# Bridging Gates over Qubits

Seyed Sajad Kahani
Supervisor: Prof. Dan Browne

August 22, 2023

# Contents

**Abstract**

# Chapter 1

# Introduction

Quantum computation (QC) is an emerging field that aims to use quantum mechanics to solve problems that are intractable for classical computers. Since the earliest conceptualization of quantum computation [15], it has been believed that quantum computers could revolutionize the way we solve problems, particularly those involving simulating nature. Over time, it has become clear that quantum computers have applications far beyond physical simulations. There are algorithms for search and traversing graphs, solving linear equations [27], and methods for machine learning and optimization [21].

Despite significant efforts, we are still far from fully utilizing these algorithms. Our current hardware technology has not yet achieved the desired accuracy and number of qubits necessary for quantum computers to outperform classical computers in solving useful problems. The current situation is commonly referred to as the "noisy intermediate scale quantum" (NISQ) era [34], characterized by restricted resources, including a limited number of qubits, constrained qubit connectivity, hardware-specific gate sets, and limited circuit depth due to noise [13].

The restricted qubit resources and excessive noise susceptibility of NISQ devices necessitate optimal compilers to have any hope of useful near-term quantum computation. A huge amount of research has been conducted to tackle different aspects of the compilation problem, including qubit allocation [20, 38, 31, 49, 26], routing [7, 20, 11, 29, 23] and gate synthesis [36, 44, 45, 37, 5, 14]. These aspects are deeply intertwined and one may not distinguish between them, but all of them are in some sense a circuit transformation from a higher-level circuit (with fewer imposed constraints) to a lower-level circuit (with more imposed constraints) [18].

These transformations heavily rely on SWAPs, and the researchers usually aim to minimize the number of SWAPs [7, 35, 39, 20, 26]. While a relatively neglected fact is that, in contrast to classical compilation, SWAPs are the most expensive two-qubit gates in quantum compilation [44], there are a few works addressing the inherent cost of each SWAP gate. A few proposed techniques to reduce the cost of SWAP gates, such as embedding SWAPs within other 2-qubit gates in 2QAN compiler [25], or compiling a family of circuits directly into CNOT gates [23, 29]. In this work, we aim to address the primary usage of SWAP gates - enabling connectivity between non-adjacent qubits. We analyze the possibility of simplifications for different connectivity cases.

Here, we formally define the problem of bridging as applying gates on non-adjacent qubits, and review the previous efforts on that. Then we present a method to implement a family of two-qubit gates on non-adjacent qubits with 33% less CNOTs and 66% less depth in comparison to the baseline method. We also prove lower bounds for each family of gates, which shows that our method is optimal.

The rest of this thesis is organized as follows. Chapter 2 reviews the related works. In Chapter 3, we present the algorithms and prove their correctness and optimality for the classical and quantum cases. We implement the algorithms and benchmark against state-

of-the-art techniques in Chapter **??**. Finally, Chapter 4 concludes and discusses avenues for future work.

# Chapter 2

# Background

We should be concerned with two key findings from the literature review on quantum compilation. First, the way that they broke down the problem and the assumption behind that. Secondly, the alogrithms and techniques that has been used and will be used for our research as well. Therefore, here we try to review both of these aspects to draw a big picture of our notion of quantum compilation and also to review the existing techniques related to our approach and we do these two alongside each other.

## 2.1 Quantum Compilation

The term "quantum compilation" can refer to any process that transforms a higher-level description of a quantum algorithm into a lower-level description [18]. In the majority of works [50, 7, 12, 39, 35, 32], circuits are the description used and thereby the compilation is done by transforming a general quantum circuit into a circuit that is compatible with a specific hardware. Given that the problem of finding the most optimal circuit (with respect to a sense of complexity like depth or number of gates) is proven to be NP-hard [38], the research primarily centers on deconstructing this problem into smaller components and devising techniques to effectively balance between the agility of the process and the quality of the solution. This mirrors the approach adopted in classical compilation [3].

However, the clear structure for this breakdown has not been firmly established, with numerous diverse approaches in play. As such, our goal is to provide a comprehensive overview of the issue and identify common patterns in the existing literature. This overall picture will then serve as a reference point throughout the rest of this thesis. To achieve this overview, it's crucial to define **circuit transformation** as a process that preserves the essential meaning of the circuit, ensuring that the circuit's output remains consistent for any input before and after the transformation. Yet, this process alters the circuit to adhere to specific constraints or optimize it for particular goals.

Thus, **quantum compilation** in our essay will be a sequence of circuit transformations where each transformation either enforces a constraint or optimizes the circuit (where optimization itself can be perceived as a soft constraint). The primary constraints imposed upon the circuit arise from the characteristics of the hardware and are listed below:

- **Gate set**: The set of gates physically available in the hardware.
- **Connectivity**: The connectivity between qubits in the hardware topology.

Furthermore, the optimizations that corresponds to different degrees various degrees of freedom (e.g., qubit assignments, choosing among equivalent subcircuits) can be applied. They are commonly pursuing these goals:

- **Complexity reduction**: Reducing the number of gates or depth of the circuit.

- **Preparation for constraints**: Minimizing violation of the mentioned constraints before imposing them.
- **Error mitigation**: The error of the circuit with respect to the hardware, especially in NISQ devices.

Quantum compilation frameworks mainly use a similar approach, they introduce transformations that are each is somehow seeking one of the goals defined before, tket even uses the term predicates for the constraints that are similar to these [39]. Typical transformations in their compilation process are, decomposing gates (imposing gate set constraint), assigning qubits (optimization of connectivity as a soft constraint), routing (imposing connectivity constraint), and further optimizations (reducing complexity). Yet the order of these transformations is not the same everywhere, for example, some [25, 35] deffer the gate synthesis after the routing while some [48, 39] does it otherwise.

With this background established, we now dive into details on gate set and connectivity constraints, and circuit optimizations techniques for reducing complexity.

### 2.1.1 Gate synthesis

Imposing the gate set constraints which is known as the gate synthesis, is one of the oldest subroutines in the quantum compilation. The problem here is to decompose a general $n$-qubit gate into a sequence of gates from the physically available gates on the device. If set of available gates is able to produce any arbitrary gate, we call it a universal gate set [5]. Here while we review the results for the synthesis of one, two, and more than two-qubit gates, we try to define mathematical objects that will be used in the rest of the thesis. For this purpose, the gate set constraint is defined as below:

**Definition 1** (Gate set constraint)**.** *The gate set constaint is represented by a set of unitary operators $G$ that are the set of available gates on the device, and we say a circuit (or a sequence of gates) is compatible with the gate set constraint if it is composed of only gates from $G$.*

*For our purpose, $G$ is the set of all one-qubit gates plus CNOT gate.*

While most of the devices have the capability to perform arbitrary one-qubit gates, with a neglectible cost, there is a famous result for the synthesis of one-qubit gates, called Solovay-Kitaev theorem [14], that formalize approximate synthesis of any one-qubit gate using only two distinct one-qubit gates as the gate set.

For two-qubit gates, it is known that the set of all one-qubit gates together with CNOT can produce any two-qubit gate and it can be done by upto three CNOTs (and it is optimal) [44, 46]. One of the famous gates that needs three CNOTs is SWAP. Using CNOT as the only available two-qubit gate although is common in theoretical studies of compilation [50, 38, 26, 49, 20, 28] there are other continuous gate sets such as $\mathrm{fSim}(\theta, \phi)$ [16] or other Hamiltonians evolution [9]. Also, the evolution of XYZ Heisenberg interaction Hamiltonian, has been used as an intermediate gate set or a tool for analytical analysis in some works [40, 46]. The importance of this Hamiltonian is because of a decomposition (Cartan's KAK decomposition, Cirac-Kraus decomposition, or Khaneja-Glaser decomposition) that states any two-qubit gate can be made by only one evolution of $XYZ$ Hamiltonian and four one-qubit gates [24, 22]. Here we state this fruitful decomposition as a lemma:

**Lemma 1** (KAK decomposition [43])**.** *For any $U \in U(4)$, there exist $V_1, V_2, V_3, V_4 \in U(2)$ together with $\alpha, \beta, \gamma \in \mathbb{R}$ such that*

$$U = (V_1 \otimes V_2) e^{i\alpha X \otimes X + i\beta Y \otimes Y + i\gamma Z \otimes Z} (V_3 \otimes V_4). \tag{2.1}$$

For more than two-qubit gates, first TOFOLLI gate was used to prove the universality [5], later it was proven that it is not necessary. Although there are a few ways to synthesis a

general $n$-qubit gate [40, 36], they are not commonly used because of its inherent inefficiency. It is known that it could not be better than $O(4^n)$ gates. [36]

Synthesis of local Hamiltonian evolution, as a special case of $n$-qubit gate, is also studied as it is important for simluation purposes [10]. Most of the researchs are built upon Suzuki-Trotter decomposition [42, 41], that approximates the time evolution of a Hamiltonian with a sequence of time evolutions of its terms. While most of the works rely on the first-order Suzuki-Trotter decomposition (ak Lie-Trotter formula) [39, 35], there are a few efforts to analysis higher-order errors [8] or to use other decomposition, such as a random sampling decomposition (QDRIFT) [6]. Beside the gate synthesis, Hamiltonian compilation has implications that can be used in other parts of the compilation process, for example [25] defines routing and scheduling algorithms specifically for Hamiltonian compilation.

### 2.1.2 Qubit Allocation and Routing

The implementation of the qubits into a physical substance needs to overcome so many technical difficulties, and it will be impossible to have a fully connected device with a large number of qubits in the near future. Therefore, the connectivity constraint is always a part of the compilation process in the NISQ era. A typical device looks like Figure 2.1, that the nodes are qubits and the edges are showing the pairs that a two-qubit gate can be applied on them.

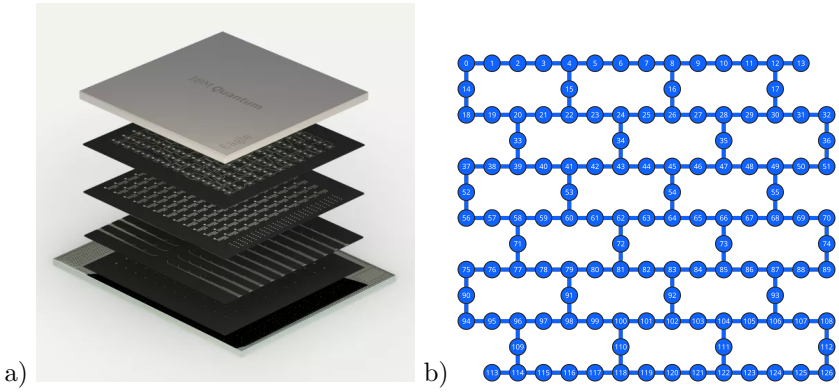

Figure 2.1: a) IBM Eagle processor, a 127 qubit device, b) IBM Eagle connectivity graph [19]

Connectivity constraint will be similarly defined and then, we will review its literature which has drawn so much attention in the recent years.

**Definition 2** (Connectivity constraint)**.** *The connectivity constraint is represented by a graph $G = (V, E)$ where $V$ is the set of qubits and $E$ is the set of edges between qubits. We say a circuit (or a sequence of gates) is compatible with the connectivity constraint if there is a map from qubits to vertices of $G$ and for each two-qubit gate between two qubits, there is an edge between the corresponding vertices in $G$.*

Researchers often tackle the problem of resolving connectivity constraint, by defining two subproblems, qubit allocation and routing. The definition of them may vary in the literature, but we can roughly define the **qubit allocation** qubit allocation we roughly mean a mapping from logical qubits of a quantum circuit to physical qubits of a device in way that minimizes the circuit's complexity overhead due to routing. And by **routing** we mean the problem of finding an optimal circuit that is compatible with device connectivity and is equivalent to the circuit. Results suggest that the qubit allocation could affect the complexity of the circuit by 10% in realistic scenarios [31]. This fact justifies such a break-down of connectivity constraint into two subproblems furthermore.

Qubit allocation shares common traits with other resource allocation problems [2],[3, pp. 440-444] and it is not wondering that it is NP-hard for arbitrary connectivity graphs, this can be shown easily by a reduction from graph isomorphism [38]. Note that real-world devices are not arbitrary graphs and it might be exactly solvable in reasonable time for certain families of graphs.

There have been attempts to find the optimal solution by searching for arbitrary connectivity graphs, which can be feasible for small devices [38]. But, the most common approach in the research is to use a heuristic [49, 20, 11, 31, 28] together with a search algorithm (such as BFS, $A^*$[50], or others[26]) to find a reasonable solution. Some of these efforts have also been implmented in current quantum compilers [35, 39].

Treating routing problem, could be done similarly by thinking of routing as a subsequent qubit reallocations. By introducing a search space of partial allocations (that are partial functions), we can use a unified search space for both qubit allocation and routing [50, 7]. While this will unlock the possibility of using the same algorithm for both qubit allocation and routing that is used in [50], the most of the recent works prefer to treat them separately [26, 25, 7].

Using heuristic alogrithms for routing is a common practice in the literature [50, 20, 11, 26], while there are efforts to find the an exact algorithm [20] that will be inefficient for general case, and by imposing some restrictions on the connectivity graph, we can solve the problem in polynomial time [7]. For example, the problem is solvable in polynomial time for path graphs, complete graphs, tree graphs, and product graphs. These results has been built upon classical problems that are already known like token swapping and routing via matching [4].

While the SWAP gate plays a crucial role in most of the routing algorithms, there are a few efforts that goes beyond the SWAP gates, like using bridged CNOT (sometime called remote CNOT) gate over 1 qubit [20, 38, 39]. By bridged CNOT gate, we mean any sequence of gates that acts as CNOT between two non-adjacent qubits without explicitly having a SWAP gate in it. An example of such gate is depicted in Figure 2.1.2a and it can be compared with the case that are explicitly used, which is shown in Figure 2.1.2b.

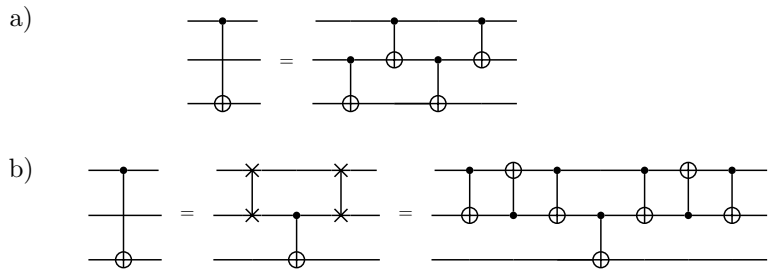


Figure 2.2: a) A bridged CNOT gate over one qubit, b) The baseline implementation of bridged CNOT gate over one qubit

This idea could be broaden to any two-qubit gate, even SWAP itself, (see Figure 2.1.2). This is the problem people have been working on under different names [36] and, here we define bridged gates for an arbitrary connectivity constraint in the definition below:

**Definition 3** (Bridged gate)**.** *Given a connectivity constraint $G = (V, E)$, a target two-qubit gate called $T$ defined on two qubits $a, b \in V$ then, a **bridged** $T$ gate is a sequence of gates that obeys the connectivity constraint and is identical to $T$.*

For the sake of readability, we may omit specifying the connectivity constraint when it is a linear connectivity constraint. From this definition, we can see that the baseline implementation of a bridged gate can be done using SWAPs (like Figure 2.1.2b and Figure 2.1.2b),

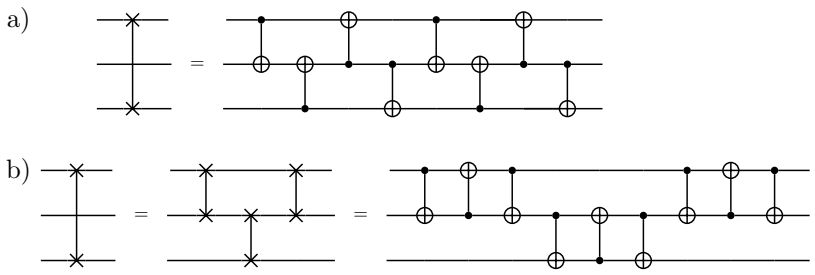


Figure 2.3: a) A bridged SWAP gate over one qubit, b) The baseline implementation of bridged SWAP gate over one qubit

however we are looking for more efficient implementations that are not necessarily trivial (like Figure 2.1.2a and Figure 2.1.2a). The baseline implementation is also shown in the following corollary.

**Corollary 1** (Baseline implemention of a bridged gate)**.** *For any two-qubit gate $T$, A bridged $T$ gate defined on qubits $1, n \in V$ where every $(i, i+1) \in E$, can be implemented for an even $n$ using the following sequence of gates:*

$$T^{1,n} = \prod_{i=n/2-1}^{1} \mathrm{SWAP}^{i,i+1} \otimes \mathrm{SWAP}^{n-i+1,n-i}\, T^{n/2,n/2+1} \prod_{i=1}^{n/2-1} \mathrm{SWAP}^{i,i+1} \otimes \mathrm{SWAP}^{n-i+1,n-i}, \tag{2.2}$$

*and for an odd $n$ we can respectively use:*

$$T^{1,n} = \mathrm{SWAP}^{n,n-1}\, T^{1,n-1}\, \mathrm{SWAP}^{n,n-1}\,. \tag{2.3}$$

*Where $\mathrm{SWAP}^{i,j}$ is a SWAP gate between qubits $i$ and $j$, $T^{i,j}$ is a $T$ gate applied on $i$ and $j$, and note that $\prod_{i=a}^{b} A_i = A_b A_{b-1} \dots A_a$.*

In addition to those efforts, there are tries to totally avoid the SWAP gates and directly apply the routing process by transforming the circuit into a compatible circuit using CNOTs and $R_z$s [29, 23]. Note that this process is not easily expandable to arbitrary gates and circuits, and because of its nature (of working directly with a representation of the whole circuit), its scalability is questionable. Despite that, an important result of this approach is to generate bridged CNOT gates over more than one qubits in a systematic way [29]. The gate that can be generated is written in Equation 2.4 and is shown in Figure 2.1.2. This generalization uses $4n + O(1)$ CNOTs and has a depth of $4n + O(1)$ for a bridged CNOT over $n$ qubits, which is not generally better in comparison to the baseline implementation that uses $6n + O(1)$ CNOTs, with a depth of $3n + O(1)$.

$$\mathrm{CNOT}^{1\to n} = \left[ \prod_{i=n-2}^{2} \mathrm{CNOT}^{i+1\to i} \prod_{i=1}^{n-1} \mathrm{CNOT}^{i+1\to i} \right]^2, \tag{2.4}$$

where $\mathrm{CNOT}^{i\to j}$ is a CNOT gate with control on qubit $i$ and target on qubit $j$.

### 2.1.3 Optimizations

Circuit optimization is the set of techniques to transform the circuit into a more efficient one, without imposing or lifting any constraint. Circuit optimization is often achieved by applying simplification rules to the circuit [33]. These simplification rules are usually based on the commutation relations of the gates [20, 35, 39].

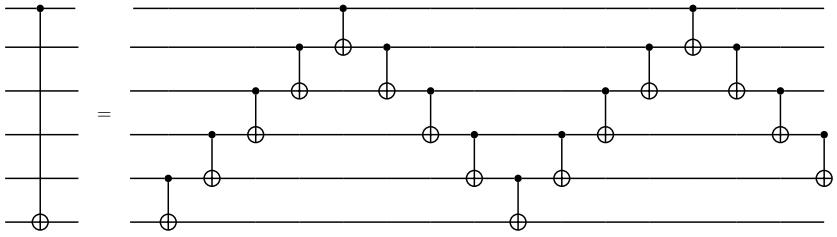

Figure 2.4: A bridged CNOT gate over 6 qubits

Therefore the optimization could be seen as a search problem in the space of equivalent circuits that are explored by applying the simplification rules. The matter that to which extent the search space is explored is a degree of freedom that is called optimization level [35, 39], which is shared with classical compilers as well [1].

# Chapter 3

# Discussion

## 3.1 Problem Statement

We have already reviewed the main research themes around quantum compilation in Chapter 2. We have seen that one of the main challenges is in imposing connectivity constraints on the circuit. Furthermore, because of the huge dependence of most of the research on swapping qubits, putting together with the fact that swapping is the most expensive two-qubit gate, it is important to study the alternatives for swapping. In this chapter, we introduce our research problem based on the definition of bridging that we discussed earlier.

While our aim in the most broad term is to find alternatives to SWAP gate, to apply gates with respect to connectivity constraints, we start with a more specific problem. We seek for optimal bridged gate, and by optimal we mean both the least number of CNOTs and the least depth. Note that, this problem is going to be studied for an arbitrary connectivity constraint $G$.

The relation of this problem to the approach of [29, 23] is that we are looking for bridged gates that can be adopted in the any compilation process, in contrast to theirs that is defining a whole new compilation process. In terms of the results, while some of our preliminary results such as a simple bridged CNOT gate can be derived from their techniques too, however, the main contribution of our work is to show the optimal bridged gate for any two-qubit gate, while those efforts are limited to compiling the gates that are combination of CNOTs and $R_z$s.

## 3.2 Two-qubit Gate Classes

Our discussion heavily relies on the KAK decomposition of two-qubit gates. We have already introduced it in Lemma 1. Here we introduce a few classes of two-qubit gates, based on their KAK decomposition.

**Definition 4** (Two-qubit Gate Classes)**.** *Based on the KAK decomposition of two-qubit gates, that is*

$$U = (V_1 \otimes V_2)e^{i\alpha X\otimes X + i\beta Y\otimes Y + i\gamma Z\otimes Z}(V_3 \otimes V_4), \tag{3.1}$$

*we define the following classes of two-qubit gates:*

- ***Class nil**: Two-qubit gates that $\alpha = \beta = \gamma = 0$.*
- ***Class I:** Two-qubit gates that only one of $\alpha, \beta, \gamma \neq 0$.*
- ***Class II:** Two-qubit gates that at least two of $\alpha, \beta, \gamma \neq 0$.*

The gates in class nil are local gates that are applied on each qubit separately, so they are irrelevant to our discussion about bridging. The gates in class I, such as CNOT, CZ and

etc., can be decomposed to one $\mathrm{C}R_x$ gate upto local gates. This fact is shown in Lemma 2 and enables us to bridge them.

**Lemma 2.** *For any two-qubit gate $U$ in class I, it can be decomposed as*

$$U = W_1 \otimes W_2 \mathrm{C}R_x(\theta) W_3 \otimes W_4, \tag{3.2}$$

*where $\mathrm{C}R_x(\theta)$ is a controlled rotation gate around $x$ axis, with control on the first qubit and target on the second qubit.*

*Proof.* Without lose of generality, we can assume that $\gamma \neq 0$ and $\alpha = \alpha = 0$. Now, we can write

$$U = (V_1 \otimes V_2) e^{i\gamma Z \otimes Z} (V_3 \otimes V_4). \tag{3.3}$$

Then, similar to the way that we derive $\mathrm{C}Z$ and CNOT gates, we can write

$$\begin{aligned} U &= (V_1 \otimes V_2 R_z(-2\gamma)) CR_z(4\theta)(V_3 \otimes V_4), \\ &= (V_1 \otimes V_2 R_z(-2\gamma) H) CR_x(4\theta)(V_3 \otimes HV_4). \end{aligned} \tag{3.4}$$

Now we $W_i$s are easily derived from $V_i$s. □

The gates in class II, such as SWAP, cannot be bridged any better than the baseline implementation mentioned in Corollary 1. This fact is shown later using an special trait of these gates in Lemma 4.

## 3.3 Bridged Gates

Here we introduce the bridged $T$ gate for any $T$ that belongs to class I. This gate is shown under a linear connectivity constraints, so it can be used for any other connectivity constraint (by using the connecting path between the qubits). An example of this gate (bridging over 4 qubits) is shown in Figure 3.3. Note that $V_i$s and $\theta$ can be computed using Lemma 2

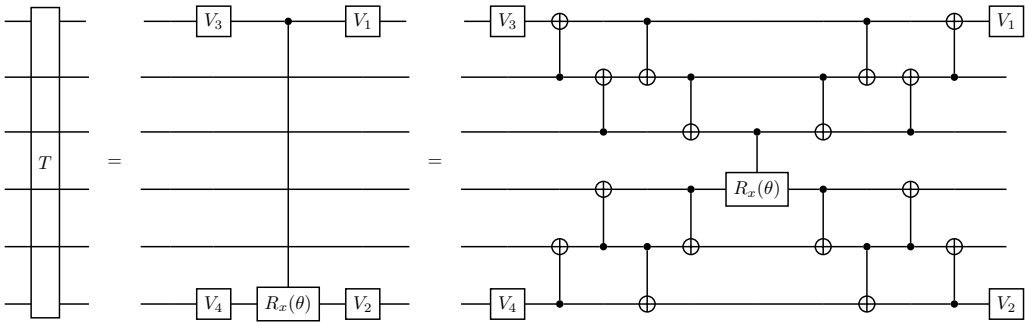


Figure 3.1: A bridged class I gate over 6 qubits

This circuit above can be mathematically defined as follows:

**Theorem 1** (Bridged class I gate)**.** *For any class I two-qubit gate $T$, A bridged $T$ gate defined on qubits $1, n \in V$, where every $(i, i+1) \in E$, can be implemented by the following sequence of gates:*

$$T = (V_1 \otimes V_2) B_n^\dagger \mathrm{C}R_x^{\lceil n/2 \rceil \to \lceil n/2 \rceil + 1}(\theta) B_n (V_3 \otimes V_4), \tag{3.5}$$

*where $V_i$s and $\theta$ are defined in Lemma 2, and $B_n$ for $n = 4$ is defined as follows:*

$$B_4 = (\mathrm{CNOT}^{1\to 2} \otimes \mathrm{CNOT}^{3\to 4})(\mathrm{CNOT}^{2\to 1} \otimes \mathrm{CNOT}^{4\to 3}), \tag{3.6}$$

*and for an even $n \geq 6$ it could be written as follows:*

$$\begin{aligned} B_n =& (\mathrm{CNOT}^{n/2-1\to n/2} \otimes \mathrm{CNOT}^{n/2+1\to n/2+2}) \\ & (\mathrm{CNOT}^{n/2-2\to n/2-1} \otimes \mathrm{CNOT}^{n/2+2\to n/2+3}) \\ & \prod_{i=1}^{n/2-3} (\mathrm{CNOT}^{i\to i+1} \otimes \mathrm{CNOT}^{i+3\to i+2} \otimes \mathrm{CNOT}^{n-i-1\to n-i-2} \otimes \mathrm{CNOT}^{n-i\to n-i+1}) \\ & (\mathrm{CNOT}^{3\to 2} \otimes \mathrm{CNOT}^{n-1\to n-2}) \\ & (\mathrm{CNOT}^{2\to 1} \otimes \mathrm{CNOT}^{n\to n-1}). \end{aligned} \tag{3.7}$$

*and, for any odd $n$ it could be recovered by removing $n$th qubit and its corresponding CNOTs from $B_n$.*

*Proof.* In order to prove the correctness of this circuit, we will show that it is equivalent to the baseline implementation of bridged $T$ gate (Corollary 1).

First, using the fact that $[\mathrm{CNOT}^{i\to j}, \mathrm{CNOT}\, i \to k] = 0$ and $[\mathrm{CNOT}^{i\to j}, \mathrm{CNOT}^{k\to j}] = 0$, we can show that $B_n$ (for even $n$) could be rearranged as:

$$B_n = \prod_{i=1}^{n/2-1} (\mathrm{CNOT}^{i\to i+1}\, \mathrm{CNOT}\, i+1 \to i) \otimes (\mathrm{CNOT}^{n-i\to n-i+1}\, \mathrm{CNOT}\, n-i+1 \to n-i). \tag{3.8}$$

This rearrangement for $n = 6$ is shown in Figure 3.3.

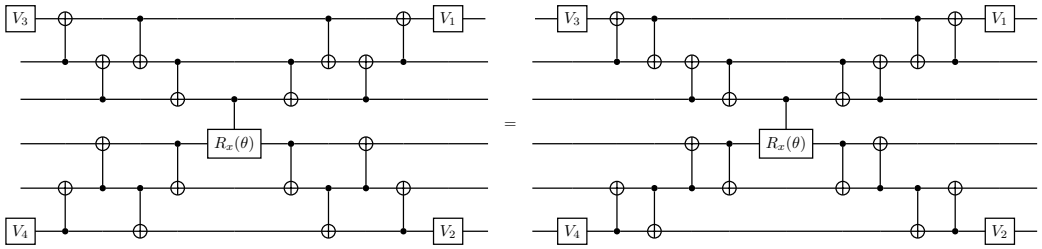


Figure 3.2: Rearrangement of CNOTs that is used in the proof

Then, using the fact that

$$\mathrm{SWAP}^{i,j} = \mathrm{CNOT}^{j\to i}\, \mathrm{CNOT}^{i\to j}\, \mathrm{CNOT}^{j\to i}, \tag{3.9}$$

it could be simplified further to

$$B_n = \prod_{i=1}^{n/2-1} (\mathrm{CNOT}^{i+1\to i}\, \mathrm{SWAP}^{i,i+1}) \otimes (\mathrm{CNOT}\, n-i+1 \to n-i\, \mathrm{SWAP}^{n-i,n-i+1}). \tag{3.10}$$

Now, by moving the rearranging the CNOTs to be applied after the SWAPs, we will have

$$B_n = \prod_{i=1}^{n/2-1} \mathrm{CNOT}^{n/2\to i} \otimes \mathrm{CNOT}^{n-i+1\to n/2+1} \prod_{i=1}^{n/2-1} \mathrm{SWAP}^{i,i+1} \otimes \mathrm{SWAP}^{n-i,n-i+1}. \tag{3.11}$$

This process is shown in Figure 3.3 for $n = 6$.

Finally, the second product term is similar to the one in Corollary 1, and the first product term is a sequence of CNOTs that all commute with $\mathrm{C}R_x^{n/2\to n/2+1}(\theta)$, so they will be canceled in $B_n^\dagger \mathrm{C}R_x^{n/2\to n/2+1}(\theta) B_n$.

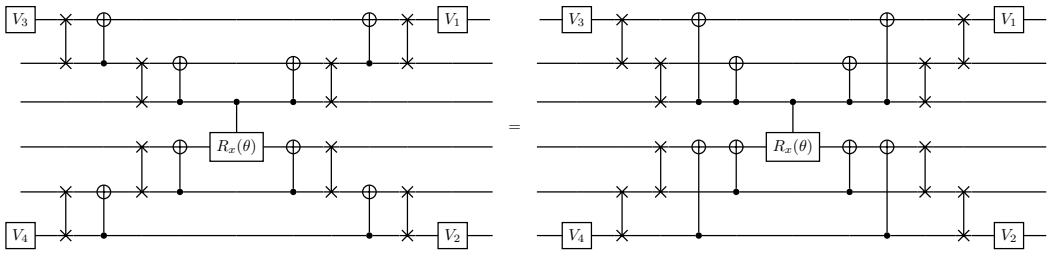


Figure 3.3: Moving CNOTs across the SWAPs that is used in the proof

$$
\begin{aligned}
B_n^\dagger \mathrm{C}R_x^{n/2\to n/2+1}(\theta) B_n = &\prod_{i=n/2-1}^{1} \mathrm{SWAP}^{i,i+1} \otimes \mathrm{SWAP}^{n-i,n-i+1} \\
&\mathrm{C}R_x^{n/2\to n/2+1}(\theta) \\
&\prod_{i=1}^{n/2-1} \mathrm{SWAP}^{i,i+1} \otimes \mathrm{SWAP}^{n-i,n-i+1} \\
&= \mathrm{C}R_x^{1\to n}(\theta)
\end{aligned}
\tag{3.12}
$$

Finally, by recalling Lemma 2,

$$(V_1 \otimes V_2) B_n^\dagger \mathrm{C}R_x^{\lceil n/2\rceil \to \lceil n/2\rceil+1}(\theta) B_n (V_3 \otimes V_4) = T. \tag{3.13}$$

In order to prove the correctness of the circuit for odd $n$ as well, we can simply remove the $n$th qubit and its corresponding CNOTs from $B_n$. This fact will be propagated through the proof and every other thing will be the same. □

The formula above shows that the circuit uses $4n+O(1)$ CNOTs with the depth $n+O(1)$ (noting that $\mathrm{C}R_x$ needs two CNOTs [30, chapter 4]). While in comparison to the baseline implementation (see Corollary 1), this circuit has 33% improvement in the number of CNOTs and 66% improvement in the depth. Even for the special case of CNOT, this circuit has approximately the same number of CNOTs as the bridged CNOT gate in Equation 2.4 and Figure 2.1.2, however, its depth is improved by 75%.

## 3.4 Proof of Optimality

Here we aim to proof the optimality of the bridged $T$ gate described in the previous section for class I gates, and the baseline implementation for class II gates. By optimality we mean that the number of CNOTs and the depth of the circuit is the minimum possible (upto an $O(1)$ constant).

We start by introducing the the tools and the intuition behind their definitions, in addition to the core lemma that is used in the proofs. Then, we will prove the results for class I and class II gates separately. The first tool that we need is conjugation that is commonly used in group theory [47].

**Definition 5** (Conjugation)**.** *For any two gates $A$ and $B$, we define the conjugation of $A$ by $B$ as*

$$B^\dagger A B. \tag{3.14}$$

The intuition behind the conjugation of gates is that if the conjugation of $A$ by $B$ is $C$, then applying $A$ before $A$ is equivalent to applying $C$ after $B$. Informally this could be phrased as by crossing $A$ over $B$, $A$ will be transformed to $C$. Figure 3.4 shows a few conjugations that we will use in the proofs.

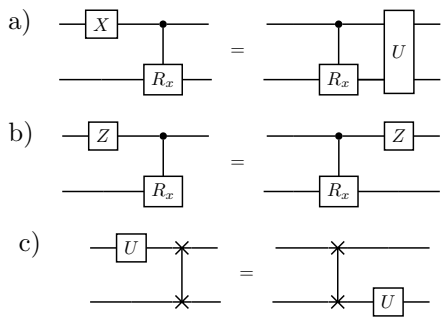


Figure 3.4: a) Conjugation of $X \otimes I$ by $\mathrm{C}R_x$ is a two-qubit gate. b) Conjugation of $Z \otimes I$ by $\mathrm{C}R_x$ is $Z \otimes I$. c) Conjugation of $U \otimes I$ by SWAP is $I \otimes U$

.

Furthermore, this effect, will help us to develop a tool to properly analyse the flow of information in a circuit. To complete this idea we define a sense of locality, using the following simple definition:

**Definition 6** (Trivial/Nontrivial action)**.** *A gate $U$ defined on a set of qubits $Q$ is acting trivially on qubit $q \in Q$ if $U$ can be written as*

$$U = U' \otimes I_q \tag{3.15}$$

*where $U'$ is a gate defined on $Q \setminus \{q\}$ and $I_q$ is the identity gate on qubit $q$. Respectively, $U$ is acting nontrivially on $q$ if it cannot be written in the above form.*

Now, these proofs are based on the idea that if two gate sequence $A = A_n \dots A_2 A_1$ and $B = B_m \dots B_2 B_1$ are equal, then the conjugation of any arbitrary gate $U$ by $A$ is equal to the conjugation of $U$ by $B$. Or in another word,

**Corollary 2.** *Assume $A = A_n \dots A_2 A_1$ and $B = B_m \dots B_2 B_1$ are two gate sequences, then the necessary condition for $A = B$ is that for any arbitrary gate $U$, the conjugation of $U$ by $A$ must be equal to the conjugation of $U$ by $B$.*

This corollary will be amazingly useful by noting the fact that the conjugation of $U$ by $A$ can be iteratively computed as follows:

$$\begin{cases} U_1 = \text{conjugation of } U \text{ by } A_1 \\ U_{i+1} = \text{conjugation of } U_i \text{ by } A_{i+1} \end{cases} \Rightarrow U_n = \text{conjugation of } U \text{ by } A. \tag{3.16}$$

Now, we can start proving the optimality of the bridged $T$ gate for different classes, starting by class I.

### 3.4.1 Class I

The sketch of this proof consists of first reducing the problem to bridging a $\mathrm{C}R_x^{1\to n}$ gate. Then because of the nontrivial (on 1 and $n$) conjugations of $X_1$ and $Z_n$ by $\mathrm{C}R_x^{1\to n}$, we can conclude that any sequence of gates that is bridged $\mathrm{C}R_x^{1\to n}$ needs to have a minimum number of CNOTs, and a minimum depth, to produce such nontrivial conjugations. These two bounds are achieved by the bridged $T$ gate.

So, to start the proof, we first establish two facts about the conjugations:

**Corollary 3** (No-go for one-qubit gates)**.** *For any gate acting trivially on $t$, its conjugation by any one-qubit gate on $t$ is still trivial on $t$.*

This corollary informally states that if we want to create nontrivial action on $t$ by conjugation, we need use two-qubit gates. This fact can be easily shown by simply combining definition of trivial action and conjugation.

**Lemma 3** (No-move for class I gates)**.** *For any gate $A$ acting nontrivially on $n$ and trivially on $t$, its conjugation $B$ by any class I gate $U$ acting on $n$ and $t$, is nontrivial on $n$.*

*Proof.* Noting that $A$ must be trivial on $t$ and nontrivial on $n$, $A$ could be decomposed as $A' \otimes I_t$. Using the definition of conjugation in form of $U^\dagger AU = B$, we can write

$$\begin{aligned} B &= U^\dagger (A' \otimes I_t) U \\ &= (V_3^\dagger \otimes V_4^\dagger) \mathrm{C}R_x(-\theta)(V_1^\dagger A' V_1 \otimes I) \mathrm{C}R_x(\theta)(V_3 \otimes V_4) \\ &= (V_3^\dagger \otimes V_4^\dagger)(|0\rangle\langle 0| + R_x(-\theta) \otimes |1\rangle\langle 1|)(V_1^\dagger A' V_1 \otimes I)(|0\rangle\langle 0| + R_x(\theta) \otimes |1\rangle\langle 1|)(V_3 \otimes V_4) \\ &= (V_3^\dagger \otimes V_4^\dagger)(V_1^\dagger A' V_1 \, |0\rangle\langle 0| + R_x(-\theta) V_1^\dagger A' V_1 R_x(\theta) \, |1\rangle\langle 1|)(V_3 \otimes V_4) \end{aligned} \tag{3.17}$$

We have used the decomposition of Lemma 2 for $U$ and expanded $\mathrm{C}R_x$ in terms of $|0\rangle\langle 0|$ and $|1\rangle\langle 1|$. Then, in order to proof by contradiction, we can assume that $B = I_n \otimes B'$, then $\mathrm{tr}_n(B) = B'$, so

$$B' = \mathrm{tr}_n(B) = \mathrm{tr}_n(A') \, |0\rangle\langle 0|_n + \mathrm{tr}_n(A') \, |1\rangle\langle 1|_n = \mathrm{tr}_n(A') I_t \tag{3.18}$$

which means that $B = \mathrm{tr}_n(A') \otimes I_n \otimes I_t$, that results in $A = \mathrm{tr}_n(A') \otimes I_n \otimes I_t$, as well. Which is a contradiction with the assumption that $A$ is nontrivial on $n$. □

Now, we can prove the optimality of the bridged $T$ gate for class I gates in the theorem below.

**Theorem 2.** *Any bridged $T$ gate where $T$ is a class I gate acting on qubits 1 and $n$ with linear connectivity, needs at least $4n + O(1)$ CNOTs and has at least $n$ depth.*

*Proof.* Without losing generality, we can assume that $T = \mathrm{C}R_x^{1 \to n}(\theta)$, because of Lemma 2. We can also show that the conjugation of $X_1$ and $Z_n$ (one qubit gates acting on qubits 1 and $n$ respectively) by $T$ is nontrivial on 1 and $n$ Then, we assume that a sequence of gates $A = A_\ell \dots A_2 A_1$ is equivalent to $T$, the conjugation of $X_1$ and $Z_n$ under $A$ must be nontrivial on 1 and $n$ as well.

In order to simplify the notation, we name the conjugation of $X_1$ and $Z_n$ by $A_i \dots A_1$ as $\xi_i$ and $\zeta_i$ respectively. Now, based on the fact that one-qubit gates could not create nontrivial action (Corollary 3), and because of the connectivity constraint, in order to have nontrivial action on $n$ by $\xi_n$, we need to a CNOT gate acting on $n$ and $n-1$, or in another word $A_j = \mathrm{CNOT}^{n \rightleftharpoons n-1}$ for some $j \leq \ell$. Also $\xi_{j-1}$, must act nontrivially on $n-1$. Moreover, using Lemma 3 we know that using one CNOT ($A_j$) will leave $\xi_j$ acting nontrivially on $n-1$, which is not desired. This means that at least two CNOTs are necessary, or in another word $A_k = \mathrm{CNOT}^{n-1 \rightleftharpoons n-1 \pm 1}$ for $j < k \leq \ell$. By recursively applying this argument, we can conclude that a sequence of CNOT pairs acting on subsequent qubits from 1 to $n$ is necessary solely to make $\xi_n$ to have nontrivial action on $n$. A similar argument could hold for $\zeta_n$. Now because of the fact that these two sequences are going in opposite directions, and they cannot have more than $O(1)$ gates in common, we can conclude that at least $4(n-1) + O(1)$ CNOTs are necessary to have nontrivial action on 1 and $n$ by $\xi_n$ and $\zeta_n$. Note that the minimum depth could be easily derived from the fact that a CNOT between $n$ and $n-1$ should take place after a CNOT between $n-1$ and $n-2$ and so on. □

### 3.4.2 Class II

For the class II, we will prove in a similar manner that there is no solution better than the baseline implementation (Corollary 1). Here we start by a lemma that shows the distinctive feature of the class II gates to be used in the proof.

**Lemma 4.** *for any gate $A$ acting nontrivially on $n$ and trivially on $t$, its conjugation $B$ by a class II gate $U$ acting on $n$ and $t$, is nontrivial on $t$.*

*Proof.* Similar to the approach in proving Lemma 3, using proof by contradiction, we can assume that $B = B' \otimes I_t$, then

$$\begin{gathered}(A' \otimes I_t)U = U(B' \otimes I_t) \\ (A'V_1 \otimes V_2)e^{i\alpha X\otimes X+i\beta Y\otimes Y+i\gamma Z\otimes Z}(V_3 \otimes V_4) = (V_1 \otimes V_2)e^{i\alpha X\otimes X+i\beta Y\otimes Y+i\gamma Z\otimes Z}(V_3B' \otimes V_4).\end{gathered} \tag{3.19}$$

Then by multiplying $(V_1^\dagger \otimes V_2^\dagger)$ from left and $(V_3^\dagger \otimes V_4^\dagger)$ from right, we can write

$$(V_1^\dagger A'V_1 \otimes I_t)e^{i\alpha X\otimes X+i\beta Y\otimes Y+i\gamma Z\otimes Z} = e^{i\alpha X\otimes X+i\beta Y\otimes Y+i\gamma Z\otimes Z}(V_3B'V_3^\dagger \otimes I_t). \tag{3.20}$$

Now, by expanding the exponential, we will have

$$\begin{aligned}(V_1^\dagger A'V_1 \otimes I_t)&(c_1I \otimes I + ic_2X \otimes X + ic_3Y \otimes Y + ic_4Z \otimes Z) \\ &=(c_1I \otimes I + ic_2X \otimes X + ic_3Y \otimes Y + ic_4Z \otimes Z)(V_3B'V_3^\dagger \otimes I_t).\end{aligned} \tag{3.21}$$

Where $c_i$s are some constants. Finally, by multiplying both sides by $(I_n \otimes P)$ where $P$ is one of $I, X, Y, Z$, and then by tracing out the qubit $t$, we will have the following equations:

$$\begin{cases} V_1^\dagger A'V_1 = V_3B'V_3^\dagger \\ V_1^\dagger A'V_1X = XV_3B'V_3^\dagger & \text{if } \alpha \neq 0 \\ V_1^\dagger A'V_1Y = YV_3B'V_3^\dagger & \text{if } \beta \neq 0 \\ V_1^\dagger A'V_1Z = ZV_3B'V_3^\dagger & \text{if } \gamma \neq 0. \end{cases} \tag{3.22}$$

It means that if two of $\alpha, \beta, \gamma$ are non-zero, then $A$ must have the form $A' = A'' \otimes I_n$ which is against the assumption that $A$ is nontrivial on $n$. □

Now, we can prove the main theorem.

**Theorem 3.** *Any bridged gate where $T$ is a class II gate acting on qubits 1 and $n$ with linear connectivity, needs at least $6n + O(1)$ CNOTs and has at least $3n$ depth.*

*Proof.* This proof is similar to the proof of Theorem 2, assume that $A_j = \text{CNOT}^{1\to2}$ is the only CNOT applying on qubits 1 and 2. We have assumed the direction without loose of generality as the other direction could be proved similarly. Now, the conjugation of $Z_1$ (acting on qubit 1) by $A_1^\dagger \dots A_{j-1}^\dagger$ that we call it $U_1$ must be nontrivial on qubit 1. Using Lemma 4, the conjugation of $U_1$ by $T$ must act nontrivial on $n$. On the other hand the conjugation of $U_1$ under $A_j \dots A_1$ is equal to the conjugation of $Z_1$ under $A_j$ which is trivial on 2 and will be trivial on any $i \geq 2$ as $A_j$ is the only CNOT between 1 and 2, which is not compatible with the assumption of acting nontrivially on $n$. This means that at least two CNOTs are necessary. We call the second CNOT as $A_k$ where $j < k$. Now, using Lemma 3 we know that the conjugation of $U_1$ by $A_k \dots A_1$ is still nontrivial on 2 that is not desired, so the third CNOT is necessary as well. Recursively we can conclude that a sequence made of three CNOTs is necessary. All of these arguments could be applied in the other direction, so we can conclude that at least $6n + O(1)$ CNOTs are necessary. □

### 3.4.3 Extension to Arbitrary Connectivity

The extension to arbitrary connectivity is can be done by defining layers. Layers are used to linearizing the vertices of the graph with respect to their distance from a node. Then, the proof of Theorem 2 and Theorem 3 Given a connectivity graph $G = (V, E)$, and a target gate $T$ that is defined on $a, b \in V$, we can define a set of layers $L_1 \dots L_k$, defined as

$$L_i = \{v \in V \mid \text{dist}(v, a) = i\}. \tag{3.23}$$

Where $\text{dist}(v, a)$ is the distance between $v$ and $a$ in $G$. Then, the whole proof could be re used, with the only difference that instead of acting nontrivially on $i$th qubit, now we care about acting nontrivially on one of the qubits in $L_i$. This means that the number of CNOTs and the depth will be $4\text{dist}(a, b) + O(1)$ and $\text{dist}(a, b) + O(1)$ respectively.

# Chapter 4

# Conclusion

## 4.1 Future works

As like as many other efforts, we assumed that any qubit in the device is used and is continuously carrying information. This might not be a bad idea in NISQ devices, where any single qubit is precious and the number of qubits is limited. However, because of its interesting results and its potential to be used in future devices, we will briefly discuss the case where we have ancilla qubits as well.

Bridging class I gates over $n$ ancilla qubit needs $2n + O(1)$ CNOTs and this number for class II gates is $4n + O(1)$, which shows a significant improvement in comparison to th previous results. TODO: show circuits for these cases mention network coding as well [17]

# Bibliography


[1] Alfred V. Aho, Ravi Sethi, and Jeffrey D. Ullman. *Compilers - Principles, Techniques and Tools*. English. 1st edition. Reading, Mass: Pearson, Jan. 1986. ISBN: 9780201100884.

[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.

[3] Randy Allen and Ken Kennedy. *Optimizing Compilers for Modern Architectures: A Dependence-based Approach*. English. 1st edition. San Francisco: Morgan Kaufmann, Oct. 2001. ISBN: 9781558602861.

[4] Indranil Banerjee and Dana Richards. "New Results on Routing via Matchings on Graphs". In: *Fundamentals of Computation Theory*. Springer Berlin Heidelberg, 2017, pp. 69–81. DOI: 10.1007/978-3-662-55751-8_7.

[5] A. Barenco et al. "Elementary gates for quantum computation". In: *Physical Review A* 52.5 (Nov. 1, 1995). ZSCC: 0005107, pp. 3457–3467. ISSN: 1050-2947, 1094-1622. DOI: 10.1103/PhysRevA.52.3457. arXiv: quant-ph/9503016. URL: http://arxiv.org/abs/quant-ph/9503016 (visited on 08/10/2023).

[6] 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).

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

[8] Andrew M. Childs et al. "A Theory of Trotter Error". In: *Physical Review X* 11.1 (Feb. 1, 2021). ZSCC: 0000002, p. 011020. ISSN: 2160-3308. DOI: 10.1103/PhysRevX.11.011020. arXiv: 1912.08854[cond-mat,physics:physics,physics:quant-ph]. URL: http://arxiv.org/abs/1912.08854 (visited on 03/24/2023).

[9] Andrew M. Childs et al. "Characterization of universal two-qubit Hamiltonians". In: *Quantum Information and Computation* 11.1 (). ZSCC: 0000037. ISSN: 15337146, 15337146. DOI: 10.26421/QIC11.1-2. arXiv: 1004.1645[quant-ph]. URL: http://arxiv.org/abs/1004.1645 (visited on 08/11/2023).

[10] Andrew M. Childs et al. "Toward the first quantum simulation with quantum speedup". In: *Proceedings of the National Academy of Sciences* 115.38 (Sept. 18, 2018). ZSCC: 0000363, pp. 9456–9461. DOI: 10.1073/pnas.1801723115. URL: https://www.pnas.org/doi/abs/10.1073/pnas.1801723115 (visited on 03/17/2023).

[11] Alexander Cowtan et al. "On the qubit routing problem". en. In: (2019). arXiv:1902.08091 [quant-ph], 32 pages. DOI: 10.4230/LIPIcs.TQC.2019.5. arXiv: 1902.08091[quant-ph]. URL: http://arxiv.org/abs/1902.08091 (visited on 02/03/2023).

[12] 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).

[13] Andrew W. Cross et al. "Validating quantum computers using randomized model circuits". In: *Phys. Rev. A* 100 (3 Sept. 2019), p. 032328. DOI: 10.1103/PhysRevA.100.032328. URL: https://link.aps.org/doi/10.1103/PhysRevA.100.032328.

[14] C.M. Dawson and M.A. Nielsen. "The Solovay-Kitaev algorithm". In: *Quantum Information and Computation* 6.1 (Jan. 2006), pp. 81–95. DOI: 10.26421/qic6.1-6.

[15] Richard P. Feynman. "Quantum mechanical computers". In: *Foundations of Physics* 16.6 (June 1, 1986). ZSCC: 0001758, pp. 507–531. ISSN: 1572-9516. DOI: 10.1007/BF01886518. URL: https://doi.org/10.1007/BF01886518 (visited on 03/23/2023).

[16] B. Foxen et al. "Demonstrating a Continuous Set of Two-qubit Gates for Near-term Quantum Algorithms". In: *Phys. Rev. Lett. 125, 120504 (2020)* 125.12 (Jan. 23, 2020), p. 120504. DOI: 10.1103/physrevlett.125.120504. arXiv: 2001.08343 [quant-ph].

[17] Tracey Ho and Desmond Lun. *Network coding: an introduction*. Cambridge University Press, 2008.

[18] Robert Hundt. "Quantum Languages, Compilers, and Tools". In: *Quantum Computing for Programmers*. Cambridge University Press, 2022, pp. 292–321. DOI: 10.1017/9781009099974.010.

[19] IBM. *IBM Unveils Breakthrough 127-Qubit Quantum Processor*. Nov. 16, 2021. URL: https://newsroom.ibm.com/2021-11-16-IBM-Unveils-Breakthrough-127-Qubit-Quantum-Processor.

[20] 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].

[21] S. Jordan. *Quantum Algorithm Zoo*. Mar. 17, 2023. URL: https://quantumalgorithmzoo.org.

[22] Navin Khaneja and Steffen J. Glaser. "Cartan decomposition of SU(2n) and control of spin systems". In: *Chemical Physics* 267.1-3 (June 2001), pp. 11–23. DOI: 10.1016/s0301-0104(01)00318-4.

[23] Aleks Kissinger and Arianne Meijer-van de Griend. *CNOT circuit extraction for topologically-constrained quantum memories*. arXiv:1904.00633. ZSCC: 0000062 type: article. arXiv, May 26, 2019. arXiv: 1904.00633[quant-ph]. URL: http://arxiv.org/abs/1904.00633 (visited on 07/19/2023).

[24] B. Kraus and J. I. Cirac. "Optimal Creation of Entanglement Using a Two–Qubit Gate". In: *Physical Review A* 63.6 (May 2001). arXiv:quant-ph/0011050, p. 062309. ISSN: 1050-2947, 1094-1622. DOI: 10.1103/PhysRevA.63.062309. URL: http://arxiv.org/abs/quant-ph/0011050 (visited on 02/13/2023).

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

[26] 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).

[27] Ashley Montanaro. "Quantum algorithms: an overview". In: *npj Quantum Information* 2.1 (Jan. 2016). DOI: 10.1038/npjqi.2015.23.

[28] Prakash Murali et al. "Noise-Adaptive Compiler Mappings for Noisy Intermediate-Scale Quantum Computers". In: *Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems*. ACM, Apr. 2019. DOI: 10.1145/3297858.3304075.

[29] Beatrice Nash, Vlad Gheorghiu, and Michele Mosca. "Quantum circuit optimizations for NISQ architectures". In: *Quantum Science and Technology* 5.2 (Mar. 31, 2020). ZSCC: 0000081, p. 025010. ISSN: 2058-9565. DOI: 10.1088/2058-9565/ab79b1. arXiv: 1904.01972[quant-ph]. URL: http://arxiv.org/abs/1904.01972 (visited on 03/24/2023).

[30] Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information: 10th Anniversary Edition*. ZSCC: NoCitationData[s0]. Dec. 9, 2010. DOI: 10.1017/CBO9780511976667. URL: https://www.cambridge.org/highereducation/books/quantum-computation-and-quantum-information/01E10196D0A682A6AEFFEA52D53BE9AE (visited on 08/14/2023).

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

[32] Alexandru Paler, Alwin Zulehner, and Robert Wille. "NISQ circuit compilation is the travelling salesman problem on a torus". In: *Quantum Science and Technology* 6.2 (Mar. 2021). ZSCC: 0000010, p. 025016. ISSN: 2058-9565. DOI: 10.1088/2058-9565/abe665. URL: https://dx.doi.org/10.1088/2058-9565/abe665 (visited on 08/10/2023).

[33] Jessica Pointing et al. "Quanto: Optimizing Quantum Circuits with Automatic Generation of Circuit Identities". In: (Nov. 22, 2021). DOI: 10.48550/ARXIV.2111.11387. arXiv: 2111.11387 [quant-ph].

[34] John Preskill. "Quantum computing in the NISQ era and beyond". In: *Quantum* 2 (2018), p. 79.

[35] Qiskit contributors. *Qiskit: An Open-source Framework for Quantum Computing*. 2023. DOI: 10.5281/zenodo.2573505.

[36] Vivek V. Shende, Stephen S. Bullock, and Igor L. Markov. "Synthesis of Quantum Logic Circuits". In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 25.6 (June 2006). arXiv:quant-ph/0406176, pp. 1000–1010. ISSN: 0278-0070, 1937-4151. DOI: 10.1109/TCAD.2005.855930. URL: http://arxiv.org/abs/quant-ph/0406176 (visited on 02/03/2023).

[37] Vivek V. Shende, Igor L. Markov, and Stephen S. Bullock. *Minimal Universal Two-qubit Quantum Circuits*. ZSCC: NoCitationData[s0]. Mar. 12, 2004. DOI: 10.1103/PhysRevA.69.062321. arXiv: quant-ph/0308033. URL: http://arxiv.org/abs/quant-ph/0308033 (visited on 07/25/2023).

[38] 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).

[39] Seyon Sivarajah et al. "t$—$ket$\rangle$ : A Retargetable Compiler for NISQ Devices". In: *Quantum Science and Technology* 6.1 (Jan. 1, 2021). ZSCC: NoCitationData[s0], p. 014003. ISSN: 2058-9565. DOI: 10.1088/2058-9565/ab8e92. arXiv: 2003.10611[quant-ph]. URL: http://arxiv.org/abs/2003.10611 (visited on 03/24/2023).

[40] P. B. M. Sousa and R. V. Ramos. *Universal quantum circuit for n-qubit quantum gate: A programmable quantum gate*. arXiv:quant-ph/0602174. ZSCC: 0000045 type: article. arXiv, May 29, 2006. DOI: 10.48550/arXiv.quant-ph/0602174. arXiv: quant-ph/0602174. URL: http://arxiv.org/abs/quant-ph/0602174 (visited on 08/11/2023).

[41] Masuo Suzuki. "General theory of fractal path integrals with applications to many-body theories and statistical physics". In: *Journal of Mathematical Physics* 32.2 (Feb. 1991), pp. 400–407. DOI: 10.1063/1.529425.

[42] Hale F Trotter. "On the product of semi-groups of operators". In: *Proceedings of the American Mathematical Society* 10.4 (1959), pp. 545–551.

[43] Robert R. Tucci. *An Introduction to Cartan's KAK Decomposition for QC Programmers*. arXiv:quant-ph/0507171. ZSCC: 0000059 type: article. arXiv, July 18, 2005. DOI: 10.48550/arXiv.quant-ph/0507171. arXiv: quant-ph/0507171. URL: http://arxiv.org/abs/quant-ph/0507171 (visited on 07/19/2023).

[44] Farrokh Vatan and Colin Williams. "Optimal Quantum Circuits for General Two-Qubit Gates". In: *Physical Review A* 69.3 (Mar. 22, 2004). ZSCC: 0000332, p. 032315. ISSN: 1050-2947, 1094-1622. DOI: 10.1103/PhysRevA.69.032315. arXiv: quant-ph/0308006. URL: http://arxiv.org/abs/quant-ph/0308006 (visited on 03/24/2023).

[45] Farrokh Vatan and Colin P. Williams. *Realization of a General Three-Qubit Quantum Gate*. arXiv:quant-ph/0401178. ZSCC: 0000035 type: article. arXiv, Jan. 30, 2004. DOI: 10.48550/arXiv.quant-ph/0401178. arXiv: quant-ph/0401178. URL: http://arxiv.org/abs/quant-ph/0401178 (visited on 07/19/2023).

[46] G. Vidal and C. M. Dawson. "A universal quantum circuit for two-qubit transformations with three CNOT gates". In: *Physical Review A* 69.1 (Jan. 2004). arXiv:quant-ph/0307177, p. 010301. ISSN: 1050-2947, 1094-1622. DOI: 10.1103/PhysRevA.69.010301. URL: http://arxiv.org/abs/quant-ph/0307177 (visited on 02/13/2023).

[47] Eric W. Weisstein. *Conjugation*. ZSCC: NoCitationData[s0]. URL: https://mathworld.wolfram.com/ (visited on 08/21/2023).

[48] Robert Wille, Stefan Hillmich, and Lukas Burgholzer. "Efficient and Correct Compilation of Quantum Circuits". In: *2020 IEEE International Symposium on Circuits and Systems (ISCAS)*. 2020 IEEE International Symposium on Circuits and Systems (ISCAS). ZSCC: 0000010 ISSN: 2158-1525. Oct. 2020, pp. 1–5. DOI: 10.1109/ISCAS45731.2020.9180791.

[49] 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).

[50] 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).