, then the total execution cost on the processor p is

$$EXEC_p = \sum_{task \in tasks_p} C_{task,p} \tag{1}$$

.

The other cost is the communication cost, which can be calculated by the following formula.

$$COMM_p = \sum_{task \in tasks_p} \sum_{\substack{(i=j),(i,j) \in E, \\ A(j) \neq p}} d_{ij} * cc_{avg} \tag{2}$$

.

d_{ij} is the data sent between two communicating tasks between i, j and cc_{avg} is the average amount of transferring a data unit through the network transmission media. Therefore, the total cost for the processor p is

$$COST_p = EXEC_p + COMM_p \tag{3}$$

.

Because the total execution time is equal to the execution of the most heavily loaded processor, the total execution time can be described as following.

$$COST = \max\{COST_p | 1 \leq p \leq n\} \tag{4}$$

.

Therefore, the object function [26] is

$$\min \ COST \tag{5}$$

.

2.4 Quantum Distributed Task Allocation Algorithms

Partitioning the given quantum circuit into several fragments is one of the main roles for a distributed quantum compiler. In the process of partitioning the given quantum circuit, the distributed quantum compiler has to minimize the number of inter-processor communication in order to reduce the delay in the entire circuit execution.

Severals algorithm for finding the quantum circuit partition with minimum number of inter-processor communications have been proposed and these algorithms are based on exhaustive search [10], graph partitioning [11], hypergraph partitioning [12], genetic algorithm [13], dynamic programming [14], window-based quantum circuit partitioning [15], and connectivity matrix of the quantum circuit [16].

2.5 Experimental Works Related To Distributed Quantum Computing

An optical remote quantum gate is already experimentally implemented and it successfully implemented four Bell states [27].

Quantum computing architecture for a large-scale quantum computing that combines ion-trap and optical technologies [28] and a superconducting chiplet that connects the physical processor and microwave links are also proposed.

Table 1 Comparison of Execution Times Between Superconducting and Ion trap [29]

Physical Architecture	One qubit gate time (ns)	Two qubit gate time (ns)
Superconducting	1.3×10^2	$2.5 \times 10^2 - 4.5 \times 10^2$
Ion trap	2.0×10^4	2.5×10^5

2.6 Toward Realization of Heterogeneous Quantum Computing

Distributed quantum computing is superior to a single large quantum processor in terms of not only its scalability, but also its heterogeneity. In other words, the whole quantum computing clusters can use quantum processors with different physical architectures. As shown in the Table 1, the execution time of quantum qubit gates significantly changes in various physical architecture. Also, the concept and challenges for building a quantum computing cluster with various physical platforms or quantum interconnects (QuICs) are discussed in [30].

3. Problem Definition

Suppose a distributed quantum computing system consists of N quantum processors connected via communication links. Each quantum processor has limited number of qubits and execution time.

A quantum circuit in the program consists of several qubits and M gates, including CNOTs which corresponds to an interaction graph $G(V, E)$. V represents a set of qubits and E represents set of a connection between two qubits involved in each CNOT gate. $q_i \in V$ is labeled by the qubit index, and $(control, target) \in E$ is labeled by control-target relationship of all the CNOT gates.

The problem is how to allocate each qubits in the given quantum circuit to which processor with varying execution time in order to minimize the total execution time. This problem can be formulated as an optimization problem, which requires a cost function, which is the value to either maximize or minimize to acquire the optimal solution.

4. Proposal

4.1 Objective Function

Suppose A be the optimal assignment such that $A(q_i) = p_j$ if a qubit q_i in the given quantum circuit to a quantum processor p_j.

A group of qubits allocated to a quantum processor p_j is denoted as $qubits_j$, local single qubit gates allocated to a particular qubit q_i is $SingleQubitGates_i$, local two qubit gates allocated to a particular qubit q_i is $TwoQubitGates_i$.

The execution time of a single qubit gate and two qubit gate on a particular quantum processor p_j is $SingleQubitGateTime_j$ and $TwoQubitGateTime_j$, respectively.

The cost of executing all local quantum gates on a qubit q_i on a quantum processor p_j is

$$GATECOST = \sum_{gate \in SingleQubitGates_i} SingleQubitGateTime_j + \sum_{gate \in TwoQubitGates_i} TwoQubitGateTime_j \tag{6}$$

.

Suppose a CNOT gate CNOT(control, target) involves two qubits, which are control qubit and target qubit, and they are a part of a quantum processor p_s and p_t respectively. Also, all the CNOT gates in a quantum processor p_j are denoted as $CNOTs_j$.

The communication cost in a quantum processor p_j is

$$COMMCOST_j = \sum_{\substack{CNOT(control,target) \in CNOTs_j \\ control \in q_s \\ target \in q_t}} PathLength(s, k) \tag{7}$$

.

PathLength is the length of the path between the processor s and the processor t on the given network topology, and the processor j is same as at least either the processor s or the processor t.

Thus, the total cost on a quantum processor p_j is

$$COST_j = GATECOST_j + COMMCOST_j \tag{8}$$

.

Both execution of single qubit gates and communication with other processors affect the total execution time on each processor, and because the processor with the greatest cost will decide the total execution time on the whole distributed quantum system, the following value should be calculated.

$$MAXCOST = \max\{COST_j | 1 \leq j \leq N\} \tag{9}$$

.

Also, minimizing this value will reduce both the execution time for quantum gates execution and interprocessor communication, and the objective function of this problem is

$$\min\ MAXCOST \tag{10}$$

4.2 Simulated Annealing

Simulated annealing [31] is a heuristic algorithm which reaches to the global optimal solution in some cases. It requires two given values, which are *temperature* (how long it takes to reach the optimal solution) and *energy* (how close the current answer is to the optimal solution).

The solution changes randomly and even the answer after randomization process is accepted if the current energy value becomes higher than its previous one in order to avoid being stuck in the local minimum.

Algorithm 1 Finding a neighbor state

Require:
Processor list P $\{P_0, P_1, \ldots P_N\}$
Initial allocation A $\{P_0 : \text{qubits}_0, P_1 : \text{qubits}_1, \ldots P_n : \text{qubits}_n\}$
Ensure: New allocation A
function MOVE
    $P_i \leftarrow$ a randomly selected processor
    $P_j \leftarrow$ another randomly selected processor
    $\text{qindex}_i \leftarrow$ a randomly selected qubit index from 0 to len(qubits_i)
    $\text{qindex}_j \leftarrow$ a randomly selected qubit index from 0 to len(qubits_j)
    $A[P_i][\text{qindex}_i], A[P_j][\text{qindex}_j] = A[P_j][\text{qindex}_j], A[P_i][\text{qindex}_i]$
end function

Algorithm 2 Calculating the acceptance probability

Require: current energy value cur_eng, new energy value new_eng, current temperature temp
Ensure: an acceptance probability prob
function ACCEPTPROB(cur_eng, new_eng, temp)
    if cur_eng < new_eng **then**
        return 1
    else
        return exp (-(new_eng-cur_eng)/temp)
    end if
end function

Algorithm 3 Simulated Annealing

Require:
A random allocation A
Initial temperature T
Iteration number IterNum
Ensure: The optimal allocation A'
function SIMULATEDANNEALING(A, T, IterNum)
    A' = A
    for iter := 1 to IterNum **do**
        temp := T ∗(1 − iter/IterNum)
        copyA ← Copy(A')
        newA ← Move(copyA)
        eng ← CalcEnergy(copyA)
        neweng ← CalcEnergy(newA)
        if AcceptProb(eng, neweng, temp) > randomvalue(0, 1) **then**
            A' = newA
        end if
    end for
    return A'
end function

Algorithm 4 Calculating the energy value

Require:
Initial qubit allocation A $\{P_0 : \text{qubits}_0, P_1 : \text{qubits}_1, \ldots P_n : \text{qubits}_n\}$
A list of quantum gates gate_list $\{\text{gate}_0, \ldots, \text{gate}_N\}$
A list of execution time of a single qubit gate on each quantum processor single_qubit_gate_time_list [time_0, . . . time_N]
A list of execution time of a CNOT gate on each quantum processor CNOT_gate_time_list [time_0, . . . time_N]
Network topology N
Ensure: An energy value *E*
function CALCENERGY(A, gate_list)
    processor_list ← [keys in A]
    gate_cost_list ← [0 for processor in processor_list]
    comm_cost_list ← [0 for processor in processor_list]
    for processor_id := 0 to length of processor_list −1 **do**
        for gate := gate_0 to gate_N **do**
            single_qubit_gate_time ← single_qubit_gate_time_list[processor_id]
            CNOT_gate_time ← CNOT_gate_time_list[processor_id]
            if gate. name ≠ CNOT and gate. index ∈ A[processor_id] **then**
                gate_cost_list[processor_id] += single_qubit_gate_time
            else if gate. name = CNOT and gate. index ∈ A[processor_id] and gate. target_index ∈ A[processor_id] **then**
                gate_cost_list[processor_id] += CNOT_gate_time
            end if
        end for
    end for
    distance_matrix ← N_distance_matrix
    for processor_id := 0 to len(processor_list) −1 **do**
        for gate := gate_0 to gate_N **do**
            if gate_name = CNOT **then**
                if gate_index, gate_target_index ∈ A[processor_id] **then**
                    comm_cost_list[processor_id] += 0
                else if gate_index ∈ A[processor_id] **then**
                    for processor'_id := 0 to length of processor_list −1 **do**
                        if gate_target_index ∈ A[processor'_id] **then**
                            distance ← distance_matrix[processor_id][processor'_id]
                            comm_cost_list[processor_id] += distance
                        end if
                    end for
                else if gate_gate_index ∈ A[processor_id] **then**
                    for processor'_id := 0 to length of processor_list −1 **do**
                        if gate_index ∈ A[processor'_id] **then**
                            distance ← distance_matrix[processor'_id][processor_id]
                            comm_cost_list[processor_id] += distance
                        end if
                    end for
                end if
            end if
        end for
    end for
    for processor_id := 0 to length of processor_list −1 **do**
        gate_cost ← gate_cost_list[processor_id]
        comm_cost ← comm_cost_list[processor_id]
        cost_list[processor_id] ← gate_cost + comm_cost
    end for
    return max cost_list
end function

5. Evaluation

This chapter investigates the efficiency of the allocation method proposed in the previous chapter, under the distributed quantum computing system with several processors with different execution time in a limited network topology.

In this experiment, three 2-qubit quantum processors with different execution time are connected in a linear topology, which are shown in Table2 and Fig2.

The author executed the quatum circuit in Fig3 on these three quantum processors. This circuit is a quantum circuit for encoding 5-qubit quantum repetition code, which is one of the circuits used for benchmarking purposes [32].

The code used for this experiment is uploaded in [33]

.

6. Result

the total execution time of four different allocation cases

- when qubits are randomly allocated
- when only the gate execution cost is optimized
- when only the communication cost is optimized
- when both costs are optimized

.

are compared and the comparison result is shown in Fig4. I measured the total execution time in the four cases, ten trials for each, and compared the average execution time. The figure 4 shows that the total execution time becomes the shortest when the both gate execution cost and communication cost are optimized.

7. Discussion

In this experiment, the total execution time of the gate-cost-based case and that of the communication-cost-based-case are almost the same. However, there should be some cases that where either the gate-execution-cost-based optimization or the communication-cost-based optimization works better than the other. For example, gate-cost-based optimization would work better if the given circuit have more single qubit gates than CNOT gates and these gates are fairly allocated to each qubit. On the other hand, the communication-cost-based optimization yields a better performance if the given quantum circuit has more CNOT gates than one-qubit gates, and each processor has less neighboring processors, such as linear topology.

Table 2 Details of Each Processor

Processor name	One qubit gate time (s)	Two qubit gate time (s)
P1	0.2	1.0
P2	0.4	2.0
P3	0.6	3.0

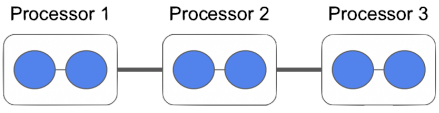

Fig. 2 Network Topology of Quantum Processors

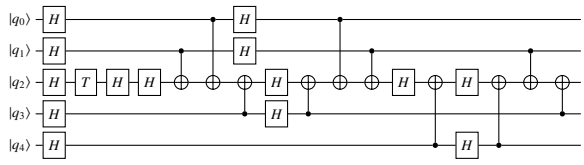

Fig. 3 Quantum circuit for encoding 5-qubit quantum repetition code

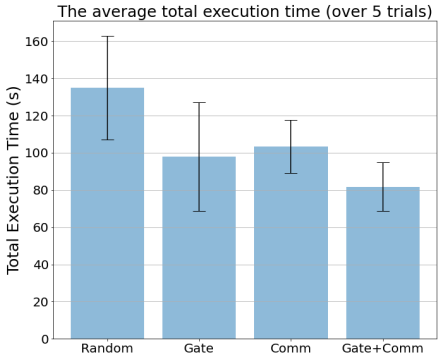

Fig. 4 Experiment result

8. Conclusion

This thesis aims to propose an effective scheme for qubit allocation for distributed quantum computing to reduce the total execution time, and the chart in the previous chapter clearly demonstrates that the case when the both gate execution cost and communication cost are optimized performs the best. This fact also states that, similar to the task allocation algorithm for classical distributed computing, people have to take both the task (quantum gate) execution cost and communication cost into account in order to come up with an (nearly-) optimal qubit allocation in terms of reducing the total execution time.

9. Acknowledgement

I would like to express my gratitude to Master's students in the same research group Ryosuke Satoh, and Yasuhiro Okura, who provide constructive feedbacks and valuable discussion.

References

[1] Shor, Peter W. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. SIAM review 41.2 (1999): 303-332.
[2] Grover, Lov K. A fast quantum mechanical algorithm for database search." Proceedings of the twenty-eighth annual ACM symposium on Theory of computing. 1996.
[3] Gidney, Craig, and Martin Ekerå. How to factor 2048 bit RSA integers in 8 hours using 20 million noisy qubits. Quantum 5 (2021): 433.
[4] Lov K Grover. Quantum telecomputation. arXiv preprint quant-ph/9704012, 1997.
[5] R. Van Meter and S. J. Devitt, The Path to Scalable Distributed Quantum Computing, in Computer, vol. 49, no. 9, pp. 31-42, Sept. 2016, doi: 10.1109/MC.2016.291.
[6] D. Ferrari, A. S. Cacciapuoti, M. Amoretti, and M. Caleffi, "Compiler design for distributed quantum computing," IEEE Transactions on Quantum Engineering, vol. 2, pp. 1–20, 2021.
[7] V.M. Lo. Heuristic algorithms for task assignment in distributed systems. IEEE Transactions on Computers, 37(11):1384–1397, 1988.
[8] Perng-Yi Richard Ma, Lee, and Tsuchiya. A task allocation model for distributed computing systems. IEEE Transactions on Computers, C-31(1):41–47, 1982.
[9] Gamal Attiya and Yskandar Hamam. Task allocation for maximizing reliability of distributed systems: A simulated annealing approach. Journal of Parallel and Distributed Computing, 66(10):1259–1266, 2006.
[10] Mariam Zomorodi-Moghadam, Mahboobeh Houshmand, and Monireh Houshmand. Optimizing teleportation cost in distributed quantum circuits. International Journal of Theoretical Physics, 57(3):848–861, 2018.

[11] Omid Daei, Keivan Navi, and Mariam Zomorodi-Moghadam. Optimized quantum circuit partitioning. International Journal of Theoretical Physics, 59(12):3804–3820, 2020.
[12] Pablo Andres-Martinez and Chris Heunen. Automated distribution of quantum cir- cuits via hypergraph partitioning. Physical Review A, 100(3):032308, 2019.
[13] Mahboobeh Houshmand, Zahra Mohammadi, Mariam Zomorodi, and Monireh Houshmand. An evolutionary approach to optimizing teleportation cost in distributed quantum computation. International Journal of Theoretical Physics, 59, 04 2020.
[14] Zohreh Davarzani, Mariam Zomorodi-Moghadam, Mahboobeh Houshmand, and Mostafa Nouri-baygi. A dynamic programming approach for distributing quantum circuits by bipartite graphs. Quantum Information Processing, 19(10):1–18, 2020.
[15] Eesa Nikahd, Naser Mohammadzadeh, Mehdi Sedighi, and Morteza Saheb Zamani. Automated window-based partitioning of quantum circuits. Physica Scripta, 96, 12 2020.
[16] Ismail Ghodsollahee, Zohreh Davarzani, Mariam Zomorodi, PawfB QBawiak, Monireh Houshmand, and Mahboobeh Houshmand. Connectivity matrix model of quantum circuits and its application to distributed quantum circuit optimization. Quantum Inf. Process., 20:1–21, 2021.
[17] Cuomo Daniele, Marcello Caleffi, and Angela Sara Cacciapuoti. Towards a distributed quantum computing ecosystem. IET Quantum Communication 1.1 (2020): 3-8.
[18] Anocha Yimsiriwattana, and Samuel J. Lomonaco Jr. Distributed quantum computing: A distributed Shor algorithm. Quantum Information and Computation II. Vol. 5436. SPIE, 2004.
[19] Iaakov Exman and Efrat Levy. Quantum probes reduce measurements: Application to distributed grover algorithm. arXiv:1208.3905, 08 2012.
[20] Jens Eisert, Kurt Jacobs, Polykarpos Papadopoulos, and Martin B Plenio. Optimal local implementation of nonlocal quantum gates. Physical Review A, 62(5):052317, 2000.
[21] Rodney Van Meter, WJ Munro, Kae Nemoto, and Kohei M Itoh. Arithmetic on a distributed-memory quantum multicomputer. ACM Journal on Emerging Technologies in Computing Systems (JETC), 3(4):1–23, 2008.
[22] Hong, Sabrina S., Alexander T. Papageorge, Prasahnt Sivarajah, Genya Crossman, Nicolas Didier, Anthony M. Polloreno, Eyob A. Sete, Stefan W. Turkowski, Marcus P. da Silva, and Blake R. Johnson. "Demonstration of a parametrically activated entangling gate protected from flux noise." Physical Review A 101, no. 1 (2020): 012302.
[23] Grzesiak, Nikodem, Reinhold Blümel, Kenneth Wright, Kristin M. Beck, Neal C. Pisenti, Ming Li, Vandiver Chaplin et al. "Efficient arbitrary simultaneously entangling gates on a trapped-ion quantum computer." Nature communications 11, no. 1 (2020): 1-6.
[24] Li, Rui, Shurui Li, Dongmin Yu, Jing Qian, and Weiping Zhang. "Optimal model for fewer-qubit CNOT gates with Rydberg atoms." Physical Review Applied 17, no. 2 (2022): 024014.
[25] Yichuan Jiang, "A Survey of Task Allocation and Load Balancing in Distributed Systems," in IEEE Transactions on Parallel and Distributed Systems, vol. 27, no. 2, pp. 585-599, 1 Feb. 2016, doi: 10.1109/TPDS.2015.2407900.
[26] Gamal Attiya and Yskandar Hamam. Task allocation for minimizing programs completion time in multicomputer systems. volume 3044, pages 97–106, 05 2004.
[27] Severin Daiss, Stefan Langenfeld, Stephan Welte, Emanuele Distante, Philip Thomas, Lukas Hartung, Olivier Morin, and Gerhard Rempe. A quantum-logic gate between distant quantum-network modules. Science, 371(6529):614–617, 2021.
[28] Jungsang Kim and Changsoon Kim. Integrated optical approach to trapped ion quantum computation. arXiv preprint arXiv:0711.3866, 2007.
[29] Linke, Norbert M., Dmitri Maslov, Martin Roetteler, Shantanu Debnath, Caroline Figgatt, Kevin A. Landsman, Kenneth Wright, and Christopher Monroe. "Experimental comparison of two quantum computing architectures." Proceedings of the National Academy of Sciences 114, no. 13 (2017): 3305-3310.
[30] Awschalom, David, Karl K. Berggren, Hannes Bernien, Sunil Bhave, Lincoln D. Carr, Paul Davids, Sophia E. Economou et al. "Development of quantum interconnects (quics) for next-generation information technologies." PRX Quantum 2, no. 1 (2021): 017002.
[31] S. Kirkpatrick, C.D. Gelatt, M.P. Vecchi,Optimization by Simulated Annealing,Readings in Computer Vision,Morgan Kaufmann,1987
[32] Ang Li, Samuel Stein, Sriram Krishnamoorthy, and James Ang. Qasmbench: A low-level QASM benchmark suite for NISQ evaluation and simulation, 2021.
[33] Nakai, Makoto. (2022). heqsim: Heterogeneous Quantum Computing Simulator (Version 0.0.1) [Computer software]