Project Progress Report

Seyed Sajad Kahani
22222815

February 3, 2023

1 Literature Review

The literature review is divided into three parts: general-purpose quantum compilers, specific-purpose quantum compilers for Hamiltonians, and classical problems and algorithms related to the compilation.

1.1 General Quantum Compilers

In the first part, we review papers directly related to quantum compilation. While most of the papers are useful for our case, we are focusing on two specific subjects: the approaches to the gate-set, especially two-qubit gates, during the compilation process, and the algorithms and principles for qubit allocation and routing gates through the device.

We have analyzed almost all of the papers listed, with a focus on the gate-set. However, we are still reviewing the papers to organize and categorize the algorithms and corresponding problems in qubit allocation and routing.

Seyed Sajad Kahani can you list your changes? Yes, here are the changes made to the text:

The literature review is divided into three parts: general-purpose quantum compilers, specific-purpose quantum compilers for Hamiltonians, and classical problems and algorithms related to the compilation.

Subsection: General Quantum Compilers

The first part consists of papers that are directly related to quantum compilation. While most of the papers are useful, we are also searching for two specific subjects.

Approaches to the gate-set, especially two-qubit gates, during the compilation process. Algorithms and principles for qubit allocation and routing gates through the device.

We have analysed almost all of the papers listed below with respect to the gate-set. However, we are still reviewing the papers to categorize their algorithms and corresponding problems in qubit allocation and routing.

[5] Andrew M Childs, Eddie Schoute, and Cem M Unsal. “Circuit Transformations for Quantum Architectures”. In: (), p. 29.

[7] Andrew Cross et al. "OpenQASM 3: A Broader and Deeper Quantum Assembly Language". In: *ACM Transactions on Quantum Computing* 3.3 (Sept. 30, 2022), pp. 1–50. ISSN: 2643-6809, 2643-6817. DOI: `10.1145/3505636`. (Visited on 11/11/2022).

[10] Toshinari Itoko et al. "Optimization of Quantum Circuit Mapping using Gate Transformation and Commutation". In: (July 2019). DOI: `10.48550/ARXIV.1907.02686`. arXiv: `1907.02686 [quant-ph]`.

[14] Gushu Li, Yufei Ding, and Yuan Xie. "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.* ASPLOS '19: Architectural Support for Programming Languages and Operating Systems. Providence RI USA: ACM, Apr. 4, 2019, pp. 1001–1014. ISBN: 978-1-4503-6240-5. DOI: `10.1145/3297858.3304023`. (Visited on 11/11/2022).

[16] Alexandru Paler. *On the Influence of Initial Qubit Placement During NISQ Circuit Compilation.* Jan. 30, 2019. arXiv: `1811.08985`.

[17] Alexandru Paler, Alwin Zulehner, and Robert Wille. "NISQ circuit compilers: search space structure and heuristics". In: (), p. 9.

[21] Marcos Yukio Siraichi et al. "Qubit allocation". In: *Proceedings of the 2018 International Symposium on Code Generation and Optimization.* CGO '18: 16th Annual IEEE/ACM International Symposium on Code Generation and Optimization. Vienna Austria: ACM, Feb. 24, 2018, pp. 113–125. ISBN: 978-1-4503-5617-6. DOI: `10.1145/3168822`. (Visited on 11/11/2022).

[23] Chi Zhang et al. "Time-optimal Qubit mapping". In: *Proceedings of the 26th ACM International Conference on Architectural Support for Programming Languages and Operating Systems.* ASPLOS '21: 26th ACM International Conference on Architectural Support for Programming Languages and Operating Systems. Virtual USA: ACM, Apr. 19, 2021, pp. 360–374. ISBN: 978-1-4503-8317-2. DOI: `10.1145/3445814.3446706`. (Visited on 11/11/2022).

[24] Xiangzhen Zhou, Sanjiang Li, and Yuan Feng. "Quantum Circuit Transformation Based on Simulated Annealing and Heuristic Search". In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 39.12 (Dec. 2020), pp. 4683–4694. ISSN: 0278-0070, 1937-4151. DOI: `10.1109/TCAD.2020.2969647`. URL: `https://ieeexplore.ieee.org/document/8970267/` (visited on 11/11/2022).

[25] Alwin Zulehner, Alexandru Paler, and Robert Wille. "Efficient mapping of quantum circuits to the IBM QX architectures". In: *2018 Design, Automation & Test in Europe Conference & Exhibition (DATE).* 2018 Design, Automation & Test in Europe Conference & Exhibition (DATE). Dresden, Germany: IEEE, Mar. 2018, pp. 1135–1138. ISBN: 978-3-9819263-0-9. DOI: `10.23919/DATE.2018.8342181`. URL: `http://ieeexplore.ieee.org/document/8342181/` (visited on 11/11/2022).

1.2 Hamiltonian Quantum Compilers

Compilation of Hamiltonians has its own literature, though it is not as extensive as that of general quantum compilers. Any paper related to the compilation of Hamiltonians is considered as a part of this section and should be carefully studied.

As a baseline for our project, we are using the paper [13] which is a recent work on the compilation of Hamiltonians.

[4] Earl Campbell. "Random Compiler for Fast Hamiltonian Simulation". In: *Physical Review Letters* 123.7 (Aug. 2019), p. 070503. DOI: `10.1103/PhysRevLett.123.070503`. URL: `https://link.aps.org/doi/10.1103/PhysRevLett.123.070503` (visited on 01/22/2023).

[13] Lingling Lao and Dan E. Browne. *2QAN: A quantum compiler for 2-local qubit Hamiltonian simulation algorithms.* Nov. 7, 2021. arXiv: `2108.02099`.

1.3 Classical Compilation and Related Problems

Many of the challenges in designing and implementing quantum compilers have comparable solutions in the classical world that have been well-studied. It is important to conduct a thorough review of classical compilation for this project, as well as explore algorithms from related fields such as resource allocation in distributed systems and scheduling in operating systems to possibly find the standard solutions to the problems we face.

We have initiated our review with two textbooks in classical compilation [1] and [3]. We will also continue to search for any additional relevant papers.

[2] Mansoor Alicherry and T.V. Lakshman. "Network aware resource allocation in distributed clouds". In: *2012 Proceedings IEEE INFOCOM.* IEEE, Mar. 2012. DOI: `10.1109/infcom.2012.6195847`.

2 The Aim

The objective of this project is to survey the current literature on compiling local Hamiltonians, and use this information to enhance existing approaches, such as 2QAN. In order to effectively tackle the complex compilation process, the sub-problems must be decomposed into smaller and more manageable tasks.

The results of this project will then be tested and evaluated for performance and quality, with the goal of creating a prototype implementation that can be used and further developed by other researchers. It should be noted that the focus will be on exploring solutions rather than creating a polished and highly-engineered implementation.

Note that this objective, is carried on from the beginning of the project, and it is still valid by this time.

3 Research Method

The research methodology for this project involves a three-step process:

In the first step, a new algorithm or approach to the problem is introduced. This could be an idea that has not been explored before, or an improvement on an existing method.

The next step is to evaluate the correctness and optimality of the new idea. This is done by using mathematical tools such as complexity theory, to ensure that the new approach will work as intended and that it is optimal in terms of performance.

Finally, the new idea is simulated to see how it performs in practice. This is an important step, as it allows us to see the real-world performance of the new approach and identify any potential issues or limitations.

4 Results

Besides all the insights and the knowledge that we have gained from the literature review, we have also a few ideas that can result in improving the existing compilers.

The main idea which is developed pretty well, is related to the routing problem.

When we have a two-qubit gate, (for example, a CNOT gate), and it shall be applied between two qubits that their corresponding qubits on the device are not connected, we will use a chain of SWAP operators to move the qubits and make them adjacent to each other, and then apply the gate. This is a very common approach [6].

Here we introduce a possible simplification of this approach, using simplified bridge gates, instead of using a chain of SWAP operators.

4.1 Bridge Gate

For a simple case, that we have three qubits, called a, b, c, and we want to apply a CNOT gate on (a, c), but the connectivity only allows us to apply a CNOT gate on (a, b) and (b, c), the first solution would be to use a SWAP gate, which is shown in figure 4.1 (7 gates, depth of 7) While another approach is to use a bridge gate, which is shown in figure 4.1 (4 gates, depth of 4). [20]

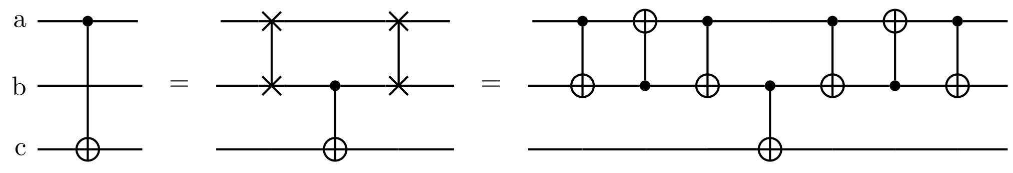


Figure 1: Applying a CNOT gate on (a, c) using a SWAP gate

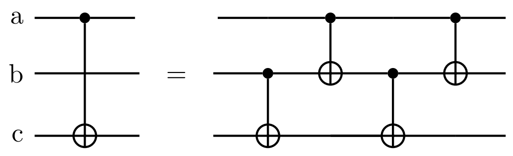


Figure 2: Applying a CNOT gate on (a, c) using a bridge gate

The bridge gate is already generalized using a recursive approach [20], with $4n + O(1)$ depth and $4n + O(1)$ gates complexity. (where n is the number of qubits in the chain). Here we present a simplified version of the bridge gate, with $n + O(1)$ depth and $4n + O(1)$ gates complexity.

Note that a simple swapping approach will have $3n + O(1)$ depth and $6n + O(1)$ gates complexity.

Definition 1. *Simplified Bridge Gate We can define a simplified bridge gate, recursively, as follows:*

$$\begin{aligned}\mathrm{Bridge}_n(1,n) =& \mathrm{CNOT}(2,1)\mathrm{CNOT}(1,2)\mathrm{CNOT}(n,n-1)\mathrm{CNOT}(n-1,n)\\ & \mathrm{Bridge}_{n-1}(2,n-1)\\ & \mathrm{CNOT}(2,1)\mathrm{CNOT}(1,2)\mathrm{CNOT}(n,n-1)\mathrm{CNOT}(n-1,n)\end{aligned} \tag{1}$$

where Bridge_2 *is defined as fig. 4.1 and* Bridge_1 *is just a CNOT.*

We ignore to show that this gate is equivalent to the recursive definition of the bridge gate, since it is not the main focus of the progress report.

It is easy to show that such circuit contains $4n + O(1)$ gates, but in order to show the depth, we need to use the fact that two CNOTs applying on the same gates, commute with each other, therefore, like for the case of even ns, we can show the recursive formula is equivalent to the following:

$$\begin{aligned}\mathrm{Bridge}_n(1,n) =& \prod_{i=1}^{n/2-1} \mathrm{CNOT}(i+1,i)\mathrm{CNOT}(n-i+1,n-i)\\ & \prod_{i=1}^{n/2-1} \mathrm{CNOT}(i,i+1)\mathrm{CNOT}(n-i,n-i+1)\\ & \mathrm{Bridge}_1(n/2-1,n/2)\\ & \left(\prod_{i=1}^{n/2-1} \mathrm{CNOT}(i+1,i)\mathrm{CNOT}(n-i+1,n-i)\right)^\dagger\\ & \left(\prod_{i=1}^{n/2-1} \mathrm{CNOT}(i,i+1)\mathrm{CNOT}(n-i,n-i+1)\right)^\dagger\end{aligned} \tag{2}$$

where each term of each products in the RHS and the LHS can be applied simultaneously (4 gates at a time). Therefore, the depth of the circuit is $n+O(1)$. Hereby we can see an example of the bridge gate for $n = 6$ in fig. 3.

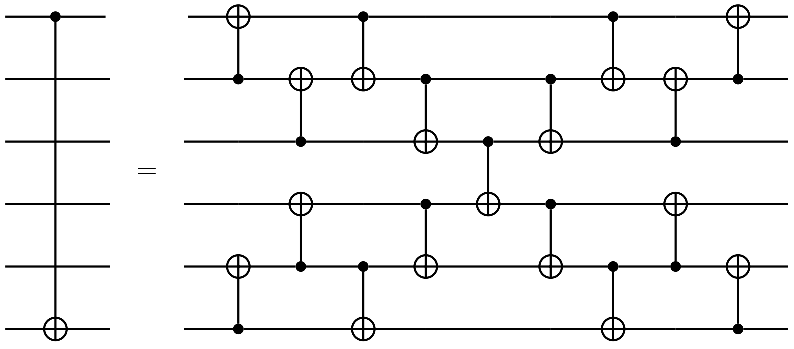


Figure 3: The bridge gate for $n = 6$

Furthermore, we can use the same idea and create a bridge gate to express $e^{iZ^1X^n}$ using CNOTs and $e^{iZ^1X^2}$ gates. Then we are working on the most general way to bridge a qubit gate between two qubits in a chain.

It is also proved that any effort to decompose a CNOT into a chain of CNOTs, cannot result in any complexity better than the simplified bridge gate. Therefore, the simplified bridge gate is the best possible gate to bridge a qubit gate between two qubits in a chain.

As far as we know, the family of bridge gates (the previous one or the simplified one) is not implemented (explicitly or implicitly) in any quantum compiler. It can be easily shown that for a simple case like the one in the figure 4 that we need to swap and return the qubits, using bridge gate can reduce both depth and number of gates.

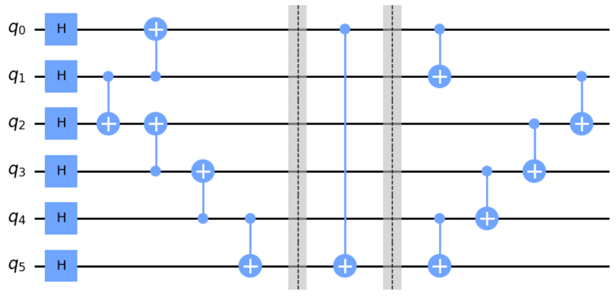
5 Challenges

This project as an interdisciplinary project, has the challenge of the diversity of the literature. There are a vast range of keywords and languages in the papers. And in the first step, to find and understand the literature in this field, the general knowledge of quantum compilation, physical implementations of quantum computers, and classical algorithms for compilation is required.

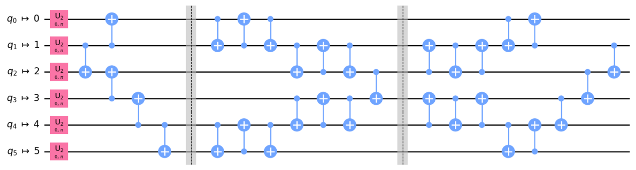
The second challenge is the subtle line between theoretical works and implementation efforts. We are in an era that both efforts are meaningful and important. And yet any theoretical work should consider the applicability of the results. Also, any new implementation should be theoretically sound and should be able to be used in the future.

To help to overcome with this challenge, we specified that this, as a master project, shall be a theoretical work. And any implementations will only be for the sake of evaluation and comparison.

a)



b)



c)

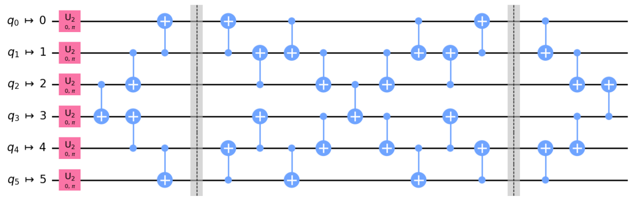


Figure 4: a) The original circuit, consisting of some local operations, then a far away CNOT and then some local operations. b) The circuit after transpiling using Qiskit. c) The circuit after transpiling using bridge gate as an intermediate gate.

Besides these general challenges, in the near future, we should be able to understand and generalize all of the efforts in the allocation and routing subproblems of quantum compilation. Which is a huge analytical task. Also, yet we have a small demonstrated improvement and a theoretical proof, we should be able to introduce more powerful results to the field.