

Bridging Gates over Qubits

Seyed Sajad Kahani
Supervisor: Prof. Dan Browne

August 24, 2023

Contents

| | | |
|----------|-----------------------------------------------|-----------|
| 1 | Introduction | 1 |
| 2 | Background | 3 |
| 2.1 | Quantum Compilation | 3 |
| 2.1.1 | Gate synthesis | 4 |
| 2.1.2 | Qubit Allocation and Routing | 5 |
| 2.1.3 | Optimisations | 7 |
| 3 | Discussion | 9 |
| 3.1 | Problem Statement | 9 |
| 3.2 | Two-qubit Gate Classes | 9 |
| 3.3 | Bridged Gates | 10 |
| 3.4 | Proof of Optimality | 12 |
| 3.4.1 | Class I | 15 |
| 3.4.2 | Class II | 16 |
| 3.4.3 | Extension to Arbitrary Connectivity | 16 |
| 3.5 | Summary of Results | 18 |
| 4 | Conclusion | 19 |
| 4.1 | Future works | 19 |
| 4.1.1 | Implementation | 19 |
| 4.1.2 | Simultaneous Bridging | 19 |
| 4.1.3 | Ancillas | 19 |
| 4.2 | Final Remark | 19 |

Abstract

Quantum computers promise to revolutionise computing by harnessing the power of quantum mechanics. However, current quantum hardware is limited by restricted qubit connectivity that necessitates the extensive use of expensive SWAP gates during compilation. This paper introduces an optimal method to implement two-qubit gates between non-adjacent qubits. We formally define the bridging problem and present circuits that bridge a class of two-qubit gates over multiple qubits. Our circuit uses only $4n + O(1)$ CNOTs to bridge key entangling gates like CNOT, improving by 33% over baseline SWAP methods. We further prove the optimality of this circuit, and the optimality of the baseline SWAP method for other gates like SWAP. Beyond reducing gate counts, our techniques offer depth improvements up to 66%. The results generalise to arbitrary device graphs, where n is the distance between the target qubits. Our bridging techniques will help mitigate connectivity constraints and may serve as useful components for routing in near-term quantum computers. Further research directions are outlined including simultaneous bridging, ancilla-assisted bridging, and practical implementations.

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 conceptualisation of quantum computation [15], it has been believed that quantum computers could revolutionise 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. Algorithms for search, graph traversal, solving linear equations [27], machine learning and optimisation [21] have been developed.

Despite significant efforts, we are still far from fully utilising 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], characterised 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 substantial 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 researchers typically try to minimise the number of SWAPs [7, 35, 39, 20, 26]. While a relatively overlooked 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 that address the inherent cost of each SWAP gate. There are a few proposed techniques to reduce the cost of SWAP gates, such as embedding SWAPs within other 2-qubit gates in the 2QAN compiler [25], or compiling a class of circuits directly into CNOT gates [23, 29]. In this study, our focus is on the primary usage of SWAP gates - enabling connectivity between non-adjacent qubits. We analyse the possibility of simplification in various connectivity scenarios.

In the following, we formally define the problem of bridging as applying gates to non-adjacent qubits and review the previous efforts on that. We then introduce a method for implementing a class of two-qubit gates on non-adjacent qubits with 33% fewer CNOTs and 66% less depth compared to the baseline method. Additionally, we prove tight lower bounds for bridging each class of gates, demonstrating the optimality of our method. The introduced method could be beneficial for different approaches to impose connectivity constraints, serving as a subroutine in many widely-used routing algorithms. Furthermore, several directions for further research exist beyond our method, such as exploring the utili-

sation of ancillas in bridging or the simultaneous bridging of gates, all of which may lead to further useful results.

The remainder of this thesis is organised as follows. Chapter 2 reviews essential background about quantum compilation and related works. In Chapter 3, we present our method, along with its proofs of correctness and optimality for different classes. Finally, Chapter 4 concludes and discusses avenues for future work.

Chapter 2

Background

We should be concerned with two key findings from this chapter. First, the methodology employed to deconstruct the problem of quantum compilation, along with its underlying assumptions. Secondly, the facts and techniques that will also be used for our research. 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 specific hardware. Given that the problem of finding the most optimal circuit (with respect to a sense of complexity such as depth or number of gates) has been proven to be NP-hard [38], the research primarily centres 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 is 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 optimises 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 optimises the circuit (where the optimisation 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, optimisations that correspond to various degrees of freedom (e.g., qubit assignments, choosing among equivalent subcircuits) can be applied. They are commonly pursuing these goals:

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

- **Preparation for constraints:** Minimise violation of the mentioned constraints before imposing them.
- **Error mitigation:** Reduce 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 (optimisation of connectivity as a soft constraint), routing (imposing connectivity constraint), and further optimisations (reducing complexity). Yet the order of these transformations is not the same everywhere, for example, some [25, 35] defer the gate synthesis after the routing while some [48, 39] do it otherwise.

With this background established, we now dive into details on gate set and connectivity constraints, and circuit optimisation 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 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 the 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 constraint 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 negligible cost, there is a famous result for the synthesis of one-qubit gates, called Solovay-Kitaev theorem [14], that formalises 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 up to 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 $\text{fSim}(\theta, \phi)$ [16] or other Hamiltonian evolutions [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*

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

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 synthesise 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 simulation purposes [10]. Most of the research is 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 (aka Lie-Trotter formula) [39, 35], there are a few efforts to analyse higher-order errors [8] or to use other decomposition, such as a random sampling decomposition (QDRIFT) [6]. Besides 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.

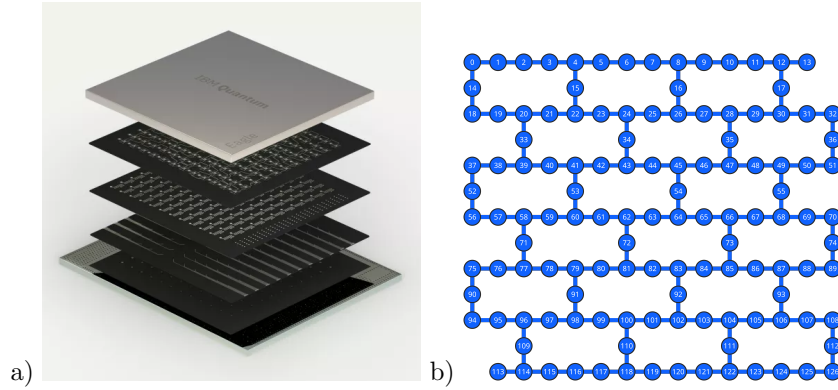


Figure 2.1: a) IBM Eagle processor, a 127 qubit device, b) IBM Eagle connectivity graph [19]

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 minimises 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 breakdown 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 surprising 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 implemented 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 algorithms 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 have 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 go beyond the SWAP gates, like using bridged CNOT (sometimes 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 SWAPs are explicitly used, which is shown in Figure 2.1.2b.

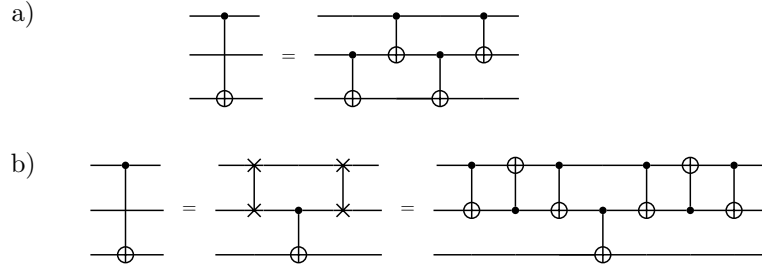


Figure 2.2: a) A bridged CNOT gate over one qubit, b) The baseline implementation of bridged CNOT gate over one qubit

This idea could be broadened 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),

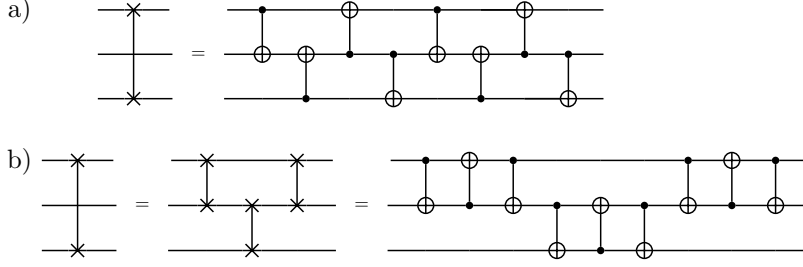


Figure 2.3: a) A bridged SWAP gate over one qubit, b) The baseline implementation of bridged SWAP gate over one qubit

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 implementation 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:*

$$T^{1,n} = \prod_{i=n/2-1}^1 \text{SWAP}^{i,i+1} \otimes \text{SWAP}^{n-i+1,n-i} T^{n/2,n/2+1} \prod_{i=1}^{n/2-1} \text{SWAP}^{i,i+1} \otimes \text{SWAP}^{n-i+1,n-i}, \quad (2.2)$$

and for an odd n we can respectively use:

$$T^{1,n} = \text{SWAP}^{n,n-1} T^{1,n-1} \text{SWAP}^{n,n-1}. \quad (2.3)$$

Where $\text{SWAP}^{i,j}$ 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 generalisation 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 $3n + O(1)$.

$$\text{CNOT}^{1 \rightarrow n} = \left[\prod_{i=n-2}^2 \text{CNOT}^{i+1 \rightarrow i} \prod_{i=1}^{n-1} \text{CNOT}^{i+1 \rightarrow i} \right]^2, \quad (2.4)$$

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

2.1.3 Optimisations

Circuit optimisation is the set of techniques to transform the circuit into a more efficient one, without imposing or lifting any constraint. Circuit optimisation 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].

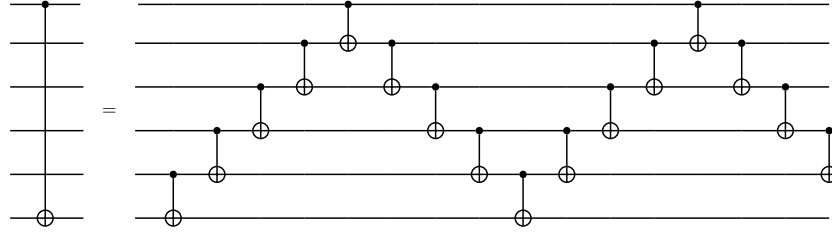


Figure 2.4: A bridged CNOT gate over 6 qubits

Therefore the optimisation 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 optimisation level [35, 39], which is shared with classical compilers as well [1].

Chapter 3

Discussion

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, along 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.

3.1 Problem Statement

While our aim in the broadest term is to find alternatives to the SWAP gate to apply gates with respect to connectivity constraints, we start with a more specific problem. We seek the 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 any compilation process, in contrast to theirs that defines 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 combinations 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*

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

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 CR_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*

$$U = W_1 \otimes W_2 CR_x(\theta) W_3 \otimes W_4, \quad (3.2)$$

where $CR_x(\theta)$ is a controlled rotation gate around x axis, with control on the first qubit and target on the second qubit.

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

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

Then, similar to the way that we derive CZ and CNOT gates, we can write

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

Now W_i s are easily derived from V_i s. \square

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 constraint, 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 θ can be computed using Lemma 2

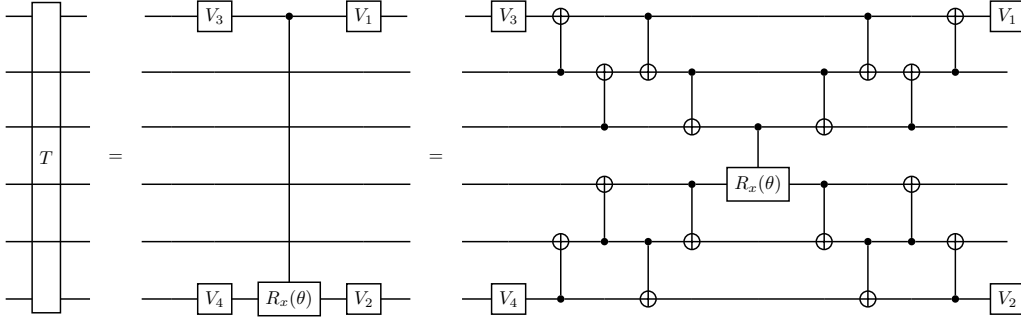


Figure 3.1: A bridged class I gate over 6 qubits

This circuit above can be mathematically defined as follows:

Proposition 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:*

$$T = (V_1 \otimes V_2) B_n^\dagger CR_x^{[n/2] \rightarrow [n/2]+1}(\theta) B_n (V_3 \otimes V_4), \quad (3.5)$$

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

$$B_4 = (\text{CNOT}^{1 \rightarrow 2} \otimes \text{CNOT}^{3 \rightarrow 4}) (\text{CNOT}^{2 \rightarrow 1} \otimes \text{CNOT}^{4 \rightarrow 3}), \quad (3.6)$$

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

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

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 the bridged T gate (Corollary 1).

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

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

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

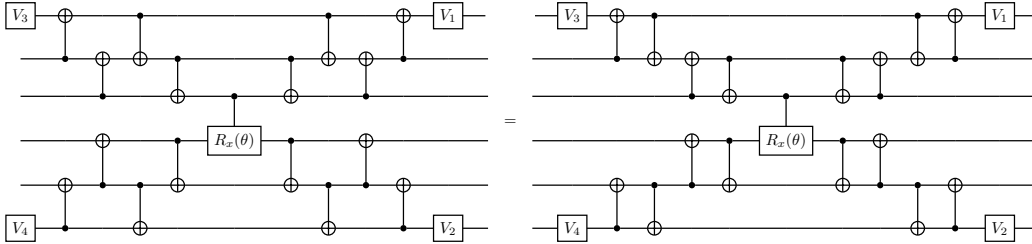


Figure 3.2: Rearrangement of CNOTs that is used in the proof

Then, using the fact that

$$\text{SWAP}^{i,j} = \text{CNOT}^{j \rightarrow i} \text{CNOT}^{i \rightarrow j} \text{CNOT}^{j \rightarrow i}, \tag{3.9}$$

it could be simplified further to

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

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

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

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 $CR_x^{n/2 \rightarrow n/2+1}(\theta)$, so they will be canceled in $B_n^\dagger CR_x^{n/2 \rightarrow n/2+1}(\theta) B_n$.

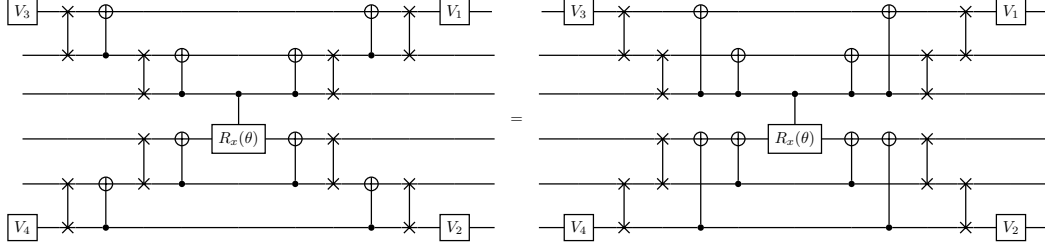


Figure 3.3: Moving CNOTs across the SWAPs that is used in the proof

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

Finally, by recalling Lemma 2,

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

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. \square

The formula above shows that the circuit uses $4n + O(1)$ CNOTs with the depth $n + O(1)$ (noting that $C R_x$ 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

Now, we aim to prove 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).

Here we first introduce the tools and the intuition behind their definitions, in addition to the core lemma that is used in the proofs. Then, we will prove a general lower bound for the depth, followed by the lower bounds for the number of CNOTs for class I and class II gates separately. Starting by 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

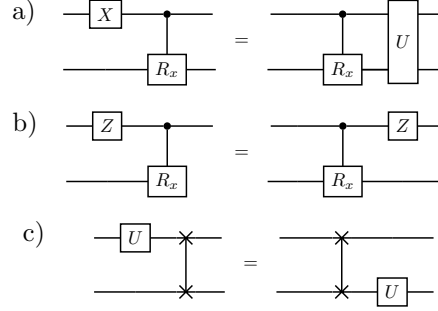


Figure 3.4: a) Conjugation of $X \otimes I$ by CR_x is a two-qubit gate. b) The conjugation of $Z \otimes I$ by CR_x is $Z \otimes I$. c) The conjugation of $U \otimes I$ by SWAP is $I \otimes U$.

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.

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*

$$U = U' \otimes I_q \quad (3.15)$$

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 sequences $A = A_n \dots A_2 A_1$ 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 of incredible useful if one notices that the conjugation of U by A can be computed iteratively as follows:

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

Now, here present a set of facts that use definition of trivial/nontrivial action to specify the conjugation of gates by each class of gates.

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.

Then, we independently prove the following lemma for class I gates, that informally states a single class I gate, cannot move nontrivial action (creating on one qubit and destroying on another).

Lemma 3 (No move for class I). *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 $U^\dagger A U = B$, we can write

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

We have used the decomposition of Lemma 2 for U and expanded $C R_x$ 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 $\text{tr}_n(B) = B'$, so

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

which means that $B = \text{tr}_n(A') \otimes I_n \otimes I_t$, that results in $A = \text{tr}_n(A') \otimes I_n \otimes I_t$. This contradicts the assumption that A is not trivial in n . \square

Afterward, by using a similar approach, we can prove that there is no gate can stay on just one of the qubits after conjugation by a class II gate, while this fact is not true for class I gates and has been shown in Figure 3.4.

Lemma 4 (No stationary for class II). *For any gate A that acts 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

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

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

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

Now, by expanding the exponential, we will have

$$\begin{aligned}
&(V_1^\dagger A' V_1 \otimes I_t) (c_1 I \otimes I + i c_2 X \otimes X + i c_3 Y \otimes Y + i c_4 Z \otimes Z) \\
&= (c_1 I \otimes I + i c_2 X \otimes X + i c_3 Y \otimes Y + i c_4 Z \otimes Z) (V_3 B' V_3^\dagger \otimes I_t).
\end{aligned} \tag{3.21}$$

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:

$$\begin{cases} V_1^\dagger A' V_1 = V_3 B' V_3^\dagger \\ V_1^\dagger A' V_1 X = X V_3 B' V_3^\dagger & \text{if } \alpha \neq 0 \\ V_1^\dagger A' V_1 Y = Y V_3 B' V_3^\dagger & \text{if } \beta \neq 0 \\ V_1^\dagger A' V_1 Z = Z V_3 B' V_3^\dagger & \text{if } \gamma \neq 0. \end{cases} \tag{3.22}$$

It means that if two of α, β, γ 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 . \square

Finally, using the previous lemma, we can prove a less restrictive claim for both classes I and II, that states that there is always a gate that will not stay on its qubit after conjugation.

Lemma 5 (Nonstationary for classes I and II). *For an arbitrary two-qubit gate T that is not a class nil gate and is defined on qubits a and b , there exists one-qubit gate U defined on a that the conjugation of U under T acts nontrivially on b .*

Proof. For class I, if we use Lemma 2 and the fact that the conjugation of a gate is a bijective function of that gate, then the problem reduces to the existence of such U for $T = CR_x^{a \rightarrow b}(\theta)$, which is trivially true by choosing $U = X$.

For class II, this is the direct result of Lemma 4. \square

Now, for the first result we can prove the minimum necessary depth for bridging any two-qubit gate that does not belong to class nil.

Proposition 2. *Any bridged T gate where T is not a class nil gate and acts on qubits 1 and n , and bridging is done with a linear connectivity constraint, the bridged gate has at least $n + O(1)$ depth.*

Proof. Using Lemma 5, there exists a U defined on qubit 1 where the conjugation of U by T acts nontrivially on qubit n . Now, if the bridged T gate is implemented by the gate sequence $A = A_\ell \dots A_2 A_1$, then the conjugation of U by A is nontrivial on n .

We name the conjugation of U by $A_j \dots A_2 A_1$ as u_j , and we want u_ℓ to be nontrivial on n . We know that the gates that can be applied on n are one-qubit gates and $\text{CNOT}^{n-1 \rightleftharpoons n}$. Corollary 3 states that one-qubit gates could not introduce nontrivial action in this case, so the only way is to apply $\text{CNOT}^{n-1 \rightleftharpoons n}$ at some point, which means that

$$A_j = \text{CNOT}^{n-1 \rightleftharpoons n} \otimes A'_j \text{ for some } j. \quad (3.23)$$

By $\text{CNOT}^{a \rightleftharpoons b}$ we mean the CNOT gate that can have arbitrary direction. Then, note that if u_{j-1} acts trivially on $n-1$, it would not be possible to make u_j nontrivial on n , therefore we can conclude that u_{j-1} acts nontrivially on $n-1$.

Now, we can repeat the same argument for $n-1$ and u_{j-1} , and we can conclude that

$$A_k = \text{CNOT}^{n-2 \rightleftharpoons n-1} \otimes A'_k \text{ for some } k < j, \quad (3.24)$$

and u_{k-1} acts nontrivially on $n-2$. Continuing this argument, we can find c_{n-1}, \dots, c_1 such that $c_i < c_{i+1}$ and

$$A_{c_i} = \text{CNOT}^{i \rightleftharpoons i+1} \otimes A'_{c_i}, \quad (3.25)$$

and u_{c_i-1} acts nontrivially on i . The existence of c_i guarantees that the depth of the circuit is at least $n-1$. \square

This lower bound is tight for class I gates, as the introduced bridged T gate for class I gates has the same depth. However, for class II gates, this lower bound might not be tight. Next, we will prove separate results for each class.

3.4.1 Class I

The sketch of this proof consists of first reducing the problem to bridging a $CR_x^{1 \rightarrow n}$ gate. Then because of the nontrivial (on 1 and n) conjugations of X_1 and Z_n by $CR_x^{1 \rightarrow n}$, we can conclude that any sequence of gates that is bridged $CR_x^{1 \rightarrow n}$ 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.

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

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

Proof. Without losing generality, we can assume that $T = CR_x^{1 \rightarrow n}(\theta)$, 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 also 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 ξ_i and ζ_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 ξ_n , we need a CNOT gate acting on n and $n-1$, or in other words

$$A_j = \text{CNOT}^{n-1 \Rightarrow n} \otimes A'_j \text{ for some } j. \quad (3.26)$$

Also ξ_{j-1} , must act nontrivially on $n-1$. Moreover, using Lemma 3 we know that using one CNOT (A_j) will leave ξ_j acting nontrivially on $n-1$, which is not desired. This means that at least two CNOTs are necessary, so

$$A_k = \text{CNOT}^{n-1 \Rightarrow n-1 \pm 1} \otimes A'_k \text{ for some } j < k. \quad (3.27)$$

Similarly ξ_{k-1} must act nontrivially on $n-1 \pm 1$.

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 ξ_n have nontrivial action on n . A similar argument could hold for ζ_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 ξ_n and ζ_n . \square

3.4.2 Class II

For the class II, we will prove in a similar way to class I that there is no solution better (in terms of number of CNOTs) than the baseline implementation (Corollary 1). Now, we can prove the main theorem for class II which states a lower bound for the CNOT count.

Theorem 2. *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.*

Proof. This proof shares some of its core ideas with the proof of Theorem 1. Assume that $A_j = \text{CNOT}^{1 \rightarrow 2}$ is the only CNOT applying on qubits 1 and 2. We have assumed the direction without losing generality as the other direction can be converted by local operations. Now, the conjugation of Z_1 (acting on qubit 1) by $A_1^\dagger \dots A_{j-1}^\dagger$ that we call U_1 must be nontrivial on qubit 1 and trivial everywhere else. 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. \square

3.4.3 Extension to Arbitrary Connectivity

Note that our circuit is valid for general connectivity and its correctness has been proved for an arbitrary connectivity constraint, but the proofs for optimality need to be extended. This extension can be done by defining layers to group the vertices of the graph with respect to their distance from a vertex $a \in V$. A visualization of these layers is shown in Figure 3.5. Formally, layers are defined as below.

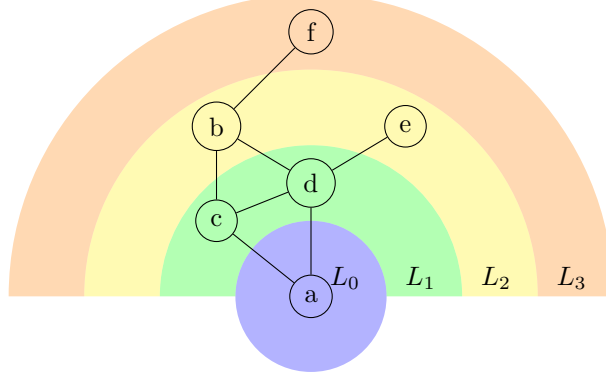


Figure 3.5: Visualization of layers

Definition 7 (Layers). *Given a graph $G = (V, E)$, and the vertex $a \in V$, we can define disjoint sets $L_1, L_k \subset V$ as*

$$L_i = \{b \mid \text{dist}(a, b) = i\}, \quad (3.28)$$

where $\text{dist}(a, b)$ is the distance between a and b .

By this definition, there is no edge between i and j if $|i - j| > 1$. This fact is used to prove the generalization of Proposition 2 and Theorems 1 and 2 for an arbitrary connectivity constraint. The lower bound for the depth is stated in the theorem below.

Proposition 3. *For any arbitrary connectivity constraint $G = (V, E)$, a bridged T gate where T is a class I or class II gate, acting on qubits $a, b \in V$, needs at least $\text{dist}(a, b) + O(1)$ depth.*

Proof. Using Lemma 5, we find U defined on a where the conjugation of U by T acts nontrivially on b . Now, we assume that $T = A_\ell \dots A_1$ and u_i is the conjugation of U by $A_i \dots A_1$.

Also, we define layers with respect to distance from a as L_i^a . Now if we define trivial action on layer L_i^a as the action of a gate that is trivial on every qubit in L_i^a , and otherwise nontrivial, now, u_ℓ is nontrivial on $L_{\text{dist}(a, b)}^a$. The only way to make u_ℓ nontrivial on a layer is to apply a CNOT gate between a qubit in that layer and a qubit in another layer (assume it happens in A_j). Moreover, because of the fact that non-adjacent layers are not connected using edges, this CNOT must be between a qubit in $L_{\text{dist}(a, b)}^a$ and a qubit in $L_{\text{dist}(a, b) \pm 1}^a$. And another requirement is that u_{j-1} must be trivial on $L_{\text{dist}(a, b) \pm 1}^a$. With these requirements, by a recursive argument, we can conclude that the minimum depth is $\text{dist}(a, b) + O(1)$. \square

Theorem 3. *For any arbitrary connectivity constraint $G = (V, E)$, a bridged T gate where T is a class I gate, acting on qubits $a, b \in V$, needs at least $4\text{dist}(a, b) + O(1)$ CNOTs.*

Proof. Without losing generality, we can assume T is a CR_x with control on a and target on b . We start by defining layers with respect to distance from a as L_i^a . Then, we say a gate U is acting trivially on L_i^a if it is acting trivially on every qubit inside the set, otherwise, we say it is acting nontrivially. Then, noting that $b \in L_{\text{dist}(a, b)}^a$ and non-adjacent layers are not connected using edges, we can treat these layers similarly to qubits with linear connectivity. So, by a similar argument, we can prove that $2\text{dist}(a, b) + O(1)$ CNOTs are needed to make Z_b defined on b , conjugated by T act nontrivially on qubit a (and trivially on qubits in layers in between) and similarly another $2\text{dist}(a, b) + O(1)$ for X_a . Using the fact that layers are disjoint one can conclude these CNOTs can have $O(1)$ in common. So, the total number of CNOTs is $4\text{dist}(a, b) + O(1)$. \square

Finally, the generalised theorem for class II is written below.

Theorem 4. *For any arbitrary connectivity constraint $G = (V, E)$, a bridged T gate where T is a class II gate, acting on qubits $a, b \in V$, needs at least $6\text{dist}(a, b) + O(1)$ CNOTs.*

Proof. By defining layers with respect to distance from a as L_i^a , we can treat these layers similarly to qubits with linear connectivity.

Assume that there exists a CNOT between layer $L_0^a = \{a\}$ and L_1^a that in the sequence of $T = A_\ell \dots A_1$, will take place in A_j . Then, we call the conjugation of Z_a (defined on a) by $A_1^\dagger \dots A_{j-1}^\dagger$ as U_a , that is definitely nontrivial on L_0^a . Lemma 4 states that the conjugation of U_a by T (which is equal to the conjugation of Z_a by $A_\ell \dots A_j$) must act nontrivial on n . On the other hand we know that Z_a commutes with A_j (which without losing generality we assumed is a CNOT with control gate in L_0^a). In order that U_a has nontrivial action on b , we need to have another CNOT between these two layers. Denoting the second as A_k , with $j < k$, Lemma 3 implies U_a remains nontrivial on layer 2, but undesirable for our goal, necessitating a third CNOT. By recursion, at least three CNOTs are needed. These arguments apply in reverse too, showing a need for $6 \cdot \text{dist}(a, b) + O(1)$ CNOTs for a bridged T gate between layers a and b . \square

3.5 Summary of Results

In this chapter, we have introduced a circuit for bridging class I two-qubit gates, and we have proved that this circuit is optimal in terms of CNOT count and depth. We have also proved that the baseline implementation is optimal for class II gates, in terms of CNOT count. We have also generalised these results for arbitrary connectivity constraints. The results are summarised in Table 3.1.

| Class | Lower bound | | Ours | | Baseline | |
|----------|-------------|------------|-------------|------------|-------------|-------------|
| | CNOT | Depth | CNOT | Depth | CNOT | Depth |
| Class I | $4n + O(1)$ | $n + O(1)$ | $4n + O(1)$ | $n + O(1)$ | $6n + O(1)$ | $3n + O(1)$ |
| Class II | $6n + O(1)$ | $n + O(1)$ | - | - | $6n + O(1)$ | $3n + O(1)$ |

Table 3.1: Summary of results

Chapter 4

Conclusion

In this essay, we have studied the problem of bridging that has not been directly addressed yet. We presented an optimal implementation for bridging class I gates, such as CNOT, that uses only $4n + O(1)$ CNOTs and depth $n + O(1)$. This outperforms previous bridging methods by 33% in CNOT count and 66% in depth. We also proved that the baseline SWAP implementation is optimal for bridging class II gates like SWAP, requiring $6n + O(1)$ CNOTs. Furthermore, we generalised these results to arbitrary connectivity graphs, showing the bridging complexity depends on the distance between qubits.

4.1 Future works

Here we discuss some possible future works that can be done in this field.

4.1.1 Implementation

None of existing quantum compilers have implemented bridging for more than one qubit in between yet. The implementation of bridging can be done as an alternative way of routing, with a decision criteria for choosing between the two methods, or as a simplification that is done after routing. The question of how-to-do and when-to-do here needs meaningful engineering efforts and will be interesting to see.

4.1.2 Simultaneous Bridging

One of the possible future works is to consider the case where we have multiple gates to be bridged simultaneously. A special case of this problem is when we have multiple SWAP gates on non-adjacent qubits that we want to do in the shortest time. This problem is approached in [7] by reducing it to ROUTINGVIAMATCHING problem [4].

4.1.3 Ancillas

Bridging gates over ancilla qubits will be doubtlessly easier, as we should not worry about preserving an arbitrary state in the qubits in between. A simple case for CNOT and SWAP gates is shown in Figure 4.1.3 and Figure 4.1.3 respectively. This situation might be related to the field of network coding [17], that in simplest terms, aims to transmit information in a classical network made of bits, that are all initially zero.

4.2 Final Remark

In conclusion, we have demonstrated asymptotically optimal constructions for bridging key two-qubit gate classes, with significantly reduced CNOT count and depth over prior meth-

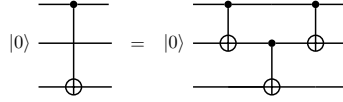


Figure 4.1: A bridged CNOT gate over one ancilla qubit

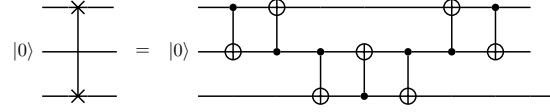


Figure 4.2: A bridged SWAP gate over one ancilla qubit

ods. Our work helps enable more efficient routing in NISQ devices by avoiding explicit SWAPs. Further developing bridging techniques could provide a valuable tool for mitigating connectivity constraints in quantum architectures.

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 Transmutation 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/CB09780511976667. 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. “ $\text{\texttt{t}}\text{\texttt{--}}\text{\texttt{sket}}\text{\texttt{r}}\text{\texttt{angle}}\text{\texttt{s}}$: 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).