





# Acknowledgments

I would like to express my gratitude to Harry Buhrman for giving me the opportunity to work in his group in CWI and for his various valuable advice during this project. I am very grateful to Ronald de Wolf for many enjoyable and sometimes frustrating discussions. For continuously correcting my English skills, I also would like to thank Ronald very much. I am grateful to Richard Jozsa for his very useful and friendly discussions during and after the conference ‘Computational Complexity of Quantum Hamiltonian Systems’ in Leiden.

For advice, and interesting, often funny discussions I would like to thank Christian Schaffner, Ben Toner, Jop Briët, Wouter Koolen-Wijkstra, Tim van Erven, Peter Harremoës, Stephanie Wehner, and especially Falk Unger. Many thanks to Jop Briët for proofreading this thesis. These INS4-ers made my time here such an enjoyable time.

Furthermore, I would like to thank Prof. Huy Trung Phan for recommending me to UvA. I also would like to thank Peter de Goeje, Niels Molenaar, Linh Ngoc Ngo and Pieter Hammingh for helping me out with many things since I came to Amsterdam. For useful advice, and various funny parties and chats, I would like to thank ‘chi’<sup>1</sup> Duong, Nga, Chi, Hong, Hanh, Van Anh, Thuy and ‘anh’<sup>1</sup> Thang, Trung, Ha, Thieu, Huy, Kien, and especially Thanh and Duc for being my great roommates. Last but not least, I would like to thank my family and my girl-friend for being who they are.

Amsterdam, June 2008  
Dung Manh Chu

---

<sup>1</sup>Vietnamese ways to call your dear female/male friends, who are older than you.



# Contents

<b>Acknowledgments</b>	<b>ii</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Introduction . . . . .	1
1.2 Outline . . . . .	2
1.3 Basics of Quantum Computation . . . . .	2
1.3.1 Qubits and Quantum States . . . . .	2
1.3.2 Quantum Gates . . . . .	3
1.3.3 Measurements . . . . .	4
1.3.4 The Density Matrix . . . . .	5
1.3.5 The Schmidt Decomposition . . . . .	7
1.4 Other Preliminaries . . . . .	8
1.4.1 Group Theory . . . . .	8
1.4.2 The Matrix Exponential . . . . .	9
1.5 Summary . . . . .	10
<b>2 The Matrix Product Formalism</b>	<b>11</b>
2.1 Introduction . . . . .	11
2.2 Notation . . . . .	13
2.3 Construction of the Matrix Product Representation . . . . .	14
2.4 Space Required for Storing a Matrix Product State . . . . .	15
2.5 Reconstructing Schmidt Decompositions from a Matrix Product State . . . . .	16
2.6 Properties of the Matrix Product Representation . . . . .	16
2.7 Computations Using the Matrix Product Representation . . . . .	17
2.7.1 One-Qubit Unitary Operations . . . . .	17
2.7.2 Two-Qubit Unitary Operations . . . . .	18
2.7.3 One-Qubit Measurements . . . . .	18
2.8 Summary . . . . .	18
<b>3 The Contracting Tensor Network Formalism</b>	<b>19</b>
3.1 Introduction . . . . .	19
3.2 Notation . . . . .	20
3.3 The Contracting Tensor Network Formalism . . . . .	22
3.4 Simulation of Quantum Circuits . . . . .	25
3.5 Contraction Complexity and Treewidth . . . . .	27
3.6 A Class of Simulable Quantum Circuits . . . . .	30

3.7	Relation between the Contracting Tensor Network and the Matrix Product Formalisms . . . . .	32
3.8	Summary . . . . .	35
<b>4</b>	<b>The Stabilizer Formalism</b>	<b>37</b>
4.1	Introduction . . . . .	37
4.2	Notation . . . . .	37
4.3	The Stabilizer Formalism . . . . .	39
4.4	Unitary Gates in the Stabilizer Formalism . . . . .	43
4.5	Clifford Gates in the Stabilizer Formalism . . . . .	45
4.5.1	Applying the Hadamard Gate . . . . .	45
4.5.2	Applying the Phase Gate . . . . .	46
4.5.3	Applying the CNOT Gate . . . . .	46
4.6	Measurements in the Stabilizer Formalism . . . . .	47
4.7	Improved Simulation of Clifford Circuits . . . . .	49
4.7.1	Applying the Clifford Gates . . . . .	50
4.7.2	Applying Measurements . . . . .	50
4.7.3	Correctness and Complexity of the Improved Simulation . . . . .	51
4.8	The Power of The Clifford Gates . . . . .	52
4.9	Summary . . . . .	52
<b>5</b>	<b>The Matchgate Formalism</b>	<b>53</b>
5.1	Introduction . . . . .	53
5.2	Notation . . . . .	54
5.3	Some Properties of Matchgates . . . . .	57
5.4	Simulation of Quantum Circuits of Gaussian operations . . . . .	59
5.5	Relation between Gaussian Operations and Matchgates . . . . .	62
5.6	The Power of Matchgates . . . . .	64
5.7	Comparison of the Matchgate, Matrix Product, Contracting Tensor Network and Stabilizer Formalisms . . . . .	66
5.7.1	The Matchgate Formalism and the Matrix Product Formalism . . . . .	66
5.7.2	The Matchgate Formalism and The Stabilizer Formalism . . . . .	67
5.7.3	The Matchgate Formalism and the Contracting Tensor Network Formalism . . . . .	68
5.8	Summary . . . . .	68
	<b>Appendices</b>	<b>69</b>
A-1	Proof of Theorem 2.7.2 . . . . .	69
A-2	Proof of Theorem 2.7.3 . . . . .	71
	<b>Bibliography</b>	<b>75</b>

# Chapter 1

## Introduction

### 1.1 Introduction

We have seen the rapid development of classical computers and their unprecedented influence to human lives since the middle of the 20<sup>th</sup> century. Physically, classical computers are based on the rules of classical physics. In contrast, quantum computers are based on the rules of quantum physics. It is generally believed that quantum computers are more powerful than classical computers. One clear piece of evidence for this viewpoint is Shor's factorization algorithm [18], which requires  $O(n^3)$  steps to factorize an  $n$ -bit integer, whereas the best known classical algorithm is exponentially slower. It is, thus, interesting to ask the question: which classes of quantum computation are computationally equivalent to classical computation? The answer to this question will provide a clear picture about the relationship between quantum and classical computations, which will give more insights into the quantum computation model. In this thesis, we survey this area by reviewing the existing simulation methods and comparing these methods together.

In this thesis, we will focus on the quantum computation model based on quantum circuits and quantum gates. A class of quantum circuits is said to be classically simulable if there exists an algorithm that can reproduce the outcome of any circuit in the class in polynomial time with polynomial space required to run the simulation. Hence the question of classical simulability of quantum computation is closely related to the question of the real power of quantum computers. People usually regard entanglement as a key difference between quantum computation and classical computation. It is true physically, i.e. entanglement only exists in quantum systems. However, based on the results from the area of classical simulation, Jozsa et al. [10] showed that it is misleading to view entanglement as a key resource for quantum computational power.

The study of simulation of quantum computing involves representing quantum states, quantum operators, and computational steps. In such a representation formalism, classes of quantum computation with 'small' descriptions will be efficiently simulated. By 'small' descriptions, we mean polynomial-size representations of input states and computational steps. We can categorize the existing simulation methods into two different types. The first type is related to topological structures of quantum circuits. Within the simulation methods of

this type, there is no restriction on the type of gates that can be used in circuits. The matrix product formalism and the contracting tensor network formalism are of this type. The second type is related to properties of gates used in circuits. There is no restriction on the topological structures of circuits, but gates that can be used must have some specific properties. The stabilizer formalism and the matchgate formalism are of this type.

## 1.2 Outline

The research done in this project was driven by the wish to compare the simulation methods and combine some of them if possible. Chapter 1 gives an introduction to quantum computation and preliminaries, which will be used during the thesis. The rest of the thesis can be divided into two parts. The first part includes Chapters 2 and 3, which cover the first simulation type. Chapter 2 is devoted to studying the matrix product formalism, in which we will investigate the role of entanglement in quantum computation. Chapter 3 deals with the contracting tensor network formalism. The second part includes the last two chapters, which cover the second simulation type, in which we focus on some classes of gates with certain properties. Chapter 4 is a study of the stabilizer formalism. The improved simulation proposed by Aaronson et al. [1] can be found here. Chapter 5 is devoted to investigating the matchgate formalism. At the end of each chapter, we have a section aiming to compare the formalism studied in the chapter with other formalisms.

In the following sections, we will introduce basic concepts of quantum computation, which are needed to understand the rest of the thesis. For more details, we refer to Chapters 1 and 2 in the book [15] of Nielsen and Chuang. We also introduce some other mathematical preliminaries. Those who are familiar with these concepts can skip this chapter.

## 1.3 Basics of Quantum Computation

### 1.3.1 Qubits and Quantum States

Like classical computing with the elementary information unit called bits, which can have value either 0 or 1, quantum computing also has the analog of a bit, which is called *qubit*. A qubit can also have value 0 or 1. Moreover, a qubit can have value other than these values. Mathematically, a qubit is described by a unit vector in the Hilbert space  $\mathbb{C}^2$ , where  $\mathbb{C}$  is the complex field. The inner product between two vectors  $\begin{pmatrix} x \\ y \end{pmatrix}$  and  $\begin{pmatrix} z \\ t \end{pmatrix}$  in this Hilbert space is defined to be  $xy + zt$ . We always use the following basis in  $\mathbb{C}^2$ ,

$$|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \quad |1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \quad (1.1)$$

where  $|\cdot\rangle$  is called a ket vector. The complex conjugate transpose of a ket vector is called a bra vector, denoted by  $\langle \cdot |$ , for example

$$\langle 0| = |0\rangle^\dagger = (1, 0), \quad \langle 1| = |1\rangle^\dagger = (0, 1). \quad (1.2)$$

The state of a qubit can be any linear combination of the basis states of the form  $\alpha|0\rangle + \beta|1\rangle$  such that  $|\alpha|^2 + |\beta|^2 = 1$ .

To describe larger quantum systems, we can combine several qubits together. In classical computing, bits are combined by using the Cartesian product of the binary set  $\{0, 1\}$ , whereas qubits in quantum computing are combined by using the tensor product of the Hilbert space  $\mathbb{C}^2$ . There is a crucial difference between the two products. Let us consider two systems  $S_A$  and  $S_B$ . Throughout this thesis, whenever we speak about a system, we mean some linear space. Every state in the Cartesian product system  $S_A \times S_B$  can be decomposed into two substates, one lying in  $S_A$  and the other lying in  $S_B$ . However, this statement is not true in general for states in the tensor product system of  $S_A$  and  $S_B$ . If we want to have an  $n$ -qubit system, we need to combine the space  $\mathbb{C}^2$   $n$  times by taking the tensor product  $\mathbb{C}^2 \otimes \cdots \otimes \mathbb{C}^2$ , also denoted by  $(\mathbb{C}^2)^{\otimes n}$ . This is a  $2^n$ -dimensional linear space with a basis consisting of the following vectors:

$$\begin{aligned} &|0\rangle \otimes |0\rangle \otimes \cdots \otimes |0\rangle \otimes |0\rangle, \\ &|0\rangle \otimes |0\rangle \otimes \cdots \otimes |0\rangle \otimes |1\rangle, \\ &|0\rangle \otimes |0\rangle \otimes \cdots \otimes |1\rangle \otimes |0\rangle, \\ &\vdots \\ &|1\rangle \otimes |1\rangle \otimes \cdots \otimes |1\rangle \otimes |1\rangle. \end{aligned} \tag{1.3}$$

The basis vector  $|i_1\rangle \otimes |i_2\rangle \otimes \cdots \otimes |i_n\rangle$  is denoted by  $|i_1 i_2 \dots i_n\rangle$  or  $|i\rangle$ , where  $i = i_1 i_2 \dots i_n$ . An  $n$ -qubit quantum state is described by a unit vector in  $(\mathbb{C}^2)^{\otimes n}$ , and hence can be expressed in the form  $c_0|0\rangle + c_1|1\rangle + \cdots + c_{2^n-1}|2^n-1\rangle$ , where  $\sum_{i=0}^{2^n-1} |c_i|^2 = 1$ .

We will be interested in quantum states that are not decomposable into a tensor product of two substates. These states are called *entangled states*. One way of quantifying the amount of entanglement presence in a state is using the Schmidt rank according to a certain partition. We will consider this issue in Subsection 1.3.5. Interesting examples are the so-called *Bell states* (or *EPR states*), which we will use frequently in the thesis:

$$\frac{|00\rangle + |11\rangle}{\sqrt{2}}, \frac{|00\rangle - |11\rangle}{\sqrt{2}}, \frac{|01\rangle + |10\rangle}{\sqrt{2}}, \frac{|01\rangle - |10\rangle}{\sqrt{2}}. \tag{1.4}$$

### 1.3.2 Quantum Gates

There are two types of operations that can act on a quantum system: unitary operations and measurements. We will consider measurements in the next section and unitary operations in this section.

A *unitary operation* on  $n$  qubits is a matrix  $U$  of dimension  $2^n \times 2^n$ , which satisfies  $U^\dagger = U^{-1}$ . The notation ‘ $\dagger$ ’ denotes the complex conjugate transpose operator. A unitary operation on  $n$  qubits can be considered as a quantum gate on  $n$  qubits. There are some basic gates, which are commonly used in quantum computing.

The **NOT** gate: like the classical NOT gate, the quantum NOT gate flips  $|0\rangle$  to  $|1\rangle$ , and  $|1\rangle$  to  $|0\rangle$ . The matrix of the quantum NOT gate is

$$NOT = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}. \tag{1.5}$$

The NOT gate is usually called the X gate, or Pauli-X gate.

The **Hadamard** gate: the Hadamard gate takes  $|0\rangle$  to  $(|0\rangle + |1\rangle)/\sqrt{2}$  and  $|1\rangle$  to  $(|0\rangle - |1\rangle)/\sqrt{2}$ . The matrix of the Hadamard gate is

$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}. \quad (1.6)$$

The **phase** gate: the phase gate takes  $|0\rangle$  to  $|0\rangle$  and  $|1\rangle$  to  $i|1\rangle$ . The matrix of the phase gate is

$$P = \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix}. \quad (1.7)$$

The **Pauli-Z** gate: the Pauli-Z gate takes  $|0\rangle$  to  $|0\rangle$  and  $|1\rangle$  to  $-|1\rangle$ . The matrix of the Pauli-Z gate is

$$Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}. \quad (1.8)$$

The **Pauli-Y** gate: the Pauli-Y gate takes  $|0\rangle$  to  $i|1\rangle$  and  $|1\rangle$  to  $-i|0\rangle$ . The matrix of the Pauli-Y gate is

$$Y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}. \quad (1.9)$$

The  $\pi/8$  gate: the  $\pi/8$  gate takes  $|0\rangle$  to  $|0\rangle$  and  $|1\rangle$  to  $e^{i\pi/4}|1\rangle$ . The matrix of the  $\pi/8$  gate is

$$Y = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\pi/4} \end{pmatrix}. \quad (1.10)$$

The **controlled-NOT** (CNOT) gate: the CNOT gate is a 2-qubit gate and it takes  $|00\rangle$  to  $|00\rangle$ ,  $|01\rangle$  to  $|01\rangle$ ,  $|10\rangle$  to  $|11\rangle$ ,  $|11\rangle$  to  $|10\rangle$ . Its matrix is

$$\text{CNOT} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}. \quad (1.11)$$

The **SWAP** gate: the SWAP gate is a 2-qubit gate and it swaps the two qubits it acts on, i.e. it takes  $|ij\rangle$  to  $|ji\rangle$  for all  $i, j \in \{0, 1\}$ . Its matrix is

$$\text{SWAP} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}. \quad (1.12)$$

### 1.3.3 Measurements

A quantum measurement can be described by a collection  $\{M_1, \dots, M_k\}$ , where  $M_i$  are called measurement operators, the indices  $i$  refer to the measurement outcomes that might occur. The measurement operators must satisfy the completeness equation, namely  $\sum_{i=1}^k M_i^\dagger M_i = I$ , where  $I$  is the identity operator. If we apply the measurement to a state  $|\phi\rangle$ , then the probability of observing outcome  $i$  is given by  $p(i) = \langle \phi | M_i^\dagger M_i | \phi \rangle$ , and the post-measurement state is

$$\frac{M_i |\phi\rangle}{\sqrt{p(i)}}.$$

There is a special class of measurements, which are called projective measurements. A projective measurement is described by an observable,  $M$ , a Hermitian operator on the state space being observed. Consider the spectral decomposition of  $M$ ,

$$M = \sum_m \lambda_m P_m,$$

where  $P_m$  is the projector onto the eigenspace of  $M$  with eigenvalue  $\lambda_m$ . Note that  $\lambda_m$  runs over all the possible real eigenvalues of  $M$ . The possible outcomes are the possible values of the eigenvalues of  $M$ . If we measure the state  $|\phi\rangle$ , then the probability of observing outcome  $\lambda_m$  is  $p(m) = \langle\phi|P_m|\phi\rangle$ , and the post-measurement state is

$$\frac{P_m|\phi\rangle}{\sqrt{p(m)}}.$$

In Chapter 5 we will need to compute the expectation of the outcomes of a projective measurement. Let  $\mathbf{E}(M)$  denote the expectation of the outcome values of the projective measurement  $M$ , then we have

$$\begin{aligned} \mathbf{E}(M) &= \sum_m \lambda_m p(m) \\ &= \sum_m \lambda_m \langle\phi|P_m|\phi\rangle \\ &= \langle\phi|\left(\sum_m \lambda_m P_m\right)|\phi\rangle \\ &= \langle\phi|M|\phi\rangle. \end{aligned} \tag{1.13}$$

### 1.3.4 The Density Matrix

The description of quantum states given in Section 1.3.1 is called the state vector formulation of quantum states. The density matrix is an alternative formulation, which is more convenient to use in some scenarios.

The density matrix of a quantum state  $|\phi\rangle$  is defined by the equation

$$\rho = |\phi\rangle\langle\phi|. \tag{1.14}$$

The density matrix formulation is a convenient means of describing mixed quantum states. A mixed quantum state is a quantum system consisting of a number of states  $|\phi_i\rangle$ , where  $i = 1, \dots, k$  for some integer  $k$ , with respective probabilities  $p_i$  such that  $\sum_i p_i = 1$ . Such a mixed state is denoted by  $\{(p_1, |\phi_1\rangle), \dots, (p_k, |\phi_k\rangle)\}$ . The density matrix of this mixed state is defined by

$$\rho = \sum_{i=1}^k p_i |\phi_i\rangle\langle\phi_i|. \tag{1.15}$$

We now consider how the action of a quantum gate to a quantum state described in the density matrix formulation. Suppose we apply a unitary operator  $U$  to the mixed state  $\rho$  in Equation (1.15).  $U$  takes each  $|\phi_i\rangle$  to  $U|\phi_i\rangle$ . Hence the resulting density matrix is given by

$$\rho' = \sum_i p_i U|\phi_i\rangle\langle\phi_i|U^\dagger = U\rho U^\dagger.$$

Suppose we perform a measurement described by measurement operators  $M_m$  to  $\rho$ . If the initial state is  $|\phi_i\rangle$ , then the probability of observing result  $m$  is

$$p(m|i) = \langle \phi_i | M_m^\dagger M_m | \phi_i \rangle.$$

The probability of observing result  $m$  from the mixed state  $\rho$ , hence, is

$$p(m) = \sum_i p(m|i) p_i = \sum_i p_i \langle \phi_i | M_m^\dagger M_m | \phi_i \rangle. \quad (1.16)$$

To get a neater formula for  $p(m)$ , we introduce the trace operator. This operator will be also used frequently. The trace of an  $n \times n$  matrix  $A$ , denoted by  $tr(A)$ , is defined to be the sum of its diagonal elements, i.e.  $tr(A) = \sum_{k=1}^n A_{kk}$ . It is easily seen that  $tr(A+B) = tr(A) + tr(B)$ , and  $tr(cA) = c \cdot tr(A)$  where  $A, B$  are arbitrary square matrices of the same size and  $c$  is a complex number. The trace operator is also cyclic, i.e.  $tr(AB) = tr(BA)$  since  $tr(AB) = \sum_{k,j} A_{kj} B_{jk} = \sum_{j,k} A_{jk} B_{kj} = tr(BA)$ . Equation (1.16) becomes

$$\begin{aligned} p(m) &= \sum_i p_i tr \left( M_m^\dagger M_m | \phi_i \rangle \langle \phi_i | \right) \\ &= tr \left( M_m^\dagger M_m \sum_i p_i | \phi_i \rangle \langle \phi_i | \right) \\ &= tr \left( M_m^\dagger M_m \rho \right). \end{aligned} \quad (1.17)$$

We sometimes describe subsystems of a composite system. For this purpose, we use the reduced density matrix formulation. Suppose we have two systems  $A$  and  $B$ , whose composite density matrix is  $\rho^{AB}$ . The reduced density matrix for system  $A$  is defined by

$$\rho^A = tr_B(\rho^{AB}), \quad (1.18)$$

where  $tr_B$  is called the partial trace over system  $B$ . The partial trace is a linear operator and its action on state  $|a_1\rangle\langle a_2| \otimes |b_1\rangle\langle b_2|$  is defined by

$$tr_B(|a_1\rangle\langle a_2| \otimes |b_1\rangle\langle b_2|) = |a_1\rangle\langle a_2| tr(|b_1\rangle\langle b_2|),$$

where  $|a_1\rangle, |a_2\rangle$  are any two vectors in the state space of system  $A$ , and  $|b_1\rangle, |b_2\rangle$  are any two vectors in the state space of system  $B$ .

Suppose  $|\Phi\rangle$  is a quantum state on a bipartite system  $A \otimes B$  and  $|\Phi\rangle$  can be written as

$$|\Phi\rangle = \sum_{i,j} \lambda_{ij} |\phi_i^A\rangle |\psi_j^B\rangle,$$

where  $i, j$  run over some index sets;  $|\phi_i^A\rangle$  are orthonormal,  $|\psi_j^B\rangle$  are also orthonormal. Then the reduced density matrix for system  $A$  is

$$\rho^A = \sum_{i,i'} \left( \sum_j \lambda_{ij} \lambda_{i'j}^* \right) |\phi_i^A\rangle \langle \phi_{i'}^A|, \quad (1.19)$$

and the reduced density matrix for system  $B$  is

$$\rho^B = \sum_{j,j'} \left( \sum_i \lambda_{ij} \lambda_{ij'}^* \right) |\psi_j^B\rangle \langle \psi_{j'}^B|, \quad (1.20)$$

where ‘\*’ denotes the complex conjugate operation.

### 1.3.5 The Schmidt Decomposition

Suppose  $|\Phi\rangle$  is a pure state in a composite system  $AB$ . The Schmidt decomposition states that there exist  $r$  orthonormal states  $|\phi_i^A\rangle$  for system  $A$ , and  $r$  orthonormal states  $|\phi_i^B\rangle$  for system  $B$  such that

$$|\Phi\rangle = \sum_{i=1}^r \lambda_i |\phi_i^A\rangle |\phi_i^B\rangle, \quad (1.21)$$

where  $\lambda_i$  are non-negative real numbers satisfying  $\sum_i \lambda_i^2 = 1$ .  $|\phi_i^A\rangle$  and  $|\phi_i^B\rangle$  are called Schmidt vectors of  $|\Phi\rangle$  with respect to systems  $A$  and  $B$ .  $\lambda_i$  are called Schmidt coefficients and  $r$  is called Schmidt rank. The following theorem lists some properties of the Schmidt decomposition.

**Theorem 1.3.1.** *Suppose  $|\Phi\rangle$  is a quantum state in the system  $AB$ . Consider a Schmidt decomposition of  $|\Phi\rangle$ :*

$$|\Phi\rangle = \sum_{i=1}^r \lambda_i |\phi_i^A\rangle |\phi_i^B\rangle.$$

The following properties always hold:

1. The reduced density matrices of systems  $A$  and  $B$  are  $\rho^A$  and  $\rho^B$ , given by

$$\rho^A = \sum_{i=1}^r \lambda_i^2 |\phi_i^A\rangle \langle \phi_i^A|, \quad \rho^B = \sum_{i=1}^r \lambda_i^2 |\phi_i^B\rangle \langle \phi_i^B|,$$

respectively. Furthermore,  $\rho^A$  and  $\rho^B$  have the same eigenvalues, namely  $\lambda_i^2$ .

2. If  $|\xi^B\rangle$  is a state in  $B$ , which is orthogonal to all the  $|\phi_i^B\rangle$ , then  $|\Phi\rangle$  and  $|\psi^A\rangle |\xi^B\rangle$  are orthogonal for any state  $|\psi^A\rangle$  in  $A$ .
3. Applying local unitary operations on either  $A$  or  $B$  does not change the Schmidt vectors of the other system, the Schmidt coefficients, or the Schmidt rank of the composite system. By local operations, we mean operations that act only on either  $A$  or  $B$ .

*Proof.* 1. By Equation (1.19), we have

$$\rho^A = \sum_{i=1}^r \lambda_i \lambda_i^* |\phi_i^A\rangle \langle \phi_i^A| = \sum_{i=1}^r \lambda_i^2 |\phi_i^A\rangle \langle \phi_i^A|.$$

In a basis containing  $|\phi_i^A\rangle$ ,  $\rho^A$  will be a diagonal matrix with non-zero diagonal elements  $\lambda_1^2, \dots, \lambda_r^2$ . Hence,  $\lambda_i^2$  are all the eigenvalues of  $\rho^A$ .

The statement for system  $B$  is proved in the same way.

2. The inner product between  $|\psi^A\rangle |\xi^B\rangle$  and  $|\phi_i^A\rangle |\phi_i^B\rangle$  is equal to  $\langle \psi^A | \phi_i^A \rangle \cdot \langle \xi^B | \phi_i^B \rangle = 0$ . Therefore the inner product between  $|\psi^A\rangle |\xi^B\rangle$  and  $|\Phi\rangle$  is equal to 0 as well. This implies that  $|\Phi\rangle$  and  $|\psi^A\rangle |\xi^B\rangle$  are orthogonal.

3. Consider performing a unitary operation  $U$  locally on system  $A$ . Let  $|\phi'_i{}^A\rangle$  denote  $U|\phi_i{}^A\rangle$ . Since unitary operations preserve both the length of a vector and the inner product between two vectors, the states  $|\phi'_i{}^A\rangle$  are also orthonormal. Hence the Schmidt decomposition of the resulting state is

$$|\Phi'\rangle = \sum_{i=1}^r \lambda_i |\phi'_i{}^A\rangle |\phi_i{}^B\rangle,$$

which proves Statement 3. □

We see that if  $|\Phi\rangle$  has Schmidt rank  $r = 1$ , then it can be decomposed into the tensor product of one state in system  $A$  and one state in system  $B$ . If  $r$  is strictly larger than 1, then this decomposition is not possible. The bigger  $r$  is, the more entangled  $|\Phi\rangle$  is. For that reason, the Schmidt rank can be thought of as a measure of entanglement of quantum states with respect to the partition  $A|B$ . Statement 3 in Theorem 1.3.1 says that, with this measure, the amount of entanglement of a quantum state does not change after performing local unitary operations. Another entanglement measure for general  $n$ -qubit states can be constructed as follows: first, compute Schmidt ranks of  $|\Phi\rangle$  with respect to all possible bipartite partitions of the system of qubits. Then, we compute the maximum Schmidt rank among these Schmidt ranks. The maximum Schmidt rank can be used as a measure of entanglement.

## 1.4 Other Preliminaries

### 1.4.1 Group Theory

A group  $(G, \cdot)$  is a non-empty set  $G$  with a binary group operation ‘ $\cdot$ ’ with the following four properties:

- Closure:  $a \cdot b \in G$  for all  $a, b \in G$ .
- Associativity:  $(a \cdot b) \cdot c = a \cdot (b \cdot c)$  for all  $a, b, c \in G$ .
- Identity: there exists  $e \in G$  such that  $a \cdot e = e \cdot a = a$  for all  $a \in G$ .
- Inverse: for all  $a \in G$ , there exists  $a^{-1} \in G$  such that  $a \cdot a^{-1} = a^{-1} \cdot a = e$ .

We often omit ‘ $\cdot$ ’ and write  $a \cdot b$  as  $ab$ . For example, the set  $G_1 = \{\pm I, \pm X, \pm Y, \pm Z, \pm iI, \pm iX, \pm iY, \pm iZ\}$ , where  $X, Y, Z$  are Pauli matrices, together with the matrix multiplication operation forms a group with identity element  $I$ . This group is called the *Pauli group* on 1-qubit.

Consider a set of elements  $g_1, \dots, g_l$  in  $G$ . An element  $h \in G$  is said to be dependent on  $\{g_1, \dots, g_l\}$  if  $h = g_{i_1} g_{i_2} \dots g_{i_k}$ , where  $k$  is some integer and  $i_j \in \{1, \dots, l\}$  for all  $j = 1, \dots, k$ . The group  $G$  is said to be generated by  $\{g_1, \dots, g_l\}$  if every element of  $G$  is dependent on  $\{g_1, \dots, g_l\}$ . It is easily seen that  $G_1$  is generated by three elements  $X, Y$  and  $iI$ .

Given two groups  $(G, \cdot)$  and  $(H, *)$ , a group homomorphism from  $(G, \cdot)$  to  $(H, *)$  is a function  $f : G \rightarrow H$  such that  $f(a \cdot b) = f(a) * f(b)$  for all  $a, b \in G$ . Group homomorphisms are very important tools since they provide structural relations between the two groups. They allow us to transform actions in one group to the other. We use this tool when we study the stabilizer formalism.

## 1.4.2 The Matrix Exponential

The exponential of a square matrix  $X$ , denoted by  $e^X$ , is defined by the power series

$$e^X = \sum_{n=0}^{\infty} \frac{X^n}{n!}, \quad (1.22)$$

where  $X^0 = I$ .

The following theorem states some basic properties of this function.

**Theorem 1.4.1.** *For square matrices  $X, Y$ :*

1. *The series (1.22) converges.*
2.  *$(e^X)^\dagger = e^{X^\dagger}$ .*
3. *If  $XY = YX$ , then  $e^{X+Y} = e^X e^Y = e^Y e^X$ .*
4.  *$e^X$  is invertible and  $(e^X)^{-1} = e^{-X}$ .*
5.  *$e^{(\alpha+\beta)X} = e^{\alpha X} e^{\beta X}$  for any complex numbers  $\alpha, \beta$ .*
6. *If  $C$  is invertible, then  $e^{CX C^{-1}} = C e^X C^{-1}$ .*
7. *If  $x$  is an eigenvector of  $X$  with eigenvalue  $\alpha$ , then  $x$  is also an eigenvector of  $e^X$  with eigenvalue  $e^\alpha$ .*
8. *The function  $f(t) = e^{tX}$  is a smooth function of complex variable  $t$  and*

$$\frac{d}{dt} f(t) = X e^{tX} = e^{tX} X. \quad (1.23)$$

9. *If  $X$  is Hermitian, then  $e^{iX}$  is unitary.*

*Proof.* 1. Let us consider a matrix norm  $\|\cdot\|$  such that  $\|A\| \cdot \|B\| \leq \|AB\|$  for all square matrices  $A, B$  of the same size. We see that  $\|X^n\| \leq \|X\|^n$ , and hence

$$\sum_{n=0}^{\infty} \left\| \frac{X^n}{n!} \right\| \leq \sum_{n=0}^{\infty} \frac{\|X\|^n}{n!} = e^{\|X\|}.$$

Therefore, the series (1.22) is absolutely convergent. It implies that the series converges.

2. This follows from the fact that  $(X^n)^\dagger = (X^\dagger)^n$  for every square matrix  $X$  and integer number  $n$ .
3. By multiplying out the two series representing  $e^X$  and  $e^Y$ , we get

$$e^X e^Y = \sum_{n=0}^{\infty} \sum_{k=0}^n \frac{X^k}{k!} \frac{Y^{n-k}}{(n-k)!} = \sum_{n=0}^{\infty} \frac{1}{n!} \sum_{k=0}^n \frac{n!}{k!(n-k)!} X^k Y^{n-k}.$$

Since  $X$  and  $Y$  commute, we have

$$(X + Y)^n = \sum_{k=0}^n \frac{n!}{k!(n-k)!} X^k Y^{n-k},$$

and, thus,

$$e^X e^Y = \sum_{n=0}^{\infty} \frac{1}{n!} (X + Y)^n = e^{X+Y}.$$

It is easily seen that  $e^Y e^X = e^{Y+X} = e^{X+Y}$ .

4. This follows directly from Statement 3 and the fact that  $X$  commutes with  $-X$ .
5. This follows directly from Statement 3 and the fact that  $\alpha X$  commutes with  $\beta X$  for any complex numbers  $\alpha$  and  $\beta$ .
6. We see that  $(CXC^{-1})^n = CX^nC^{-1}$ , and hence

$$e^{CXC^{-1}} = \sum_{n=0}^{\infty} C \frac{X^n}{n!} C^{-1} = C e^X C^{-1}.$$

7. Since  $Xx = \alpha x$ , we have  $X^n x = \alpha^n x$ , and, thus,

$$e^X x = \sum_{n=0}^{\infty} \frac{\alpha^n}{n!} x = e^\alpha x.$$

Therefore,  $x$  is an eigenvector of  $e^X$  with eigenvalue  $e^\alpha$ .

8. Differentiating the power series  $e^{tX}$  term by term with respect to  $t$  gives

$$\frac{d}{dt} f(t) = \sum_{n=1}^{\infty} \frac{t^{n-1} X^n}{(n-1)!} = \sum_{n=1}^{\infty} \frac{t^{n-1} X^{n-1}}{(n-1)!} X = e^{tX} X.$$

We can also do the above computation as follows:

$$\frac{d}{dt} f(t) = \sum_{n=1}^{\infty} \frac{t^{n-1} X^n}{(n-1)!} = \sum_{n=1}^{\infty} X \frac{t^{n-1} X^{n-1}}{(n-1)!} = X e^{tX}.$$

9. This follows from the fact that  $(e^{iX})^\dagger = e^{(iX)^\dagger} = e^{-iX} = (e^{iX})^{-1}$ .

□

## 1.5 Summary

We have introduced essential concepts of quantum computing, that are fundamentally different from classical computing. We pointed out differences between bits and qubits, also between the way bits and qubits are combined to build larger systems. We gave two different formulations for describing quantum states. We also considered an informal definition of entangled states and gave two different ways of quantifying entanglement. The evolution of quantum states was described by unitary operations and measurements, which we treated in two different subsections due to their fundamental differences. We introduced the Schmidt decomposition, some elements of group theory and the matrix exponential, which we use in coming chapters.

## Chapter 2

# The Matrix Product Formalism

The matrix product representation (MPR) of quantum states is a way of expressing quantum states using matrices. Instead of representing a quantum state based on the computational basis, in the matrix product formalism, we try to associate each qubit in the quantum state with a matrix. Using these matrices, the quantum state can be uniquely determined. Moreover, the evolution of the quantum state through a quantum circuit can be efficiently computed by using its MPR as long as the sizes of the matrices are small.

### 2.1 Introduction

Quantum states are often expressed in the conventional way as follows

$$\sum_{i_1, \dots, i_n=0}^1 c_{i_1 \dots i_n} |i_1 \dots i_n\rangle, \quad (2.1)$$

where  $c_{i_1 i_2 \dots i_n}$  are complex numbers satisfying  $\sum_{i_1, \dots, i_n} |c_{i_1 i_2 \dots i_n}|^2 = 1$ . The main advantage of this representation is that one can easily describe a quantum state in terms of complex coefficients corresponding to the basis vectors in the computational basis. This advantage, however, directly causes a big disadvantage: the number of coefficients needed to describe a state is exponential in  $n$ . It would be naive if we want to simulate quantum computations using the conventional representation since it always requires exponential amount of space to just store a quantum state. It is known that BQP<sup>1</sup> is contained in PSPACE<sup>2</sup>. Unfortunately, it does not mean that we can efficiently simulate quantum computations in polynomial time. Therefore, any new representation of quantum states that we want to use should be not only efficient (i.e., it can efficiently describe a big class of quantum states), but also suitable to simulate quantum operations.

---

<sup>1</sup>BQP is the class of problems efficiently solvable on a quantum computer with bounded error probability.

<sup>2</sup>PSPACE is the class of problems solvable on a classical computer using polynomial amount of resources.

Consider an  $n$ -qubit product quantum state, i.e., an  $n$ -qubit state that can be decomposed as a product of  $n$  1-qubit states

$$|\phi^{[1]}\rangle \otimes \cdots \otimes |\phi^{[n]}\rangle, \quad (2.2)$$

where  $|\phi^{[i]}\rangle$  is a quantum state on the  $i^{\text{th}}$  qubit. Each state  $|\phi^{[i]}\rangle$  requires 2 complex coefficients to describe. In total, to describe the state (2.2) we need only  $2n$  complex coefficients. Of course, the simple form (2.2) can not be used to describe all quantum states since there do exist  $n$ -qubit quantum states that are not product states (e.g., EPR states). We need to extend (2.2) to a more general form. The matrix product representation (MPR) is such a generalization: instead of using one state for each qubit as in (2.2), we use a number of states which form a matrix of states for each qubit

$$|\psi\rangle = \sum_{\alpha_1, \dots, \alpha_n} |\phi_{\alpha_1 \alpha_2}^{[1]}\rangle \otimes \cdots \otimes |\phi_{\alpha_n \alpha_1}^{[n]}\rangle, \quad (2.3)$$

where each  $|\phi_{\alpha_i \alpha_{i+1}}^{[i]}\rangle$  (we implicitly mean  $\alpha_{n+1} = \alpha_1$  from now on) is a 1-qubit state of the  $i^{\text{th}}$  qubit. The crucial point is that the ranges of the indices  $\alpha_1, \dots, \alpha_n$  would be small so that the number of coefficients needed to describe  $|\psi\rangle$  is polynomial in  $n$ . If this is the case, then we only need a polynomial amount of space to store a matrix product state.

Note that the first and the last indices in (2.3) are the same and also summed. Let  $\Phi^{[i]}$  be the matrix whose entries are  $|\phi_{\alpha_i \alpha_{i+1}}^{[i]}\rangle$ . Define a product between two matrices whose entries are quantum states as follows: suppose  $A = [a_{ij}]_{n \times m}$ , and  $B = [b_{ij}]_{m \times p}$ . Then  $AB := C$  where  $C = [c_{ij}]_{n \times p}$  and  $c_{ij} = \sum_{k=1}^m |a_{ik}\rangle \otimes |b_{kj}\rangle$ .

**Lemma 2.1.1.** (Jozsa [9]) *The following equality always holds*

$$|\psi\rangle = \text{tr}(\Phi^{[1]} \cdots \Phi^{[n]}).$$

Note that the definitions of  $\Phi^{[i]}$  ensure that the number of columns of matrix  $\Phi^{[i]}$  equals the number of rows of matrix  $\Phi^{[i+1]}$ . We, therefore, can properly take the product of  $\Phi^{[1]}, \dots, \Phi^{[n]}$ .

*Proof of Lemma 2.1.1.* For the sake of simplicity, we consider the case  $n = 3$ . Suppose that the range of  $\alpha_i$  is  $\{1, \dots, r_i\}$  where  $r_i$  is some integer. Let

$$A = \Phi^{[1]}\Phi^{[2]} = [a_{ik}]_{r_1 \times r_3} \text{ where } a_{ik} = \sum_{l=1}^{r_2} |\phi_{il}^{[1]}\rangle |\phi_{lk}^{[2]}\rangle. \text{ Let } B = A\Phi^{[3]} = [b_{ij}]_{r_1 \times r_1}. \text{ We get}$$

$$|b_{ij}\rangle = \sum_{k=1}^{r_3} |a_{ik}\rangle |\phi_{kj}^{[3]}\rangle = \sum_{k=1}^{r_3} \left( \sum_{l=1}^{r_2} |\phi_{il}^{[1]}\rangle |\phi_{lk}^{[2]}\rangle \right) |\phi_{kj}^{[3]}\rangle = \sum_{k=1}^{r_3} \sum_{l=1}^{r_2} |\phi_{il}^{[1]}\rangle |\phi_{lk}^{[2]}\rangle |\phi_{kj}^{[3]}\rangle.$$

$$\text{Therefore } \text{tr}(\Phi^{[1]}\Phi^{[2]}\Phi^{[3]}) = \sum_{i=1}^{r_1} |b_{ii}\rangle = \sum_{i,k,l} |\phi_{il}^{[1]}\rangle |\phi_{lk}^{[2]}\rangle |\phi_{ki}^{[3]}\rangle = |\psi\rangle.$$

This proves the lemma. The proof can be easily extended to the general case.  $\square$

Note that the state (2.2) is indeed a special case of (2.3). This can be seen easily when all  $r_i$  (defined in the above proof) equal 1.

So far, we've assumed that there exists the so-called MPR ((2.3)). One would ask a question: is it possible to describe any quantum state by using the MPR? The next sections will mention a method ([23]) to construct MPR from any quantum state. Before that, we introduce some notation that is often used throughout this chapter.

## 2.2 Notation

Qubits in quantum systems are indexed from 1 to  $n$ .

The *subsystem* that contains qubits  $1, \dots, i$  (or  $i + 1, \dots, n$ ) is denoted by  $[1 \dots i]$  (or  $[i + 1 \dots n]$ ).

The  $i^{\text{th}}$  *splitting* (or decomposition) of the qubits is obtained by dividing the qubits into two subsystems  $[1 \dots i]$  and  $[i + 1 \dots n]$ .

The  $i^{\text{th}}$  *Schmidt decomposition* of an  $n$ -qubit quantum state  $|\psi\rangle$  is denoted by

$$|\psi\rangle = \sum_{\alpha_i=1}^{\chi_i} \lambda_{\alpha_i}^{[i]} |\phi_{\alpha_i}^{[1 \dots i]}\rangle |\phi_{\alpha_i}^{[i+1 \dots n]}\rangle,$$

where  $\lambda_{\alpha_i}^{[i]}$  are the Schmidt coefficients;  $\chi_i$  is the Schmidt rank;  $|\phi_{\alpha_i}^{[1 \dots i]}\rangle$  are the Schmidt vectors corresponding to the first subsystem;  $|\phi_{\alpha_i}^{[i+1 \dots n]}\rangle$  are the Schmidt vectors corresponding to the second subsystem. Let  $\chi$  be the maximal number among  $\{\chi_1, \dots, \chi_{n-1}\}$ . Then we can write

$$|\psi\rangle = \sum_{\alpha_i=1}^{\chi} \lambda_{\alpha_i}^{[i]} |\phi_{\alpha_i}^{[1 \dots i]}\rangle |\phi_{\alpha_i}^{[i+1 \dots n]}\rangle.$$

(we implicitly mean that  $\lambda_{\alpha_i}^{[i]} = 0$  if  $\alpha_i > \chi_i$ )

*Inner product of vectors acting on different systems:* consider two subsystems  $[1 \dots k]$  and  $[1 \dots l]$  where  $1 \leq k < l \leq n$ ; suppose  $|i_1 \dots i_k\rangle$  are vectors in the computational basis of subsystem  $[1 \dots k]$  and  $|j_1 \dots j_l\rangle$  be vectors in the computational basis of subsystem  $[1 \dots l]$ . Consider two vectors  $|a\rangle, |b\rangle$  in these subsystems,

$$\begin{aligned} |a\rangle &= \sum_{i_1, \dots, i_k} \alpha_{i_1 \dots i_k} |i_1 \dots i_k\rangle, \\ |b\rangle &= \sum_{j_1, \dots, j_l} \beta_{j_1 \dots j_l} |j_1 \dots j_l\rangle. \end{aligned}$$

The inner product between two vectors  $|a\rangle, |b\rangle$  is defined by the equation

$$\langle a|b\rangle = \sum_{j_{k+1}, \dots, j_l} \left( \sum_{j_1, \dots, j_k} \alpha_{j_1 \dots j_k}^* \beta_{j_1 \dots j_l} \right) |j_{k+1} \dots j_l\rangle.$$

It means that the inner product between  $|a\rangle$  and  $|b\rangle$  is a state on subsystem  $[k + 1 \dots l]$ .

## 2.3 Construction of the Matrix Product Representation

Suppose  $|\psi\rangle$  is an  $n$ -qubit quantum state. Consider the first Schmidt decomposition

$$|\psi\rangle = \sum_{\alpha_1=1}^{\chi} \lambda_{\alpha_1}^{[1]} |\phi_{\alpha_1}^{[1]}\rangle |\phi_{\alpha_1}^{[2\dots n]}\rangle. \quad (2.4)$$

We always implicitly mean that the range of  $\alpha_i$  is from 1 to  $\chi$ . Therefore, from now on, we ignore the ranges of  $\alpha_i$  in the sum notation.

Writing  $|\phi_{\alpha_1}^{[1]}\rangle$  in terms of the computational basis  $\{|0\rangle, |1\rangle\}$  gives

$$|\phi_{\alpha_1}^{[1]}\rangle = \sum_{i_1} \Gamma_{\alpha_1}^{[1]i_1} |i_1\rangle.$$

We also implicitly mean that the range of indices  $i_k$  is  $\{0, 1\}$ . By substituting this equation back into Equation (2.4) we get

$$|\psi\rangle = \sum_{\alpha_1; i_1} \Gamma_{\alpha_1}^{[1]i_1} \lambda_{\alpha_1}^{[1]} |i_1\rangle |\phi_{\alpha_1}^{[2\dots n]}\rangle. \quad (2.5)$$

Consider the second Schmidt decomposition

$$|\psi\rangle = \sum_{\alpha_2} \lambda_{\alpha_2}^{[2]} |\phi_{\alpha_2}^{[12]}\rangle |\phi_{\alpha_2}^{[3\dots n]}\rangle.$$

The crucial point in construction of MPR is to express each Schmidt vector  $|\phi_{\alpha_1}^{[2\dots n]}\rangle$  of the first Schmidt decomposition in terms of all Schmidt vectors  $\{|\phi_{\alpha_2}^{[3\dots n]}\rangle\}$  of the second Schmidt decomposition. To do this, notice that  $|\phi_{\alpha_1}^{[2\dots n]}\rangle$  can always be written as

$$|\phi_{\alpha_1}^{[2\dots n]}\rangle = \sum_{i_2} |i_2\rangle |\tau_{\alpha_1 i_2}^{[3\dots n]}\rangle, \quad (2.6)$$

where  $|\tau_{\alpha_1 i_2}^{[3\dots n]}\rangle$  are possibly unnormalized vectors. Equation (2.6) is obtained by writing  $|\phi_{\alpha_1}^{[2\dots n]}\rangle$  in terms of the vectors in the computational basis, then grouping together all components that share the same value of qubit  $i_2$ . Also recall that  $\{|\phi_{\alpha_2}^{[3\dots n]}\rangle\}$  is orthonormal. Hence, if vectors  $|\tau_{\alpha_1 i_2}^{[3\dots n]}\rangle$  are in the subspace spanned by  $\{|\phi_{\alpha_2}^{[3\dots n]}\rangle\}$ , then we can write each vector  $|\phi_{\alpha_1}^{[2\dots n]}\rangle$  in terms of  $\{|\phi_{\alpha_2}^{[3\dots n]}\rangle\}$ .

**Lemma 2.3.1.** ([23]) *All vectors  $|\tau_{\alpha_1 i_2}^{[3\dots n]}\rangle$  lie in the subspace spanned by  $\{|\phi_{\alpha_2}^{[3\dots n]}\rangle\}$ .*

*Proof.* We recall that every vector in the subspace corresponding to qubit system  $[3\dots n]$  can always be expressed in terms of a vector spanned by  $\{|\phi_{\alpha_2}^{[3\dots n]}\rangle\}$  and a vector in subspace  $\{|\phi_{\alpha_2}^{[3\dots n]}\rangle\}^\perp$  (where  $\{|\phi_{\alpha_2}^{[3\dots n]}\rangle\}^\perp$  is the subspace that consists of all vectors that are orthogonal to  $\{|\phi_{\alpha_2}^{[3\dots n]}\rangle\}$ ). Let  $\{|\theta_{\alpha_2'}^{[3\dots n]}\rangle\}$  be an orthonormal basis of subspace  $\{|\phi_{\alpha_2}^{[3\dots n]}\rangle\}^\perp$ . Suppose that  $|\tau_{\alpha_1 i_2}^{[3\dots n]}\rangle = \sum_{\alpha_2} \Omega_{\alpha_1 \alpha_2}^{[2]i_2} |\phi_{\alpha_2}^{[3\dots n]}\rangle + \sum_{\alpha_2'} \Delta_{\alpha_1 \alpha_2'}^{[2]i_2} |\theta_{\alpha_2'}^{[3\dots n]}\rangle$ . We need to show that all  $\Delta_{\alpha_1 \alpha_2'}^{[2]i_2}$  equal 0.

Substituting  $|\tau_{\alpha_1 i_2}^{[3\dots n]}\rangle$  back into Equation (2.6) gives

$$|\phi_{\alpha_1}^{[2\dots n]}\rangle = \sum_{\alpha_2; i_2} \Omega_{\alpha_1 \alpha_2}^{[2]i_2} |i_2\rangle |\phi_{\alpha_2}^{[3\dots n]}\rangle + \sum_{\alpha_2'; i_2} \Delta_{\alpha_1 \alpha_2'}^{[2]i_2} |i_2\rangle |\theta_{\alpha_2'}^{[3\dots n]}\rangle.$$

Then, substituting  $|\phi_{\alpha_1}^{[2\dots n]}\rangle$  into Equation (2.4) we get

$$|\psi\rangle = \sum_{\alpha_1, \alpha_2'; i_2} \lambda_{\alpha_1}^{[1]} \Omega_{\alpha_1 \alpha_2'}^{[2]i_2} |\phi_{\alpha_2'}^{[1]}\rangle |i_2\rangle |\phi_{\alpha_2'}^{[3\dots n]}\rangle + \sum_{\alpha_1, \alpha_2'; i_2} \lambda_{\alpha_1}^{[1]} \Delta_{\alpha_1 \alpha_2'}^{[2]i_2} |\phi_{\alpha_2'}^{[1]}\rangle |i_2\rangle |\theta_{\alpha_2'}^{[3\dots n]}\rangle.$$

Therefore,  $\langle \theta_{\alpha_2}^{[3\dots n]} | \psi \rangle = \sum_{\alpha_1; i_2} \lambda_{\alpha_1}^{[1]} \Delta_{\alpha_1 \alpha_2}^{[2]i_2} |\phi_{\alpha_2}^{[1]}\rangle |i_2\rangle$ . By Theorem 1.3.1, we must

have  $\langle \theta_{\alpha_2}^{[3\dots n]} | \psi \rangle = 0$ . Moreover,  $\{|\phi_{\alpha_2}^{[1]}\rangle |i_2\rangle\}$  is orthonormal. Hence we get  $\lambda_{\alpha_1}^{[1]} \Delta_{\alpha_1 \alpha_2}^{[2]i_2} = 0$  for all  $i_2, \alpha_1, \alpha_2'$ . This is the case iff  $\Delta_{\alpha_1 \alpha_2}^{[2]i_2} = 0$  since  $\lambda_{\alpha_1}^{[1]}$  are non-zero for  $\alpha_i \leq \chi_i$ . This proves Lemma 2.3.1.  $\square$

**Corollary 2.3.2.** *There are complex coefficients  $\Gamma_{\alpha_1 \alpha_2}^{[2]i_2}$  such that*

$$|\tau_{\alpha_1 i_2}^{[3\dots n]}\rangle = \sum_{\alpha_2} \Gamma_{\alpha_1 \alpha_2}^{[2]i_2} \lambda_{\alpha_2}^{[2]} |\phi_{\alpha_2}^{[3\dots n]}\rangle.$$

Substituting the above equation into (2.6) gives

$$|\phi_{\alpha_1}^{[2\dots n]}\rangle = \sum_{i_2} |i_2\rangle \sum_{\alpha_2} \Gamma_{\alpha_1 \alpha_2}^{[2]i_2} \lambda_{\alpha_2}^{[2]} |\phi_{\alpha_2}^{[3\dots n]}\rangle = \sum_{i_2; \alpha_2} \Gamma_{\alpha_1 \alpha_2}^{[2]i_2} \lambda_{\alpha_2}^{[2]} |i_2\rangle |\phi_{\alpha_2}^{[3\dots n]}\rangle.$$

Now we can expand Equation (2.5) as follows to get an intermediate state in our MPR construction process

$$|\psi\rangle = \sum_{\alpha_1, \alpha_2; i_1, i_2} \Gamma_{\alpha_1}^{[1]i_1} \lambda_{\alpha_1}^{[1]} \Gamma_{\alpha_1 \alpha_2}^{[2]i_2} \lambda_{\alpha_2}^{[2]} |i_1 i_2\rangle |\phi_{\alpha_2}^{[3\dots n]}\rangle. \quad (2.7)$$

By repeating the procedure transforming from (2.5) to (2.7)  $(n-2)$  times, we'll arrive at

$$|\psi\rangle = \sum_{\substack{\alpha_1, \dots, \alpha_{n-1} \\ i_1, \dots, i_{n-1}}} \Gamma_{\alpha_1}^{[1]i_1} \lambda_{\alpha_1}^{[1]} \Gamma_{\alpha_1 \alpha_2}^{[2]i_2} \lambda_{\alpha_2}^{[2]} \dots \Gamma_{\alpha_{n-2} \alpha_{n-1}}^{[n-1]i_{n-1}} \lambda_{\alpha_{n-1}}^{[n-1]} |i_1 i_2 \dots i_{n-1}\rangle |\phi_{\alpha_{n-1}}^{[n]}\rangle.$$

Rewriting  $|\phi_{\alpha_{n-1}}^{[n]}\rangle = \sum_{i_n} \Gamma_{\alpha_{n-1}}^{[n]i_n} |i_n\rangle$  gives us the MPR of  $|\psi\rangle$

$$|\psi\rangle = \sum_{\substack{\alpha_1, \dots, \alpha_{n-1} \\ i_1, \dots, i_n}} \Gamma_{\alpha_1}^{[1]i_1} \lambda_{\alpha_1}^{[1]} \Gamma_{\alpha_1 \alpha_2}^{[2]i_2} \lambda_{\alpha_2}^{[2]} \dots \Gamma_{\alpha_{n-2} \alpha_{n-1}}^{[n-1]i_{n-1}} \lambda_{\alpha_{n-1}}^{[n-1]} \Gamma_{\alpha_{n-1}}^{[n]i_n} |i_1 i_2 \dots i_n\rangle. \quad (2.8)$$

## 2.4 Space Required for Storing a Matrix Product State

Let  $\Gamma^{[l]}$  denote the tensor corresponding to qubit  $l$ . Let  $\lambda^{[l]}$  denote the vector whose components are the Schmidt coefficients of the  $l^{\text{th}}$  Schmidt decomposition. By Equation (2.8),  $|\psi\rangle$  can be written as

$$\Gamma^{[1]} \lambda^{[1]} \Gamma^{[2]} \lambda^{[2]} \dots \Gamma^{[n-1]} \lambda^{[n-1]} \Gamma^{[n]}.$$

Since  $\chi$  is the maximal value among  $\{\chi_1, \dots, \chi_{n-1}\}$ , the number of entries in tensor  $\Gamma^{[l]}$  is at most  $2\chi^2$  and the number of entries in vector  $\lambda^{[l]}$  is at most  $\chi$ . Therefore we need at most  $(2\chi^2 + \chi)n$  complex coefficients to describe the matrix product state. Obviously, MPR requires only a polynomial amount of resources iff the maximal Schmidt rank is polynomial in  $n$ .

## 2.5 Reconstructing Schmidt Decompositions from a Matrix Product State

From section 2.3, we know that a matrix product state was constructed by applying Schmidt decompositions on different decompositions. Can we reconstruct Schmidt decompositions from a matrix product state? The answer is yes.

Suppose we are given matrix product state  $|\psi\rangle$ . In the construction of MPR, we used the following relation between the Schmidt vectors of the  $(l-1)^{th}$  Schmidt decomposition and the Schmidt vectors of the  $l^{th}$  Schmidt decomposition

$$|\phi_{\alpha_{l-1}}^{[l\dots n]}\rangle = \sum_{\alpha_l; i_l} \Gamma_{\alpha_{l-1}\alpha_l}^{[l]i_l} \lambda_{\alpha_l}^{[l]} |i_l\rangle |\phi_{\alpha_l}^{[l+1\dots n]}\rangle.$$

Applying this relation multiple times, we end up with

$$|\phi_{\alpha_{l-1}}^{[l\dots n]}\rangle = \sum_{\substack{\alpha_1 \dots \alpha_{n-1} \\ i_1, \dots, i_n}} \Gamma_{\alpha_{l-1}\alpha_l}^{[l]i_l} \lambda_{\alpha_l}^{[l]} \Gamma_{\alpha_l\alpha_{l+1}}^{[l+1]i_{l+1}} \lambda_{\alpha_{l+1}}^{[l+1]} \dots \Gamma_{\alpha_{n-1}}^{[n]i_n} |i_l i_{l+1} \dots i_n\rangle. \quad (2.9)$$

We also need to compute  $|\phi_{\alpha_{l-1}}^{[1\dots l-1]}\rangle$ . Consider the  $(l-1)^{th}$  Schmidt decomposition

$$|\psi\rangle = \sum_{\alpha_{l-1}} \lambda_{\alpha_{l-1}}^{[l-1]} |\phi_{\alpha_{l-1}}^{[1\dots l-1]}\rangle |\phi_{\alpha_{l-1}}^{[l\dots n]}\rangle.$$

We also have the following equation (it was an intermediate state in the process of constructing MPR)

$$|\psi\rangle = \sum_{\substack{\alpha_1, \dots, \alpha_{l-1} \\ i_1, \dots, i_{l-1}}} \Gamma_{\alpha_1}^{[1]i_1} \lambda_{\alpha_1}^{[1]} \Gamma_{\alpha_1\alpha_2}^{[2]i_2} \lambda_{\alpha_2}^{[2]} \dots \Gamma_{\alpha_{l-2}\alpha_{l-1}}^{[l-1]i_{l-1}} \lambda_{\alpha_{l-1}}^{[l-1]} |i_1 i_2 \dots i_{l-1}\rangle |\phi_{\alpha_{l-1}}^{[l\dots n]}\rangle.$$

Since  $|\phi_{\alpha_{l-1}}^{[l\dots n]}\rangle$  are orthonormal, the above two equations give us

$$|\phi_{\alpha_{l-1}}^{[1\dots l-1]}\rangle = \sum_{\substack{\alpha_1, \dots, \alpha_{l-2} \\ i_1, \dots, i_{l-1}}} \Gamma_{\alpha_1}^{[1]i_1} \lambda_{\alpha_1}^{[1]} \Gamma_{\alpha_1\alpha_2}^{[2]i_2} \lambda_{\alpha_2}^{[2]} \dots \Gamma_{\alpha_{l-2}\alpha_{l-1}}^{[l-1]i_{l-1}} \lambda_{\alpha_{l-1}}^{[l-1]} |i_1 i_2 \dots i_{l-1}\rangle. \quad (2.10)$$

By Equations (2.9) and (2.10), we have finished reconstructing the  $(l-1)^{th}$  Schmidt decomposition since the Schmidt coefficients already are known from the MPR.

## 2.6 Properties of the Matrix Product Representation

**Lemma 2.6.1.** *The following properties of Schmidt vectors always hold*

Property 1:

$$|\phi_{\alpha_{l-1}}^{[l\dots n]}\rangle = \sum_{\alpha_l, i_l} \Gamma_{\alpha_{l-1}\alpha_l}^{[l]i_l} \lambda_{\alpha_l}^{[l]} |i_l\rangle |\phi_{\alpha_l}^{[l+1\dots n]}\rangle.$$

Property 2:

$$|\phi_{\alpha_l}^{[1\dots l]}\rangle = \sum_{\alpha_{l-1}, i_l} \Gamma_{\alpha_{l-1}\alpha_l}^{[l]i_l} \lambda_{\alpha_l}^{[l]} |\phi_{\alpha_{l-1}}^{[1\dots l-1]}\rangle |i_l\rangle.$$

*Proof.* Property 1 can be directly implied from the construction of MPR. Property 2 is a corollary of Equation (2.10) by using  $|\phi_{\alpha_l}^{[1\dots l]}\rangle$  and  $|\phi_{\alpha_{l-1}}^{[1\dots l-1]}\rangle$ .  $\square$

## 2.7 Computations Using the Matrix Product Representation

In this section, we investigate quantum computations using the MPR. The results show that computations related to 1-qubit gates, 2-qubit gates and 1-qubit measurements cost polynomially many steps in  $n$  and in  $\chi$ .

### 2.7.1 One-Qubit Unitary Operations

Suppose  $U$  is a 1-qubit unitary operator acting on qubit  $l$ .

**Theorem 2.7.1.** (Vidal [23]) *The computational cost of  $U$  is of  $O(\chi^2)$  steps.*

*Proof of Theorem 2.7.1.* Consider the  $(l-1)^{th}$  Schmidt decomposition

$$\begin{aligned} |\psi\rangle &= \sum_{\alpha_{l-1}} \lambda_{\alpha_{l-1}}^{[l-1]} |\phi_{\alpha_{l-1}}^{[1\dots l-1]}\rangle |\phi_{\alpha_{l-1}}^{[l\dots n]}\rangle \\ &= \sum_{\alpha_{l-1}} \lambda_{\alpha_{l-1}}^{[l-1]} |\phi_{\alpha_{l-1}}^{[1\dots l-1]}\rangle \sum_{i_l} |i_l\rangle |\tau_{\alpha_{l-1}i_l}^{[l+1\dots n]}\rangle \\ &= \sum_{\alpha_{l-1}} \lambda_{\alpha_{l-1}}^{[l-1]} |\phi_{\alpha_{l-1}}^{[1\dots l-1]}\rangle \sum_{i_l} |i_l\rangle \sum_{\alpha_l} \Gamma_{\alpha_{l-1}\alpha_l}^{[l]i_l} \lambda_{\alpha_l}^{[l]} |\phi_{\alpha_l}^{[l+1\dots n]}\rangle. \end{aligned}$$

We get

$$|\psi\rangle = \sum_{\alpha_{l-1}, \alpha_l; i_l} \lambda_{\alpha_{l-1}}^{[l-1]} \Gamma_{\alpha_{l-1}\alpha_l}^{[l]i_l} \lambda_{\alpha_l}^{[l]} |\phi_{\alpha_{l-1}}^{[1\dots l-1]}\rangle |i_l\rangle |\phi_{\alpha_l}^{[l+1\dots n]}\rangle. \quad (2.11)$$

By Theorem 1.3.1, all the Schmidt vectors  $|\phi_{\alpha_{l-1}}^{[1\dots l-1]}\rangle$  do not change after performing  $U$ . Neither do  $|\phi_{\alpha_l}^{[l+1\dots n]}\rangle$  and  $\lambda_{\alpha_{l-1}}^{[l-1]}, \lambda_{\alpha_l}^{[l]}$  since  $U$  is local under every Schmidt splitting. Therefore the only components that are affected by  $U$  are tensors  $\Gamma^{[1]}, \dots, \Gamma^{[n]}$ .

Suppose we index  $U$ 's entries using binary indices,  $U_{ij}$ , where  $i$  is the row index,  $j$  is the column index. We have  $U|i\rangle = U_{0i}|0\rangle + U_{1i}|1\rangle = \sum_{k=0;1} U_{ki}|k\rangle$ ;

$U \left( \Gamma_{\alpha_{l-1}\alpha_l}^{[l]i_i} |i_i\rangle \right) = \sum_k U_{ki} \Gamma_{\alpha_{l-1}\alpha_l}^{[l]i_i} |k\rangle$ . Therefore

$$\begin{aligned} U \left( \sum_{i_i} \Gamma_{\alpha_{l-1}\alpha_l}^{[l]i_i} |i_i\rangle \right) &= \sum_{i_i} U \left( \Gamma_{\alpha_{l-1}\alpha_l}^{[l]i_i} |i_i\rangle \right) = \sum_{i_i} \sum_k U_{ki} \Gamma_{\alpha_{l-1}\alpha_l}^{[l]i_i} |k\rangle \\ &= \sum_k \left( \sum_{i_i} U_{ki} \Gamma_{\alpha_{l-1}\alpha_l}^{[l]i_i} \right) |k\rangle. \end{aligned}$$

Hence  $\Gamma^{[l]}$  changes according to

$$\Gamma'_{\alpha_{l-1}\alpha_l} = \sum_{i=0;1} U_{ki} \Gamma_{\alpha_{l-1}\alpha_l}^{[l]i}. \quad (2.12)$$

This update procedure needs to be done with all entries of  $\Gamma^{[l]}$ . Since there are  $2\chi^2$  entries and each update needs 3 steps, the whole update procedure requires  $3 \times 2\chi^2 = 6\chi^2 = O(\chi^2)$  steps.  $\square$

Note that if we view  $\Gamma^{[l]}$  as a  $2 \times \chi^2$  matrix, Equation (2.12) can be written as a multiplication between the two matrices

$$\Gamma'^{[l]} = U\Gamma^{[l]}.$$

## 2.7.2 Two-Qubit Unitary Operations

Suppose  $V$  is a 2-qubit unitary operator acting on qubits  $l, l+1$ .

**Theorem 2.7.2.** (Vidal [23]) *Updating the MPR of  $|\psi\rangle$  after performing operator  $V$  costs  $O(\chi^3)$  steps.*

The detailed proof of this theorem is given in Appendix A-1.

## 2.7.3 One-Qubit Measurements

**Theorem 2.7.3.** (Yoran and Short [24]) *If a single qubit measurement is applied to a qubit of  $|\psi\rangle$ , then the outcome probabilities can be calculated in  $O(\chi^2)$  steps and the updated MPR of the state can be calculated in  $O(n\chi^4)$  steps.*

The detailed proof of this theorem is given in Appendix A-2.

## 2.8 Summary

In this section, we considered the matrix product formalism, in which an  $n$ -qubit quantum state is represented by a set of  $n$  matrices. The sizes of the matrices depend only on the maximal Schmidt rank over all the possible partitions of the  $n$  qubits. We also studied updating of a matrix product state after performing one-qubit gates, two-qubit gates acting on nearest-neighbor qubits and one-qubit measurements. All these updates were shown to be polynomial in  $\chi$ , the maximal Schmidt rank. Therefore, if the maximal Schmidt rank is always kept polynomially small during the execution of a circuit, then we can simulate the action of the circuit in polynomial time.

## Chapter 3

# The Contracting Tensor Network Formalism

Within the tensor formalism, Markov and Shi [13] describe quantum states and quantum operators by a common mathematical object, the so-called tensor. With this representation, a quantum circuit and computational steps are translated to a graph and edge contractions, respectively. The complexity of the contraction process on the graph can be upper-bounded by the treewidth of the graph. If  $d$  denotes the graph's treewidth, and  $T$  denotes the number of gates in the circuit, then the contraction of the whole graph can be done classically in  $T^{O(1)}exp[O(d)]$  time. Hence, quantum circuits whose corresponding graphs are of logarithmic treewidth can be efficiently simulated.

### 3.1 Introduction

Within the tensor formalism, quantum states and quantum operators are described by tensors, which are direct generalizations of matrices, and computational steps are translated to so-called tensor contractions. Hence a quantum circuit is mapped to a graph whose vertices represent quantum gates, and whose edges represent segments of qubit wires connecting two gates. The contraction of the whole graph faithfully describes the evolution of the quantum state in the circuit.

In Section 3.2, we will start by introducing the tensor and tensor network concepts. Section 3.3 considers the representations of quantum states, quantum operators and quantum circuits using tensors and tensor networks. The translation of computations on quantum circuits to contractions on graphs is studied in Section 3.4. Section 3.5 is devoted to the relation of the complexity of a graph's contraction to the graph's treewidth. In Section 3.6, we study an upper bound of a circuit-graph's treewidth based on the so-called qubit-crossing concept, which is easily computed in the circuit. We then derive a class of quantum circuits which can be classically simulated. These sections are mainly based on paper [13]. Finally, we study some relations between the tensor formalism and the matrix product formalism in the last section; and give a conjecture about a relation between the amount of entanglement of the final state in a circuit and the circuit's treewidth.

### 3.2 Notation

**Definition 3.2.1.** A rank- $k$  tensor  $g = [g_{i_1, i_2, \dots, i_k}]$ , each of whose indices ranges over a set of size  $m$ , is an  $m^k$ -dimensional array of complex numbers. We also denote by  $r(g)$  the rank of  $g$ , i.e.  $r(g) = k$ .

For example, a rank-0 tensor is a tensor with no index, which is simply a complex number. A rank-1 tensor in an  $m$ -dimensional space is a complex vector with  $m$  elements. A rank-2 tensor in an  $m$ -dimensional space is a complex matrix of size  $m \times m$ . The tensor concept is obviously a generalization of the traditional matrix concept. The tensor formalism aims to represent quantum states and quantum operators by means of tensors. We also need to be able to describe the evolution of a quantum circuit by means of some operation between tensors. To this end, the tensor formalism introduces the so-called tensor contraction operation.

**Definition 3.2.2.** Given two tensors sharing a common subset of indices,  $g = [g_{i_1, \dots, i_l, j_1, \dots, j_p}]$  and  $h = [h_{i_1, \dots, i_l, k_1, \dots, k_n}]$ . Contracting these two tensors gives a new tensor  $f = [f_{j_1, \dots, j_p, k_1, \dots, k_n}]$  and

$$f_{j_1, \dots, j_p, k_1, \dots, k_n} = \sum_{i_1, \dots, i_l} g_{i_1, \dots, i_l, j_1, \dots, j_p} \cdot h_{i_1, \dots, i_l, k_1, \dots, k_n}, \quad (3.1)$$

where the indices  $i_1, \dots, i_l$  run over their ranges.

We consider the complexity of computing the new tensor  $f$  in Definition 3.2.2. Computing one element of  $f$  costs us  $O(m^l)$  steps. The tensor consists of  $m^{p+n}$  elements. Computing the whole tensor, therefore, takes  $O(m^{l+p+n})$  steps, which is at most  $O(m^{r(g)+r(h)})$ .

There exists another way of interpreting a tensor using graph theory. We consider graphs with open edges (an open edge is an edge which has one end point is a vertex of the graph, and the other end point is free). A rank- $k$  tensor  $g$  can be graphically represented as a vertex labelled with  $g$ , and having  $k$  open wires which are labeled by the  $k$  indices of  $g$ . Part (a) of Figure 3.1 illustrates the graphic representation of tensor  $g = [g_{i_1, i_2, i_3, i_4}]$ .

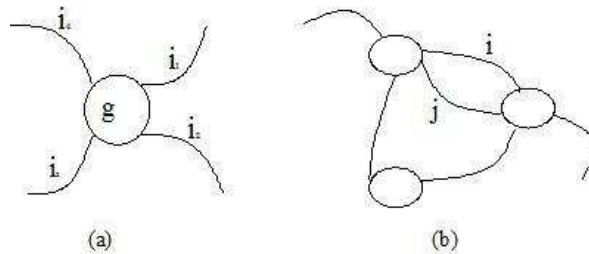


Figure 3.1: (a) Graphic representation of the tensor  $g = [g_{i_1, i_2, i_3, i_4}]$ ; (b) A network of three tensors.

**Definition 3.2.3.** A tensor network is a collection of tensors, in which some of the indices may be used by two tensors.

Graphically, a tensor network is represented by an undirected graph, whose vertices correspond to the tensors and whose edges correspond to the indices of the tensors. An edge between two vertices corresponds to a common index shared by the two corresponding tensors. Part (b) of Figure 3.1 illustrates a network consisting of three tensors.

The tensor contraction operation can be also interpreted using graph theory. Given a graph  $G$ , the contraction of a pair of vertices  $u$  and  $v$  produces a graph in which the two nodes  $u$  and  $v$  are replaced with a single node such that it is adjacent to the nodes to which  $u$  and  $v$  were originally adjacent. If there exist edges between  $u$  and  $v$ , then these edges are simply removed upon contraction. Consider the tensor contraction of two tensors  $g = [g_{i_1, \dots, i_l, j_1, \dots, j_p}]$  and  $h = [h_{i_1, \dots, i_l, k_1, \dots, k_n}]$  based on Equation (3.1). Graphically, this operation is illustrated in Figure 3.2. The tensor contraction, hence, is graphically the vertex contraction of the two vertices corresponding to  $g$  and  $h$ .

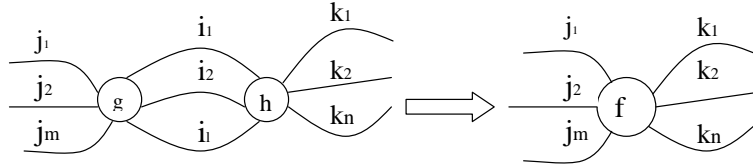


Figure 3.2: Graphical representation of the contraction of two tensors sharing some common indices.

We now introduce a property of contractions on tensor networks, which will be used later as the main idea that the contracting tensor formalism is based on.

**Theorem 3.2.4.** *Contracting all the closed edges of a tensor network in any order will give us the same resulting tensor.*

*Proof.* Let us consider a graph  $G$  which might contain open edges. We index edges of a graph  $G$  using  $i_1, i_2, \dots, i_N$  where  $N$  is the number of edges in  $G$ . The tensor of a vertex is indexed with increasing order of the subscripts of the indices. For examples, consider a vertex  $v$  and denote by  $i_{v_1}, i_{v_2}, \dots, i_{v_{d(v)}}$  its adjacent edges, where  $d(v)$  means the degree of  $v$ , and  $v_1 < v_2 < \dots < v_{d(v)}$ . The tensor of  $v$  will be denoted by  $T_{i_{v_1} i_{v_2} \dots i_{v_{d(v)}}}^v$ . All merged tensors are also indexed in this way. Let  $O = \{i_{o_1}, i_{o_2}, \dots, i_{o_k}\}$  denote all open edges with  $o_1 < o_2 < \dots < o_k$ .

Consider an arbitrary edge ordering  $\pi$  of the closed edges. We know that the final vertex left at the end of the contraction process according to  $\pi$  only has  $k$  open edges  $i_{o_1}, i_{o_2}, \dots, i_{o_k}$ . Let denote by  $T_{i_{o_1} i_{o_2} \dots i_{o_k}}^\pi$  the tensor of this vertex.  $T_{i_{o_1} i_{o_2} \dots i_{o_k}}^\pi$  is the tensor contraction of  $T^{\pi(1)}, T^{\pi(2)}, \dots, T^{\pi(N-k-1)}$  and  $T^{\pi(N-k)}$ . Equation (3.1) tells us that

$$T_{i_{o_1} i_{o_2} \dots i_{o_k}}^\pi = \sum T_{i_{\pi(1)1} i_{\pi(1)2} \dots i_{\pi(1)d(\pi(1))}}^{\pi(1)} T_{i_{\pi(2)1} i_{\pi(2)2} \dots i_{\pi(2)d(\pi(2))}}^{\pi(2)} \dots T_{i_{\pi(N-k)1} i_{\pi(N-k)2} \dots i_{\pi(N-k)d(\pi(N-k))}}^{\pi(N-k)}.$$

The sum above is taken over all indices corresponding to the closed edges. A permutation of  $\pi(1), \dots, \pi(N-k)$  results in changes of positions of  $T_{i_{\pi(1)1} i_{\pi(1)2} \dots i_{\pi(1)d(\pi(1))}}^{\pi(1)}$ ,

$T_{i_{\pi(2)1} i_{\pi(2)2} \dots i_{\pi(2)d(\pi(2))}}^{\pi(2)}, \dots, T_{i_{\pi(N-k)1} i_{\pi(N-k)2} \dots i_{\pi(N-k)d(\pi(N-k))}}^{\pi(N-k)}$  in the sum, which do not change the final result. Therefore,  $T_{i_{o_1} i_{o_2} \dots i_{o_k}}^{\pi}$  does not depend on the ordering  $\pi$ . This proves the theorem.  $\square$

### 3.3 The Contracting Tensor Network Formalism

In this section, we introduce the tensor formalism in which quantum states and quantum operations are represented by tensors. We also look at the correspondence between the evolution of a quantum circuit and the contraction process of the tensor network describing the circuit.

We focus on tensors whose indices can take 4 values which are set to be elements of the set  $\Pi = \{|i\rangle\langle j| : i, j \in \{0, 1\}\}$ .

**Definition 3.3.1.** *Let  $\rho$  be an  $a$ -qubit density operator. The tensor representation of  $\rho$  is a rank- $a$  tensor, denoted by  $[T_{\sigma_1, \sigma_2, \dots, \sigma_a}^\rho]$ , where  $\sigma_1, \sigma_2, \dots, \sigma_a$  run over  $\Pi$  and*

$$T_{\sigma_1, \sigma_2, \dots, \sigma_a}^\rho = \text{tr}(\rho \cdot (\otimes_{i=1}^a \sigma_i)^\dagger),$$

where the above dot operation is interpreted as the matrix multiplication.

Graphically, the tensor of an  $a$ -qubit density operator is drawn as a vertex with  $a$  open wires as depicted in Figure 3.3.

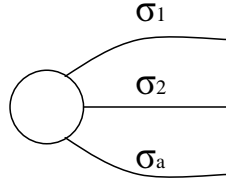


Figure 3.3: Graphical representation of the tensor of a density operator on  $a$  qubits.

Let us consider the tensor of a general pure quantum state on 1 qubit  $|\psi\rangle = c_0|0\rangle + c_1|1\rangle$ , whose density operator is  $\rho = |\psi\rangle\langle\psi|$ . By definition, the tensor of  $\rho$  is a rank-1 tensor, denoted by  $T_\sigma^\rho$  where  $\sigma \in \Pi$ , and

$$T_\sigma^\rho = \text{tr}(\rho \cdot \sigma^\dagger).$$

Wherever  $\sigma = |i\rangle\langle j|$  is used as subscript, we decode it as a 2-bit string  $ij$ . We have

$$\begin{aligned} T_{ij}^\rho &= \text{tr}(\rho \cdot (|i\rangle\langle j|)^\dagger) \\ &= \text{tr}(|\psi\rangle\langle\psi| \cdot |j\rangle\langle i|) \\ &= \langle\psi|j\rangle\langle i|\psi\rangle \\ &= c_j^* c_i. \end{aligned}$$

Therefore, if we consider the tensor  $T^\rho$  as a vector, it would be  $[c_0 c_0^*, c_0 c_1^*, c_1 c_0^*, c_1 c_1^*]$ . Let us look at the matrix  $\rho$

$$\rho = \begin{pmatrix} c_0 c_0^* & c_0 c_1^* \\ c_1 c_0^* & c_1 c_1^* \end{pmatrix}.$$

We see that there is a one-one correspondence between the elements of the tensor  $T^\rho$  and the elements of  $\rho$ . Indeed, this relation is stated in Proposition 3.3.2. Before that, we consider two maps mapping an element in  $\Pi^n$  to  $n$ -bit binary strings. The first map takes all ket vectors of the elements in  $\Pi^n$

$$f : \Pi^n \rightarrow \{0, 1\}^n$$

$$(|a_1\rangle\langle b_1|, \dots, |a_n\rangle\langle b_n|) \mapsto (a_1, \dots, a_n)$$

The second map takes all the bra vectors

$$s : \Pi^n \rightarrow \{0, 1\}^n$$

$$(|a_1\rangle\langle b_1|, \dots, |a_n\rangle\langle b_n|) \mapsto (b_1, \dots, b_n)$$

We usually write  $a_1 \dots a_n$  instead of  $(a_1, \dots, a_n)$ .

**Proposition 3.3.2.** *Given an  $n$ -qubit density operator  $\rho$  and its tensor  $T_{\sigma_1, \dots, \sigma_n}^\rho$ . Then the following equality holds*

$$T_{\sigma_1, \dots, \sigma_n}^\rho = \rho_{f(\sigma_1, \dots, \sigma_n), s(\sigma_1, \dots, \sigma_n)}.$$

*Proof.* For the sake of simplicity,  $f(\sigma_1, \dots, \sigma_n)$  and  $s(\sigma_1, \dots, \sigma_n)$  are denoted by  $f$  and  $s$ , respectively. Notice that  $\otimes_{i=1}^n \sigma_i = |f\rangle\langle s|$  and, hence,

$$\begin{aligned} \rho \cdot (\otimes_{i=1}^n \sigma_i)^\dagger &= \rho(|s\rangle\langle f|) \\ &= \left( \sum_{i,j \in \{0,1\}^n} \rho_{i,j} |i\rangle\langle j| \right) |s\rangle\langle f| \\ &= \sum_{i,j \in \{0,1\}^n} \rho_{i,j} |i\rangle\langle j|s\rangle\langle f| \\ &= \sum_{i \in \{0,1\}^n} \rho_{i,s} |i\rangle\langle f|. \end{aligned}$$

This gives  $\text{tr}(\rho \cdot (\otimes_{i=1}^n \sigma_i)^\dagger) = \rho_{f,s}$ , which proves the proposition.  $\square$

We now consider the tensor representation of a quantum operation. Let  $Q$  be a general quantum operator acting on  $a$  input qubits and  $b$  output qubits, i.e. it takes an  $a$ -qubit density operator to a  $b$ -qubit density operator.

**Definition 3.3.3.** *The tensor representation of  $Q$  is a rank- $(a+b)$  tensor on a  $4$ -dimensional space, denoted by  $[T_{\sigma_1, \sigma_2, \dots, \sigma_a, \tau_1, \tau_2, \dots, \tau_b}^Q]$ , where  $\sigma_1, \sigma_2, \dots, \sigma_a, \tau_1, \tau_2, \dots, \tau_b$  run over  $\Pi$  and*

$$T_{\sigma_1, \sigma_2, \dots, \sigma_a, \tau_1, \tau_2, \dots, \tau_b}^Q = \text{tr}(Q \cdot (\otimes_{i=1}^a \sigma_i) \cdot Q^\dagger \cdot (\otimes_{j=1}^b \tau_j)^\dagger).$$

The tensor  $T^Q$  is graphically described by a single vertex with  $(a+b)$  open wires as depicted in Figure 3.4.

Consider the tensor of the NOT gate. It is a tensor with two indices, denoted as  $T_{\sigma, \tau}^{NOT} = \text{tr}(N\sigma N^\dagger \tau^\dagger)$ , where

$$N = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}.$$

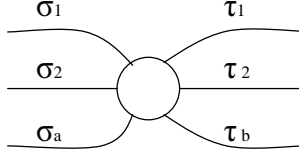


Figure 3.4: Graphical representation of the tensor of a quantum operator acting on  $a$  input qubits and  $b$  output qubits.

We denote by  $T_{ij,ab}^{NOT}$  the tensor  $T_{\sigma,\tau}^{NOT}$  where  $\sigma = |i\rangle\langle j|$  and  $\tau = |a\rangle\langle b|$ . We get

$$\begin{aligned} T_{ij,ab}^{NOT} &= \text{tr}(N|i\rangle\langle j|N^\dagger(|a\rangle\langle b|)^\dagger) \\ &= \text{tr}(|\bar{i}\rangle\langle\bar{j}||b\rangle\langle a|) \\ &= \langle\bar{j}|b\rangle\langle a|\bar{i}\rangle. \end{aligned}$$

(where the bar notation denotes the bit complement operation).  
The tensor  $T^{NOT}$  is given in Table 3.1.

$\tau \backslash \sigma$	00	01	10	11
00	0	0	0	1
01	0	0	1	0
10	0	1	0	0
11	1	0	0	0

Table 3.1: The tensor representation of the NOT gate

The following proposition shows a relation between the tensor representation and the tensor product of two operators.

**Proposition 3.3.4.** *Given two quantum operators  $A, B$  where  $A$  acts on  $a_1$  input qubits and  $b_1$  output qubits, and  $B$  acts on  $a_2$  input qubits and  $b_2$  output qubits. Let  $T_{\sigma_1, \dots, \sigma_{a_1}, \tau_1, \dots, \tau_{b_1}}^A, T_{\sigma_{a_1+1}, \dots, \sigma_{a_1+a_2}, \tau_{b_1+1}, \dots, \tau_{b_1+b_2}}^B, T_{\sigma_1, \dots, \sigma_{a_1+a_2}, \tau_1, \dots, \tau_{b_1+b_2}}^{A \otimes B}$  represent the tensors of  $A, B$  and  $A \otimes B$  respectively. The following equality holds*

$$T_{\sigma_1, \dots, \sigma_{a_1+a_2}, \tau_1, \dots, \tau_{b_1+b_2}}^{A \otimes B} = T_{\sigma_1, \dots, \sigma_{a_1}, \tau_1, \dots, \tau_{b_1}}^A \cdot T_{\sigma_{a_1+1}, \dots, \sigma_{a_1+a_2}, \tau_{b_1+1}, \dots, \tau_{b_1+b_2}}^B.$$

*Proof.* We recall a property of the *trace* operation: given two square matrices  $X, Y$ , the trace of their tensor product equals the product of their traces. This is true since  $\text{tr}(A) = \sum_i a_{ii}, \text{tr}(B) = \sum_j b_{jj}$  and  $\text{tr}(A \otimes B) = \sum_{ij} a_{ii} b_{jj}$ . Using this property, we get

$$\begin{aligned} T_{\sigma_1, \dots, \sigma_{a_1+a_2}, \tau_1, \dots, \tau_{b_1+b_2}}^{A \otimes B} &= \text{tr} \left( (A \otimes B) (\otimes_{i=1}^{a_1+a_2} \sigma_i) (A \otimes B)^\dagger (\tau_{i=1}^{b_1+b_2} \sigma_j)^\dagger \right) \\ &= \text{tr} \left( \left( A (\otimes_{i=1}^{a_1} \sigma_i) A^\dagger (\otimes_{j=1}^{b_1} \tau_j)^\dagger \right) \otimes \left( B (\otimes_{i=a_1+1}^{a_1+a_2} \sigma_i) B^\dagger (\otimes_{j=b_1+1}^{b_1+b_2} \tau_j)^\dagger \right) \right) \\ &= \text{tr} \left( A (\otimes_{i=1}^{a_1} \sigma_i) A^\dagger (\otimes_{j=1}^{b_1} \tau_j)^\dagger \right) \text{tr} \left( B (\otimes_{i=a_1+1}^{a_1+a_2} \sigma_i) B^\dagger (\otimes_{j=b_1+1}^{b_1+b_2} \tau_j)^\dagger \right) \\ &= T_{\sigma_1, \dots, \sigma_{a_1}, \tau_1, \dots, \tau_{b_1}}^A T_{\sigma_{a_1+1}, \dots, \sigma_{a_1+a_2}, \tau_{b_1+1}, \dots, \tau_{b_1+b_2}}^B. \end{aligned}$$

□

In the tensor formalism, measurement operators are described by the so-called measurement scenario.

**Definition 3.3.5.** *Let  $m$  be an integer. Consider  $m$  single-qubit measurements  $\{M(1), I - M(1)\}, \{M(2), I - M(2)\}, \dots, \{M(m), I - M(m)\}$ , where each  $M(i)$  is a measurement operator. A measurement scenario on  $m$  qubits is a function  $S : \{1, \dots, m\} \rightarrow GL(2)$  such that  $S(i)$  is either  $M(i)$  or  $I - M(i)$ .*

( $GL(2)$  denotes the group of invertible  $2 \times 2$  matrices over the complex field).

We note that if a qubit  $i$  is not measured, we set  $S(i)$  to be the identity operator.

In the next section, we show how to use the tensor formalism to simulate the computation of a circuit with a given input state.

### 3.4 Simulation of Quantum Circuits

We begin this section with the definition of quantum circuit graphs.

**Definition 3.4.1.** *Given a quantum circuit  $C$  with an input  $\rho$ . Each segment of qubit wires is marked by a distinct label. We index the tensor of each gate in  $C$  by the labels of its incoming qubit segments and outgoing qubit segments. The circuit graph of  $C$  is the tensor network of all these tensors.*

By a segment of a qubit wire, we mean a part of the qubit wire lying in between two consecutive gates on that wire. Figure 3.5 illustrates an example.

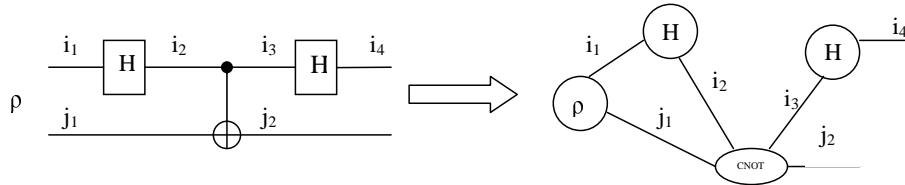


Figure 3.5: Graph representation of a quantum circuit.

Notice that open edges could appear in circuit graphs, e.g. the edges  $i_4, j_2$  in the example in Figure 3.5. If we contract all closed edges in the circuit graph of a quantum circuit  $C$  on  $n$  qubits, we will end up with a single tensor consisting of  $n$  open edges. The idea of simulating quantum circuits using the tensor formalism is that this final tensor indeed represents the outcome state of the circuit  $C$  executed on the input  $\rho$ . The main aim of this section is to prove this statement, which enables us to work on the circuit graph instead of the quantum circuit.

We first consider the situation when the circuit  $C$  has only one gate.

**Theorem 3.4.2.** *Let  $C$  be a quantum circuit that has only one gate  $U$  acting on  $n$  input qubits and  $m$  output qubits, and consider an input  $\rho$ . Then the contraction of the two tensors of  $U$  and  $\rho$  gives us the tensor of the outcome state  $U\rho U^\dagger$ .*

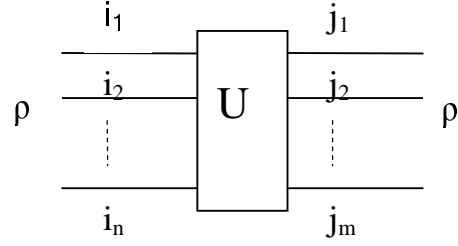


Figure 3.6: Simulation of a quantum circuit consisting of one gate.

*Proof.* We first rewrite the tensors of  $U$ ,  $\rho$  and the outcome  $\rho' = U\rho U^\dagger$ .

$$\begin{aligned} T_{i_1 \dots i_n}^\rho &= \text{tr} \left( \rho (\otimes_{k=1}^n i_k)^\dagger \right), \\ T_{i_1 \dots i_n, j_1 \dots j_m}^U &= \text{tr} \left( U (\otimes_{k=1}^n i_k) U^\dagger (\otimes_{k=1}^m j_k)^\dagger \right), \\ T_{j_1 \dots j_m}^{\rho'} &= \text{tr} \left( \rho' (\otimes_{k=1}^m j_k)^\dagger \right) = \text{tr} \left( U \rho U^\dagger (\otimes_{k=1}^m j_k)^\dagger \right). \end{aligned}$$

(where the indices  $i_1, \dots, i_n, j_1, \dots, j_m$  run over the set  $\Pi$ ).

Suppose that  $i_k = |a_k\rangle\langle b_k|$  and  $j_k = |c_k\rangle\langle d_k|$ , where  $a_k, b_k, c_k, d_k$  are binary, and  $a = a_1 \dots a_n, b = b_1 \dots b_n, c = c_1 \dots c_m, d = d_1 \dots d_m$ . By Proposition 3.3.2, we get  $T_{i_1 \dots i_n}^\rho = \rho_{a,b}$  and

$$T_{i_1 \dots i_n, j_1 \dots j_m}^U = \left( U (\otimes_{k=1}^n i_k) U^\dagger \right)_{c,d} = \left( U |a\rangle\langle b| U^\dagger \right)_{c,d}.$$

(where the notation  $(X)_{m,n}$  means the  $(m, n)$  element of matrix  $X$ )

Contracting the tensor  $T^\rho$  to the tensor  $T^U$  gives us a new tensor, denoted by  $T_{j_1 \dots j_m}^{U\rho}$ , and

$$\begin{aligned} T_{j_1 \dots j_m}^{U\rho} &= \sum_{i_1, \dots, i_n} T_{i_1 \dots i_n}^\rho T_{i_1 \dots i_n, j_1 \dots j_m}^U \\ &= \sum_{a,b} \rho_{a,b} \left( U |a\rangle\langle b| U^\dagger \right)_{c,d} \\ &= \sum_{a,b} \left( U (\rho_{a,b} |a\rangle\langle b|) U^\dagger \right)_{c,d} \\ &= \left( U \left( \sum_{a,b} \rho_{a,b} |a\rangle\langle b| \right) U^\dagger \right)_{c,d} \\ &= (U \rho U^\dagger)_{c,d} \\ &= T_{j_1 \dots j_m}^{\rho'} \end{aligned}$$

which proves the theorem.  $\square$

We now generalize Theorem 3.4.2 to circuits of more than one gate.

**Theorem 3.4.3.** *Let  $C$  be a quantum circuit and  $\rho$  be an input. Then the contraction of the whole circuit graph of  $C$  with the input  $\rho$  gives us the tensor of the outcome state when we apply  $C$  to the input.*

*Proof.* We can always order the gates in  $C$  by the time order (from left to right):  $G_1, G_2, \dots, G_k$  for some  $k$ . Applying Theorem 3.4.2 repeatedly, we get: contracting the tensor of  $\rho$  to the tensor of  $G_1$  gives us the tensor of the quantum state after executing  $G_1$ , i.e. the state  $G_1\rho G_1^\dagger$ ; contracting the new tensor to the tensor of  $G_2$  gives us the tensor of the quantum state after executing  $G_2$ , and so on. Finally, after contracting  $G_k$ , we get the final tensor which is the tensor of the outcome.  $\square$

Theorem 3.4.3 can be generalized to general quantum operators. Each general quantum operation  $\Phi$  can be decomposed to a number of Kraus operators  $K_i$  such that, for every density operator  $\rho$  that  $\Phi$  can act on, we have  $\Phi(\rho) = \sum_i K_i \rho K_i^\dagger$ . Notice that the tensor representation of  $\Phi$  will be

$$T_{\sigma_1, \sigma_2, \dots, \sigma_a, \tau_1, \tau_2, \dots, \tau_b}^\Phi = \text{tr} \left( \sum_i (K_i \cdot (\otimes_{k=1}^a \sigma_k) \cdot K_i^\dagger) \cdot (\otimes_{j=1}^b \tau_j)^\dagger \right),$$

which can be written as  $\sum_i \text{tr}(K_i \cdot (\otimes_{k=1}^a \sigma_k) \cdot K_i^\dagger \cdot (\otimes_{j=1}^b \tau_j)^\dagger)$ . Theorem 3.4.3 does not require the unitary property of quantum gates. Hence it still holds with Kraus operators. The linearities of the sum and trace operators guarantee that the theorem will hold for general quantum operators.

We now consider measurement operators on the tensor formalism. The probability of observing a measurement outcome can also be computed as follows: first attach the tensor of the corresponding measurement scenario to the circuit graph, and then perform contraction of the whole resulting tensor network.

**Theorem 3.4.4.** (Markov and Shi [13]) *Let  $C$  be a quantum circuit,  $\rho$  be an input, and  $\tau$  be a measurement scenario. Contracting the tensor network of  $G_C$  together with  $\rho$  and  $\tau$  gives a rank-0 tensor which is the probability that  $\tau$  is realized at the end of the circuit.*

We have seen that computations on quantum circuits can be translated to contractions of circuit graphs. One advantage of using this translation in simulating quantum circuits is that contractions of circuit graphs can be performed with any ordering of edges, whereas computations on the circuit model must follow gates in the time order. This order-independent advantage gives us ways to optimize the contraction processes by looking for an optimal ordering in which the contraction process requires the least number of computational steps.

**Theorem 3.4.5.** *Contracting all the closed edges of a circuit graph in any order will give us the same resulting tensor.*

*Proof.* Theorem 3.4.5 is a special case of Theorem 3.2.4 when the tensor network is a circuit graph.  $\square$

In the next section, we will make use of the order-independent property of the contraction operation to estimate the contraction complexity of a quantum circuit graph.

### 3.5 Contraction Complexity and Treewidth

We begin this section by introducing the treewidth concept. Unless otherwise stated, graphs mentioned here are undirected and may have multiple edges or

loops. For every graph  $G$ , the set of its edges is denoted by  $E(G)$  and the set of its vertices is denoted by  $V(G)$ .

**Definition 3.5.1.** Let  $G$  be a graph. A tree decomposition of  $G$  is a tree  $T$ , together with a function that maps each vertex  $w \in V(T)$  to a subset  $B_w \subseteq V(G)$ , which is called a bag, such that the following three conditions hold:

- (C1) Each vertex of  $G$  must appear in at least one bag, i.e.  $\cup_{v \in V(T)} B_v = V(G)$ .
- (C2) For each edge of  $G$ , there is at least one bag containing both of its end vertices.
- (C3) All bags (vertices) in  $T$  containing a given vertex in  $G$  must form a connected subtree of  $T$ .

The width of a tree decomposition is the maximum size of a bag minus one, i.e.  $\max_{w \in V(T)} |B_w| - 1$ . The treewidth of  $G$ , denoted by  $tw(G)$  is the minimum width among all tree decompositions of  $G$ .

The  $-1$  in the definition of the width of a tree decomposition is introduced for convenience only: the smallest treewidth value is one and it happens when the graph is a tree. Intuitively, the smaller a graph's treewidth, the closer it is to a tree, and a tree decomposition is a drawing of the graph to make it look like a tree as much as possible. Figure 3.7 illustrates an example of a tree decomposition.

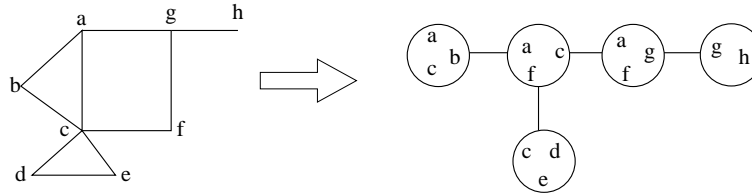


Figure 3.7: An example of a graph with a tree decomposition of width two.

Computing the treewidth of a graph is a hard problem. Arnborg et al. [2] showed that determining whether a graph has treewidth at most  $k$  is NP-complete. Nevertheless, varieties of heuristic algorithms have been proposed to find upper bounds or lower bounds on a graph's treewidth. Bodlaender [3] gave a survey of these algorithms. Instead of explaining these algorithms here, we give another definition of treewidth which is equivalent to Definition 3.5.1. This provides another way of interpreting a graph's treewidth.

**Definition 3.5.2.** Given a graph  $G$ . A contraction ordering  $\pi$  is an ordering of all the edges of  $G$ . The complexity of  $\pi$ , denoted by  $cc(\pi)$ , is the maximal rank of a merged vertex during the contraction process. The contraction complexity of  $G$ , denoted by  $cc(G)$ , is defined as the minimum complexity among all contraction orderings.

This type of contraction is called one-edge-at-a-time [13] since at every step you contract exactly one edge. This differs from the tensor contraction which contract all edges between two tensors at one time. The one-edge-at-a-time contraction, however, increases the maximal rank of a merged vertex by at most

two-fold. The reason is: if we need to contract edges between two vertices  $u$  and  $v$  of degrees  $l + k$  and  $l + k'$ , respectively, which share  $l$  common edges, the one-edge-at-a-time contraction will create vertices of degrees  $k + k' + l - 1, k + k' + l - 2, \dots, k + k'$  which all do not exceed  $r(u) + r(v)$ .

To introduce the relation between the treewidth and the contraction complexity of a graph  $G$ , we consider the so-called line graph of  $G$ .

**Definition 3.5.3.** *The line graph of  $G$ , denoted by  $G^*$ , is the graph whose vertex set is the set of edges of  $G$ , and two vertices of  $G^*$  are adjacent if and only if the two corresponding edges in  $G$  share a common vertex.*

Figure 3.8 illustrates an example.

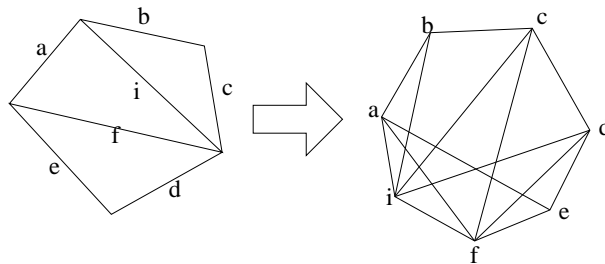


Figure 3.8: An example of a graph (on the left hand side) with 7 edges and its line graph (on the right hand side) with 3 vertices.

The following theorem states that the treewidth and the contraction complexity measures are indeed equivalent as long as we consider classes of graphs with bounded maximal degrees.

**Theorem 3.5.4.** (Markov and Shi [13]) *For any graph  $G$  of maximal degree  $\Delta G$ , we have*

1.  $cc(G) = tw(G^*)$ .
2.  $(tw(G) - 1)/2 \leq tw(G^*) \leq \Delta(G)(tw(G) + 1) - 1$ .

We now have all necessary concepts to study the complexity of contracting a whole circuit graph. We begin with Theorem 3.5.5 which gives an upper bound for that.

**Theorem 3.5.5.** (Markov and Shi [13]) *Given a tensor network  $N$  of a size- $T$  quantum circuit with a classical input and single qubit measurements at the end of the circuit, and a contraction process specified by an ordering of edges in  $N$ ; let  $d$  be the maximum rank of all the tensors that appear in the process. The contraction takes  $T \cdot O(16^d)$  steps; and there exists a contraction process that takes  $T \cdot O[16^{cc(G)}]$  steps.*

*Proof.* The algorithm scans edge by edge in the given order and at each time compute the new tensor based on Equation (3.1). The complexity of computing a new tensor is at most  $O(4^{2d})$  since  $d$  is the maximum rank of all the tensors. The total runtime, hence, is bounded by  $2T \cdot O(4^{2d}) = O(T \cdot 16^d)$ .

Let us denote by  $D$  the maximum among all  $d$ 's. It is obvious that there exists a contraction process taking  $T \cdot O(16^D)$  steps. We know, furthermore, that  $cc(G) \leq D$ . Hence, this contraction costs us  $T \cdot O(16^D)$  steps.  $\square$

Theorem 3.5.5 does not show us how to find an optimal contraction process. Nevertheless, given a tree decomposition of  $G^*$ , Markov and Shi show that we can find a contraction ordering of  $G$  of complexity not bigger than the width of the tree decomposition.

**Theorem 3.5.6.** (Markov and Shi [13]) *There is an efficient deterministic algorithm that given a tree decomposition of  $G^*$  of width  $d$  will output a contraction ordering  $\pi$  of  $G$  with  $cc(\pi) \leq d$ .*

Finding an optimal tree decomposition is an NP-hard problem ([2]). However, Robertson and Seymour [17] gave us an algorithm for finding a "good" tree decomposition of any given graph.

**Theorem 3.5.7.** (Robertson and Seymour [17]) *There is a deterministic algorithm that given a graph  $G$  will output a tree decomposition of width  $O(tw(G))$  in time  $|V(G)|^{O(1)} \exp[O(tw(G))]$ .*

We have had enough materials for our main simulation algorithm

**Theorem 3.5.8.** (Markov and Shi [13]) *Let  $C$  be a quantum circuit of size  $T$ , with  $n$  input and  $m$  output qubits,  $x \in \{0, 1\}^n$  be an input, and  $\tau$  be a measurement scenario. Denote by  $G$  the circuit graph of  $C$ . Then the probability that  $\tau$  is observed at the end of the circuit can be computed deterministically in time  $T^{O(1)} \exp[O(\Delta(G) \cdot tw(G))]$ .*

*Proof.* The algorithm is described below.

1. Apply the Robertson-Seymour algorithm to compute a tree decomposition  $T$  of  $G^*$  of width  $w = O(tw(G^*))$ .
2. Find a contraction ordering  $\pi$  of complexity  $w$  from  $T$  by using Theorem 3.5.6.
3. Contract  $G$  using the ordering  $\pi$ . The unique element of the final rank-0 tensor is output as the desired probability of observing  $\tau$ .

The most computation-consuming parts of the algorithm are step 2 and step 4. Theorem 3.5.7 tells us that step 2 costs  $|V(G^*)|^{O(1)} \exp[O(tw(G^*))]$  which equals  $|V(G^*)|^{O(1)} \exp[O(\Delta(G) \cdot tw(G))]$  (by Theorem 3.5.4). We know, furthermore, that the number of vertices of  $G^*$  is the number of edges of  $G$ , which is  $T$ . Step 2 of the algorithm, hence, costs us  $T^{O(1)} \exp[O(\Delta(G) \cdot tw(G))]$  steps. By Theorem 3.5.5, the complexity of step 4 is  $T \cdot O(\exp(cc(\pi))) = T \cdot O(\exp(tw(G^*))) = T \cdot O(\exp(\Delta(G) \cdot tw(G)))$ . So the overall complexity is  $T^{O(1)} \exp[O(\Delta(G) \cdot tw(G))]$ .  $\square$

## 3.6 A Class of Simulable Quantum Circuits

Theorem 3.5.8 gives us a simulation whose complexity depends solely on the treewidth measure. Considering simulation of a quantum circuit, we prefer to use parameters related to the circuit itself. To this end, Theorem 3.6.1 provides a treewidth's estimation that is based on the so-called qubit-crossing concept. Let us consider a quantum circuit  $C$  whose qubits are indexed by  $1, 2, \dots, n$ . If a gate  $G$  acts on qubits  $j_1, j_2, \dots, j_k$  (and suppose that  $j_1 < j_2 < \dots < j_k$ ), then  $G$  is said to cross the qubit  $i$  if and only if  $j_1 \leq i \leq j_k$ .

**Theorem 3.6.1.** (Markov and Shi [13]) *Let  $C$  be a quantum circuit. Define  $r$  to be the maximum number of gates that cross any given qubit. Then the treewidth of the circuit graph  $G$  of  $C$  is bounded by  $r - 1$ ; and therefore the circuit can be simulated deterministically in time  $T^{O(1)} \exp[O(r \cdot \Delta(G))]$ .*

*Proof.* We construct a tree decomposition of  $G$  as follows: for each qubit  $i$ , where  $1 \leq i \leq n$ , define bag  $B_i$  to be the collection of all gates that cross qubit  $i$ . Consider the tree  $T$  consisting of one path of  $n$  vertices (bags)  $B_1 - B_2 - \dots - B_n$ . We now check the three conditions in Definition 3.5.2: Condition (C1) holds since each gate  $G$  must act on some qubit, say qubit  $i$ , then  $G$  is contained in the bag  $B_i$ . Condition (C2) holds since every segment of a qubit, say qubit  $j$ , connecting two gates,  $B_j$  contains both the two gates. Condition (C3) holds since for each gate acting on qubits  $j_1 < j_2 < \dots < j_k$ , the set of bags that contain the gate is  $\{B_{j_1}, B_{j_1+1}, \dots, B_{j_k}\}$ , which forms a subtree of  $T$ . The tree  $T$ , hence, is a tree decomposition of  $G$ .

By the definition of the parameter  $r$ , we have  $r = \max_{1 \leq i \leq n} |B_i|$ . Therefore the width of  $T$  is  $r - 1$  and  $tw(G_c) < r$ .  $\square$

We notice that  $O(r)$  is not always an optimal bound on the treewidth of  $G_C$ . Let us consider the following example where the treewidth is indeed very small, but  $r$  might be as large as we want: 2-qubit circuit  $C$  consisting of  $n$  CNOT gates. The circuit graph will be a path connecting the  $n$  gates, and have treewidth 1. However, the maximum qubit-crossing number of this circuit is  $n$ .

We now consider an alternative qubit-cross concept which we have not been able to compare with the one above. Consider a quantum circuit  $C$ . Instead of indexing the qubits by  $1, \dots, n$ , we arrange the qubits in a virtual tree of arbitrary shape (in the original setup, the tree is of line shape). We notice that this tree arrangement has no physical meaning. For a gate  $G$  acting on qubits  $j_1 < j_2 < \dots < j_k$ , it is said that  $G$  crosses the qubit  $i$  if and only if there exist  $j_p, j_q$ , where  $1 \leq p, q \leq k$ , such that, in the qubit tree, the qubit  $i$  is contained in the path connecting  $j_p$  to  $j_q$ . Theorem 3.6.2 can be proven similarly as the proof of Theorem 3.6.1.

**Theorem 3.6.2.** *Let  $C$  be a quantum circuit in which each gate has an equal number of input and output qubits, and the qubits are arranged in a tree. With respect to this arrangement, define  $d$  to be the maximum number of gates that cross any given qubit. Then the treewidth of the circuit graph  $G$  of  $C$  is upper bounded by  $d - 1$ ; and therefore the circuit can be simulated deterministically in time  $T^{O(1)} \exp[O(d \cdot \Delta(G))]$ .*

Theorem 3.6.2 and Theorem 3.6.1 mean that if  $d \cdot \Delta(G)$  or  $r \cdot \Delta(G)$  of a quantum circuit is polylogarithmic, then the circuit can be classically simulated. Let us consider a quantum circuit of size  $T$ . In any qubit-crossing concept above, each gate crosses at least one qubit. The number of crosses, hence, is at least  $T$ . By the Pigeon Principle, both  $d$  and  $r$  are not smaller than  $T/n$ . Therefore, if  $T$  is equal to  $n \cdot \text{poly}(n)$ , then both theorems do not provide us any conclusion on classical simulability of the circuit.

### 3.7 Relation between the Contracting Tensor Network and the Matrix Product Formalisms

In this section, we exploit a relation between the matrix product formalism and the tensor formalism. It turns out that a matrix product state can be interpreted as a tensor network if we do not restrict ourselves to tensor whose indices are in the set  $\Pi$  as defined in Definition 3.3.1.

Let us consider a tensor product state  $|\psi\rangle$  represented by  $n$  tensors  $\Gamma_{\alpha_1}^{[1]i_1}, \Gamma_{\alpha_1\alpha_2}^{[2]i_2}, \dots, \Gamma_{\alpha_{n-2}\alpha_{n-1}}^{[n-1]i_{n-1}}, \Gamma_{\alpha_{n-1}}^{[n]i_n}$  where each index  $i_k$  can take two values, either 0 or 1, and each index  $\alpha_k$  can take  $\chi$  values in the set  $\{1, \dots, \chi\}$ . Then if in the computational basis  $|\psi\rangle$  is described as  $\sum_{i_1, \dots, i_n} \psi_{i_1 \dots i_n} |i_1 \dots i_n\rangle$ , we have

$$\psi_{i_1 \dots i_n} = \sum_{\alpha_1, \dots, \alpha_n} \Gamma_{\alpha_1}^{[1]i_1} \Gamma_{\alpha_1\alpha_2}^{[2]i_2} \dots \Gamma_{\alpha_{n-2}\alpha_{n-1}}^{[n-1]i_{n-1}} \Gamma_{\alpha_{n-1}}^{[n]i_n}. \quad (3.2)$$

Therefore if we consider  $|\psi\rangle$  as a rank- $n$  tensor, then Equation (3.2) states that the tensor  $|\psi\rangle$  is exactly the contraction product of the tensors  $\Gamma_{\alpha_1}^{[1]i_1}, \Gamma_{\alpha_1\alpha_2}^{[2]i_2}, \dots, \Gamma_{\alpha_{n-2}\alpha_{n-1}}^{[n-1]i_{n-1}}, \Gamma_{\alpha_{n-1}}^{[n]i_n}$ . This interpretation is illustrated in Figure 3.9. Note, how-

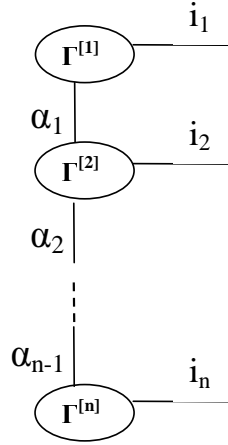


Figure 3.9: Tensor interpretation of an n-qubit tensor product state.

ever, that in order to have an efficient computation, we now expect not only the rank of each tensor in the network but also the domain of each index to be small. The range of each index  $i_k$  is fixed, viz. the set  $\{0, 1\}$ . But the range of  $\alpha_k$  depends on the maximal Schmidt rank  $\chi$ .

Applying a 1-qubit gate  $U$  to the qubit  $j$  now is just the contraction of the tensor  $\Gamma^{[j]}$  to  $U$  which might be interpreted as a rank-2 tensor on a 2-dimensional space  $[U_{ki}]_{k,i \in \{0,1\}}$ . A justification of this statement is Equation (2.12). Figure 3.10 depicts this contraction.

Applying a 2-qubit gate  $U$  is rather more complicated. The idea is: suppose that we are applying  $U$  to two consecutive qubits  $k$  and  $k+1$ , we have a cycle between the three tensors  $\Gamma^{[k]}, \Gamma^{[k+1]}$  and  $U$ . The first step is to make contraction of this cycle to get only one resulting tensor. Now what we get is a

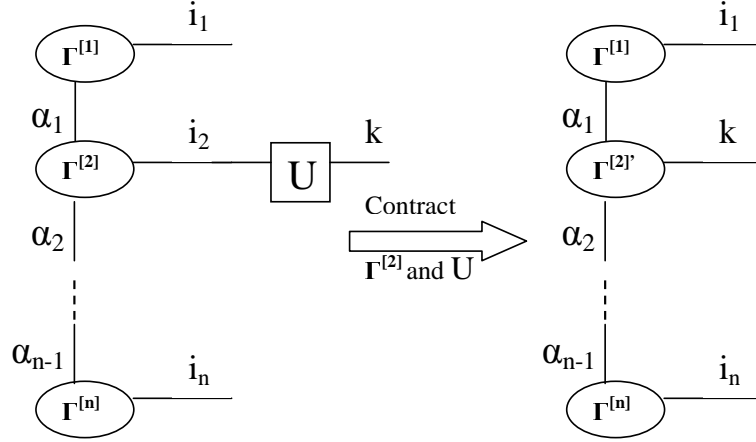


Figure 3.10: Tensor interpretation of applying a 1-qubit gate to a tensor product state where the tensor  $\Gamma^{[2]'}$  is the contraction product of  $\Gamma^{[2]}$  and  $U$ .

network of  $(n - 1)$  tensors with  $n$  open edges. To restore to the standard setting which consists of  $n$  tensors, we decompose the resulting tensor in the first step into two tensors connected by a new edge. This 2-step procedure is depicted in Figure 3.11.

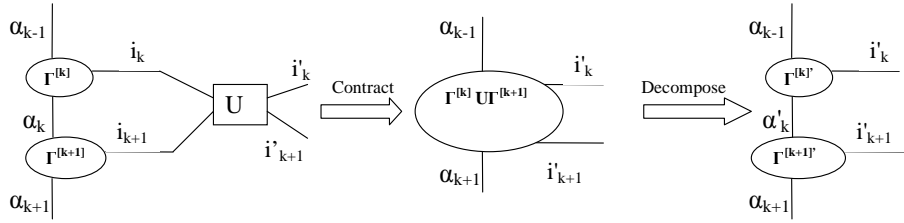


Figure 3.11: Contraction process corresponding to the process of applying a 2-qubit gate to two consecutive qubits on a tensor product state.

The tensor interpretation of the matrix product formalism also allows us to compute the inner product of two matrix product states. Let us consider two states  $|\psi\rangle$  and  $|\phi\rangle$  which are represented by  $\Gamma_{\alpha_1}^{[1]i_1}, \Gamma_{\alpha_1\alpha_2}^{[2]i_2}, \dots, \Gamma_{\alpha_{n-2}\alpha_{n-1}}^{[n-1]i_{n-1}}, \Gamma_{\alpha_{n-1}}^{[n]i_n}$  and  $\Omega_{\beta_1}^{[1]i_1}, \Omega_{\beta_1\beta_2}^{[2]i_2}, \dots, \Omega_{\beta_{n-2}\beta_{n-1}}^{[n-1]i_{n-1}}, \Omega_{\beta_{n-1}}^{[n]i_n}$ , respectively, in the matrix product formalism.

**Theorem 3.7.1.** *The inner product of  $|\psi\rangle$  and  $|\phi\rangle$  is the result of the contraction of the whole tensor network in Figure 3.12. Furthermore, this can be achieved in  $O(n\alpha^2\beta^2)$  steps, where  $\alpha, \beta$  are the maximal domain size among domains of  $\alpha_1, \dots, \alpha_n$  and  $\beta_1, \dots, \beta_n$ , respectively.*

*Proof.* We notice that the tensor network consisting of  $\Omega_{\beta_1}^{[1]*}, \Omega_{\beta_1\beta_2}^{[2]*}, \dots, \Omega_{\beta_{n-2}\beta_{n-1}}^{[n-1]*}, \Omega_{\beta_{n-1}}^{[n]*}$  on the right-hand side of Figure 3.12 represents the state  $\langle\phi|$ . The tensor network, therefore, represents the product  $\langle\phi|\psi\rangle$  which is the inner product between  $|\psi\rangle$  and  $|\phi\rangle$ . The contraction process with the edge order  $\{i_1, \dots, i_n, \alpha_1, \beta_1, \dots, \alpha_n, \beta_n\}$  costs us  $O(n\alpha^2\beta^2)$  steps.  $\square$

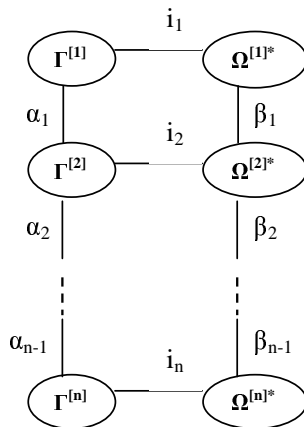


Figure 3.12: Tensor network computing the inner product between two matrix product states, where the star \* denotes the complex conjugation.

As consequence, Corollary 3.7.2 allows us to compute the length of an unnormalized MPR state.

**Corollary 3.7.2.** *The length of an unnormalized vector  $|\psi\rangle$ , which is  $\langle\psi|\psi\rangle$ , can be computed in  $O(n\alpha^4)$  steps.*

Let us consider two MPR states  $|\psi\rangle$ ,  $|\phi\rangle$ , and a measurement  $\{M_0, M_1\}$  where  $M_0 = |\phi\rangle\langle\phi|$ ,  $M_1 = I - |\phi\rangle\langle\phi|$ . The probability of observing 0 is  $|\langle\psi|\phi\rangle|^2$ , which can be efficiently computed as long as the matrix product representations of  $|\psi\rangle$ ,  $|\phi\rangle$  are efficient. We, thus, have shown the following corollary of Theorem 3.7.1.

**Corollary 3.7.3.** *If  $|\psi\rangle, |\phi\rangle$  can be represented efficiently in the MPR, then the probability distribution of the outcomes of the measurement  $\{M_0, M_1\}$  can be computed efficiently as well.*

We are currently working on the following combination of the contracting tensor network formalism and the MPR formalism: consider a quantum circuit with a given input state. We use the MPR formalism to represent the input state. The tensor representation of this MPR will then be connected to the open input edges of the circuit graph of the quantum circuit. Instead of updating quantum gates with the left-to-right ordering, we might do some edge contractions first. Updating the new tensors to the MPR state will then be performed. We know that the maximum rank of a tensor is bounded by the treewidth of the circuit. Furthermore, Theorem 3.7.4, as a generalization of Lemma 4 in [9], tells us that applying a tensor of rank  $k$  to a state increases the maximum Schmidt rank of the state by a factor of at most  $4^{k/2}$ . These facts suggest that the ratio between the maximum Schmidt ranks of the initial state and the final state might be bounded by  $\exp(w)$ , where  $w$  is the treewidth of the circuit. This conjecture, in cases it is true, would imply that starting with an initial product state, the maximum Schmidt rank of the final state is bounded by  $\exp(w)$ .

**Theorem 3.7.4.** *Let  $|\psi\rangle$  have Schmidt rank  $r$  for the partition at the  $i$ th qubit  $1 \dots i | (i+1) \dots n$ , and  $|\psi'\rangle$  be obtained from  $|\psi\rangle$  by applying a  $k$ -qubit gate  $U$ . Let*

$r'$  be the maximum Schmidt rank of  $|\psi'\rangle$  for the same partition. The following inequality holds

$$r' \leq 4^{\lfloor k/2 \rfloor} \cdot r,$$

where  $\lfloor x \rfloor$  means the largest integer not bigger than  $x$ .

*Proof.* We consider two cases:

Case 1: all affected qubits are in one half of the partition. Hence, the Schmidt rank according to this partition does not change, i.e.  $r' = r$ .

Case 2: qubit  $i$  lies in between the affected qubits. We use SWAP gates to move the affected qubits next to the border of the partition, as illustrated in Figure 3.13. These moves do not change the Schmidt rank since they act locally in the partition. If there are  $m$  affected qubits in the first half, and  $k - m$  in the second half, then after the movements, the affected qubits are qubits ranging from  $i - m + 1, \dots, i, i + 1, i + k - m$ .

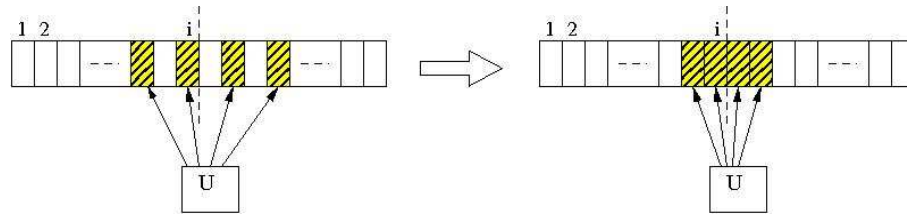


Figure 3.13: Movement of the affected qubits to the border of the partition at qubit  $i$ .

Let  $A$  denote the subsystem consisting of  $m$  affected qubits on the left hand side of qubit  $i$  and  $B$  denote the system consisting of  $k - m$  affected qubits on the right hand side. Let  $|\psi\rangle = \sum_{l=1}^r |a_l\rangle|b_l\rangle$  be the Schmidt decomposition for the partition at qubit  $i$  after applying the SWAP gates. For each  $l$ , let  $\sigma_l$  be the reduced state of subsystem  $B$  in  $|b_l\rangle$ , which is a mixed density matrix of  $k - m$  qubits. Therefore, there exist at most  $2^{k-m}$  eigenstates of  $\sigma_l$ , denoted as  $|e_l^1\rangle, |e_l^2\rangle, \dots, |e_l^{2^{k-m}}\rangle$ .

In the new state  $|\psi'\rangle = U|\psi\rangle$ , we consider the subsystem  $C$  consisting of qubits  $1, \dots, i$ , plus all the qubits in  $B$ . The reduced state of subsystem  $C$  will be spanned by the  $2^{k-m} \cdot r$  vectors  $U(|a_l\rangle|e_l^1\rangle), \dots, U(|a_l\rangle|e_l^{2^{k-m}}\rangle)$  for  $l = 1, \dots, r$ . For each state  $U(|a_l\rangle|e_l^p\rangle)$ , tracing out all  $(k - m)$  qubits in  $B$  gives a reduced density matrix of qubits  $1, \dots, i$ , which has rank at most  $2^{k-m}$ . Hence, the reduced density matrix of qubits  $1, \dots, i$  in the state  $|\psi'\rangle$  will have rank at most  $2^{k-m}(2^{k-m} \cdot r) = 4^{k-m} \cdot r$ . This implies  $r' \leq 4^{k-m} \cdot r$ .

Reasoning in the same way with the qubits in subsystem  $A$  will give us:  $r' \leq 4^m \cdot r$ . Since either  $k - m$  or  $m$  must be at most  $\lfloor k/2 \rfloor$ , we conclude that  $r' \leq 4^{\lfloor k/2 \rfloor} \cdot r$ .  $\square$

### 3.8 Summary

In this chapter, we consider the contracting tensor network formalism, first introduced by Markov and Shi in [13]. Within the formalism, elements of a quantum circuits are represented by tensors. We also study the translation of the computational evolution of a quantum circuit to the contraction evolution

of the corresponding circuit graph. Analysis on complexity shows that the contraction process can be estimated using the treewidth of the graph. After that, we relate the treewidth parameter to the maximal qubit-crossing parameter, which can be more easily interpreted on quantum circuits. Finally, we show a relation between the previous formalism, the matrix product formalism, and the tensor formalism. From the point of view of this relation, a matrix product state can be interpreted as a tensor network. Consequently, some operations can be computed using this new interpretation, such as the inner product of two matrix product states, and the length of an unnormalized matrix product state. We also propose a conjecture about the amount of entanglement (defined as maximum Schmidt ranks) of the final quantum state associate with a circuit and the treewidth of the circuit graph.

# Chapter 4

## The Stabilizer Formalism

### 4.1 Introduction

The stabilizer formalism was, at the beginning, used in quantum error-correcting codes to construct stabilizer codes. However people soon realized that the formalism was also very useful in many other problems. In classical simulation of quantum circuits, it plays a very important role as it gives us an interesting class of quantum circuits that can be simulated efficiently by classical computers. This chapter focuses on the Gottesman-Knill theorem, from which a class of classically simulable quantum circuits can be derived. This chapter is based mainly on [1, 8].

### 4.2 Notation

Throughout this note, we will frequently use the four Pauli matrices

$$I = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \quad X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \quad Y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}, \quad Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}.$$

It is easy to check that these matrices have the following relations

$$\begin{aligned} X^2 &= Y^2 = Z^2 = I, \\ XY &= iZ, YZ = iX, ZX = iY, \\ YX &= -iZ, ZY = -iX, XZ = -iY. \end{aligned} \tag{4.1}$$

Note by these identities that the Pauli matrices either commute or anti-commute.

**Definition 4.2.1.** *The Pauli group on 1 qubit is defined as*

$$G_1 = \{\pm I, \pm X, \pm Y, \pm Z, \pm iI, \pm iX, \pm iY, \pm iZ\}.$$

The group  $G_1$  is generated by  $\{X, Y, iI\}$ . We will frequently write  $G_1 = \langle X, Y, iI \rangle$  where the pointed brackets denote the group generated by the enclosed elements.

**Definition 4.2.2.** *The Pauli group on  $n$  qubits, denoted by  $G_n$ , is defined as  $(G_1)^{\otimes n}$ .*

Every element in  $G_n$  can be written in the form  $\alpha\sigma_1 \otimes \sigma_2 \otimes \cdots \otimes \sigma_n \in G_n$ , where  $\alpha \in \{\pm 1, \pm i\}$ ,  $\sigma_i \in G_1$  for all  $i = 1, \dots, n$ . Theorem 4.2.3 lists some properties of the Pauli group.

**Theorem 4.2.3.** *The following statements are true:*

1.  $G_n$  is a subgroup of  $U(2^n)$ , where  $U(2^n)$  is the group of unitary  $2^n \times 2^n$  matrices.
2.  $\text{tr}(M) = 0$  for every  $M \in G_n \setminus \{\pm I, \pm iI\}$ , where  $\text{tr}(M)$  is the trace of the matrix  $M$ .
3. Every two Pauli operators either commute or anticommute.
4. If  $M = \alpha\sigma_1 \otimes \sigma_2 \otimes \cdots \otimes \sigma_n \in G_n$  has a real overall phase  $\alpha \in \{1, -1\}$ , then  $M$  is Hermitian; otherwise  $M$  is anti-Hermitian.
5. If  $M$  is an  $n$ -qubit Hermitian Pauli operator, then  $M^2 = I^{\otimes n}$ ; if  $M$  is an anti-Hermitian Pauli operator, then  $M^2 = -I^{\otimes n}$ .

*Proof.* 1. It follows from the fact that all the Pauli matrices are unitary.

2. It is easy to check that  $\text{tr}(X) = \text{tr}(Y) = \text{tr}(Z) = 0$ . Suppose that  $M = \alpha\sigma_1 \otimes \cdots \otimes \sigma_n$ . We have  $\text{tr}(M) = \alpha \cdot \prod_{i=1}^n \text{tr}(\sigma_i)$ . Since  $M \neq \pm I, \pm iI$ , at least one of the  $\sigma$  matrices must differ from  $I$  and thus has trace 0. Therefore  $\text{tr}(M) = 0$ .
3. This follows from the fact that two Pauli matrices either commute or anticommute.
4. We notice that all the Pauli matrices are Hermitian. We have  $M^\dagger = \alpha^* \sigma_1^\dagger \otimes \cdots \otimes \sigma_n^\dagger = \alpha^* \sigma_1 \otimes \cdots \otimes \sigma_n$ , which equals  $M$  if  $\alpha = \pm 1$  and equals  $-M$  if  $\alpha = \pm i$ .
5. We have  $M^2 = \alpha^2 \sigma_1^2 \otimes \cdots \otimes \sigma_n^2 = \alpha^2 I \otimes \cdots \otimes I$ , which equals  $I^{\otimes n}$  if  $\alpha = \pm 1$  and equals  $-I^{\otimes n}$  if  $\alpha = \pm i$ . □

There exists a very useful homomorphism between  $G_n$  and the additive binary group  $\mathbb{F}_2^{2n}$ . This homomorphism maps an element of  $G_n$  to a  $2n$ -dimensional binary vector. Firstly, we consider the following mapping ([22]) for the 1-qubit case:

$$\begin{aligned}
 B : G_1 &\rightarrow \mathbb{F}_2^2 \\
 \alpha I &\mapsto (0, 0) \\
 \alpha X &\mapsto (1, 0) \\
 \alpha Z &\mapsto (0, 1) \\
 \alpha Y &\mapsto (1, 1)
 \end{aligned} \tag{4.2}$$

for every  $\alpha \in \{\pm 1, \pm i\}$ . It can be easily checked that the above mapping is a homomorphism from the matrix group  $G_1$  to the additive group  $\mathbb{F}_2^2$  (this follows immediately from the equalities in Equation (4.1)). For convenience, we

sometimes use  $\sigma_{00}, \sigma_{10}, \sigma_{01}, \sigma_{11}$  for  $I, X, Z, Y$ , respectively. The homomorphism  $B$  can be extended to the general  $n$ -qubit case:

$$B : G_n \rightarrow \mathbb{F}_2^{2n}$$

$$\alpha \sigma_{x_1 z_1} \otimes \sigma_{x_2 z_2} \otimes \cdots \otimes \sigma_{x_n z_n} \mapsto (x_1, \dots, x_n, z_1, \dots, z_n)$$

We define a  $2n \times 2n$  matrix  $\Lambda$  as

$$\Lambda = \begin{pmatrix} 0 & I \\ I & 0 \end{pmatrix},$$

where the  $I$  matrices on the off-diagonal are identity matrices of size  $n$ . For any two vectors  $u, v$  in  $\mathbb{F}_2^{2n}$ , the formula  $u\Lambda v^T$  defines the so-called symplectic inner product between them. The first structural relation between the Pauli group and the binary group  $\mathbb{F}_2^{2n}$  is stated in the following proposition.

**Proposition 4.2.4.** (*van den Nest [22]*)

1. The mapping  $B$  is an homomorphism from the multiplicative group  $G_n$  to the additive group  $\mathbb{F}_2^{2n}$ .
2. Elements  $g, g'$  of the Pauli group  $G_n$  commute if and only if the symplectic inner product of  $B(g)$  and  $B(g')$  vanishes, i.e.  $B(g)\Lambda B(g')^T = 0$ . They anti-commute if and only if  $B(g)\Lambda B(g')^T = 1$ .

*Proof.* 1. It follows from the fact that the mapping (4.2) for the 1-qubit case is a homomorphism.

2. Suppose that  $g = \alpha \sigma_{x_1 z_1} \otimes \sigma_{x_2 z_2} \otimes \cdots \otimes \sigma_{x_n z_n}$  and  $g' = \alpha' \sigma_{x'_1 z'_1} \otimes \sigma_{x'_2 z'_2} \otimes \cdots \otimes \sigma_{x'_n z'_n}$ . We get  $B(g) = (x_1, \dots, x_n, z_1, \dots, z_n)$ ,  $B(g') = (x'_1, \dots, x'_n, z'_1, \dots, z'_n)$  and  $B(g)\Lambda B(g')^T = x_1 z'_1 \oplus \cdots \oplus x_n z'_n \oplus z_1 x'_1 \oplus \cdots \oplus z_n x'_n$ . By Equation (4.1) we observe that  $\sigma_{x_i z_i}$  commutes with  $\sigma_{x'_i z'_i}$  if and only if they are identical or one of them is  $I$ .  $\sigma_{x_i z_i}$  and  $\sigma_{x'_i z'_i}$  are identical if  $x_i = x'_i, z_i = z'_i$ . One of them is  $I$  if either  $x_i = z_i = 0$  or  $x'_i = z'_i = 0$ . We deduce that  $\sigma_{x_i z_i}$  commutes with  $\sigma_{x'_i z'_i}$  if and only if  $x_i z'_i \oplus x'_i z_i = 0$ . Hence, the symplectic inner product of  $B(g)$  and  $B(g')$  is 0 if and only if the number of commuting pairs  $(\sigma_{x_i z_i}, \sigma_{x'_i z'_i})$  is even. This happens if and only if  $g$  commutes with  $g'$ . It implies that  $B(g)\Lambda B(g')^T = 0$  if and only if  $g$  and  $g'$  commute, and  $B(g)\Lambda B(g')^T = 1$  if and only if  $g$  and  $g'$  anticommute. □

Later on, we'll see that by using the homomorphism  $B$ , we can translate actions on Pauli operators to appropriate actions on elements of  $\mathbb{F}_2^{2n}$ . This enables us to simulate them easily on classical computers. Proposition 4.2.4 also gives us a powerful tool when we study properties of Pauli operators. Two Pauli operators in  $G_n$  commute if and only if their images in  $\mathbb{F}_2^{2n}$  are orthogonal with respect to the symplectic inner product.

### 4.3 The Stabilizer Formalism

In the stabilizer formalism, a quantum system is described by following the evolution, not of its state as normal, but instead of the set of operators that

could act on the system ([8]). The set of operators needs to be chosen so that in every time step, it is sufficient to use this set to uniquely determine the state of the system (maybe up to an unimportant global phase). For our simulation purpose, we look for sets of operators which can be efficiently represented.

Firstly, we consider how a quantum state is described by a set of operators. Let us consider an example: suppose we are given the state  $|\psi\rangle = (|00\rangle + |01\rangle)/\sqrt{2}$ . It is easy to see that the following equalities

$$\begin{aligned}(Z \otimes X)|\psi\rangle &= |\psi\rangle, \\ (Z \otimes I)|\psi\rangle &= |\psi\rangle\end{aligned}\tag{4.3}$$

hold since  $Z|0\rangle = |0\rangle, X|0\rangle = |1\rangle, X|1\rangle = |0\rangle$ . Furthermore, if we consider  $|\phi\rangle = c_{00}|00\rangle + c_{01}|01\rangle + c_{10}|10\rangle + c_{11}|11\rangle$  as a general 2-qubit state and look for  $|\phi\rangle$  satisfying both equations in (4.3), we must have

$$\begin{aligned}|\phi\rangle &= (Z \otimes X)|\psi\rangle = c_{00}|01\rangle + c_{01}|00\rangle - c_{10}|11\rangle - c_{11}|10\rangle, \\ |\phi\rangle &= (Z \otimes I)|\psi\rangle = c_{00}|00\rangle + c_{01}|01\rangle - c_{10}|10\rangle - c_{11}|11\rangle,\end{aligned}$$

which imply that

$$\begin{aligned}c_{00} &= c_{01}, c_{10} = -c_{11}, \\ c_{10} &= -c_{10}, c_{11} = -c_{11}.\end{aligned}$$

We get  $c_{00} = c_{01}$  and  $c_{10} = c_{11} = 0$ . It means that  $|\phi\rangle = c_{00}(|00\rangle + |01\rangle)$ , which differs from  $|\psi\rangle$  only by a global phase. The 2-qubit states which satisfy both equalities in (4.3) are called stabilized states of  $\{ZX, ZI\}$ . We notice that, if we ignore global phases, the state  $|\psi\rangle$  is the only stabilized state of  $\{Z \otimes X, Z \otimes I\}$ . In this sense, we say that the state  $|\psi\rangle$  is represented by  $\{Z \otimes X, Z \otimes I\}$ .

We now consider the general stabilizer formalism in which quantum states, such as  $|\psi\rangle$  above, are represented by sets of Pauli operators.

**Definition 4.3.1.** *Given a quantum state  $|\psi\rangle$  and a Pauli operator  $U$ . We say  $U$  stabilizes  $|\psi\rangle$  if  $|\psi\rangle$  is an eigenvector of  $U$  with eigenvalue  $+1$ . If so, we also say that  $|\psi\rangle$  is a stabilized state of  $U$ .*

We notice that given a quantum state  $|\psi\rangle$ , if  $U$  and  $V$  are its stabilizers, then so are  $U^{-1}, UV$ . Therefore, the set of operators which stabilize a quantum state forms a multiplicative group of matrices. Also, if  $|\phi\rangle$  and  $|\phi'\rangle$  are stabilized states of a unitary operator  $W$ , then so is every linear combination of them. The set of stabilized states of a unitary operator, hence, forms a vector space.

**Definition 4.3.2.** *Given a subgroup  $S$  of  $G_n$ . Suppose  $V_S$  is the set of  $n$ -qubit states which are stabilized by all elements of  $S$ . Then  $V_S$  is called the stabilized vector space of  $S$ , and  $S$  is called the stabilizer of the space  $V_S$ .*

If we have a subgroup  $S$  such that its stabilized vector space  $V_S$  is non-trivial, then the elements of  $V_S$  can be represented by the group  $S$ . This representation might not be efficient since in many cases, the size of  $S$  is exponentially big. Theorem 4.3.4, however, tells us that a group can be faithfully described by using a set of at most  $\log_2(|G|)$  generators.

**Definition 4.3.3.** *A set of elements  $g_1, \dots, g_\ell$  in a group  $G$  is said to generate the group  $G$  if every element of  $G$  can be written as a product of elements from  $g_1, \dots, g_\ell$ . If  $G$  is generated by  $g_1, \dots, g_\ell$ , we write  $G = \langle g_1, \dots, g_\ell \rangle$ .*

**Theorem 4.3.4.** (Nielsen and Chuang [15]) A group  $G$  of size  $|G|$  has a set of at most  $\log_2(|G|)$  generators.

*Proof.* Suppose that  $g_1, \dots, g_\ell$  is a set of elements in  $G$ . Denote by  $\langle g_1, \dots, g_\ell \rangle$  the subgroup spanned by  $g_1, \dots, g_\ell$  and suppose, furthermore, that the subgroup is not the entire group  $G$ . Let us consider an element  $u \notin \langle g_1, \dots, g_\ell \rangle$ . For each element  $g$  in the subgroup,  $ug$  must lie outside the subgroup since, if otherwise, then  $u = (ug)g^{-1}$  would be an element of the subgroup. For ever two distinct elements  $g, g'$  in the subgroup,  $ug$  and  $ug'$  must also be different. Therefore, the size of  $\langle g_1, \dots, g_\ell, u \rangle$  is at least twice as big as the size of  $\langle g_1, \dots, g_\ell \rangle$ , from which we conclude that  $G$  must have a set of generators containing at most  $\log_2(|G|)$  elements.  $\square$

We are interested in quantum states that can be described by a subgroup of Pauli operators. It would mean that the stabilized vector space of the subgroup is of dimension 1. We do not yet have all the tools to figure out in which conditions, a Pauli subgroup will give us a 1-dimensional stabilized vector space. We will answer this question at the end of this section. It is easy to show that if  $V_S$  is non-trivial, i.e.  $V_S$  is of dimension bigger than 0, then  $S$  is Abelian and  $-I \notin S$ . If, otherwise,  $S$  is not Abelian, then there exist  $g, g'$  anti-commuting in  $S$ . For any vector  $x \in V_S$ ,  $x = gg'x = -g'gx = -x$ , which implies that  $x = 0$  and thus  $V_S$  is trivial. If  $-I$  is an element of  $S$ , then for any  $x \in V_S$ ,  $x = -Ix = -x$ . This would imply that  $V_S$  is trivial.

Suppose we are given a subgroup  $S$  generated by  $l$  commuting elements  $g_1, \dots, g_\ell$ , which does not contain  $-I$ . The generating set would also contain redundant elements if  $g_1, \dots, g_\ell$  are dependent. We, hence, want to have a way of checking whether they are independent or not. To this end, we consider the so-called check matrix of a generating set.

**Definition 4.3.5.** Given a generating set  $g_1, \dots, g_\ell$  of a subgroup of  $G_n$ . The check matrix of the set is the  $l \times 2n$  matrix whose rows are binary vectors  $B(g_1), \dots, B(g_\ell)$ .

For example, let us consider the generating set consisting of two Pauli operators  $Z \otimes X, Z \otimes I$ . The check matrix of this set is given in (4.4)

$$\begin{bmatrix} 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}. \quad (4.4)$$

Theorem 4.3.6 gives us a way of checking if a generating set is independent or not.

**Theorem 4.3.6.** (Nielsen and Chuang [15]) Given a generating set  $g_1, \dots, g_\ell$  of  $G_n$  such that  $-I$  is not an element of  $\langle g_1, \dots, g_\ell \rangle$ . The generators  $g_1$  through  $g_\ell$  are independent if and only if the corresponding check matrix is of rank  $l$ .

*Proof.* The check matrix of  $g_1, \dots, g_\ell$  is of rank smaller than  $l$  if and only if its rows are linearly dependent, i.e. there exist  $a_1, \dots, a_\ell$  such that  $\sum_i a_i B(g_i) = 0$ , and  $a_j \neq 0$  for some  $j$ . It can be shown that  $B$  is in fact a surjective homomorphism, and if we ignore global phases in  $G_n$ ,  $B$  would be an isomorphism. Hence,  $\sum_i a_i B(g_i)$  is equal to 0 if and only if  $\prod_i g_i^{a_i} = I$  up to a global phase. Since  $-I \notin S$  and  $(\pm iI)^2 = -I$ ,  $\pm iI$  is not in  $S$  neither. Therefore,  $\prod_i g_i^{a_i} = I$ , which implies that  $g_j = g_j^{-1} = \prod_{i \neq j} g_i^{a_i}$  and thus  $g_1, \dots, g_\ell$  are dependent.  $\square$

We notice that determining the rank of a  $p \times q$  binary matrix can be done efficiently in  $O(\min\{p, q\}^3)$  steps.

We now turn to the main question in this section. Given a subgroup  $S = \langle g_1, \dots, g_\ell \rangle$  of  $G_n$ , where all  $g_i$  are independent and  $-I \notin S$ , what is the dimension of  $V_S$ . We need to gather one more result before giving the solution to the question.

**Proposition 4.3.7.** *(Nielsen and Chuang [15]) Let  $S = \langle g_1, \dots, g_\ell \rangle$  be generated by  $l$  independent generators and satisfy  $-I \notin S$ . Fix a subset  $E$  of  $\{1, \dots, l\}$ . There exists  $g \in G_n$  such that it anti-commutes with  $g_i$  for all  $i \in E$  and commutes with all  $g_j$  for  $j \notin E$ .*

*Proof.* Let  $G$  be the check matrix of  $g_1, \dots, g_\ell$ . Its rows are linearly independent by Theorem 4.3.6, and thus there exists a  $2n$ -dimensional vector  $y$  such that  $Gy = e_E$ , where  $e_E$  is the  $\ell$ -dimensional vector with 1s in the  $i^{\text{th}}$  position for all  $i \in E$  and 0s elsewhere. Denote by  $x$  the vector  $\Lambda^{-1}y$ , so  $G\Lambda x = e_E$ . Let  $g$  be a Pauli operator such that  $B(g) = x^T$ . Since  $G\Lambda x = e_E$ ,  $B(g_i)\Lambda B(g)^T = 1$  for all  $i \in E$  and  $B(g_j)\Lambda B(g)^T = 0$  for all  $j \notin E$ . Therefore, by Proposition 4.2.4,  $g$  anti-commutes with  $g_i$  for all  $i \in E$  and commutes with all  $g_j$  for  $j \notin E$ .  $\square$

We now have all the tools to answer the main question in this section.

**Theorem 4.3.8.** *(Nielsen and Chuang [15]) Let  $S = \langle g_1, \dots, g_{n-k} \rangle$  be generated by  $n-k$  independent and commuting elements from  $G_n$  such that  $-I \notin S$ . Then the stabilized vector space  $V_S$  of  $S$  is of dimension  $2^k$ .*

*Proof.* Let  $x = (x_1, \dots, x_{n-k})$  be a vector of  $\mathbb{F}_2^{n-k}$ . Define

$$P_S^x = \frac{\prod_{j=1}^{n-k} (I + (-1)^{x_j} g_j)}{2^{n-k}}.$$

We notice that since  $g_i, g_j$  commute, so do  $I + (-1)^{x_i} g_i$  and  $I + (-1)^{x_j} g_j$ . We know furthermore that  $(I + g_i)/2$  is the projector onto the  $+1$  eigenspace of  $g_i$ , hence  $P_S^{(0, \dots, 0)}$  is the projector onto all  $+1$  eigenspaces of  $g_1, \dots, g_{n-k}$ . In other words,  $P_S^{(0, \dots, 0)}$  is the projector onto  $V_S$ . The dimension of  $P_S^{(0, \dots, 0)}$ , hence, is the dimension of  $V_S$ .

By Proposition 4.3.7, for each  $x$  there exists  $g_x$  such that it anti-commutes with  $g_i$  for all  $i$  such that  $x_i = 1$  and commutes with the other  $g_j$ . We, hence, get  $g_x(I + g_i) = (I + (-1)^{x_i} g_i)g_x$  for all  $i \in \{1, \dots, n-k\}$  and thus  $g_x P_S^{(0, \dots, 0)} g_x^\dagger = P_S^x$ . The dimension of  $P_S^x$ , therefore, is the same as the dimension of  $V_S$ .

We notice that for two different binary bits  $a, b$ , we always have  $(I + (-1)^a g_i)(I + (-1)^b g_i) = I + (-1)^{a+b} I + ((-1)^a + (-1)^b)g_i = 0$ . For two differences  $x, y \in \mathbb{F}_2^{n-k}$ , there exists  $i$  such that  $x_i \neq y_i$ . Since  $I + (-1)^{x_i} g_i, I + (-1)^{y_i} g_i$  commute for all  $i$  and  $j$ , the matrix product  $P_S^x \cdot P_S^y$  contains  $(I + (-1)^{x_i} g_i)(I + (-1)^{y_i} g_i)$  as a factor. Hence we get to have  $P_S^x \cdot P_S^y = 0$ , which implies that  $P_S^x$  are the projectors on orthogonal subspaces.

The proof is concluded by the following observation,

$$\begin{aligned}
 \sum_{x \in \mathbb{F}_2^{n-k}} P_S^x &= \sum_{x^{(n-k-1)} \in \mathbb{F}_2^{n-k}} \left( P_S^{x^{(n-k-1)}0} + P_S^{x^{(n-k-1)}1} \right) \\
 &= \sum_{x^{(n-k-1)} \in \mathbb{F}_2^{n-k}} P_S^{x^{(n-k-1)}} \\
 &\dots \\
 &= \sum_{x^{(0)} \in \mathbb{F}_2} P_S^{x^{(0)}} = I
 \end{aligned} \tag{4.5}$$

We know that the identity matrix  $I$  is a projector onto a  $2^n$ -dimensional space, while the left hand side of Equation (4.5) is a sum over  $2^{n-k}$  orthogonal projectors of the same dimension as the dimension of  $V_S$ . Hence the dimension of  $V_S$  must be  $2^k$ . This concludes the proof.  $\square$

Theorem 4.3.8 tells us that the dimension of the stabilized space  $V_S$  depends only on the number of generating elements of  $S$ . For our simulation purpose, an important implication of Theorem 4.3.8 is the following: if  $S$  is a subgroup of  $G_n$  and generated by  $n$  independent and commuting elements such that  $-I \notin S$ , then the stabilized space  $V_S$  is a 1-dimensional space. This space contains only one pure state  $|\psi\rangle$  if we ignore an unimportant global phase.  $|\psi\rangle$  is called a stabilizer state on  $n$  qubits and can be faithfully represented by the stabilizer  $S$ .

As an example, we consider again the stabilizer  $S = \langle Z \otimes X, Z \otimes I \rangle$  in  $G_2$ . It is easy to check that  $Z \otimes X, Z \otimes I$  are commuting since  $(Z \otimes X) \cdot (Z \otimes I) = (Z \cdot Z) \otimes (X \cdot I) = (Z \otimes I) \cdot (Z \otimes X)$ . By Equation (4.3),  $(|00\rangle + |01\rangle)/\sqrt{2}$  is a stabilized state of  $S$ . Furthermore, Theorem 4.3.8 implies that  $S$  stabilizes a vector space of dimension 1. We, hence, conclude that  $(|00\rangle + |01\rangle)/\sqrt{2}$  is the stabilizer state of  $S$ .

We notice that highly entangled states could be also stabilizer states. For example, let us consider the maximally entangled 2-qubit state  $|\psi\rangle = (|00\rangle + |11\rangle)/\sqrt{2}$ . It is easily verified that  $X \otimes X, Z \otimes Z$  stabilize  $|\psi\rangle$ , and they are commuting. Hence,  $\langle X \otimes X, Z \otimes Z \rangle$  is the stabilizer of the stabilizer state  $|\psi\rangle$ .

In the next section we will study how to use the stabilizer formalism to simulate quantum circuits with Clifford gates<sup>1</sup>. In general, we can use the stabilizer formalism to simulate any quantum circuit. However, if a circuit contains only Clifford gates, Gottesman [8] showed that the simulation can be done efficiently.

## 4.4 Unitary Gates in the Stabilizer Formalism

In this section we consider an  $n$ -qubit stabilizer state  $|\psi\rangle$  of the subgroup  $S = \langle g_1, \dots, g_n \rangle$  of  $G_n$ , where  $g_1, \dots, g_n$  are independent and commuting, and  $-I \notin S$ . Results from the previous section show that  $n$  elements  $g_1, \dots, g_n$  uniquely determine the state  $|\psi\rangle$ . Suppose that we want to apply an  $n$ -qubit unitary gate  $U$  to  $|\psi\rangle$ . We are interested in the questions: Is the resulting state  $|\psi'\rangle = U|\psi\rangle$  a stabilizer state? If so, what is its stabilizer?

<sup>1</sup>Clifford gates will be introduced in the next section.

For any element  $g$  of  $S$ ,

$$|\psi'\rangle = U|\psi\rangle = Ug|\psi\rangle = UgU^\dagger U|\psi\rangle = UgU^\dagger |\psi'\rangle, \quad (4.6)$$

and thus the state  $|\psi'\rangle$  is stabilized by  $UgU^\dagger$ . This suggests considering the set  $S' = USU^\dagger := \{UgU^\dagger : g \in S\}$ . Theorem 4.4.1 lists some properties of this set.

**Theorem 4.4.1.** *The following statements are true:*

1.  $Ug_1U^\dagger, \dots, Ug_nU^\dagger$  are independent.
2.  $Ug_1U^\dagger, \dots, Ug_nU^\dagger$  are commuting.
3.  $Ug_1U^\dagger, \dots, Ug_nU^\dagger$  generate  $S'$ .

*Proof.* 1. It is easy to check that the mapping

$$\begin{aligned} H_U : U(2^n) &\rightarrow U(2^n) \\ g &\mapsto UgU^\dagger, \end{aligned}$$

is an isomorphism. Hence the independence of  $g_1, \dots, g_n$  implies the independence of  $Ug_1U^\dagger, \dots, Ug_nU^\dagger$ .

2. For different  $i, j$ , we have  $(Ug_iU^\dagger)(Ug_jU^\dagger) = Ug_iU^\dagger Ug_jU^\dagger = Ug_i g_j U^\dagger$  and also  $(Ug_jU^\dagger)(Ug_iU^\dagger) = Ug_j g_i U^\dagger$ . Since  $g_i$  and  $g_j$  commute, so do  $Ug_iU^\dagger$  and  $Ug_jU^\dagger$ .
3. It follows from the fact that  $g_1, \dots, g_n$  generate  $S$ . □

If we assume that all  $Ug_1U^\dagger, \dots, Ug_nU^\dagger$  are in  $G_n$ , then we can conclude that  $|\psi'\rangle$  is a stabilizer state and  $S' = \langle Ug_1U^\dagger, \dots, Ug_nU^\dagger \rangle$  is its stabilizer. The assumption would mean that  $U$  takes elements of  $S$  to elements of  $G_n$  under conjugation. If we consider all subsets  $S$  in  $G_n$ , then  $U$  leaves the group  $G_n$  fixed under conjugation. The set of matrices which leave the group  $G_n$  fixed under conjugation is called the normalizer of  $G_n$ , denoted by  $N(G_n)$ .  $N(G_n)$  is also called the Clifford group, denoted by  $\mathcal{C}_n$ , for its relationship to the usual Clifford groups and Clifford algebras ([8]). We will use  $N(G_n)$  and  $\mathcal{C}_n$  interchangeably. We, hence, have shown the following theorem:

**Theorem 4.4.2.** *Let  $|\psi\rangle$  be a stabilizer state stabilized by  $n$  independent, commuting elements  $g_1, \dots, g_n$  from  $G_n$ ; and  $U$  be a Clifford operator in  $\mathcal{C}_n$ . Applying  $U$  to  $|\psi\rangle$  will give us a new state  $|\psi'\rangle$  satisfying the following properties:*

1.  $|\psi'\rangle$  is an  $n$ -qubit stabilizer state.
2.  $S' = \langle Ug_1U^\dagger, \dots, Ug_nU^\dagger \rangle$  is the stabilizer of  $|\psi'\rangle$ .

Gottesman [8] analyzed the structure of the Clifford group  $\mathcal{C}_n$ . It turns out that the group can be analytically described by a set of three generating operators.

**Theorem 4.4.3.** *(Gottesman [8]) The Clifford group  $\mathcal{C}_n$  is generated by three gates: Hadamard, Phase and CNOT, in the sense that every operator in  $\mathcal{C}_n$  can be decomposed as a (matrix multiplication/tensor) product of gates from these three kinds of gates.*

For that reason, these three gates are called Clifford gates. For our simulation purpose, we want to analyze in details how to represent a stabilizer state in the most efficient way and how to update the representation when performing a Clifford gate to the state. The next section will study these issues.

## 4.5 Clifford Gates in the Stabilizer Formalism

In all our later discussion of the stabilizer formalism, we will use the convention that an  $n$ -qubit stabilizer  $S$  is described in terms of independent commuting generators  $g_1, \dots, g_n$  such that  $-I \notin S$ . We want to represent  $g_1, \dots, g_n$  in an efficient way. Let us consider, for instance, the first generator  $g_1$ . Suppose that  $g_1 = \alpha_1 \sigma_{x_{11}z_{11}} \otimes \dots \otimes \sigma_{x_{1n}z_{1n}}$ . The mapping  $B$  allows us to represent  $\sigma_{x_{11}z_{11}} \otimes \dots \otimes \sigma_{x_{1n}z_{1n}}$  using its binary representation. For the scalar coefficient  $\alpha_1$ , we notice that it can be either 1 or  $-1$ . The reason is: suppose that  $\alpha_1 = \pm i$ , we would get  $g_1^2 = -I$ . Hence  $-I \in S$  which contradicts the assumption of  $S$ . For our simulation purpose, if  $\alpha_1 = -1$ , we set it to 0. We, therefore, can represent  $S$  using its check matrix, plus an additional binary column representing scalar coefficients of  $g_1, \dots, g_n$ . All entries of the matrix are binary. The matrix looks like (4.7)

$$\begin{pmatrix} x_{11} & \dots & x_{1n} & z_{11} & \dots & z_{1n} & r_1 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots \\ x_{n1} & \dots & x_{nn} & z_{n1} & \dots & z_{nn} & r_n \end{pmatrix}, \quad (4.7)$$

where  $r_i$  represents the scalar coefficient  $\alpha_i$ . We notice that this representation is extremely efficient since it contains all binary data. To represent an  $n$ -qubit stabilizer state, we need only  $n(2n + 1)$  bits.

We now turn to the question: how to update the representation of the state  $|\psi\rangle$  after applying a Clifford gate  $U$ . Since the Clifford group is generated by the Hadamard, Phase and CNOT gates, it is sufficient to consider only these three Clifford gates.

### 4.5.1 Applying the Hadamard Gate

It is easy to check the following equalities:

$$HXH^\dagger = Z; HZH^\dagger = X; HYH^\dagger = -Y. \quad (4.8)$$

As a consequence we have Figure 4.1, listing the transformations of the Pauli matrices under conjugation by the Hadamard gate. The transformations in the binary form obey the following rules:  $(x, z, r) \mapsto (x', z', r')$ , where  $x' = z, z' = x, r' = r \oplus (xz)$ .

	Pauli basis				Binary Representation			
Input	I	X	Z	Y	001	011	101	111
Ouput	I	Z	X	-Y	001	101	011	110

Figure 4.1: Transformations of the Pauli matrices under conjugation by the Hadamard gate. Binary representations of inputs and outputs are also considered.

We now consider how the generators  $g_1, \dots, g_n$  are changed after performing  $H$  on qubit  $a$ . The generator  $g_i = r_i \sigma_{x_{i1}z_{i1}} \otimes \dots \otimes \sigma_{x_{in}z_{in}}$  will be transformed

to  $g'_i = r'_i \sigma_{x_{i1}z_{i1}} \otimes \cdots \otimes \sigma_{x'_{ia}z'_{ia}} \otimes \cdots \otimes \sigma_{x_{in}z_{in}}$ , where  $x'_{ia} = z_{ia}$ ,  $z'_{ia} = x_{ia}$  and  $r'_i = r_i \oplus (x_{ia}z_{ia})$ . We, therefore, have Algorithm 4.1 to update the state after applying the Hadamard gate on qubit  $a$ :

---

**Algorithm 4.1** Update the state  $|\psi\rangle$  after applying the Hadamard gate to qubit  $a$ .

---

```

for  $i = 1$  to  $n$  do
     $r_i \leftarrow r_i \oplus (x_{ia}z_{ia});$ 
     $temp \leftarrow x_{ia};$ 
     $x_{ia} \leftarrow z_{ia};$ 
     $z_{ia} \leftarrow temp;$ 
end for

```

---

### 4.5.2 Applying the Phase Gate

It is easy to check the following equalities

$$PXP^\dagger = Y; PZP^\dagger = Z; PYP^\dagger = -X. \quad (4.9)$$

As a consequence we have Figure 4.2, listing the transformations of the Pauli matrices under conjugation by the Phase gate. The transformations in the binary form obey the following rules:  $(x, z, r) \mapsto (x', z', r')$ , where  $z' = z$ ,  $x' = x \oplus z$  and  $r' = r \oplus (xz)$ .

	Pauli basis				Binary Representation			
Input	I	X	Z	Y	001	011	101	111
Ouput	I	Y	Z	-X	001	111	101	010

Figure 4.2: Transformations of the Pauli matrices under conjugation by the Phase gate. Binary representations of inputs and outputs are also considered.

We now consider how the generators  $g_1, \dots, g_n$  are changed after performing  $P$  on qubit  $a$ . The generator  $g_i = r_i \sigma_{x_{i1}z_{i1}} \otimes \cdots \otimes \sigma_{x_{in}z_{in}}$  will be transformed to  $g'_i = r'_i \sigma_{x_{i1}z_{i1}} \otimes \cdots \otimes \sigma_{x'_{ia}z'_{ia}} \otimes \cdots \otimes \sigma_{x_{in}z_{in}}$ , where  $x'_{ia} = x_{ia} \oplus z_{ia}$ ,  $z'_{ia} = z_{ia}$  and  $r'_i = r_i \oplus (x_{ia}z_{ia})$ . We, therefore, have Algorithm 4.2 to update the state after applying the Phase gate on qubit  $a$ :

---

**Algorithm 4.2** Update the state  $|\psi\rangle$  after applying the Phase gate to qubit  $a$ .

---

```

for  $i = 1$  to  $n$  do
     $r_i \leftarrow r_i \oplus (x_{ia}z_{ia});$ 
     $x_{ia} \leftarrow x_{ia} \oplus z_{ia};$ 
end for

```

---

### 4.5.3 Applying the CNOT Gate

Figure 4.3 lists the transformations of the 2-qubit Pauli operators under conjugation by the CNOT gate, where the first qubit is the control qubit and the second qubit is the target qubit. The transformations in the binary form

Pauli Basis		Binary Form	
Input	Output	Input ( $x_1x_2z_1z_2r$ )	Output ( $x'_1x'_2z'_1z'_2r'$ )
II	II	00001	00001
IX	IX	01001	01001
IY	ZY	01011	01111
IZ	ZZ	00011	00111
XI	XX	10001	11001
XX	XI	11001	10001
XY	YZ	11011	10111
XZ	-YY	10011	11110
YI	YX	10101	11101
YX	YI	11101	10101
YY	-XZ	11111	10010
YZ	XY	10111	11011
ZI	ZI	00101	00101
ZX	ZX	01101	01101
ZY	IY	01111	01011
ZZ	IZ	00111	00011

Figure 4.3: Transformations of the Pauli matrices under conjugation by the CNOT gate. We here denote by  $AB$  the tensor between  $A$  and  $B$ . Binary representations of inputs and outputs are also considered.

obey the following rules ([1]):  $(x_1, x_2, z_1, z_2, r) \mapsto (x'_1, x'_2, z'_1, z'_2, r')$ , where  $x'_1 = x_1, x'_2 = x_1 \oplus x_2, z'_1 = z_1 \oplus z_2, z'_2 = z_2, r' = r \oplus x_1z_2(x_2 \oplus z_1 \oplus 1)$ .

We now consider how the generators  $g_1, \dots, g_n$  are changed after performing  $CNOT$  on control qubit  $a$  and target qubit  $b$ . The generator  $g_i = r_i \sigma_{x_{i1}z_{i1}} \otimes \dots \otimes \sigma_{x_{in}z_{in}}$  will be transformed to  $g'_i = r'_i \sigma_{x_{i1}z_{i1}} \otimes \dots \otimes \sigma_{x'_{ia}z'_{ia}} \otimes \dots \otimes \sigma_{x_{in}z_{in}}$ , where  $x'_{ia} = x_{ia}, x'_{ib} = x_{ia} \oplus x_{ib}, z'_{ia} = z_{ia} \oplus z_{ib}, z'_{ib} = z_{ib}, r'_i = r_i \oplus x_{ia}z_{ib}(x_{ib} \oplus z_{ia} \oplus 1)$ . We, therefore, have Algorithm 4.3 to update the state after applying the CNOT gate.

---

**Algorithm 4.3** Update the state  $|\psi\rangle$  after applying the CNOT gate to control qubit  $a$  and target qubit  $b$ .

---

```

for  $i = 1$  to  $n$  do
     $r_i \leftarrow r_i \oplus x_{ia}z_{ib}(x_{ib} \oplus z_{ia} \oplus 1)$ ;
     $x_{ib} \leftarrow x_{ia} \oplus x_{ib}$ ;
     $z_{ia} \leftarrow z_{ia} \oplus z_{ib}$ ;
end for
    
```

---

## 4.6 Measurements in the Stabilizer Formalism

Theorem 4.2.3 tells us that the matrices in  $G_n$  are either Hermitian or anti-Hermitian. The Hermitian matrices can be regarded as projective measurements. For example, the Pauli operator  $Z = |0\rangle\langle 0| + (-1)|1\rangle\langle 1|$  can be thought of as a measurement of one qubit in the computational basis with two outcome

possibilities, 1 and  $-1$ . In this section we will consider how these measurements are performed in the stabilizer formalism.

Let us consider a stabilizer state  $|\psi\rangle$  with stabilizer  $\langle g_1, \dots, g_n \rangle$  and a measurement  $g \in G_n$ . Since  $g$  is Hermitian,  $g^2 = I$ . Therefore  $g$  can only have two eigenvalues 1 and  $-1$ . Let  $P_+ = (I + g)/2$ . For every  $|\phi\rangle$  from  $(\mathbb{C}^2)^{\otimes n}$ , we have  $g(P_+|\phi\rangle) = g(|\phi\rangle + g|\phi\rangle)/2 = (g|\phi\rangle + |\phi\rangle)/2 = P_+|\phi\rangle$ . It follows that  $P_+$  is the projector onto the eigenspace of eigenvalue 1. It is also easy to check that  $P_- = (I - g)/2$  is the projector onto the eigenspace of eigenvalue  $-1$ . The measurement probabilities, hence, are given by

$$\begin{aligned} p(+1) &= \text{tr} \left( \frac{I+g}{2} |\psi\rangle\langle\psi| \right), \\ p(-1) &= \text{tr} \left( \frac{I-g}{2} |\psi\rangle\langle\psi| \right). \end{aligned}$$

We distinguish two cases based on the relationship between  $g$  and the generators  $g_1, \dots, g_n$ .

**Case 1:**  $g$  commutes with all the generators. Since  $g_i g |\psi\rangle = g g_i |\psi\rangle = g |\psi\rangle$  for each  $i = 1, \dots, n$ ,  $g|\psi\rangle$  is in  $V_S$ . The 1-dimensional space  $V_S$  also contains  $|\psi\rangle$ . Thus  $g|\psi\rangle$  must be a multiple of  $|\psi\rangle$ . Therefore,  $g|\psi\rangle$  equals either  $|\psi\rangle$  or  $-|\psi\rangle$  since  $g^2 = I$ . If  $g|\psi\rangle = |\psi\rangle$ , then  $p(+1) = \text{tr}(|\psi\rangle\langle\psi|) = 1$  and thus the measurement gets result 1 with probability one with the post-measurement state  $P_+|\psi\rangle = |\psi\rangle$ . The stabilizer, hence, does not change. If  $g|\psi\rangle = -|\psi\rangle$ , we get a similar conclusion: the measurement gets result  $-1$  with probability one with the post-measurement state  $-|\psi\rangle$ . The stabilizer does not change neither.

**Case 2:**  $g$  anti-commutes with one or more of the generators. Suppose  $g$  anti-commutes with  $g_1$ . If  $g$  anti-commutes with any other generator, say  $g_j$  where  $j > 1$ , then we replace  $g_j$  by  $g'_j = g_1 g_j$ . Since  $g g'_j = g g_1 g_j = -g_1 g g_j = g_1 g_j g = g'_j g$ ,  $g$  commutes with  $g'_j$ . Using this trick, we can assume, without loss of generality, that  $g$  commutes with each of  $g_2, \dots, g_n$ .

Let us compute the probability of observing the outcome 1:

$$\begin{aligned} p(+1) &= \text{tr} \left( \frac{I+g}{2} g_1 |\psi\rangle\langle\psi| \right) = \text{tr} \left( \frac{g_1 - g_1 g}{2} |\psi\rangle\langle\psi| \right) \\ &= \text{tr} \left( g_1 \frac{I-g}{2} |\psi\rangle\langle\psi| \right) = \text{tr} \left( \frac{I-g}{2} |\psi\rangle\langle\psi| g_1 \right) \\ &= \text{tr} \left( \frac{I-g}{2} |\psi\rangle\langle\psi| \right) = p(-1), \end{aligned}$$

where the last equality holds since  $g_1 = g_1^\dagger$  and  $g_1 |\psi\rangle = |\psi\rangle$ . We deduce that  $p(+1) = p(-1) = 1/2$ . If the result 1 occurs, then the post-measurement state is  $|\psi_+\rangle = \sqrt{2} P_+ |\psi\rangle$ . Since  $|\psi_+\rangle$  is an  $+1$  eigenvalue of  $g$ , we have  $g|\psi_+\rangle = |\psi_+\rangle$ . For each  $i = 2, \dots, n$ ,  $g_i |\psi_+\rangle = (g_i + g_i g) |\psi\rangle / \sqrt{2} = (I + g) g_i |\psi\rangle / \sqrt{2} = |\psi_+\rangle$  and thus  $\langle g, g_2, \dots, g_n \rangle$  is the stabilizer of the state  $|\psi_+\rangle$ . Reasoning in the same way shows that if the result  $-1$  occurs, then  $\langle -g, g_2, \dots, g_n \rangle$  is the stabilizer of the post-measurement state  $|\psi_-\rangle = \sqrt{2} P_- |\psi\rangle$ .

For our simulation purpose, we consider the binary representation (4.7) of the stabilizer state  $|\psi\rangle$ . We also distinguish the two cases. We notice that checking whether  $g$  commutes with a generator  $g_i$  can be done in  $O(n)$  steps by comparing the symplectic inner product  $B(g)\Lambda B(g_i)^T$  to 0.

**Case 1:**  $g$  commutes with all the generators. We have showed that the generators did not change. The problem is how to determine if the measurement outcome is 1 or  $-1$ . The outcome is 1 if and only if  $g|\psi\rangle = |\psi\rangle$  meaning that  $g$  lies inside the stabilizer  $\langle g_1, \dots, g_n \rangle$ . Suppose that  $g = \alpha \sigma_{x_1 z_1} \dots \sigma_{x_n z_n}$ .  $g$  lies inside the stabilizer  $\langle g_1, \dots, g_n \rangle$  if and only if the vector  $(x_1, x_2, \dots, x_n, z_1, z_2, \dots, z_n, r)$ , where  $r = 1$  if  $\alpha = 1$  and  $r = 0$  if  $\alpha = -1$ , is linearly dependent on  $n$  row vectors of the matrix (4.7). Coppersmith et al. [7] showed that checking this dependency condition theoretically requires  $O(n^{2 \cdot 376})$  steps. However, it normally requires  $O(n^3)$  steps in practice.

**Case 2:**  $g$  anti-commutes with  $g_1$  and commutes with all other generators. We only need to replace  $g_1$  by  $g$  (or  $-g$ ) if the measurement outcome is 1 (or  $-1$ ). This requires  $O(n)$  steps.

We end this section with the Gottesman-Knill theorem which is essentially the summary of the results in this section and in the previous one.

**Theorem 4.6.1.** (Gottesman [8]) *Any quantum computer performing only: Clifford group gates, measurements of Pauli group operators, and Clifford operations conditioned on classical bits, which may be the results of earlier measurements, can be simulated efficiently on a probabilistic classical computer.*

A quantum circuit consisting gates of the above kinds is called a Clifford circuit. The theorem says that the simulation complexity of a size- $T$  Clifford circuit is  $T \cdot O(n^3)$ . The only portions of the simulation requiring  $O(n^3)$  steps are updates of measurements which commute with all the generators. These measurements would not take such time if we do not use the measurement outcome. Aaronson and Gottesman [1] improved this to  $O(n^2)$ . We will study this improvement in the next section.

## 4.7 Improved Simulation of Clifford Circuits

Aaronson and Gottesman [1] gave a significant improvement of simulating Clifford circuits by using a more informative binary representation of stabilizer states. They considered not only the stabilizer generators but also so-called destabilizer generators, which are Pauli operators that together with the stabilizer generators form a generating set of  $G_n$ . Using binary representations of these generators, we end up with a  $2n \times (2n + 1)$  binary matrix as in (4.10),

$$\begin{pmatrix} x_{11} & \dots & x_{1n} & z_{11} & \dots & z_{1n} & r_1 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots \\ x_{n1} & \dots & x_{nn} & z_{n1} & \dots & z_{nn} & r_n \\ x_{(n+1)1} & \dots & x_{(n+1)n} & z_{(n+1)1} & \dots & z_{(n+1)n} & r_{n+1} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots \\ x_{(2n)1} & \dots & x_{(2n)n} & z_{(2n)1} & \dots & z_{(2n)n} & r_{2n} \end{pmatrix}, \quad (4.10)$$

where rows 1 to  $n$  represent the destabilizer generators  $g_1, \dots, g_n$ , and rows  $n+1$  to  $2n$  represent the stabilizer generators  $g_{n+1}, \dots, g_{2n}$ . For example, we know that  $|00\rangle$  is a stabilizer state with stabilizer  $\langle Z \otimes I, I \otimes Z \rangle$ . Hence  $Z \otimes I, I \otimes Z$  are the stabilizer generators. We choose  $X \otimes I, I \otimes X$  as the destabilizer generators

since  $\{Z \otimes I, I \otimes Z, X \otimes I, I \otimes X\}$  together with multiplicative factors  $\pm 1, \pm i$  generate the full group  $G_n$ . The matrix representing  $|00\rangle$  is given below

$$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix}.$$

We will generalize this example since it will be used as the standard initial state of the improved algorithm. Let us consider the  $n$ -qubit state  $|\psi_0\rangle = |00\dots 0\rangle$ . For each 1-qubit Pauli matrix  $A$ , we denote by  $A_i$  the  $n$ -qubit Pauli matrix  $I \otimes \dots \otimes A \otimes I$ , where  $A$  occurs at the  $i^{\text{th}}$  position. It is easy to check that  $\{Z_1, Z_2, \dots, Z_n\}$  are the stabilizer generators of  $|\psi_0\rangle$  and  $\{X_1, X_2, \dots, X_n\}$  are its destabilizer generators. Let  $g_1 = X_1, \dots, g_n = X_n; g_{n+1} = Z_1, g_{n+2} = Z_2, \dots, g_{2n} = Z_n$ . Theorem 4.7.1 lists some properties of these special generators.

**Theorem 4.7.1.** *The following statements are true for the above generators:*

1.  $g_1, \dots, g_n$  commute.
2. For all  $h \in \{1, \dots, n\}$ ,  $g_h$  anti-commutes with  $g_{h+n}$ .
3. For all  $i, h \in \{1, \dots, n\}$  such that  $i \neq h$ ,  $g_i$  commutes with  $g_{h+n}$ .

*Proof.* 1. It follows because  $I$  and  $X$  commute.

2. Since  $X$  and  $Z$  anti-commute, so do  $X_h$  and  $Z_h$ , i.e.  $g_h$  and  $g_{h+n}$  anti-commute.
3. Since  $I, X$  commute and  $I, Z$  commute, so do  $X_i, Z_h$  for all  $i \neq h$ , i.e.  $g_i, g_{h+n}$  commute. □

### 4.7.1 Applying the Clifford Gates

Updating the Clifford gates is done using the original updating algorithms (Algorithms 4.1, 4.2, 4.3). The only difference is we need to scan through  $2n$  rows instead of  $n$  rows.

### 4.7.2 Applying Measurements

For the sake of simplicity, Aaronson and Gottesman introduced a subroutine called *rowsum*( $h, i$ ), which sets the generator  $g_h$  equal to  $g_i \cdot g_h$ . We consider the projective measurement  $g = Z_a$  on qubit  $a$ . We still do the same as the original algorithm. To check the commutativity of  $g$  and any stabilizer  $g_{n+i}$ , it is sufficient to look at the  $i^{\text{th}}$  Pauli matrix of  $g_{n+i}$ . If it is either  $X$  or  $Y$ , then  $g_{n+i}$  anti-commutes with  $g$ . Otherwise, they are commutative. To do that checking, we check whether there exists a  $p \in \{n+1, \dots, 2n\}$  such that  $x_{pa} = 1$ .

**Case 1:** There does not exist such a  $p$ . It means that  $g$  commutes with all the stabilizer generators. We do not need to update the representation. For determining if the measurement outcome is 1 or  $-1$ , Aaronson and Gottesman used the following procedure: use an additional  $(2n+1)^{\text{th}}$  row which is set to

be identically 0. Call  $\text{rowsum}(2n+1, i+n)$  for all  $i \in \{1, \dots, n\}$  such that  $x_{ia} = 1$ . The measurement outcome will be the final result of  $r_{2n+1}$ .

**Case 2:** Such a  $p$  exists. It means that  $g$  anti-commutes with some stabilizer generators. In case  $g$  anti-commutes with more than one stabilizer generators, we can transform it to the situation where  $g$  anti-commutes with exactly only one generator as follows: let  $p$  be the smallest such that  $x_{pa} = 1$ . For all other  $p'$ , we set  $g_{p'}$  equal to  $g_p g_{p'}$ . The analysis in Section 4.6 shows that after the replacements,  $g$  anti-commutes with  $g_p$  and commutes with all other stabilizer generators. The two measurement outcomes occur with the same probability. If the result  $+1$  occurs, we replace  $g_p$  by  $g$ . If the result  $-1$  occurs, we replace  $g_p$  by  $-g$ . To update the binary representation, we do as follows: first, call  $\text{rowsum}(i, p)$  for all  $i \in \{1, \dots, 2n\}$  such that  $i \neq p$  and  $x_{ia} = 1$ . Second, for updating the destabilizer generators, we set entire row  $p-n$  equal to row  $p$ . Third, to replace  $g_p$  by either  $g$  or  $-g$ , we set row  $p$  to be all 0, except  $z_{pa} = 1$  and  $r_p$  yet to be determined. Flip a coin and decide which the measurement outcome to occur. If it is 1, set  $r_p = 1$ . Otherwise, set  $r_p = 0$ .

### 4.7.3 Correctness and Complexity of the Improved Simulation

During the course of updating the stabilizer state, some properties of the stabilizer and destabilizer generators are invariant. These are stated in Proposition 4.7.2.

**Proposition 4.7.2.** (Aaronson and Gottesman [1]) *The followings are invariants during the course of the improved simulation.*

1.  $\langle g_{n+1}, \dots, g_{2n} \rangle$  is the stabilizer of the state  $|\psi\rangle$ ; and  $g_1, \dots, g_{2n}$  generate  $G_n$ .
2.  $g_1, \dots, g_n$  commute.
3. For all  $h \in \{1, \dots, n\}$ ,  $g_h$  anti-commutes with  $g_{h+n}$ .
4. For all  $i, h \in \{1, \dots, n\}$  such that  $i \neq h$ ,  $g_i$  commutes with  $g_{h+n}$ .

The correctness of the updates of the Clifford gates follows immediately from the analysis in Section 4.5. For the measurement's update, Case 2 has no difference from the original case. We need to analyze Case 1. The measurement  $Z_a$  commutes with all the stabilizer generators. It follows that either  $Z_a$  or  $-Z_a$  lies in the stabilizer, i.e. one of them can be generated by  $g_{n+1}, \dots, g_{2n}$ . Therefore, there exist unique binary coefficients  $c_1, \dots, c_n$  such that either  $\prod_{h=1}^n g_{h+n}^{c_h} = Z_a$  or  $\prod_{h=1}^n g_{h+n}^{c_h} = -Z_a$ . By multiplying all  $g_{h+n}$  with  $c_h = 1$ , the phase of the result will tell us the outcome: a positive phase implies the outcome to be 1, a negative phase implies the outcome to be  $-1$ .

Both  $\prod_{h=1}^n g_{h+n}^{c_h} = Z_a$  and  $\prod_{h=1}^n g_{h+n}^{c_h} = -Z_a$  imply that  $\bigoplus_{h=1}^n c_h B(g_{h+n}) = B(Z_a)$ . For each  $i \in \{1, \dots, n\}$ , Proposition 4.7.2 shows that  $g_i$  anti-commutes with  $g_{i+n}$  and commutes with all  $g_{h+n}$  for  $h \in \{1, \dots, n\}$  such that  $h \neq i$ . By Proposition 4.2.4, we get  $B(g_i) \Lambda B(g_{i+n})^T = 1$  and  $B(g_i) \Lambda B(g_{h+n})^T = 0$ . Hence

$$c_i = \bigoplus_{h=1}^n \left( c_h B(g_i) \Lambda B(g_{h+n})^T \right) = B(g_i) \Lambda \left( \bigoplus_{h=1}^n c_h B(g_{h+n})^T \right) = B(g_i) \Lambda B(Z_a)^T,$$

which is 1 if and only  $g_i$  anti-commutes with  $Z_a$ , i.e.  $x_{ia} = 1$ . Therefore, whenever  $x_{ia} = 1$  for  $i \in \{1, \dots, n\}$ ,  $rowsum(2n + 1, i + n)$  needs to be called. And the final phase  $r_{2n+1}$  indeed tells us whether the outcome 1 or  $-1$  occurs.

The procedure performed in Case 1 scans through the first  $n$  rows of the representation matrix and performs a *rowsum* if necessary. A *rowsum* costs  $O(n)$  steps. The procedure, hence, costs us  $O(n^2)$  steps.

## 4.8 The Power of The Clifford Gates

We have shown that Clifford circuits can be simulated efficiently in classical computers. How big is the class of Clifford circuits in quantum computation? Nebe et al. in [14] showed that the set of Clifford gates together with any other one-qubit gate, which is not in the Clifford group, is a universal set for quantum computation. For example, the set consisting of the CNOT, Hadamard, Phase and  $\pi/8$  gates is a quantumly universal set. Buhrman et al. in [5] gave another property about the power of the Clifford gates by showing that probabilistic mixtures of Clifford gates can simulate arbitrary one-qubit gates with depolarizing noise  $\theta = (6 - 2\sqrt{2})/7 \approx 45\%$ . By depolarizing noise  $p$ , we mean if a state undergoes this noise, then with probability  $1 - p$  the state is not affected and with probability  $p$  the state is replaced with the completely mixed state. Adding one-qubit gates with depolarizing noise  $\theta$ , hence, will not change the power of the set of Clifford gates.

## 4.9 Summary

In this chapter, we considered the stabilizer formalism, first introduced by Gottesman [8]. The formalism uses subgroups of the Pauli group  $G_n$  to represent quantum states. A quantum state is called a stabilizer state if there exist  $n$  independent commuting Pauli generators of which the state is a common  $+1$  eigenvector. Updating unitary gates to the state is done by keeping track of the Pauli generators. This can be done efficiently as long as unitary gates are in the Clifford group  $\mathcal{C}_n$ , which is the normalizer of  $G_n$ . The measurement of an observable  $g$  in  $G_n$  can also be simulated in  $O(n^3)$  steps, which was improved by Aaronson and Gottesman [1] to  $O(n^2)$ . We also noticed that the representation of a stabilizer state using the binary check matrix is very efficient since all data is binary. The space for storing data required in the original algorithm and the improved algorithms was  $n(2n + 1)$  and  $(2n + 1)^2$  bits, respectively.

## Chapter 5

# The Matchgate Formalism

### 5.1 Introduction

The matchgate concept was first introduced by Valiant in [20] as a tool to simulate a class of quantum circuits. The concept, since then, has been studied by many other authors focusing on two main directions: one is about classical simulation of quantum computing, which is the main concern of the current chapter; the other direction is about holographic algorithms (cf. [21]), which are in fact inspired by the quantum computational model. Valiant constructed matchgates by using the so-called Pfaffian and Pfaffian sum concepts in graph theory. For further details of this construction, we refer to the paper [6] of Cai, Choudhary and Lu.

After the work of Valiant, Knill [12] and Terhal et al. [19] realized a remarkable connection between matchgates and Fermionic quantum computation (cf. [4]). It turned out that matchgates can be expressed using the language of Fermions and Valiant's results can be reproved in this setting. Recently, Jozsa et al. [11] gave a different way of simulating circuits consisting of matchgates by using a very smart trick for evaluating the probability distribution of a single-qubit measurement's outcome.

In this chapter, we will study the matchgates and matchcircuits, which are constructed from matchgates, from the point of view of the language of Fermionic quantum computation. We, therefore, will not mention definitions of matchgates and matchcircuits from the graph theory point of view. Instead, we will define matchgates as matrices with specific properties. In the next section, we will introduce some notation in the area of Fermionic quantum computation. We will provide some simple examples of matchgates and their properties in Section 5.3. In Section 5.4, we study the simulation of quantum circuits consisting of Gaussian operations. Section 5.5 is devoted to explaining the relation between matchgates and Gaussian operations. In Section 5.6, we will show the universality of matchgates acting on at most next-nearest-neighbor (n.n.n. ) qubits. We compare the matchgate formalism with the matrix product formalism and the stabilizer formalism in Section 5.7. This chapter is mainly based on [11, 19].

## 5.2 Notation

In this chapter, we shall consider  $4 \times 4$  matrices of  $G(A, B)$  form (cf. [19]) as follows:

$$G(A, B) = \begin{pmatrix} a_{11} & 0 & 0 & a_{12} \\ 0 & b_{11} & b_{12} & 0 \\ 0 & b_{21} & b_{12} & 0 \\ a_{21} & 0 & 0 & a_{22} \end{pmatrix}, \quad (5.1)$$

where

$$A = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix}, \quad B = \begin{pmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{pmatrix},$$

Let us consider the action of  $G(A, B)$  on a 2-qubit state  $|\psi\rangle = c_{00}|00\rangle + c_{01}|01\rangle + c_{10}|10\rangle + c_{11}|11\rangle$ . The resulting state is  $|\psi'\rangle = c'_{00}|00\rangle + c'_{01}|01\rangle + c'_{10}|10\rangle + c'_{11}|11\rangle$  where

$$\begin{pmatrix} c'_{00} \\ c'_{11} \end{pmatrix} = A \cdot \begin{pmatrix} c_{00} \\ c_{11} \end{pmatrix}, \quad \begin{pmatrix} c'_{01} \\ c'_{10} \end{pmatrix} = B \cdot \begin{pmatrix} c_{01} \\ c_{10} \end{pmatrix}.$$

The action of  $G(A, B)$ , therefore, amounts to  $A$  acting on the even-parity subspace and  $B$  acting on the odd-parity subspace of the space of 2-qubit states. Thus  $G(A, B)$  preserves both the even and odd parity subspaces.

Let us denote by  $U(2)$  the group of  $2 \times 2$  unitary matrices and by  $SU(2)$  the group of  $2 \times 2$  unitary matrices of determinant 1. A  $G(A, B)$  matrix with both  $A, B$  in  $U(2)$  with the same determinant is called a *matchgate matrix*. For the relation between matchgate matrices and graph-theoretical matchgates defined by Valiant in [20], we will use a theorem in Cai et al. [6] to prove that every matchgate matrix can be realized by a graph-theoretical matchgate.

**Theorem 5.2.1.** (Cai, Choudhary and Lu [6]) *Consider a  $4 \times 4$  complex matrix  $U$ . Denote by  $D(ij, kl)$  the determinant of the  $2 \times 2$  submatrix of  $B$  consisting of rows  $i$  and  $j$ , and columns  $k$  and  $l$ , i.e.  $D(ij, kl) = u_{ik}u_{jl} - u_{il}u_{jk}$ . Matrix  $U$  can be realized by a graph-theoretical matchgate if and only if the following 10 identities hold:*

$$\begin{aligned} D(14, 14) &= D(23, 23), & D(24, 14) &= D(24, 23), \\ D(34, 14) &= D(34, 23), & D(14, 34) &= D(23, 34), \\ D(14, 24) &= D(23, 24), & D(12, 14) &= D(12, 23), \\ D(13, 14) &= D(13, 23), & D(14, 12) &= D(23, 12), \\ D(14, 13) &= D(23, 13), & D(14, 23) &= D(23, 14). \end{aligned} \quad (5.2)$$

For the  $G(A, B)$  matrices, except for the first identity, the other 9 identities in (5.2) are easily verified since both determinants in any of these identities are 0. The first identity requires that  $\det(A) = \det(B)$ , which holds for all matchgate matrices. Therefore, all the matchgate matrices can be thought as graph-theoretical matchgates. From now on, we restrict our attention to the matchgate matrices only. For the sake of simplicity, we call them matchgates.

In this chapter, we shall use the language of Fermionic quantum computation. To this end, we shall briefly introduce some related definitions. For more details, [4, 16] provide overviews about Fermionic quantum computation and its connection with the standard model of quantum computation. Things related to Fermions can be expressed in terms of annihilation and creation operators, denoted as  $a_j, a_j^\dagger$  for  $j = 1, \dots, n$ .

**Definition 5.2.2.** (Bravyi and Kitaev [4]) In the  $n$ -qubit system, the annihilation operator  $a_j$  acts on basis vectors as follows:

$$\begin{aligned} a_j|i_1 \dots i_{j-1} 1 i_{j+1} \dots i_n\rangle &= (-1)^{\sum_{k=1}^{j-1} i_k} |i_1 \dots i_{j-1} 0 i_{j+1} \dots i_n\rangle, \\ a_j|i_1 \dots i_{j-1} 0 i_{j+1} \dots i_n\rangle &= 0, \end{aligned}$$

and the creation operator  $a_j^\dagger$  is the conjugate transpose of  $a_j$  and acts on basis vectors as follows:

$$\begin{aligned} a_j^\dagger|i_1 \dots i_{j-1} 0 i_{j+1} \dots i_n\rangle &= (-1)^{\sum_{k=1}^{j-1} i_k} |i_1 \dots i_{j-1} 1 i_{j+1} \dots i_n\rangle, \\ a_j^\dagger|i_1 \dots i_{j-1} 1 i_{j+1} \dots i_n\rangle &= 0. \end{aligned}$$

The annihilation and creation operators satisfy the anti-commutation rules:

$$\{a_j, a_k\} = \{a_j^\dagger, a_k^\dagger\} = 0, \quad \{a_j, a_k^\dagger\} = \delta_{jk} I \quad (5.3)$$

for all  $j, k \in \{1, \dots, n\}$ , where we recall that  $\{A, B\} = AB + BA$  denotes the anti-commutator between  $A$  and  $B$ .

Given a set of annihilation and creation operators  $a_j, a_j^\dagger$ , for  $j = 1, \dots, n$ , let us consider the following  $2n$  operators:

$$c_{2j-1} = a_j + a_j^\dagger, \quad c_{2j} = (a_j - a_j^\dagger)/i, \quad (5.4)$$

for  $j = 1, \dots, n$ . The operators  $c_j$  are called Majorana Fermionic operators [4, 11]. It is easy to verify that  $c_{2j-1}^\dagger = c_{2j-1}$  and  $c_{2j}^\dagger = -(a_j^\dagger - a_j)/i = c_{2j}$ . Hence, all  $c_j$  are Hermitian. The following Proposition shows the anti-commutation property of  $c_j$ .

**Proposition 5.2.3.** Operators  $c_j$  satisfy the anti-commutation relation,

$$\{c_j, c_k\} = 2\delta_{jk} I, \quad (5.5)$$

for all  $j, k \in \{1, \dots, 2n\}$ .

*Proof.* We consider the following 4 cases:

**Case 1:**  $j, k$  are both even. Suppose that  $j = 2j', k = 2k'$ . We have

$$\begin{aligned} \{c_{2j'}, c_{2k'}\} &= -\{a_{j'} - a_{j'}^\dagger, a_{k'} - a_{k'}^\dagger\} \\ &= -\left(\{a_{j'}, a_{k'}\} + \{a_{j'}^\dagger, a_{k'}^\dagger\} - \{a_{j'}, a_{k'}^\dagger\} - \{a_{j'}^\dagger, a_{k'}\}\right) \\ &= \{a_{j'}, a_{k'}^\dagger\} + \{a_{j'}^\dagger, a_{k'}\} \\ &= 2\delta_{j'k'} I = 2\delta_{jk} I. \end{aligned}$$

**Case 2:**  $j, k$  are both odd. Suppose that  $j = 2j' - 1, k = 2k' - 1$ . We also have

$$\begin{aligned} \{c_{2j'-1}, c_{2k'-1}\} &= \{a_{j'} + a_{j'}^\dagger, a_{k'} + a_{k'}^\dagger\} \\ &= \{a_{j'}, a_{k'}\} + \{a_{j'}^\dagger, a_{k'}^\dagger\} + \{a_{j'}, a_{k'}^\dagger\} + \{a_{j'}^\dagger, a_{k'}\} \\ &= \{a_{j'}, a_{k'}^\dagger\} + \{a_{j'}^\dagger, a_{k'}\} \\ &= 2\delta_{j'k'} I = 2\delta_{jk} I. \end{aligned}$$

**Case 3:**  $j$  is even and  $k$  is odd. Suppose that  $j = 2j', k = 2k' - 1$ . We have

$$\begin{aligned} \{c_{2j'}, c_{2k'-1}\} &= -i\{a_{j'} - a_{j'}^\dagger, a_{k'} + a_{k'}^\dagger\} \\ &= -i\left(\{a_{j'}, a_{k'}\} - \{a_{j'}^\dagger, a_{k'}^\dagger\} + \{a_{j'}, a_{k'}^\dagger\} - \{a_{j'}^\dagger, a_{k'}\}\right) \\ &= -i\left(\{a_{j'}, a_{k'}^\dagger\} - \{a_{j'}^\dagger, a_{k'}\}\right) \\ &= -i(\delta_{j'k'}I - \delta_{j'k'}I) = 0 = \delta_{jk}I. \end{aligned}$$

**Case 4:**  $j$  is odd and  $k$  is even. This case is similar to Case 3.  $\square$

We are mainly interested in such operators  $c_j$  in this text. In fact, Jozsa et al. [11] gave a direct definition of them without using the Fermion language. They were defined as  $2n$  Hermitian operators satisfying the anti-commutation relation in (5.5).

Given  $2n$  Majorana Fermionic operators  $c_j$ , we will consider only quadratic Hamiltonians of the following form:

$$H = i \sum_{j \neq k; j, k=1}^{2n} h_{jk} c_j c_k, \quad (5.6)$$

where  $h = (h_{jk})$  is a  $2n \times 2n$  matrix of coefficients, which is chosen to be a real skew-symmetric matrix, i.e.  $h_{jk} = -h_{kj}$ . This choice is to guarantee that  $H$  is Hermitian. To see that, we compute  $H^\dagger$

$$H^\dagger = -i \sum_{j \neq k; j, k=1}^{2n} h_{jk} (c_j c_k)^\dagger = -i \sum_{j \neq k; j, k=1}^{2n} h_{jk} c_k^\dagger c_j^\dagger = i \sum_{j \neq k; j, k=1}^{2n} h_{kj} c_k c_j = H.$$

**Definition 5.2.4.** A unitary operation  $U$  is called a Gaussian operation if  $U$  can be written as  $e^{iH}$ , where  $H$  is a quadratic Hamiltonian of the above type.

By  $e^A$ , we mean the exponential of square matrix  $A$ , which is defined by the power series

$$I + \frac{A}{1!} + \frac{A^2}{2!} + \cdots + \frac{A^n}{n!} + \cdots$$

We refer to Theorem 1.4.1 in Section 1.4.2 for more properties of the exponential of matrices.

In showing the relation between matchgates and Gaussian operations, we will focus on the Majorana Fermionic operators constructed from the so-called Jordan-Wigner representation. In this representation, the Majorana Fermionic operators will be

$$\begin{aligned} c_1 &= X \otimes I \otimes \cdots \otimes I, \quad \dots \quad c_{2k-1} = Z \otimes \cdots \otimes Z \otimes X \otimes I \otimes \cdots \otimes I, \\ c_2 &= Y \otimes I \otimes \cdots \otimes I, \quad \dots \quad c_{2k} = Z \otimes \cdots \otimes Z \otimes Y \otimes I \otimes \cdots \otimes I \end{aligned} \quad (5.7)$$

for all  $k = 1, \dots, n$ , where in  $c_{2k-1}$  and  $c_{2k}$  Pauli matrices  $X$  and  $Y$  occur at the  $k^{\text{th}}$  position. In the Jordan-Wigner representation, we can also explicitly express the annihilation and creation operators as follows:

$$\begin{aligned} a_j &= \frac{c_{2j-1} + ic_{2j}}{2} = Z \otimes \cdots \otimes Z \otimes X_- \otimes I \otimes \cdots \otimes I, \\ a_j^\dagger &= \frac{c_{2j-1} - ic_{2j}}{2} = Z \otimes \cdots \otimes Z \otimes X_+ \otimes I \otimes \cdots \otimes I, \end{aligned} \quad (5.8)$$

where  $X_- = |0\rangle\langle 1|$ ,  $X_+ = |1\rangle\langle 0|$ . This definition of  $a_j, a_j^\dagger$  is called the Jordan-Wigner transform [16], which is the bridge between the Fermionic quantum computation model and the standard quantum computation model.

We now have all necessary notation for this chapter. The next section will provide more details about matchgates.

### 5.3 Some Properties of Matchgates

To study properties of matchgates, we first transform matchgates  $G(A, B)$  to a simpler form, namely the direct sum form  $A \oplus B$ . To this end, we consider the basis transformation based on the following matrix

$$T = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}.$$

Under this transformation, we have:  $|00\rangle \mapsto |00\rangle$ ;  $|01\rangle \mapsto |10\rangle$ ;  $|10\rangle \mapsto |11\rangle$ ;  $|11\rangle \mapsto |01\rangle$ . Therefore, the  $G(A, B)$  mapping will preserve the two subspaces spanned by  $\{|00\rangle, |01\rangle\}$  and  $\{|10\rangle, |11\rangle\}$  in the new basis. The matrix corresponding to the mapping in the new basis, hence, will be  $A \oplus B$  and we have  $G(A, B) = T(A \oplus B)T^{-1}$ .

**Theorem 5.3.1.** *The following statements are true:*

1.  $G(A, B)G(A', B') = G(AA', BB')$ .
2. If both  $A, B$  are invertible, then so is  $G(A, B)$ ; and  $G(A, B)^{-1} = G(A^{-1}, B^{-1})$ .
3. Let  $H_A, H_B$  denote the Hamiltonians corresponding to  $A, B$ , respectively, i.e.  $A = e^{iH_A}, B = e^{iH_B}$ . Then the Hamiltonian of  $G(A, B)$  is  $G(H_A, H_B)$ .

*Proof.* 1. We have

$$\begin{aligned} G(A, B)G(A', B') &= \left(T(A \oplus B)T^{-1}\right) \left(T(A' \oplus B')T^{-1}\right) = T(A \oplus B)(A' \oplus B')T^{-1} \\ &= T((AA') \oplus (BB'))T^{-1} \\ &= G(AA', BB'). \end{aligned}$$

2. This follows immediately from statement 1 above.

3. Let us consider the power series representing  $e^{iG(H_A, H_B)}$ :

$$e^{iG(H_A, H_B)} = \sum_{n=0}^{\infty} \frac{i^n G(H_A, H_B)^n}{n!}.$$

By statement 1 above, we have  $G(H_A, H_B)^n = G((H_A)^n, (H_B)^n)$ . Hence,

$$\begin{aligned} e^{iG(H_A, H_B)} &= \sum_{n=0}^{\infty} \frac{i^n G((H_A)^n, (H_B)^n)}{n!} \\ &= \sum_{n=0}^{\infty} G\left(\frac{i^n (H_A)^n}{n!}, \frac{i^n (H_B)^n}{n!}\right) \\ &= G\left(\sum_{n=0}^{\infty} \frac{(iH_A)^n}{n!}, \sum_{n=0}^{\infty} \frac{(iH_B)^n}{n!}\right) \\ &= G(e^{iH_A}, e^{iH_B}) = G(A, B). \end{aligned}$$

□

To illustrate these concepts, we now consider some simple matchgates, the matrices and the circuits corresponding to them.

$G(X, X)$  equals

$$\begin{pmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix} = X \otimes X,$$

which is realized by the circuit in Figure 5.1.

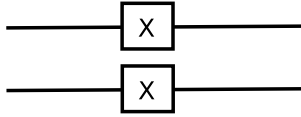


Figure 5.1: The circuit realizing the gate  $G(X, X)$

$G(Z, X)$  equals

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix} = G(Z, I)G(I, X) = CZ \cdot SWAP,$$

where CZ denotes the controlled-Z gate, and SWAP denotes the swap gate. Note that neither CZ nor SWAP is a matchgate since  $\det(Z) = \det(X) = -1 \neq \det(I)$ .  $G(Z, X)$  is realized by the circuit in Figure 5.2, where the first gate denotes the SWAP gate, the second gate denotes the CZ gate.

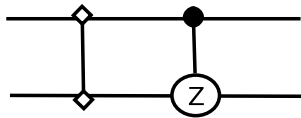


Figure 5.2: The circuit realizing the gate  $G(Z, X)$

$G(H, H)$  equals

$$\frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & -1 & 0 \\ 1 & 0 & 0 & -1 \end{pmatrix},$$

which is realized by the circuit in Figure 5.3.

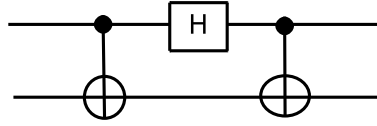


Figure 5.3: The circuit realizing the gate  $G(H, H)$

$G(X, Z)$  equals

$$\begin{pmatrix} 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix} = G(X, I)G(I, Z),$$

which is realized by the circuit in Figure 5.4.

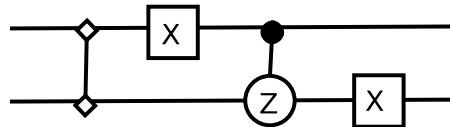


Figure 5.4: The circuit realizes the gate  $G(X, Z)$

$G(Z, Z)$  equals

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix} = G(I, Z)G(Z, I),$$

which is realized by the circuit in Figure 5.5.

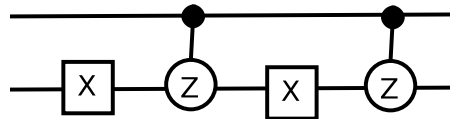


Figure 5.5: The circuit realizing the gate  $G(Z, Z)$

## 5.4 Simulation of Quantum Circuits of Gaussian operations

We start by considering actions of Gaussian operations under conjugation to Majorana Fermionic operators  $c_j$ . It turns out that the resulting operators lie

inside the  $2n$ -dimensional space spanned by  $c_1, \dots, c_{2n}$ .

**Theorem 5.4.1.** (Terhal and DiVincenzo [19], Jozsa and Miyake [11]) *Let  $H$  be any quadratic Hamiltonian and  $U = e^{iHt}$  be the corresponding Gaussian operation. Then for all  $j = 1, \dots, 2n$ , we have*

$$U^\dagger c_j U = \sum_{k=1}^{2n} R_{jk} c_k, \quad (5.9)$$

where the matrix  $R = e^{4h}$  is a matrix in  $SO(2n)$ .

*Proof.* Let us consider the function  $U(t) = e^{iHt}$  of real variable  $t$ , and the function  $c_j(t) = U(t)c_jU(t)^\dagger$ . We know that  $U(t)^\dagger = e^{-iHt}$ . We recall that if  $A$  is a matrix and  $f(t) = e^{At}$  is a function of real variable  $t$ , then  $df(t)/dt = Af(t) = f(t)A$ . We now can compute the derivative of  $c_j(t)$  with respect to  $t$ :

$$\begin{aligned} \frac{dc_j(t)}{dt} &= \frac{dU(t)}{dt} c_j U(t)^\dagger + U(t) c_j \frac{dU(t)^\dagger}{dt} \\ &= U(t)(iH)c_j U(t)^\dagger + U(t)c_j(-iH)U(t)^\dagger \\ &= iU(t)(Hc_j - c_jH)U(t)^\dagger = iU(t)[H, c_j]U(t)^\dagger, \end{aligned} \quad (5.10)$$

where  $[A, B] = AB - BA$  denotes the commutator of matrices  $A, B$ .

For distinct  $k$  and  $p$ , if  $j \neq k, p$ , then  $[c_k c_p, c_j] = c_k c_p c_j - c_j c_k c_p = 0$  since these three operators anti-commute. We also have  $[c_k c_p, c_k] = c_k c_p c_k - c_k c_k c_p = -2c_p$  and  $[c_k c_p, c_p] = c_k c_p c_p - c_p c_k c_p = 2c_k$ . Therefore,  $[H, c_j]$  equals

$$\begin{aligned} i \sum_{k \neq p; k, p=1}^{2n} h_{kp} [c_k c_p, c_j] &= i \sum_{p \neq j; p=1}^{2n} h_{jp} [c_j c_p, c_j] + i \sum_{k \neq j; k=1}^{2n} h_{kj} [c_k c_j, c_j] \\ &= i \sum_{p \neq j; p=1}^{2n} h_{jp} (-2c_p) + i \sum_{k \neq j; k=1}^{2n} h_{kj} (2c_k) \\ &= -2i \sum_{p=1}^{2n} h_{jp} c_p - 2i \sum_{k \neq j; k=1}^{2n} h_{jk} c_k \\ &= -4i \sum_{p=1}^{2n} h_{jp} c_p. \end{aligned} \quad (5.11)$$

We have used the fact that  $h_{kj} = -h_{jk}$  and  $h_{jj} = 0$  for all  $j, k \in \{1, \dots, 2n\}$ . Substituting the result of (5.11) into (5.10) gives us

$$\frac{dc_j(t)}{dt} = 4 \sum_{p=1}^{2n} h_{jp} U(t) c_p U(t)^\dagger = 4 \sum_{p=1}^{2n} h_{jp} c_p(t). \quad (5.12)$$

Consider the length- $2n$  column vector  $C(t)$  whose  $j^{\text{th}}$  entry is  $c_j(t)$ . (5.12) is equivalent to the differential equation  $\frac{dC(t)}{dt} = 4h \cdot C(t)$ , whose solution is  $C(t) = e^{4ht} C(0)$ . Hence,  $c_j(t) = \sum_{p=1}^{2n} R_{jp}(t) c_p$ , where  $R = e^{4ht}$ . Setting  $t = 1$  gives us (5.9).

To see why  $R = e^{4h}$  lies in  $SO(2n)$ , we notice that  $R^T = e^{4h^T} = e^{-4h}$  and  $RR^T = e^{4h} e^{-4h} = I$ , and by Theorem 1.4.1,  $\det(R) = e^{\text{tr}(4h)} = e^0 = 1$ .  $\square$

(5.9) is very crucial for our later study of simulation of quantum circuits. Matrix  $U$  itself might require an exponential number of coefficients when expressed in terms of  $c_1, \dots, c_{2n}$  and their products. However, the action of  $U$  under conjugation to operators  $c_j$  can be described by keeping track of only the  $2n \times 2n$  coefficient matrix  $h$ .

Let us consider a size- $m$  circuit  $C$  consisting of Gaussian gates only with the following chronological order:  $U_1, \dots, U_m$ , and an initial state  $|\psi_{in}\rangle$ . The outcome state will be  $|\psi_{out}\rangle = \tilde{U}|\psi_{in}\rangle$ , where  $\tilde{U} = U_m \cdots U_1$ . We consider applying measurements to the outcome state. Since  $c_j$  are Hermitian, they can be regarded as observables. The expectation of the outcome is

$$\mathbf{E}(c_j) = \langle \psi_{out} | c_j | \psi_{out} \rangle = \langle \psi_{in} | \tilde{U}^\dagger c_j \tilde{U} | \psi_{in} \rangle. \quad (5.13)$$

Applying (5.9) gives us

$$\begin{aligned} \tilde{U}^\dagger c_j \tilde{U} &= \sum_{k=1}^{2n} R_{jk}^{(m)} U_1^\dagger \cdots U_{m-1}^\dagger c_k U_{m-1} \cdots U_1 \\ &= \sum_{k=1}^{2n} R_{jk}^{(m)} \left( \sum_{p=1}^{2n} R_{kp}^{(m-1)} U_1^\dagger \cdots U_{m-2}^\dagger c_p U_{m-2} \cdots U_1 \right), \end{aligned} \quad (5.14)$$

where  $R^{(m)}, R^{(m-1)}$  denote the matrices corresponding to the actions of  $U_m$  and  $U_{m-1}$ , respectively. We, furthermore, denote by  $R^{(m,m-1)}$  the matrix product of  $R^{(m)}$  and  $R^{(m-1)}$ . (5.14) becomes

$$\tilde{U}^\dagger c_j \tilde{U} = \sum_{p=1}^{2n} R_{jp}^{(m,m-1)} U_1^\dagger \cdots U_{m-2}^\dagger c_p U_{m-2} \cdots U_1 \quad (5.15)$$

Repeating this computation  $m-1$  times brings us to

$$\tilde{U}^\dagger c_j \tilde{U} = \sum_{s=1}^{2n} \tilde{R}_{js} c_s, \quad (5.16)$$

where  $\tilde{R} = R^{(m)} \cdot R^{(m-1)} \cdots R^{(1)}$ . Computing  $\tilde{R}$  can be done in  $m \cdot O(n^3)$  steps. Substituting (5.16) into (5.13) gives

$$\mathbf{E}(c_j) = \sum_{s=1}^{2n} \tilde{R}_{js} \langle \psi_{in} | c_s | \psi_{in} \rangle. \quad (5.17)$$

We now switch our attention to the Majorana Fermionic operators in the Jordan-Wigner representation. Each  $c_s$  is expressed as a tensor product of Pauli matrices  $P_1^{(s)} \otimes P_2^{(s)} \otimes \cdots \otimes P_n^{(s)}$ . If we assume, furthermore, that the initial state  $|\psi_{in}\rangle$  is a product state, namely  $|\psi_{in}\rangle = |\psi_1\rangle |\psi_2\rangle \cdots |\psi_n\rangle$ , then

$$\mathbf{E}(c_j) = \sum_{s=1}^{2n} \tilde{R}_{js} \left( \prod_{k=1}^n \langle \psi_k | P_k^{(s)} | \psi_k \rangle \right), \quad (5.18)$$

which can be done in  $O(n^2)$  steps.

Let us consider the observable  $Z^{(r)}$ , i.e. the 1-qubit observable  $Z$  applied to the  $r^{\text{th}}$  qubit. In the Jordan-Wigner representation, we have  $Z^{(r)} = -ic_{2r-1}c_{2r}$ . Let  $p_0$  and  $p_1$  denote the probabilities of observing 0 and 1, respectively. We know that  $p_0 + p_1 = 1$  and, moreover,

$$\begin{aligned} p_0 - p_1 &= \mathbf{E} \left( Z^{(r)} \right) = \langle \psi_{in} | \tilde{U}^\dagger (-ic_{2r-1}c_{2r}) \tilde{U} | \psi_{in} \rangle \\ &= -i \langle \psi_{in} | \left( \tilde{U}^\dagger c_{2r-1} \tilde{U} \right) \left( \tilde{U}^\dagger c_{2r} \tilde{U} \right) | \psi_{in} \rangle \end{aligned} \quad (5.19)$$

Substituting (5.16) into (5.19) gives us

$$\begin{aligned} p_0 - p_1 &= \sum_{s,p=1}^{2n} \tilde{R}_{2r-1,s} \tilde{R}_{2r,p} \langle \psi_{in} | c_{2r-1} c_{2r} | \psi_{in} \rangle \\ &= \sum_{s,p=1}^{2n} \tilde{R}_{2r-1,s} \tilde{R}_{2r,p} \left( \prod_{k=1}^n \langle \psi_k | P_k^{(2r-1)} P_k^{(2r)} | \psi_k \rangle \right), \end{aligned} \quad (5.20)$$

which can be computed in  $O(n^3)$  steps. The probability distribution  $\{p_0, p_1\}$ , hence, can be computed in  $m \cdot O(n^3)$  steps. We have just proved the following theorem.

**Theorem 5.4.2.** (Jozsa and Miyake [11]) *Consider a class of poly-sized quantum circuits each of which contains Gaussian operations such that*

- *The input state is a product state.*
- *The output is a final measurement's outcome in the computational basis on any single qubit line.*

*Then the probability distribution of the measurement outcome can be computed in  $m \cdot O(n^3)$  steps, where  $m$  is the number of Gaussian gates, and  $n$  is the number of qubits in the circuit.*

## 5.5 Relation between Gaussian Operations and Matchgates

Gaussian operations are defined in terms of quadratic Hamiltonians. Surprisingly, there is a close relation between Gaussian operations and matchgates. In this section, we will consider this relation and show that matchgates are in fact special cases of Gaussian operations in the Jordan-Wigner representation.

**Theorem 5.5.1.** (Jozsa and Miyake [11]) *The set of matchgates acting on nearest-neighbor (n.n.) qubits is contained in the set of Gaussian operations in the Jordan-Wigner representation.*

*Proof.* We consider Gaussian operations acting on qubits 1 and 2. There are four Fermionic operators involved, which are  $c_1, c_2, c_3$  and  $c_4$ . Corresponding Hamiltonians will be linear combinations (with real coefficients) of the following 6 terms:

$$\begin{aligned} ic_1c_2 &= -Z \otimes I & ic_2c_3 &= -X \otimes X \\ ic_1c_3 &= Y \otimes X & ic_2c_4 &= -X \otimes Y \\ ic_1c_4 &= Y \otimes Y & ic_3c_4 &= -I \otimes Z \end{aligned} \quad (5.21)$$

Let  $S$  denote the set of these Hamiltonians. Then  $\{e^{iH} : H \in S\}$  is the set of Gaussian operations acting on qubits 1 and 2.

Let  $\mathcal{H}$  denote the set of  $2 \times 2$  Hermitian traceless matrices. Consider  $T = \{G(X, Y) : X, Y \in \mathcal{H}\}$ . We know that for every  $A \in SU(2)$ , there exists a Hamiltonian  $H_A$  such that  $A = e^{iH_A}$  and  $H_A$  is Hermitian. The condition  $\det(A) = 1$  requires that  $H_A$  is traceless. Suppose we have an  $H_A \in \mathcal{H}$ , then  $A = e^{iH_A}$  will be in  $SU(2)$ . We now consider a matchgate  $G(A, B)$  with  $A, B \in SU(2)$ , and suppose that  $H_A, H_B \in \mathcal{H}$  are Hamiltonians of  $A, B$ , respectively. Theorem 5.3.1 implies that  $G(H_A, H_B)$  is the Hamiltonian of  $G(A, B)$ . Hence, the Hamiltonian of  $G(A, B)$  lies in  $T$ . Conversely, consider an element  $G(X, Y)$  in  $T$ . By Theorem 5.3.1 we know that  $e^{iG(X, Y)}$  equals  $G(e^{iX}, e^{iY})$ , which is a matchgate since both  $e^{iX}, e^{iY}$  are in  $SU(2)$ . We have shown that the set  $\{e^{iH} : H \in T\}$  is exactly the set of matchgates  $G(A, B)$ .

We are going to prove that  $S = T$ , which implies that the set of the matchgates  $G(A, B)$  equals the set of Gaussian operations acting on qubits 1 and 2. It is easy to see that the action of  $Z$  to a state does not change its parity, while the action of each of  $X, Y$  to a state adds 1 to its parity, i.e. flips the parity of the state. Therefore,  $c_1c_2$  and  $c_3c_4$  preserve the even and odd parity subspaces. So do the rest of the terms in (5.21) since each of them consists of two operators that both flip the parity of the state that they are applied to. We can also easily verify that all six operators in (5.21) are traceless. We remember, furthermore, that Hamiltonians in  $S$  are always Hermitian. Therefore,  $S \subseteq T$ .

We can view  $S$  as a linear space of dimension 6 since it can be easily shown that  $c_1c_2, c_1c_3, c_1c_4, c_2c_3, c_2c_4, c_3c_4$  are linearly independent. Let us consider the following 6 matrices

$$\begin{aligned} & \begin{pmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix}, \begin{pmatrix} 0 & 0 & 0 & -i \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ i & 0 & 0 & 0 \end{pmatrix}, \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix}, \\ & \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}, \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & -i & 0 \\ 0 & i & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}, \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}. \end{aligned} \quad (5.22)$$

Linear combinations with real coefficients will generate the entire set  $T$ . Therefore,  $T$  has dimension at most 6. We have already shown that  $T$  contains  $S$  which is of dimension 6. We conclude that  $S = T$ .

If matchgates  $G(A, B)$  act on qubits  $i$  and  $i + 1$ , we consider Fermionic operators  $c_{2i-1}, c_{2i}, c_{2i+1}, c_{2i+2}$ . The same reasoning as above proves that this set of matchgates is equal to the set of Gaussian operations acting on only qubits  $i$  and  $i + 1$ . This concludes the proof.  $\square$

By Theorems 5.4.2 and 5.5.1, we have already proved the following theorem:

**Theorem 5.5.2.** (Valiant [20], Terhal and DiVincenzo [19]) Consider a class of poly-sized quantum circuits each of which contains only matchgates such that

- The matchgates act on  $n.n.$  qubit lines only.
- The input state is a product state.

- The output is a final measurement's outcome in the computational basis on any single qubit line.

Then the probability distribution of the measurement outcome can be computed in polynomial time.

We call circuits consisting of only matchgates *matchcircuits* and call the circuits mentioned in Theorem 5.5.2 *n.n. matchcircuits*. Theorem 5.5.2 says that the outcome of a single-qubit measurement in the computational basis at the end of an *n.n. matchcircuit* can be classically simulated. Note that the circuits mentioned in the theorem have gates acting only on *n.n. qubits*. The *n.n.* condition is crucial since matchgates not restricted to act on *n.n. qubits* are quantumly universal. This statement will be proved in the next section.

## 5.6 The Power of Matchgates

Jozsa et al. proved a surprising result, which showed that the set of *n.n. matchgates* and *n.n.n. matchgates* is quantumly universal. By next *n.n.n. matchgates*, we mean matchgates acting on qubits at distance 2. It implies that the set of *n.n. matchgates* and the SWAP gate is also quantumly universal. We see that, within the matchgate formalism, the SWAP gate is the key difference between classical computing and quantum computing.

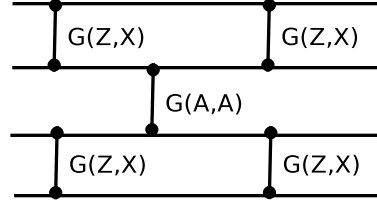
**Theorem 5.6.1.** (Jozsa and Miyake [11]) *Let  $C_n$  be any family of quantum circuits with output given by a  $Z$  basis measurement on the first line. Then  $C_n$  can be simulated by a circuit of matchgates acting on *n.n.* or next *n.n.* qubits with at most a constant increase in the size of the circuit.*

*Proof.* It is well-known that the CZ gate together with 1-qubit gates form a quantumly universal set. Hence, given any quantum circuit, we can approximate it by a circuit consisting of CZ gates and 1-qubit gates. Since the SWAP gate can be obtained from a sequence of three CNOT gates and each CNOT gate can be obtained from one CZ gate and two Hadamard gates, i.e.  $(I \otimes H)CZ(I \otimes H)$ , the SWAP gate can be obtained from a sequence of CZ gates and 1-qubit gates. Therefore, we can approximate the given circuit by a circuit  $C$  consisting of *n.n.* interacting CZ gates and 1-qubit gates.

For each qubit line in the circuit  $C$ , we replace it by four qubit lines and encode the original basis states  $|0\rangle$  and  $|1\rangle$  as logical states  $|0_l\rangle = |0000\rangle$  and  $|1_l\rangle = |1001\rangle$ . We also need to replace gates in  $C$  by encoded gates in the new circuit  $C'$ .

Suppose there is a 1-qubit gate  $A$  acting on the first qubit line of  $C$ . Let us consider the following sequence of *n.n.* matchgates:  $G(Z, X)_{12}G(Z, X)_{34}G(A, A)_{23}G(Z, X)_{12}G(Z, X)_{34}$  acting on the first four qubit lines of  $C'$  as depicted in Figure 5.6, where notation  $G(.,.)_{ij}$  means the gate  $G(.,.)$  is applied to qubit lines  $i$  and  $j$ .

By Theorem 5.3.1 we have  $G(Z, X) = G(Z, I)G(I, X) = CZ \cdot \text{SWAP}$ . Now suppose  $|\psi_{in}\rangle = \alpha|0000\rangle + \beta|1001\rangle$  is the initial state, which is the encoded state


 Figure 5.6: The circuit realizing the encoded gate of the gate  $A$ .

of  $\alpha|0\rangle + \beta|1\rangle$ . The evolution of  $|\psi_{in}\rangle$  through the circuit is as follows

$$\begin{aligned}
 |\psi_{in}\rangle &\xrightarrow{G(Z,X)_{12}} \alpha|0000\rangle + \beta|0101\rangle \\
 &\xrightarrow{G(Z,X)_{34}} \alpha|0000\rangle + \beta|0110\rangle = |0\rangle(\alpha|00\rangle + \beta|11\rangle)|0\rangle \\
 &\xrightarrow{G(A,A)_{23}} |0\rangle \left( (a_{11}\alpha + a_{12}\beta)|00\rangle + (a_{21}\alpha + a_{22}\beta)|11\rangle \right) |0\rangle \\
 &= (a_{11}\alpha + a_{12}\beta)|0000\rangle + (a_{21}\alpha + a_{22}\beta)|0110\rangle \\
 &\xrightarrow{G(Z,X)_{12}} (a_{11}\alpha + a_{12}\beta)|0000\rangle + (a_{21}\alpha + a_{22}\beta)|1010\rangle \\
 &\xrightarrow{G(Z,X)_{34}} (a_{11}\alpha + a_{12}\beta)|0000\rangle + (a_{21}\alpha + a_{22}\beta)|1001\rangle
 \end{aligned}$$

where  $a_{11}, a_{12}, a_{21}, a_{22}$  are the elements of  $A$ . Therefore, the circuit in Figure 5.6 is the encoded gate of the gate  $A$ .

We now consider the n.n. CZ gates in  $C$ . Suppose there is an n.n. CZ gate acting on qubit lines 1 and 2. Its actions on the basis states are as follows:  $|00\rangle \mapsto |00\rangle, |01\rangle \mapsto |01\rangle, |10\rangle \mapsto |10\rangle, |11\rangle \mapsto -|11\rangle$ . On the logical states, we must have

$$\begin{aligned}
 |0000\rangle|0000\rangle &\mapsto |0000\rangle|0000\rangle, \\
 |0000\rangle|1001\rangle &\mapsto |0000\rangle|1001\rangle, \\
 |1001\rangle|0000\rangle &\mapsto |1001\rangle|0000\rangle, \\
 |1001\rangle|1001\rangle &\mapsto -|1001\rangle|1001\rangle,
 \end{aligned} \tag{5.23}$$

which can be achieved by applying a CZ gate on encoded qubit lines 4 and 5.

By Theorem 5.3.1, we have  $CZ = G(Z, I) = G(H, H)G(X, I)G(H, H)$  since  $Z = HXH, I = HIH$ . Furthermore,  $CZ = G(H, H)G(X, X)G(I, X)G(H, H)$ . Except for  $G(I, X)$ , all other gates are matchgates. In fact,  $G(I, X)$  is the SWAP gate. So we have decomposed the CZ gate into a sequence of 3 matchgates and one SWAP gate.

We have shown that: if a 1-qubit  $A$  occurs in qubit  $i$  of the circuit  $C$ , then the encoded gate will consist of 5 n.n. matchgates acting on qubit lines  $4i - 3, 4i - 2, 4i - 1, 4i$  of the circuit  $C'$ . If an n.n. matchgate occurs on qubit lines  $i, i + 1$  of the circuits  $C$ , then the encoded gate will consist of 3 n.n. matchgates and one SWAP gate, which all act on qubit lines  $4i, 4i + 1$  of  $C'$ . This SWAP gate can be interpreted as swapping the two qubit lines  $4i$  and  $4i + 1$ . These swappings, however, never move one qubit line more than one position from its original position. Therefore, each matchgate on  $C'$  acts on qubit lines of at most distance 2. This concludes the proof.  $\square$

## 5.7 Comparison of the Matchgate, Matrix Product, Contracting Tensor Network and Stabilizer Formalisms

### 5.7.1 The Matchgate Formalism and the Matrix Product Formalism

We recall that the main result in the matrix product formalism says that quantum states with bounded entanglement in the sense of the Schmidt rank can be efficiently described and any circuit that does not produce highly entangled states from a given input state can be classically simulated. We ask the question: can matchcircuits produce highly entangled states from a given input? The answer is yes, and there even exist matchcircuits consisting of only n.n. matchgates that can produce maximally entangled states. This result shows that the class of n.n. matchcircuits is not included in the class of quantum circuits simulable by using the matrix product formalism. Conversely, we easily see that there is no restriction on the gates used in the matrix product formalism. Hence, the class of quantum circuits simulable by using the matrix product formalism is not included in the class of n.n. matchcircuits neither.

**Theorem 5.7.1.** *For any integer  $n$ , there exists an n.n. matchcircuit that outputs a state of Schmidt rank  $2^{\lfloor n/2 \rfloor}$  from the input state  $|0\rangle^{\otimes n}$ .*

*Proof.* It is sufficient to consider the theorem in case  $n$  is even, namely  $n = 2k$ , and show that there exists an n.n. matchcircuit that given the input state  $|0\rangle^{\otimes n}$  will outputs a state of Schmidt rank  $2^k$  with respect to the partition  $1 \dots k | (k+1) \dots 2k$ .

It is well-known that the circuit in Figure 5.7 produces the state  $(1/\sqrt{2^k}) \cdot \sum_{i=0}^{2^k-1} |i\rangle|i\rangle$  from the input state  $|0\rangle^{\otimes n}$ .

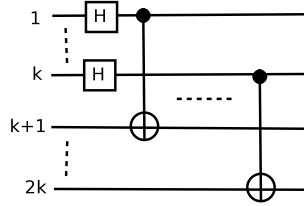


Figure 5.7: The circuit producing a maximally entangled state from the input state  $|0\rangle^{\otimes n}$

The action of the circuit can be interpreted in the following way: the first step is to create  $k$  EPR pairs  $(|00\rangle + |11\rangle)/\sqrt{2}$  on  $k$  pairs of qubits  $(1, 2), \dots, (2k-1, 2k)$ . This task can be achieved by using the composed gate  $(H \otimes I) \cdot \text{CNOT}$  as depicted in Figure 5.8.

The second step is to swap the qubit lines according to the following permutation:

$$\begin{pmatrix} 1 & 2 & \dots & k & k+1 & \dots & 2k-1 & 2k \\ 1 & k+1 & \dots & 2k-1 & 2 & \dots & k & 2k \end{pmatrix}. \quad (5.24)$$

This task can be achieved by the following sequence of the SWAP gates:  $\text{SWAP}_{2,k+1}, \text{SWAP}_{3,k+2}, \dots, \text{SWAP}_{k,2k-1}$ . There exists a sequence of n.n. SWAP gates,

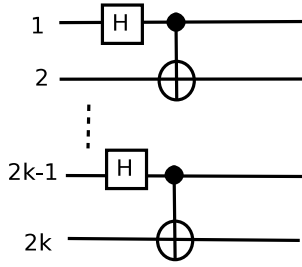


Figure 5.8: The circuit producing the state  $(|00\rangle + |11\rangle)^{\otimes k} / \sqrt{5^k}$  from the input state  $|0\rangle^{\otimes 2k}$ .

which also achieves this:  $\text{SWAP}_{k,k+1}, \text{SWAP}_{k-1,k}, \dots, \text{SWAP}_{2,3}; \text{SWAP}_{k+1,k+2}, \text{SWAP}_{k,k+1}, \dots, \text{SWAP}_{3,4}; \dots; \text{SWAP}_{2k-2,2k-1}, \text{SWAP}_{2k-3,2k-2}, \dots, \text{SWAP}_{k,k+1}$ .

We will now construct two n.n. matchcircuits realizing the two steps and then combining them sequentially will yield the desired n.n. matchcircuit. The circuit of the matchgate  $G(H, H)$  in Figure 5.3, given the input state  $|00\rangle$ , will outcome the EPR state  $(|00\rangle + |11\rangle) / \sqrt{2}$ . Therefore, if we replace each pair of the Hadamard gate and CNOT gate in Figure 5.8 by the n.n. matchgate  $G(H, H)$ , then the resulting circuit will outcome the same output state as the original state does from the input  $|0\rangle^{\otimes 2k}$ .

To realize the above sequence of n.n. SWAP gates, we consider the matchgate  $G(Z, X)$  depicted in Figure 5.2. This gate can be decomposed into  $\text{CZ} \cdot \text{SWAP}$ , whose action is the same as the action of the SWAP gate on basis states  $|00\rangle, |01\rangle, |10\rangle$ . The basis state  $|11\rangle$  is mapped to  $-|11\rangle$  by  $G(Z, X)$  and is unchanged by the SWAP gate. In this case,  $G(Z, X)$  only adds a minus sign to the state. Hence, if we replace each SWAP gate in the above sequence by a  $G(Z, X)$  gate and apply it to the state  $(|00\rangle + |11\rangle)^{\otimes k} / \sqrt{2^k}$ , the resulting state will be of the form  $(1/\sqrt{2^k}) \sum_{i=0}^{2^k-1} a_i |i\rangle |i\rangle$ , where  $a_i \in \{1, -1\}$ . This state also has Schmidt rank  $2^k$ . This concludes the proof.  $\square$

### 5.7.2 The Matchgate Formalism and The Stabilizer Formalism

It is easy to see that the set of n.n. matchcircuits is neither included nor contains the set of Clifford circuits, which is classically simulable by using the stabilizer formalism. The stabilizer formalism allows gates such as the CNOT gate that is not a matchgate. Moreover, from the proof of Theorem 5.6.1 we see that every 1-qubit gate can be realized by n.n. matchgates, while the set of Clifford operations does not cover every 1-qubit gate, for example the  $\pi/8$  gate.

Still, there is a common technique used in both formalisms. The matchgate formalism uses the important fact that the conjugation of every Gaussian operation with respect to any Majorana operator results in an operator lying in the linear span of the  $2n$  Majorana operators. If we denote by  $\mathcal{M}$  the space spanned by these operators, it will imply that  $U^\dagger M U$  lies inside  $\mathcal{M}$  for all  $M \in \mathcal{M}$  and any Gaussian operation  $U$ . Similarly, the stabilizer formalism uses the fact that operators in the Clifford group  $\mathcal{C}$  fix the Pauli group  $\mathcal{P}$  under conjugation. Explicitly, we have  $C^\dagger P C$  is an element of  $\mathcal{P}$  for any  $C \in \mathcal{C}$  and  $P \in \mathcal{P}$ . This suggests that there is some common mathematics technique behind the two

formalisms. We leave it as an open question for further study.

### 5.7.3 The Matchgate Formalism and the Contracting Tensor Network Formalism

We recall that the contracting tensor network formalism allows us to classically simulate quantum circuits with small treewidths of corresponding circuit graphs (Theorem 3.5.8) with no restriction on the types of gates. Hence, the class of simulable circuits from this formalism is not included in the class of n.n. matchcircuits. There is also no restriction on the structure of n.n. matchcircuits, i.e. we always get an n.n. matchcircuit regardless of the way we position n.n. matchgates in a circuit. Therefore, we can get matchcircuits whose circuit graphs have big treewidths. These matchcircuits are not simulable by the contracting tensor network formalism. The class of n.n. matchcircuits, hence, is not included in the class of quantum circuits simulable by the contracting tensor network formalism neither.

## 5.8 Summary

In this chapter, we considered classical simulation of quantum circuits consisting of nearest-neighbor interacting matchgates, which are 2-qubit gates acting locally on the odd-parity and even-parity subspaces. We focused on the interpretation of matchgates using the language of Majorana Fermionic operators in the Jordan-Wigner representation. To prove the classical simulability of such circuits, we first proved the simulability of quantum circuits consisting of Gaussian operations. After that, the simulability of matchgate circuits followed immediately from the fact that matchgates acting on n.n. qubits are actually Gaussian operations. We ended the chapter with some comparison of three formalisms, namely the matchgate formalism, matrix product formalism and the stabilizer formalism.

# APPENDIX

## A-1 Proof of Theorem 2.7.2

We use the following form of  $|\psi\rangle$  to prove the theorem:

$$|\psi\rangle = \sum_{\substack{\alpha_{l-1}, \alpha_l, \alpha_{l+1} \\ i_l, i_{l+1}}} \lambda_{\alpha_{l-1}}^{[l-1]} \Gamma_{\alpha_{l-1}\alpha_l}^{[l]i_l} \lambda_{\alpha_l}^{[l]} \Gamma_{\alpha_l\alpha_{l+1}}^{[l+1]i_{l+1}} \lambda_{\alpha_{l+1}}^{[l+1]} |\phi_{\alpha_{l-1}}^{[1\dots l-1]}\rangle |i_l i_{l+1}\rangle |\phi_{\alpha_{l+1}}^{[l+2\dots n]}\rangle.$$

A derivation of this equation can be done similarly as with Equation (2.11).

Notice that the  $\lambda_{\alpha_{l-1}}^{[l-1]}$  do not change after performing  $V$  since they are the coefficients of the  $(l-1)^{th}$  Schmidt decomposition under which  $V$  is local. The same reason is also applied to  $\lambda_{\alpha_{l+1}}^{[l+1]}$ ,  $|\phi_{\alpha_{l-1}}^{[1\dots l-1]}\rangle$  and  $|\phi_{\alpha_{l+1}}^{[l+2\dots n]}\rangle$ . Therefore we only need to update  $\Gamma_{\alpha_{l-1}\alpha_l}^{[l]i_l}$ ,  $\Gamma_{\alpha_l\alpha_{l+1}}^{[l+1]i_{l+1}}$  and  $\lambda_{\alpha_l}^{[l]}$ .

Let  $\Omega_{\alpha_{l-1}\alpha_{l+1}}^{i_l i_{l+1}} = \sum_{\alpha_l} \lambda_{\alpha_{l-1}}^{[l-1]} \Gamma_{\alpha_{l-1}\alpha_l}^{[l]i_l} \lambda_{\alpha_l}^{[l]} \Gamma_{\alpha_l\alpha_{l+1}}^{[l+1]i_{l+1}} \lambda_{\alpha_{l+1}}^{[l+1]}$ . We arrive at

$$|\psi\rangle = \sum_{\alpha_{l-1}, \alpha_{l+1}} \sum_{i_l, i_{l+1}} \Omega_{\alpha_{l-1}\alpha_{l+1}}^{i_l i_{l+1}} |\phi_{\alpha_{l-1}}^{[1\dots l-1]}\rangle |i_l i_{l+1}\rangle |\phi_{\alpha_{l+1}}^{[l+2\dots n]}\rangle.$$

We index  $V$ 's entries using binary indices,  $V_{ij}^{kl}$  where  $ij$  is the row index,  $kl$  is the column index. We have  $V|kl\rangle = \sum_{i,j} V_{ij}^{kl} |ij\rangle$ ;  $V\left(\Omega_{\alpha_{l-1}\alpha_{l+1}}^{kl} |kl\rangle\right) = \sum_{i,j} V_{ij}^{kl} \Omega_{\alpha_{l-1}\alpha_{l+1}}^{kl} |ij\rangle$ . Therefore

$$V\left(\sum_{kl} \Omega_{\alpha_{l-1}\alpha_{l+1}}^{kl} |kl\rangle\right) = \sum_{k,l} \sum_{i,j} V_{ij}^{kl} \Omega_{\alpha_{l-1}\alpha_{l+1}}^{kl} |ij\rangle = \sum_{i,j} \left(\sum_{k,l} V_{ij}^{kl} \Omega_{\alpha_{l-1}\alpha_{l+1}}^{kl}\right) |ij\rangle.$$

Hence  $\Omega_{\alpha_{l-1}\alpha_{l+1}}^{i_l i_{l+1}}$  change after performing  $V$  according to

$$\Omega'_{\alpha_{l-1}\alpha_{l+1}}^{i_l i_{l+1}} = \sum_{k,l} V_{i_l i_{l+1}}^{kl} \Omega_{\alpha_{l-1}\alpha_{l+1}}^{kl}. \quad (\text{A-1})$$

The resulting state is

$$|\psi'\rangle = \sum_{\alpha_{l-1}, \alpha_{l+1}} \sum_{i_l, i_{l+1}} \Omega'_{\alpha_{l-1}\alpha_{l+1}}^{i_l i_{l+1}} |\phi_{\alpha_{l-1}}^{[1\dots l-1]}\rangle |i_l i_{l+1}\rangle |\phi_{\alpha_{l+1}}^{[l+2\dots n]}\rangle.$$

Now we need to update  $\Gamma_{\alpha_{l-1}\alpha_l}^{[l]i_l}$ ,  $\Gamma_{\alpha_l\alpha_{l+1}}^{[l+1]i_{l+1}}$  and  $\lambda_{\alpha_l}^{[l]}$ . In order to update  $\Gamma_{\alpha_l\alpha_{l+1}}^{[l+1]i_{l+1}}$ , we need to compute eigenvectors of reduced system  $[l+1\dots n]$  since  $\Gamma_{\alpha_l\alpha_{l+1}}^{[l+1]i_{l+1}}$  are Schmidt vectors corresponding to the  $l^{th}$  decomposition. Rewriting  $|\psi'\rangle$  gives

$$|\psi'\rangle = \sum_{\substack{\alpha_{l-1}, i_l \\ \alpha_{l+1}, i_{l+1}}} \Omega'_{\alpha_{l-1}\alpha_{l+1}}^{i_l i_{l+1}} |\phi_{\alpha_{l-1}}^{[1\dots l-1]}\rangle |i_l\rangle |i_{l+1}\rangle |\phi_{\alpha_{l+1}}^{[l+2\dots n]}\rangle.$$

By Theorem 1.3.1, the reduced density matrix of system  $[l+1 \dots n]$  is given by

$$\rho^{[l+1 \dots n]} = \sum_{\substack{\alpha_{l+1}, i_{l+1} \\ \alpha'_{l+1}, i'_{l+1}}} \left( \sum_{\alpha_{l-1}, i_l} \Omega'_{\alpha_{l-1} \alpha_{l+1}} \Omega'_{\alpha_{l-1} \alpha'_{l+1}} \right) |i_{l+1} \phi_{\alpha_{l+1}}^{[l+2 \dots n]} \rangle \langle i'_{l+1} \phi_{\alpha'_{l+1}}^{[l+2 \dots n]}|. \quad (\text{A-2})$$

(Note that  $\alpha'_{l+1} \in \{1, \dots, \chi\}$ ,  $i_{l+1} \in \{0, 1\}$ ).

Diagonalizing  $\rho^{[l+1 \dots n]}$  gives us its eigenvalues  $\{\lambda_{\beta_l}^{[l]}\}$  and eigenvectors  $\{|\phi_{\beta_l}^{\prime[l+1 \dots n]}\rangle\}$ . The range of new index  $\beta_l$  is from 1 up to the rank of  $\rho^{[l+1 \dots n]}$ , which is at most  $2\chi$ . Expressing  $|\phi_{\beta_l}^{\prime[l+1 \dots n]}\rangle$  in terms of  $\{|\phi_{\alpha_{l+1}}^{[l+2 \dots n]}\rangle\}$  gives us a new tensor  $\Gamma^{\prime[l+1]}$

$$|\phi_{\beta_l}^{\prime[l+1 \dots n]}\rangle = \sum_{\alpha_{l+1}, i_l} \Gamma_{\beta_l \alpha_{l+1}}^{\prime[l+1] i_{l+1}} \lambda_{\alpha_{l+1}}^{[l+1]} |i_l\rangle |\phi_{\alpha_{l+1}}^{[l+2 \dots n]}\rangle. \quad (\text{A-3})$$

In order to compute  $\Gamma^{\prime[l]}$ , we first compute eigenvectors of subsystem  $[1 \dots l]$ . Then we express these vectors in terms of  $\{|\phi_{\alpha_{l-1}}^{[1 \dots l-1]} i_l\rangle\}$ . These eigenvectors can be calculated easily since we already have eigenvectors of subsystem  $[l+1 \dots n]$  and  $\lambda_{\beta_l}^{\prime[l]}$

$$|\phi_{\beta_l}^{\prime[1 \dots l]}\rangle = \frac{1}{\lambda_{\beta_l}^{\prime[l]}} \langle \phi_{\beta_l}^{\prime[l+1 \dots n]} | \psi' \rangle. \quad (\text{A-4})$$

Expressing  $|\phi_{\beta_l}^{\prime[1 \dots l]}\rangle$  in terms of  $\{|\phi_{\alpha_{l-1}}^{[1 \dots l-1]} i_l\rangle\}$  using Lemma 2.6.1 gives us  $\Gamma^{\prime[l]}$

$$|\phi_{\beta_l}^{\prime[1 \dots l]}\rangle = \sum_{\alpha_{l-1}, i_l} \Gamma_{\alpha_{l-1} \beta_l}^{\prime[l] i_l} \lambda_{\beta_l}^{\prime[l]} |\phi_{\alpha_{l-1}}^{[1 \dots l-1]} i_l\rangle. \quad (\text{A-5})$$

### Complexity of the update procedure:

The first computational step is to compute the tensor  $\Omega = \Omega_{\alpha_{l-1} \alpha_{l+1}}^{i_l i_{l+1}}$ , where  $i, j$  are binary indices,  $\alpha$  and  $\beta$  can run from 1 up to  $\chi$ . In total, there are  $4\chi^2$  entries. Each entry requires  $O(\chi)$  steps since it is computed by equation  $\Omega_{\alpha_{l-1} \alpha_{l+1}}^{i_l i_{l+1}} = \sum_{\alpha_l} \lambda_{\alpha_{l-1}}^{[l-1]} \Gamma_{\alpha_{l-1} \alpha_l}^{[l] i_l} \lambda_{\alpha_l}^{[l]} \Gamma_{\alpha_l \alpha_{l+1}}^{[l+1] i_{l+1}} \lambda_{\alpha_{l+1}}^{[l+1]}$ . Therefore we need  $20\chi^3 = O(\chi^3)$  steps to accomplish this.

The next computational step involves in computing updated  $\Omega'$  by Equation (A-1), which requires  $O(\chi^2)$  steps.

Next, we need to compute the reduced density matrix  $\rho^{[l+1 \dots n]}$  by Equation (A-2). The size of  $\rho^{[l+1 \dots n]}$  is  $2\chi \times 2\chi$ . Each entry can be computed in  $O(\chi)$  steps from entries of  $\Omega'$ . Hence  $\rho^{[l+1 \dots n]}$  can be computed in  $O(\chi^3)$  steps.

The next step is to compute eigenvalues and eigenvectors of  $\rho^{[l+1 \dots n]}$  to obtain  $\lambda_{\beta_l}^{\prime[l]}$  and  $|\phi_{\beta_l}^{\prime[l+1 \dots n]}\rangle$ . This can be done in  $O(\chi^3)$  steps. Notice that  $\lambda_{\beta_l}^{\prime[l]}$  and  $|\phi_{\beta_l}^{\prime[l+1 \dots n]}\rangle$  now are expressed in terms of the new basis  $\{|i_{l+1} \phi_{\alpha_{l+1}}^{[l+2 \dots n]}\rangle\}$  instead of in the conventional basis  $\{|i_{l+1} \dots i_n\rangle\}$ . However remember that eigenvalues of a matrix do not change under a transformation from an orthonormal basis to another orthonormal basis. Therefore,  $\lambda_{\alpha_l}^{\prime[l]}$  are indeed what we expect. This representation of  $|\phi_{\beta_l}^{\prime[l+1 \dots n]}\rangle$  helps us to compute  $\Gamma_{\beta_l \alpha_{l+1}}^{\prime[l+1] i_{l+1}}$  by Equation (A-3) directly from its coefficients (which takes only 1 division per entry). Therefore,

it takes at most  $2\chi$  steps to update  $\Gamma_{\alpha_l \alpha_{l+1}}^{[l+1]i_{l+1}}$  since  $\beta_l$  runs from 1 up to at most  $2\chi$ .

Let  $(\phi'_{\beta_l})_{i_{l+1}\alpha_{l+1}}^{[l+1\dots n]}$  denote components of vector  $|\phi'_{\beta_l}{}^{[l+1\dots n]}\rangle$ . Then, the  $|\phi'_{\beta_l}{}^{[1\dots l]}\rangle$  can be computed by (A-4) as follows

$$|\phi'_{\alpha_l}{}^{[1\dots l]}\rangle = \sum_{\alpha_{l-1}, i_l} \sum_{\alpha_{l+1}, i_{l+1}} \left[ \Omega'_{\alpha_{l-1}\alpha_{l+1}}{}^{i_l i_{l+1}} (\phi'_{\beta_l}{}^{[l+1\dots n]})_{i_{l+1}\alpha_{l+1}}^* \right] |\phi_{\alpha_{l-1}}^{[1\dots l-1]i_l}\rangle.$$

This step can obviously be done in  $O(\chi^3)$  steps.

Since  $|\phi'_{\alpha_l}{}^{[1\dots l]}\rangle$  is expressed in basis  $\{|\phi_{\alpha_{l-1}}^{[1\dots l-1]i_l}\rangle\}$ , the last update step can be achieved in  $2\chi$  steps by Equation (A-5).

All the above updates require at most  $O(\chi^3)$  steps. We conclude that updating a state after applying a 2-qubit unitary operation  $V$  costs  $O(\chi^3)$  steps.

## A-2 Proof of Theorem 2.7.3

Suppose that a measurement is applied to qubit  $l$ .

Let  $B_{\alpha_{l-1}\alpha_l}^{i_l} = \lambda_{\alpha_{l-1}}^{[l-1]i_l} \Gamma_{\alpha_{l-1}\alpha_l}^{[l]i_l} \lambda_{\alpha_l}^{[l]}$ . By Equation (2.11) we get

$$|\psi\rangle = \sum_{\alpha_{l-1}, \alpha_l; i_l} B_{\alpha_{l-1}\alpha_l}^{i_l} |\phi_{\alpha_{l-1}}^{[1\dots l-1]i_l}\rangle |i_l\rangle |\phi_{\alpha_l}^{[l+1\dots n]}\rangle. \quad (\text{A-6})$$

Since the Schmidt vectors are orthonormal, the probability of observing the outcome  $i_l$  is

$$p(i_l) = \sum_{\alpha_{l-1}, \alpha_l} |B_{\alpha_{l-1}\alpha_l}^{i_l}|^2. \quad (\text{A-7})$$

If we care only about the probability distribution of the outcome, we are done. However, if we want to reuse the MPR of the post-measurement state, we need to update all  $\Gamma^{[l]}$  and  $\lambda^{[l]}$ .

Without loss of generality, we assume that the outcome is 0. The state after the measurement is

$$|\psi'\rangle = \frac{1}{\sqrt{p(0)}} \sum_{\alpha_{l-1}, \alpha_l} B_{\alpha_{l-1}\alpha_l}^0 |\phi_{\alpha_{l-1}}^{[1\dots l-1]}\rangle |0\rangle |\phi_{\alpha_l}^{[l+1\dots n]}\rangle.$$

Let  $A_{\alpha_{l-1}\alpha_l}^{i_l} = \frac{B_{\alpha_{l-1}\alpha_l}^{i_l}}{\sqrt{p(i_l)}}$ , we get

$$|\psi'\rangle = \sum_{\alpha_{l-1}, \alpha_l} A_{\alpha_{l-1}\alpha_l}^0 |\phi_{\alpha_{l-1}}^{[1\dots l-1]}\rangle |0\rangle |\phi_{\alpha_l}^{[l+1\dots n]}\rangle. \quad (\text{A-8})$$

Tracing over qubits  $1, \dots, l-1$ , by Theorem 1.3.1, we obtain the reduced density matrix of subsystem  $[l\dots n]$

$$\rho^{[l\dots n]} = \sum_{\alpha_l, \alpha'_l} \left( \sum_{\alpha_{l-1}} A_{\alpha_{l-1}\alpha_l}^0 A_{\alpha_{l-1}\alpha'_l}^{0*} \right) |0\rangle |\phi_{\alpha_l}^{[l+1\dots n]}\rangle \langle 0| \langle \phi_{\alpha'_l}^{[l+1\dots n]}|. \quad (\text{A-9})$$

Updated Schmidt coefficients, denoted by  $\lambda'_{\beta_{l-1}}^{[l-1]}$ , and updated Schmidt vectors, denoted by  $|\phi'_{\beta_{l-1}}^{[l\dots n]}\rangle$ , for the  $(l-1)^{th}$  Schmidt decomposition are eigenvalues and eigenvectors of  $\rho^{[l\dots n]}$ . Notice that  $|\phi'_{\beta_{l-1}}^{[l\dots n]}\rangle$  are expressed in basis  $\{|0\phi_{\alpha_l}^{[l+1\dots n]}\rangle\}$ . Therefore, if we let  $M_{\beta_{l-1}\alpha_l}$  be its components, we instantly get the following equation

$$|\phi'_{\beta_{l-1}}^{[l\dots n]}\rangle = \sum_{\alpha_l} M_{\beta_{l-1}\alpha_l} |0\phi_{\alpha_l}^{[l+1\dots n]}\rangle. \quad (\text{A-10})$$

To compute  $\lambda'_{\beta_l}^{[l]}$  and  $|\phi'_{\beta_l}^{[l+1\dots n]}\rangle$ , we need the reduced density matrix of subsystem  $[l+1\dots n]$ . It can be computed using Equation (A-8) and Theorem 1.3.1

$$\rho^{[l+1\dots n]} = \sum_{\alpha_l, \alpha'_l} \left( \sum_{\alpha_{l-1}} A_{\alpha_{l-1}\alpha_l}^0 A_{\alpha_{l-1}\alpha'_l}^{0*} \right) |\phi_{\alpha_l}^{[l+1\dots n]}\rangle \langle \phi_{\alpha'_l}^{[l+1\dots n]}|. \quad (\text{A-11})$$

Let  $M_{\beta_l\alpha_l}$  be the components of vector  $|\phi'_{\beta_l}^{[l+1\dots n]}\rangle$ , we get

$$|\phi'_{\beta_l}^{[l+1\dots n]}\rangle = \sum_{\alpha_l} M_{\beta_l\alpha_l} |\phi_{\alpha_l}^{[l+1\dots n]}\rangle. \quad (\text{A-12})$$

The relation between  $|\phi'_{\beta_{l-1}}^{[l\dots n]}\rangle$  and  $|\phi'_{\beta_l}^{[l+1\dots n]}\rangle$  provides us  $\Gamma_{\beta_{l-1}\beta_l}^{[l]i_l}$

$$\Gamma_{\beta_{l-1}\beta_l}^{[l]i_l} = \frac{1}{\lambda_{\beta_{l-1}}^{[l-1]}} \langle i_l \phi'_{\beta_l}^{[l+1\dots n]} | \phi'_{\beta_{l-1}}^{[l\dots n]} \rangle. \quad (\text{A-13})$$

To compute  $\lambda'_{\beta_{l+1}}^{[l+1]}$  and  $|\phi'_{\beta_{l+1}}^{[l+2\dots n]}\rangle$ , we need the reduced density matrix of subsystem  $[l+2\dots n]$ . To do that, consider the  $l^{th}$  Schmidt decomposition of  $|\psi'\rangle$

$$\begin{aligned} |\psi'\rangle &= \sum_{\beta_l} \lambda'_{\beta_l}^{[l]} |\phi'_{\beta_l}^{[1\dots l]}\rangle |\phi'_{\beta_l}^{[l+1\dots n]}\rangle \\ &= \sum_{\beta_l, \alpha_l} \lambda'_{\beta_l}^{[l]} M_{\beta_l\alpha_l} |\phi'_{\beta_l}^{[1\dots l]}\rangle |\phi_{\alpha_l}^{[l+1\dots n]}\rangle \\ &= \sum_{\beta_l, \alpha_l} \lambda'_{\beta_l}^{[l]} M_{\beta_l\alpha_l} |\phi'_{\beta_l}^{[1\dots l]}\rangle \sum_{\alpha_{l+1}, i_l} \Gamma_{\alpha_l\alpha_{l+1}}^{[l+1]i_{l+1}} \lambda_{\alpha_{l+1}}^{[l+1]} |i_{l+1}\rangle |\phi_{\alpha_{l+1}}^{[l+2\dots n]}\rangle \\ &= \sum_{\beta_l, \alpha_l, \alpha_{l+1}} \lambda'_{\beta_l}^{[l]} M_{\beta_l\alpha_l} \Gamma_{\alpha_l\alpha_{l+1}}^{[l+1]i_{l+1}} \lambda_{\alpha_{l+1}}^{[l+1]} |i_{l+1}\rangle |\phi'_{\beta_l}^{[1\dots l]}\rangle |i_{l+1}\rangle |\phi_{\alpha_{l+1}}^{[l+2\dots n]}\rangle \\ &= \sum_{\beta_l, \alpha_{l+1}} C_{\beta_l\alpha_{l+1}}^{i_{l+1}} |i_{l+1}\rangle |\phi'_{\beta_l}^{[1\dots l]}\rangle |i_{l+1}\rangle |\phi_{\alpha_{l+1}}^{[l+2\dots n]}\rangle, \end{aligned}$$

where  $C_{\beta_l\alpha_{l+1}}^{i_{l+1}} = \sum_{\alpha_l} \lambda'_{\beta_l}^{[l]} M_{\beta_l\alpha_l} \Gamma_{\alpha_l\alpha_{l+1}}^{[l+1]i_{l+1}} \lambda_{\alpha_{l+1}}^{[l+1]}$ . By Theorem 1.3.1, the reduced density matrix of subsystem  $[l+1\dots n]$  is

$$\rho^{[l+2\dots n]} = \sum_{\alpha_{l+1}, \alpha'_{l+1}} \left( \sum_{\beta_l i_l} C_{\beta_l\alpha_{l+1}}^{i_{l+1}} C_{\beta_l\alpha'_{l+1}}^{i_{l+1}} \right) |\phi_{\alpha_{l+1}}^{[l+2\dots n]}\rangle \langle \phi_{\alpha'_{l+1}}^{[l+2\dots n]}|, \quad (\text{A-14})$$

from which we can find updated eigenvalues  $\lambda'_{\beta_{l+1}}^{[l+1]}$  and eigenvectors  $|\phi'_{\beta_{l+1}}^{[l+2\dots n]}\rangle$ .

Let  $M_{\beta_{l+1}\alpha_{l+1}}$  be the components of vectors  $|\phi'_{\beta_{l+1}}^{[l+2\dots n]}\rangle$ , that is

$$|\phi'_{\beta_{l+1}}^{[l+2\dots n]}\rangle = \sum_{\alpha_{l+1}} M_{\beta_{l+1}\alpha_{l+1}} |\phi'_{\alpha_{l+1}}^{[l+2\dots n]}\rangle.$$

The relation between  $|\phi'_{\beta_l}^{[l+1\dots n]}\rangle$  and  $|\phi'_{\beta_{l+1}}^{[l+2\dots n]}\rangle$  gives us  $\Gamma'_{\beta_l\beta_{l+1}}^{[l+1]i_{l+1}}$

$$\Gamma'_{\beta_l\beta_{l+1}}^{[l+1]i_{l+1}} = \frac{1}{\lambda'_{\beta_{l+1}}^{[l+1]}} \langle i_{l+1} | \phi'_{\beta_{l+1}}^{[l+1\dots n]} | \phi'_{\beta_l}^{[l+1\dots n]} \rangle. \quad (\text{A-15})$$

Computing  $\lambda'_{\beta_{l+2}}^{[l+2]}$ ,  $\Gamma'_{\beta_{l+1}\beta_{l+2}}^{[l+2]i_{l+2}}$ ,  $\dots$ ,  $\lambda'_{\beta_{n-1}}^{[n-1]}$ ,  $\Gamma'_{\beta_{n-1}\beta_n}^{[n]i_n}$  can be done in the same manner.

To compute new Schmidt coefficients new  $\Gamma$  for qubits  $l-2, \dots, 1$ , we notice the following equation (Lemma 2.6.1)

$$|\phi_{\alpha_l}^{[1\dots l]}\rangle = \sum_{\alpha_{l-1}, i_l} \Gamma_{\alpha_{l-1}\alpha_l}^{[l]i_l} \lambda_{\alpha_l}^{[l]} |\phi_{\alpha_{l-1}}^{[1\dots l-1]}\rangle |i_l\rangle.$$

Therefore we can update these coefficients in the same manner as we did before, just in the other direction.

### Complexity of the update procedure

Each entry of tensor  $B$  requires 2 steps. Computing whole tensor  $B$ , hence, costs  $4\chi^2$  steps.

The probability distribution  $\{p(0), p(1)\}$  computed by Equation (A-7) costs  $4\chi^2$  steps.

Computing density matrix  $\rho^{[l\dots n]}$  by Equation (A-9) costs no more than  $O(\chi^3)$  steps.

Finding eigenvalues  $\lambda'_{\beta_{l-1}}^{[l-1]}$  and eigenvectors  $\phi'_{\beta_{l-1}}^{[l\dots n]}$  costs  $O(\chi^3)$  steps.

Calculating  $\Gamma'_{\beta_{l-1}\beta_l}^{[l]i_l}$  is easy since by Equations (A-10), (A-12) and (A-13) we have

$$\Gamma'_{\beta_{l-1}\beta_l}^{[l]i_l} = \begin{cases} 0 & \text{if } i_l = 1 \\ \frac{1}{\lambda'_{\beta_{l-1}}^{[l-1]}} \sum_{\alpha_l} M_{\beta_l\alpha_l}^* M_{\beta_{l-1}\alpha_l} & \text{if } i_l = 0 \end{cases}$$

The complexity of this process is  $O(\chi^3)$ . However, updating other  $\Gamma'_{\beta_{k-1}\beta_k}^{[k]i_k}$  will be different. Let's consider computing  $\Gamma'_{\beta_l\beta_{l+1}}^{[l+1]i_{l+1}}$ . Rewriting Equation (A-12) gives

$$|\phi'_{\beta_l}^{[l+1\dots n]}\rangle = \sum_{\alpha_l, \alpha_{l+1}, i_{l+1}} M_{\beta_l\alpha_l} \Gamma_{\alpha_l\alpha_{l+1}}^{[l+1]i_{l+1}} \lambda_{\alpha_{l+1}}^{[l+1]} |i_{l+1}\rangle |\phi'_{\alpha_{l+1}}^{[l+2\dots n]}\rangle.$$

Since all  $|\phi'_{\beta_l}^{[l+1\dots n]}\rangle$  and  $|\phi'_{\beta_{l+2}}^{[l+2\dots n]}\rangle$  are expressed in terms of  $|\phi'_{\alpha_{l+1}}^{[l+2\dots n]}\rangle$ , we can compute updated  $\Gamma'_{\beta_l\beta_{l+1}}^{[l+1]i}$  by Equation (A-15) as follows

$$\Gamma'_{\beta_l\beta_{l+1}}^{[l+1]i} = \frac{1}{\lambda'_{\beta_{l+1}}^{[l+1]}} \sum_{\alpha_l, \alpha_{l+1}} M_{\beta_l\alpha_l} \Gamma_{\alpha_l\alpha_{l+1}}^{[l+1]i_{l+1}} M_{\beta_{l+1}\alpha_{l+1}}^*, \quad (\text{A-16})$$

which can be accomplished in  $O(\chi^4)$  steps.

In conclusion, the update procedure for one qubit costs at most  $O(\chi^4)$  steps. To update all  $n$  qubits, we need to perform at most  $O(n\chi^4)$  steps.

# Bibliography

- [1] Scott Aaronson and Daniel Gottesman. Improved simulation of stabilizer circuits. *Physical Review A*, 70(5):052328, Nov 2004.
- [2] Stefan Arnborg, Derek G. Corneil, and Andrzej Proskurowski. Complexity of finding embeddings in a k-tree. *SIAM Journal of Algebraic Discrete Methods*, 8(2):277–284, 1987.
- [3] H.L. Bodlaender. Treewidth: characterizations, applications, and computations. *Technical Report, UU-CS-2006-041*, 2006.
- [4] S. B. Bravyi and A. Y. Kitaev. Fermionic quantum computation. *Annals of Physics*, 298:210–226, May 2002.
- [5] Harry Buhrman, Richard Cleve, Monique Laurent, Alexander Schrijver Noah Linden, and Falk Unger. New limits on fault-tolerant quantum computation. *Proceedings of 47th IEEE FOCS*, 0:411–419, 2006.
- [6] Jin-Yi Cai, Vinay Choudhary, and Pinyan Lu. On the theory of matchgate computations. In *CCC '07: Proceedings of the Twenty-Second Annual IEEE Conference on Computational Complexity*, pages 305–318, Washington, DC, USA, 2007. IEEE Computer Society.
- [7] Don Coppersmith and Shmuel Winograd. Matrix multiplication via arithmetic progressions. *Journal of Symbolic Computation*, 9(3):251–280, 1990.
- [8] D. Gottesman. The Heisenberg representation of quantum computers. *ArXiv Quantum Physics e-prints: quant-ph/9807006*, July 1998.
- [9] Richard Jozsa. On the simulation of quantum circuits. *ArXiv Quantum Physics e-prints: quant-ph/0603163*, 2006.
- [10] Richard Jozsa and Noah Linden. On the role of entanglement in quantum-computational speed-up. *Proceedings: Mathematical, Physical and Engineering Sciences*, 459(2036):2011–2032, 2003.
- [11] Richard Jozsa and Akimasa Miyake. Matchgates and classical simulation of quantum circuits. *ArXiv Quantum Physics e-prints: quant-ph/0804.4050*, 2008.
- [12] E. Knill. Fermionic linear optics and matchgates. *ArXiv Quantum Physics e-prints: quant-ph/0108033*, August 2001.

- 
- [13] Igor L. Markov and Yaoyun Shi. Simulating quantum computation by contracting tensor networks. *ArXiv Quantum Physics e-prints: quant-ph/0511069*, 2005.
- [14] Gabriele Nebe, E. M. Rains, and N. J. A. Sloane. The invariants of the clifford groups. *Designs, Codes and Cryptography*, 24(1):99–122, 2001.
- [15] M. A. Nielsen and I. L. Chuang. *Quantum Computation and Quantum Information*. Cambridge University Press, 2000.
- [16] G. Ortiz, J. E. Gubernatis, E. Knill, and R. Laflamme. Quantum algorithms for Fermionic simulations. *Physical Review A*, 64(2):022319, Jul 2001.
- [17] Neil Robertson and P. D. Seymour. Graph minors: X. obstructions to tree-decomposition. *Journal of Combinatorial Theory, Series B*, 52(2):153–190, 1991.
- [18] Peter W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Journal on Computing*, 26(5):1484–1509, 1997.
- [19] Barbara M. Terhal and David P. DiVincenzo. Classical simulation of noninteracting-Fermion quantum circuits. *Physical Review A*, 65(3):032325, Mar 2002.
- [20] Leslie G. Valiant. Quantum circuits that can be simulated classically in polynomial time. *SIAM Journal on Computing*, 31(4):1229–1254, 2002.
- [21] Leslie G. Valiant. Holographic algorithms (extended abstract). In *FOCS '04: Proceedings of the 45th Annual IEEE Symposium on Foundations of Computer Science*, pages 306–315, Washington, DC, USA, 2004. IEEE Computer Society.
- [22] Maarten van den Nest. Local equivalence of stabilizer states and codes. PhD thesis, Katholieke Universiteit Leuven, Belgium, 2005.
- [23] Guifré Vidal. Efficient classical simulation of slightly entangled quantum computations. *Physical Review Letters*, 91(14):147902, 2003.
- [24] Nadav Yoran and Anthony J. Short. Classical simulation of limited-width cluster-state quantum computation. *Physical Review Letters*, 96(17):170503, 2006.