

Diagrammatic Design of Ansätze for Quantum Chemistry



Ayman El Amrani

St. John's College

A thesis submitted for the Honour School of Chemistry

Part II 2024

Pour ma mère et mon père.

Summary

A central challenge in computational quantum chemistry is the accurate simulation of fermionic systems. At the heart of these calculations lies the need to solve the Schrödinger equation to determine the many-electron wavefunction. An exact solution to this problem scales exponentially with the number of electrons. Classical computers have no means by which to efficiently store the increasingly large wavefunctions, making this problem computationally intractable for large and strongly-correlated systems [1]. In contrast, gate-based quantum computing presents a promising solution, offering the potential to represent electronic wavefunctions with polynomially scaling resources using quantum algorithms for the simulation of chemical systems [2]. In other words, quantum computers are a natural tool of choice for simulating processes that are inherently quantum [3].

In the last two decades, many advancements in quantum computing have been made in both hardware and software, bringing us closer to being able to simulate molecular systems. Despite these advancements, we remain in the so-called Noisy Intermediate Scale Quantum (NISQ) era [4], characterised by challenges such as poor qubit fidelity, low qubit connectivity and limited coherence times [5]. The NISQ era represents a transitional phase in quantum computing, where quantum devices are not yet error-corrected but are still capable of performing computations beyond the reach of classical computers. Overcoming the limitations of the NISQ era is crucial for realising the full potential of quantum computing in various fields, including quantum chemistry and materials science.

In this thesis, we focus on the Unitary Coupled Cluster (UCC) ansatz [6], imple-

mented on quantum devices using the Variational Quantum Eigensolver (VQE) algorithm [7]. In particular, we are concerned with the study of the excitation operators used to prepare UCC ansätze representing fermionic wavefunctions. The VQE algorithm is a method used to estimate the ground state energy of a molecular Hamiltonian by preparing a trial wavefunction, calculating its energy expectation value on a quantum device, then optimising the wavefunction parameters classically until the energy converges to the best approximation for the ground state energy [8]. It is recognised as a leading algorithm for quantum simulation on NISQ devices due to its reduced resource requirements in terms of qubit count and coherence time [9].

We build on the work of Yeung [3] on Pauli gadgets, Yordanov *et al* [10] on fermionic excitation operators and Cowtwan *et al* [11], concerning ourselves with two main questions: can we use the ZX calculus to gain insights into the structure of the UCC ansatz in the context of VQE algorithms for quantum chemistry? Secondly, in the context of NISQ devices, can we use these insights to build better ansätze with reduced circuit depth and more efficient resources? By attempting to reduce circuit depth, we are addressing the major source of error present in NISQ devices – the noise of today’s quantum hardware [11].

- **Chapter 1** develops the mathematical foundation for simulating molecules on quantum computers.
- **Chapter 2** introduces the generators of the ZX calculus and its rewrite rules.
- **Chapter 3** introduces Pauli gadgets, the basic building blocks of fermionic ansätze, and their interaction with other quantum gates.
- **Chapter 4** explores controlled rotations in terms of phase polynomials.
- **Chapter 5** applies the theory developed thus far to show how excitation operators can be expressed in terms of controlled rotations in the ZX calculus.
- **Chapter 6** introduces the software package ZxFermion that we built, demonstrating how it can be used to replicate the research done in this thesis.

Contents

1	Background	1
1.1	Context & Motivation	2
1.2	Electronic Structure Theory	4
1.3	Variational Quantum Eigensolver	9
2	ZX Calculus	11
2.1	Generators	12
2.2	Rewrite Rules	16
2.3	Clifford Conjugation	19
3	Pauli Gadgets	20
3.1	Phase Gadgets	21
3.2	Pauli Gadgets	24
3.3	Phase Polynomials	25
3.4	Commutation Relations	26
4	Controlled Rotations	29
4.1	Singly Controlled-Rotations	30
4.2	Higher Order Controlled-Rotations	31
5	Excitation Operators	33
5.1	One Body Excitation Operators	35
5.2	Two Body Excitation Operators	38

Contents

6	ZxFermion Software	40
6.1	Creating Gadgets and Circuits	41
6.2	Manipulating Circuits	42
7	Conclusion	44
7.1	Summary	44
7.2	Future Work	44
Appendices		
Bibliography		48

Chapter 1

Background

1. Background

1.1 Context & Motivation

The Variational Quantum Eigensolver (VQE) is a promising hybrid quantum-classical algorithm for achieving quantum advantage on NISQ devices [12]. Developed in 2014 by Peruzzo and McClean *et al* [13], the VQE algorithm divides the problem of estimating the ground-state energy of a molecule into two parts – computing the energy of some fermionic state on a quantum device, then classically optimising the quantum circuit representing the state until it converges to a good approximation of the true ground state.

VQE algorithms implement fermionic states on quantum devices via the Unitary Coupled Cluster (UCC) ansatz [14]. By preparing quantum states as a sequence of unitary excitation operations acting on some reference state, we define the UCC ansatz [15], allowing us to parametrically explore the Hilbert space of possible quantum states [8].

The Discretely and Continuously Optimized Variational Quantum Eigensolver (DISCO-VQE) is a specific type of VQE developed by Burton *et al.* [15]. This algorithm generates multiple distinct UCC ansätze, each yielding the same energy expectation value. This means that different sequences of unitary excitation operators are employed to rotate the reference state to a state approximating the true ground state. The identical energy expectation values of these states suggests that they equivalently capture the correlation present in the ground state, and that it may be possible to demonstrate the equivalence between these UCC ansätze through algebraic manipulation.

In this context, this thesis focuses on the diagrammatic representation of unitary excitation operators in the ZX calculus, a diagrammatic language for reasoning about quantum processes [16]. Our initial goal was to identify a generalised structure for these excitation operators within the ZX calculus, anticipating that by doing so, we might discover a way of demonstrating the equivalence of different VQE ansätze with the same energy expectation value. Through this, we aimed to uncover novel methods for optimizing ansätze that represent fermionic wavefunctions. Additionally,

1. Background

by developing a representation for these excitation operators that is independent of specific architectural constraints, we sought to gain deeper insights into the structure of UCC ansätze and the nature of correlations in molecular quantum systems. This broader understanding could potentially lead to more efficient and effective quantum simulations.

This led us to the work of Yordanov *et al* [10], which shows that excitation operators can be re-expressed in terms of controlled rotations, and that of Cowtan *et al.* [11], which shows that commuting sets of Pauli gadgets can be diagonalised. Throughout the course of our research, we were able to demonstrate diagrammatically the correspondence between these excitation operators and controlled rotations. Consequently, a significant portion of this thesis revolves around developing the diagrammatic techniques essential for replicating the findings of Yordanov *et al* in the ZX calculus.

In addition to our research goals, this thesis aims to introduce the ZX calculus in the context of quantum chemistry. While quantum chemistry is anticipated to be a principal application of quantum computing, it remains an area with limited engagement among Master’s level researchers in Chemistry. Therefore, we hope that this thesis, along with the tools developed herein, will help lower the barrier to entry for future Master’s students interested in quantum computing. By providing a solid foundation and practical insights, we aim to facilitate a smoother transition and foster greater interest in this rapidly developing field.

1. Use the ZX calculus to gain insights into the structure of UCC ansätze and the nature of correlations in molecular quantum systems.
2. Utilise these insights to rationalise the different outputs of VQE algorithms that yield the same energy.
3. Identify a general representation of excitation operators within the ZX calculus that is independent of specific architectural constraints.
4. Leverage this general representation to discover more efficient implementations of fermionic ansätze in terms of quantum resources.

1. Background

1.2 Electronic Structure Theory

Electronic Structure Problem

The main interest of electronic structure theory is finding approximate solutions to the eigenvalue equation of the full molecular Hamiltonian. Specifically, we seek solutions to the non-relativistic time-independent Schrödinger equation.

$$H = -\sum_{i=1}^N \frac{1}{2} \nabla_i^2 - \sum_{i=1}^M \frac{1}{2M_i} \nabla_i^2 - \sum_{i=1}^N \sum_{j=1}^M \frac{Z_j}{|r_i - R_j|} + \sum_{i=1}^N \sum_{j>i}^N \frac{1}{|r_i - r_j|} + \sum_{i=1}^M \sum_{j>i}^M \frac{Z_i Z_j}{|R_i - R_j|}$$

Figure 1.1: Full molecular Hamiltonian in atomic units, where Z_i is the charge of nucleus i and M_i is its mass relative to the mass of an electron.

The full molecular Hamiltonian, H , describes all of the interactions within a system of N interacting electrons and M nuclei. The first term corresponds to the kinetic energy of all electrons in the system. The second term corresponds to the total kinetic energy of all nuclei. The third term corresponds to the pairwise attractive Coulombic interactions between the N electrons and M nuclei, whilst the fourth and fifth terms correspond to all repulsive Coulombic interactions between electrons and nuclei respectively.

We are able to simplify the problem to an electronic one using the Born-Oppenheimer approximation. Motivated by the large difference in mass of electrons and nuclei, we can approximate nuclei as stationary on the timescale of electronic motion such that the electronic wavefunction depends only parametrically on the nuclear coordinates. Within this approximation, the nuclear kinetic energy term can be neglected and the nuclear repulsive term is considered to be constant. The resulting equation is the electronic Hamiltonian for N electrons.

$$H = -\sum_{i=1}^N \frac{1}{2} \nabla_i^2 - \sum_{i=1}^N \sum_{j=1}^M \frac{Z_j}{|r_i - R_j|} + \sum_{i=1}^N \sum_{j>i}^N \frac{1}{|r_i - r_j|}$$

Figure 1.2: Electronic molecular Hamiltonian in atomic units.

Throughout the remainder of this text, we will concern ourselves only with the

1. Background

electronic Hamiltonian, simply referring to it as the Hamiltonian, H . The solution to the eigenvalue equation involving the electronic Hamiltonian is the electronic wavefunction, which depends only parametrically on the nuclear coordinates. It is solved for fixed nuclear coordinates, such that different arrangements of nuclei yields different functions of the electronic coordinates. The total molecular energy can then be calculated by solving the electronic Schrödinger equation and including the constant repulsive nuclear term.

Many-Electron Wavefunctions

The many-electron wavefunction, which describes all fermions in given molecular system, must satisfy the Pauli principle. This is an independent postulate of quantum mechanics that requires the many-electron wavefunction to be antisymmetric with respect to the exchange of any two fermions.

A spatial molecular orbital is defined as a one-particle function of the position vector, spanning the whole molecule. The spatial orbitals form an orthonormal set $\{\psi_i(\mathbf{r})\}$, which if complete can be used to expand any arbitrary single-particle molecular wavefunction, that is, an arbitrary single-particle function of the position vector. In practice, only a finite set of such orbitals is available to us, spanning only a subspace of the complete space. Hence, wavefunctions expanded using this finite set are described as being ‘exact’ only within the subspace that they span.

We will now introduce the spin orbitals $\{\phi_i(\mathbf{x})\}$, that is, the set of functions of the composite coordinate \mathbf{x} , which describes both the spin and spatial distribution of an electron. Given a set of K spatial orbitals, we can construct $2K$ spin orbitals by taking their product with the orthonormal spin functions $\alpha(\omega)$ and $\beta(\omega)$. Whilst the Hamiltonian operator makes no reference to spin, it is a necessary component when constructing many-electron wavefunctions in order to correctly antisymmetrise the wavefunction with respect to fermion exchange. Constructing the antisymmetric many-electron wavefunction from a finite set of spin orbitals amounts to taking the appropriate linear combinations of symmetric products of N spin orbitals.

1. Background

A general procedure for this is achieved by constructing a Slater determinant from the finite set of spin orbitals, where each row relates to the electron coordinate \mathbf{x}_n and each column corresponds to a particular spin orbital ϕ_i [17].

$$\psi(\mathbf{x}_1, \mathbf{x}_2) = \frac{1}{\sqrt{N!}} \begin{vmatrix} \phi_i(\mathbf{x}_1) & \phi_j(\mathbf{x}_1) & \dots & \phi_k(\mathbf{x}_1) \\ \phi_i(\mathbf{x}_2) & \phi_j(\mathbf{x}_2) & \dots & \phi_k(\mathbf{x}_2) \\ \vdots & \vdots & & \vdots \\ \phi_i(\mathbf{x}_N) & \phi_j(\mathbf{x}_N) & \dots & \phi_k(\mathbf{x}_N) \end{vmatrix}$$

Figure 1.3: Slater determinant representing an antisymmetrised N -electron wavefunction.

By constructing Slater determinants and antisymmetrising the many-electron wavefunction to meet the requirements of the Pauli principle, we have incorporated exchange correlation, in that, the motion of any two electrons with parallel spins is now correlated [17].

The Hartree-Fock method yields a set of orthonormal spin orbitals, which when used to construct a single Slater determinant, gives the best variational approximation to the ground state of a system [17]. By treating electron-electron repulsion in an average way, the Hartree-Fock approximation allows us to iteratively solve the Hartree-Fock equation for spin orbitals until they become the same as the eigenfunctions of the Fock operator. This is known as the Self-Consistent Field (SCF) method and is an elegant starting point for finding approximate solutions to the many-electron wavefunction.

For an N electron system, and given a set of $2K$ Hartree-Fock spin orbitals, where $2K > N$, there exist many different single Slater determinants. The Hartree-Fock groundstate being one of these. The remainder are excited Slater determinants, recalling that all of these must be orthogonal to one-another. By treating the Hartree-Fock ground state as a reference state, we can describe the excited states relative to the reference state, as single, double, \dots , N -tuple excited states [17].

1. Background

Second Quantisation

In second quantisation, both observables and states (by acting on the vacuum state) are represented by operators, namely the creation and annihilation operators [18]. In contrast to the standard formulation of quantum mechanics, operators in second quantisation incorporate the relevant Bose or Fermi statistics each time they act on a state, circumventing the need to keep track of symmetrised or antisymmetrised products of single-particle wavefunctions [19]. Put differently, the antisymmetry of an electronic wavefunction simply follows from the algebra of the creation and annihilation operators, which greatly simplifies the discussion of systems of many identical interacting fermions [18], [19].

The Fock space is a linear abstract vector space spanned by N orthonormal occupation number vectors, each representing a single Slater determinant [18]. Hence, given a basis of N spin orbitals we can construct 2^N single Slater determinants, each corresponding to a single occupation number vector in the full Fock space. The occupation number vector for fermionic systems is succinctly denoted in Dirac notation as below, where the occupation number f_j is 1 if spin orbital j is occupied, and 0 if spin orbital j is unoccupied.

$$|\psi\rangle = |f_{n-1} f_{n-2} \dots f_1 f_0\rangle \quad \text{where } f_j \in 0, 1$$

Whilst there is a one-to-one mapping between Slater determinants with canonically ordered spin orbitals and the occupation number vectors in the Fock space, it is important to distinguish between the two since, unlike the Slater determinants, the occupation number vectors have no spatial structure and are simply vectors in an abstract vector space [18].

Operators in second quantisation are constructed from the creation and annihilation operators a_j^\dagger and a_j , where the subscripts i and j denote the spin orbital. a_j^\dagger and a_j are one another's Hermitian adjoints, and are not self-adjoint [18]. Taking the excitation of an electron from spin orbital 0 to spin orbital 1 as an example, we can

1. Background

construct the following excitation operator.

$$a_1^\dagger a_0 |0 \dots 01\rangle = |0 \dots 10\rangle$$

Due to the fermionic exchange anti-symmetry imposed by the Pauli principle, the action of the creation and annihilation operators introduces a phase to the state that depends on the parity of the spin orbitals preceding the target spin orbital.

$$\begin{aligned} a_j^\dagger |f_{n-1} \dots f_{j+1}, 0, f_{j-1} \dots f_0\rangle &= (-1)^{\sum_{s=0}^{j-1} f_s} |f_{n-1} \dots f_{j+1}, 1, f_{j-1} \dots f_0\rangle \\ a_j |f_{n-1} \dots f_{j+1}, 1, f_{j-1} \dots f_0\rangle &= (-1)^{\sum_{s=0}^{j-1} f_s} |f_{n-1} \dots f_{j+1}, 0, f_{j-1} \dots f_0\rangle \end{aligned}$$

In second quantisation, this exchange anti-symmetry requirement is accounted for by the anti-commutation relations of the creation and annihilation operators.

$$\{\hat{a}_j, \hat{a}_k\} = 0 \quad \{\hat{a}_j^\dagger, \hat{a}_k^\dagger\} = 0 \quad \{\hat{a}_j, \hat{a}_k^\dagger\} = \delta_{jk} \hat{1}$$

The phase factor required for the second quantised representation to be consistent with the first quantised representation is automatically kept track of by the anticommutation relations of the creation and annihilation operators [18].

With these creation and annihilation operators in mind, the Hamiltonian in second quantisation can be expressed as follows.

$$\hat{H} = \sum_{ij} h_{ij} a_i^\dagger a_j + \frac{1}{2} \sum_{ijkl} h_{ijkl} a_i^\dagger a_j^\dagger a_k a_l + h_{\text{Nu}}$$

Where the one-body matrix element h_{ij} corresponds to the kinetic energy of an electron and its interaction energy with the nuclei, and the two-body matrix element h_{ijkl} corresponds to the repulsive interaction between electrons i and j .

$$\begin{aligned} h_{ij} &= \int_{-\infty}^{\infty} \psi_{i(x_1)}^* \left(-\frac{1}{2} \nabla^2 + \hat{V}_{(x_1)} \right) \psi_{j(x_1)} d^3 x_1 \\ h_{ijkl} &= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \psi_{i(x_1)}^* \psi_{j(x_2)}^* \left(\frac{1}{|x_1 - x_2|} \right) \psi_{k(x_2)} \psi_{l(x_1)} d^3 x_1 d^3 x_2 \end{aligned}$$

h_{Nu} is a constant corresponding to the repulsive interaction between nuclei. These matrix elements are computed classically, allowing us to simulate only the inherently quantum aspects of the problem on a quantum computer.

1. Background

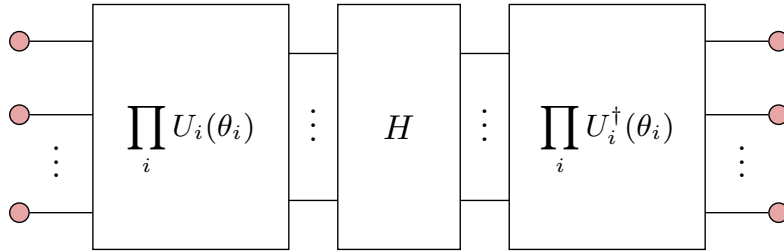
1.3 Variational Quantum Eigensolver

The Variational Quantum Eigensolver (VQE) algorithm is a particular class of variational quantum algorithm that is used to estimate the ground state energy of molecular quantum systems. It consists of a quantum subroutine, a parametrised quantum circuit that implements some UCC ansatz representing a fermionic wavefunction, and a classical subroutine that classically optimises the UCC ansatz until it converges to the best approximation of the true ground state.

Parametrised quantum circuits are similar to classical neural networks in concept, but by definition must have the same number of inputs and outputs. This requirement stems from the fact that all quantum gates must be unitary, where an n qubit quantum neural network represents a $2^n \times 2^n$ unitary map [3].

The input state for the parametrised quantum circuit is the reference state that the UCC operator $U(\boldsymbol{\theta})$ acts on, and in the case of the single Slater determinant Hartree-Fock state, is encoded as a pure quantum state $|\psi_0\rangle$. The output of the parametrised quantum circuit is an entangled state representing the UCC ansatz. That is, the UCC ansatz $|\psi(\boldsymbol{\theta})\rangle$ represents some linear combination of vectors in the Fock space, approximating the correlation present in the true ground state.

Upon measuring the output of the parametrised quantum circuit, it collapses into a single vector in the Fock space with a probability defined by that vector's weight in the UCC ansatz. The quantum subroutine computes the energy expectation value of the UCC ansatz using a quantum circuit consisting of the parametrised quantum circuit and the Hamiltonian for the system.



$$E(\boldsymbol{\theta}) = \langle 0 | U^\dagger(\boldsymbol{\theta}) H U(\boldsymbol{\theta}) | 0 \rangle$$

1. Background

Unitary Coupled Cluster Ansatz

As suggested by Peruzzo *et al* [13], the Unitary Coupled Cluster (UCC) formulation of a wavefunction can be efficiently implemented on a quantum computer using quantum gates. We will refer to this implementation as the UCC ansatz $|\psi(\boldsymbol{\theta})\rangle$, defining it as some unitary excitation operator $U(\boldsymbol{\theta})$ acting on a reference state.

$$|\psi(\boldsymbol{\theta})\rangle = U(\boldsymbol{\theta}) |\psi_0\rangle = e^{\hat{T}(\boldsymbol{\theta}) - \hat{T}^\dagger(\boldsymbol{\theta})} |\psi_0\rangle$$

Where $|\psi_0\rangle$ is a single reference Slater determinant, usually the Hartree-Fock groundstate obtained via the self-consistent field method. The operator $\hat{T}(\boldsymbol{\theta})$ is a linear combination of fermionic excitation operators, parametrised by coupled cluster amplitudes $\boldsymbol{\theta}$. The exponential of the anti-Hermitian linear combination, $\hat{T}(\boldsymbol{\theta}) - \hat{T}^\dagger(\boldsymbol{\theta})$, is therefore, by definition, unitary.

$$\hat{T}(\boldsymbol{\theta}) - \hat{T}^\dagger(\boldsymbol{\theta}) = \sum_{i,a} \theta_i^a (a_i^\dagger a_a - a_a^\dagger a_i) + \sum_{i,j,a,b} \theta_{ij}^{ab} (a_i^\dagger a_j^\dagger a_a a_b - a_a^\dagger a_b^\dagger a_i a_j) + \dots$$

Where i, j indexes occupied spin orbitals and a, b indexes virtual, or unoccupied, spin orbitals. The resulting unitary operator $U(\boldsymbol{\theta})$ cannot be directly implemented on a quantum computer since the terms of the excitation operator do not commute. Instead, we must invoke the Trotter formula to approximate the unitary. Taking a single Trotter step $\rho = 1$, since our focus is on the NISQ setting [11], we define the UCC ansatz as the following product of k parametrised unitary operators.

$$|\psi(\boldsymbol{\theta})\rangle = \prod_{m=1}^k U_m(\theta_m) |\psi_0\rangle \quad U_m(\theta_m) = e^{\theta_m(\tau_m - \tau_m^\dagger)}$$

Where m indexes all possible excitations and $\tau_m - \tau_m^\dagger$ corresponds to our anti-Hermitian fermionic excitation operators. It has been shown by Evangelista *et al* [20] that this UCC operator can exactly parametrise any state. In practice, we truncate the possible fermionic excitations to include only single and double excitations, yielding the popular UCCSD ansatz [21].

Chapter 2

ZX Calculus

In this chapter, we will introduce the ZX calculus, a diagrammatic language for reasoning about quantum processes first introduced by Coecke *et al* [16]. We will introduce the basic generators of the ZX calculus as well as the relevant rewrite rules.

This thesis uses the scalar-free ZX calculus. That is, the derivations in this thesis are correct up to some global non-zero scalar factor. All equal signs should, therefore, be interpreted as ‘equal up to a global phase’. This is done for convenience, in a similar way to how quantum chemists sometimes work with unnormalised wavefunctions. Recalling that the matrix representing our quantum circuit M is proportional to some unitary, $M^{-1} = M^\dagger$, we can efficiently compute the scalar factor by composing a given ZX diagram with its adjoint and simplifying it until it reduces to identity [22].

Note that all of the definitions in this chapter also hold for their colour-swapped counterparts, which we have chosen to omit for brevity.

2.1 Generators

By sequentially or horizontally composing the *Z Spider* (green) and *X Spider* (red) generators, we can construct undirected multigraphs known as ZX diagrams [22]. That is, graphs that allow multiple edges between vertices. Since *only connectivity matters* in the ZX calculus, a valid ZX diagram can be arbitrarily deformed (d), provided that the order of inputs and outputs is preserved.

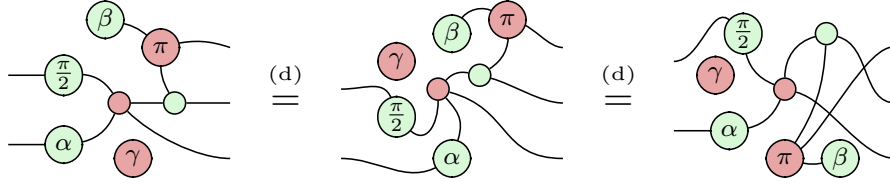


Figure 2.1: Three equivalent ZX diagrams (*only connectivity matters*).

Notation – We interpret the flow of time left to right. Hence, the wires on the left refer to inputs and the wires on the right refer to outputs.

Z Spiders (green) are defined with respect to the *Z* eigenbasis, $|0\rangle$ and $|1\rangle$, such that a *Z Spider* with n inputs and m outputs represents the following linear map.

$$n \begin{array}{c} \vdots \\ \text{---} \end{array} \begin{array}{c} \diagup \\ \text{---} \end{array} \begin{array}{c} \diagdown \\ \text{---} \end{array} m = |0\rangle^{\otimes m} \langle 0|^{\otimes n} + e^{i\alpha} |1\rangle^{\otimes m} \langle 1|^{\otimes n}$$

Figure 2.2: Interpretation of a *Z Spider* as a linear map.

X Spiders (red), are defined with respect to the *X* eigenbasis, $|+\rangle$ and $|-\rangle$.

$$n \begin{array}{c} \vdots \\ \text{---} \end{array} \begin{array}{c} \diagup \\ \text{---} \end{array} \begin{array}{c} \diagdown \\ \text{---} \end{array} m = |+\rangle^{\otimes m} \langle +|^{\otimes n} + e^{i\alpha} |-\rangle^{\otimes m} \langle -|^{\otimes n}$$

Figure 2.3: Interpretation of an *X Spider* as a linear map.

We can represent the *Z* eigenstates using an *X* spider with a phase of 0 or π .

$$\text{---} \bigcirc = |+\rangle + |-\rangle = \sqrt{2} |0\rangle \quad \text{---} \bigcirc_{\pi} = |+\rangle - |-\rangle = \sqrt{2} |1\rangle$$

Figure 2.4: $|0\rangle$ eigenstate

Figure 2.5: $|1\rangle$ eigenstate

2. ZX Calculus

Similarly, we represent the X eigenstates using the corresponding Z spiders.

$$\text{---} \bigcirc \text{---} = |0\rangle + |1\rangle = \sqrt{2} |+\rangle$$

Figure 2.6: $|+\rangle$ eigenstate

$$\text{---} \bigcirc^\pi \text{---} = |0\rangle - |1\rangle = \sqrt{2} |-\rangle$$

Figure 2.7: $|-\rangle$ eigenstate

Single qubit rotations in the Z basis are represented by a Z Spider with a single input and a single output. Arbitrary rotations in the X basis are represented by the corresponding X spider. We can view these as rotations of the Bloch sphere.

$$\begin{aligned} \text{---} \bigcirc^\alpha \text{---} &= |0\rangle\langle 0| + e^{i\alpha} |1\rangle\langle 1| = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\alpha} \end{pmatrix} \rightarrow \text{Bloch sphere with rotation around Z-axis} \\ \text{---} \bigcirc^\alpha \text{---} &= |+\rangle\langle +| + e^{i\alpha} |-\rangle\langle -| = \frac{1}{2} \begin{pmatrix} 1 + e^{i\alpha} & 1 - e^{i\alpha} \\ 1 - e^{i\alpha} & 1 + e^{i\alpha} \end{pmatrix} \rightarrow \text{Bloch sphere with rotation around X-axis} \end{aligned}$$

We can recover the Pauli Z and Pauli X matrices by setting the angle $\alpha = \pi$.

$$\text{---} \bigcirc^\pi \text{---} = |0\rangle\langle 0| + e^{i\pi} |1\rangle\langle 1| = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

$$\text{---} \bigcirc^\pi \text{---} = |+\rangle\langle +| + e^{i\pi} |-\rangle\langle -| = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$

Figure 2.8: Pauli Z and X gates in the ZX calculus.

Composition

To calculate the matrix of a ZX diagram consisting of sequentially composed spiders, we take the matrix product. Note that the order of operation of matrix multiplication is the reverse of how we have defined it for ZX diagrams.

$$\text{---} \bigcirc^\alpha \text{---} \bigcirc^\beta \text{---} \bigcirc^\gamma \text{---} = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\gamma} \end{pmatrix} \begin{pmatrix} 1 + e^{i\beta} & 1 - e^{i\beta} \\ 1 - e^{i\beta} & 1 + e^{i\beta} \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & e^{i\alpha} \end{pmatrix}$$

2. ZX Calculus

Alternatively, we could have chosen to compose the spiders in parallel, resulting in the tensor product.

$$\begin{array}{c} \textcircled{\alpha} \\ \textcircled{\beta} \end{array} = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\alpha} \end{pmatrix} \otimes \begin{pmatrix} 1 + e^{i\beta} & 1 - e^{i\beta} \\ 1 - e^{i\beta} & 1 + e^{i\beta} \end{pmatrix}$$

The CNOT gate in the ZX calculus is represented by a Z spider (control qubit) and an X spider (target qubit). We can arbitrarily deform the diagram and decompose it into matrix and tensor products as follows.

We can calculate matrix A , consisting of a single-input and two-output Z Spider (4×2 matrix) and an empty wire (identity matrix), by taking the tensor product.

$$\text{Box } A \text{ with 4 wires} = \text{Diagram with 4 wires and a green circle} = \begin{pmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 1 \end{pmatrix} \otimes \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

Similarly, to calculate the matrix B , we take the following tensor product.

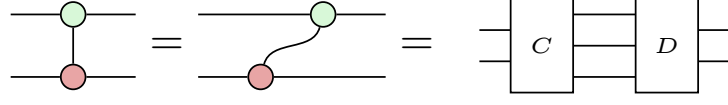
$$\text{Diagram of } B = \text{Diagram of } \text{CNOT} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \otimes \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix}$$

We can then calculate the CNOT matrix by taking the matrix product of matrix A and matrix B as follows.

$$\text{Diagram} = \left[\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \otimes \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix} \right] \left[\begin{pmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 1 \end{pmatrix} \otimes \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \right] \simeq \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

Since *only connectivity matters* (2.1), we could have equivalently calculated the matrix of the CNOT gate by deforming the diagram as follows.

2. ZX Calculus



Had we chosen to make the first qubit the target and the second qubit the control, we would have obtained the following.

$$\begin{array}{c} \text{---} \text{red circle} \text{---} \\ | \\ \text{---} \text{green circle} \text{---} \end{array} = \left[\frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix} \otimes \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \right] \left[\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \otimes \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \right] \simeq \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}$$

Hadamard Generator

All quantum gates are unitary transformations. Therefore, up to a global phase, an arbitrary single qubit rotation U can be viewed as a rotation of the Bloch sphere about some axis. We can decompose the unitary U using Euler angles to represent the rotation as three successive rotations [22].

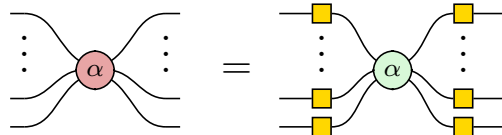


Figure 2.9: Arbitrary single-qubit rotation.

The Hadamard gate H corresponds to a rotation of the Bloch sphere π radians about the line bisecting the X and Z axes. Up to a global phase of $\exp(-i\frac{\pi}{4})$, it can be decomposed using Euler angles by choosing $\alpha = \beta = \gamma = \frac{\pi}{2}$.

$$\text{---} \text{yellow square} \text{---} = e^{-i\frac{\pi}{4}} \text{---} \text{green circle } \frac{\pi}{2} \text{---} \text{red circle } \frac{\pi}{2} \text{---} \text{green circle } \frac{\pi}{2} \text{---} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$$

There are many equivalent ways of decomposing the Hadamard gate H using Euler angles (see Appendix 7.2). The Hadamard gate switches between the Z and X bases. Hence, diagrammatically, applying Hadamard generators to all of the legs of a spider changes the colour of the spider.



2.2 Rewrite Rules

Notation – We will refer to the rules by some shorthand notation above equal signs.

Note that this could refer to applying the rule in either direction.

Spider Fusion

The most fundamental rule of the ZX calculus is the *spider fusion rule* (f). It states that spiders of the same colour connected by one or more wires fuse and their phases add modulo 2π [22].

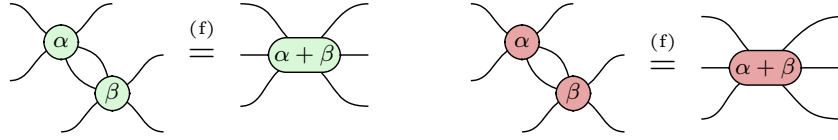
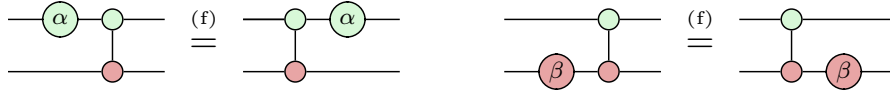


Figure 2.10: Spider fusion rule for Z spiders (left) and X spiders (right).

It is the generalisation of adding the phases of successive rotations of the Bloch sphere. We can use this rule to show that Z rotations commute through CNOT controls, and that X rotations commute through CNOT targets.



Identity Removal

The *identity rule* (id) states that any two-legged spider with no phase ($\alpha = 0$) is equivalent to a rotation by 0 radians, or identity.

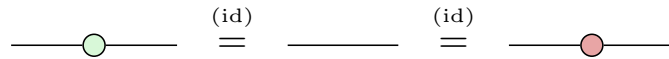
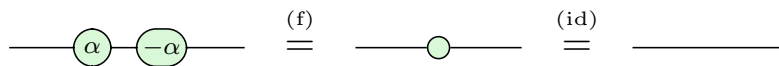


Figure 2.11: Identity removal rule.

Combining this with the spider fusion rule (2.10), we see that two successive rotations with opposite phases is equivalent to an empty wire.



State Copy and π Copy Rules

We can depict the Z and X eigenstates by assigning a phase to a Z or an X spider, respectively, through a Boolean variable a (0 or 1) multiplied by π [22].

$$\textcircled{a\pi} \text{---} = |0\rangle \text{ where } a = 0 \text{ and } |1\rangle \text{ where } a = 1$$

$$\textcircled{a\pi} \text{---} = |+\rangle \text{ where } a = 0 \text{ and } |-\rangle \text{ where } a = 1$$

The π copy rule (c) states that when a Pauli Z or Pauli X gate is pushed through a spider of the opposite colour, it is copied on all other legs and negates the spider's phase. A similar *state copy rule* (c) applies to the Z and X eigenstates.

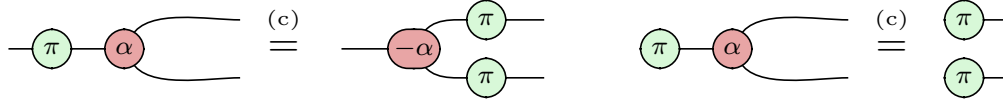


Figure 2.12: π copy (left) and state copy (right) rules for Pauli Z gate and Z eigenstates.

Using the π copy, spider fusion (2.10) and identity (2.11) rules, we show that conjugating a rotation by Pauli gates in the opposite basis negates the phase.

$$\textcircled{\pi} \text{---} \textcircled{\alpha} \textcircled{\pi} \text{---} = \textcircled{-\alpha} \text{---} \quad \textcircled{\pi} \text{---} \textcircled{\alpha} \textcircled{\pi} \text{---} = \textcircled{-\alpha} \text{---}$$

Hadamard Rules

Using that the Hadamard gate is both unitary and Hermitian, we define the *Hadamard self-inverse rule* (hi).

$$\text{---} \textcircled{\square} \text{---} \textcircled{\square} \text{---} \stackrel{(hi)}{=} \text{---}$$

Figure 2.13: Hadamard self-inverse rule.

Recalling that the Hadamard generator changes the colour of a spider and is self-inverse, we define the *Hadamard commutation rule* (hc).

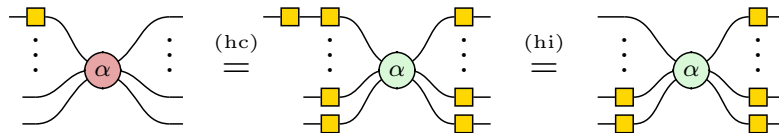


Figure 2.14: Hadamard commutation rule.

Bialgebra Rule

Using the spider fusion rule (2.10), we can show that a spider with two inputs and one output behaves like a classical XOR gate when applied to the eigenstates of the *same* basis. Whilst using the state copy rule (2.12), we can show that a spider with one input and two outputs behaves like a classical COPY gate when applied to the eigenstates of the *opposite* basis.



Figure 2.15: XOR gate (left) and COPY gate (right) with respect to the Z eigenstates.

Using the natural commutation relation of the classical XOR and COPY gates, we define the *bialgebra rule* (ba). We encourage the reader to verify this relation.

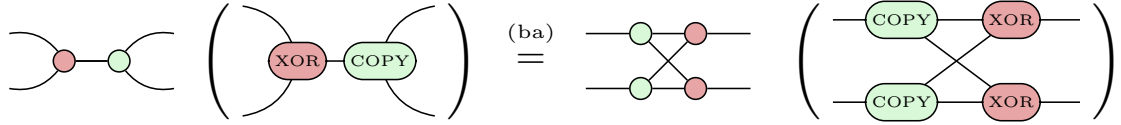


Figure 2.16: The bialgebra rule (and its classical motivation).

Hopf Rule

Like with the bialgebra rule, our motivation for this rule stems from the behaviour of the classical XOR and COPY gates. Since copying two bits then taking their XOR invariably yields 0, we can define the *Hopf rule* (hpf).

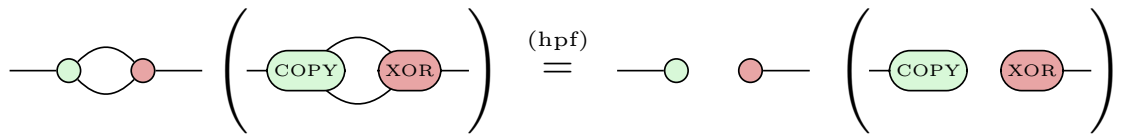
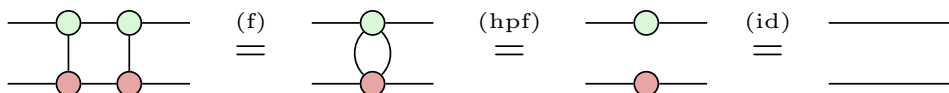


Figure 2.17: The Hopf rule (and its classical motivation).

Recall that the CNOT gate is both unitary and Hermitian, and therefore, self-inverse. The Hopf rule allows us to prove this diagrammatically as follows.



Chapter 3

Pauli Gadgets

Pauli gadgets form the building blocks for UCC ansätze representing fermionic systems in VQE algorithms. We will see in Chapter 5 how they can be used to construct excitation operators in UCC ansätze.

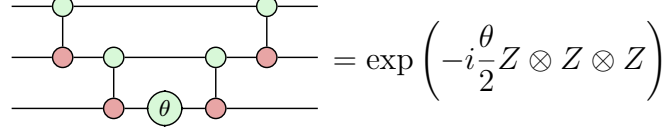
A Pauli string P is defined as a tensor product of Pauli matrices $P \in \{I, X, Y, Z\}^{\otimes n}$, where n is the number of qubits in the system. Each Pauli gate acts on a distinct qubit. Thus $Z \otimes X$ represents the Pauli Z and X gates acting on the first and second qubits respectively.

Stone's Theorem [23] states that a strongly-continuous one parameter unitary group $U(\theta) = \exp\left(-i\frac{\theta}{2}H\right)$ is generated by the Hermitian operator H . The Pauli matrices, and consequently Pauli strings, are Hermitian. We use the one-to-one correspondence between Hermitian operators and one parameter unitary groups to define Pauli gadgets as the one parameter unitary groups generated by Pauli strings.

$$\Phi_1(\theta) = \exp\left(-i\frac{\theta}{2}Z \otimes I \otimes Z\right) \quad \Phi_2(\theta) = \exp\left(-i\frac{\theta}{2}Y \otimes Z \otimes X\right)$$

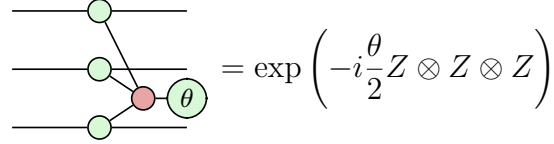
3.1 Phase Gadgets

Phase gadgets are defined as the one parameter unitary groups of Pauli strings consisting of only the Pauli I and Z matrices, $P \in \{I, Z\}^{\otimes n}$. They can be naively implemented as a Z rotation sandwiched between two ladders of CNOT gates.



$$= \exp \left(-i \frac{\theta}{2} Z \otimes Z \otimes Z \right)$$

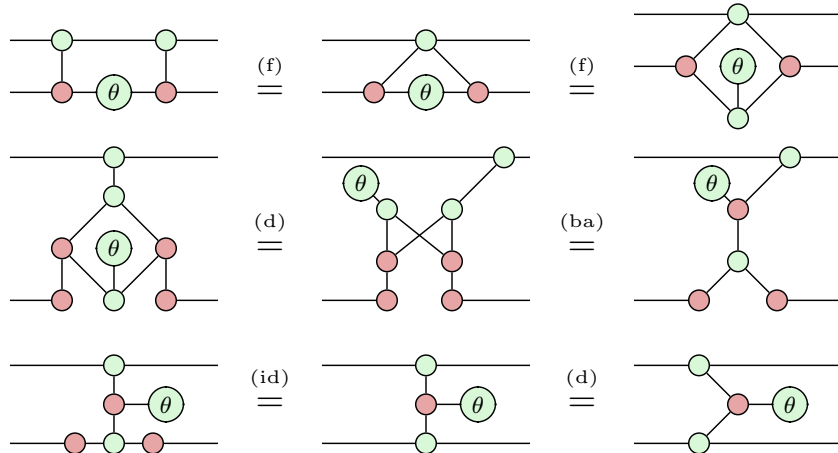
Phase gadgets correspond to unitary maps which are diagonal in the Z computational basis [11]. Consequently, they apply a global phase to a state without changing the distribution of the observed state [3]. Phase gadgets have the following representation in the ZX calculus.



$$= \exp \left(-i \frac{\theta}{2} Z \otimes Z \otimes Z \right)$$

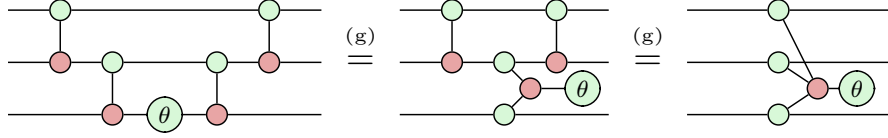
Phase gadgets can be interpreted as first copying each input in the Z basis (2.15), computing the parity of the state by taking the XOR (2.15), then multiplying the state by $\exp \left(-i \frac{\theta}{2} \right)$ or $\exp \left(i \frac{\theta}{2} \right)$ depending on the state's parity [3].

By deforming the phase gadget in quantum circuit notation and using the identity (2.11), spider fusion (2.10) and bialgebra (2.16) rules, we are able to show the correspondence with its form in the ZX calculus – *phase gadget result* (g).



3. Pauli Gadgets

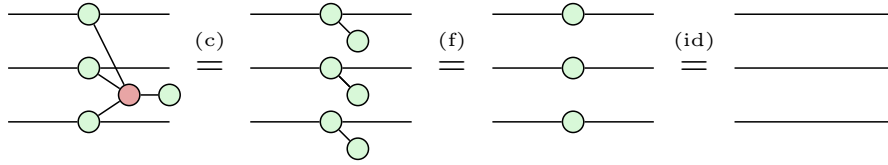
It is then a simple matter of recursively applying this result to phase gadgets in quantum circuit notation to generalise to arbitrary arity.



As well as being intuitively self-transpose, and hence diagonal, this representation comes equipped with various rules describing the interactions of phase gadgets.

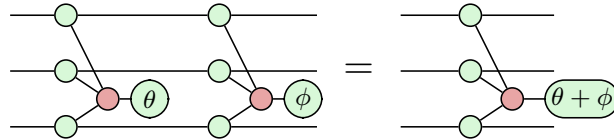
Phase Gadget Identity

Phase gadgets with an angle $\theta = 0$ can be shown to be equivalent to identity using the state copy (2.12), spider fusion (2.10) and identity removal (2.11) rules.



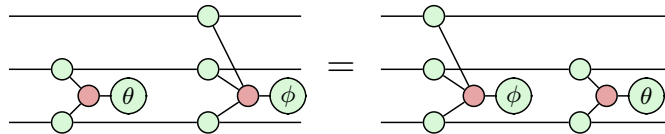
Phase Gadget Fusion

Any two adjacent phase gadgets formed from the same Pauli string fuse and their phases add. This is achieved using the spider fusion rule (2.10) and the bialgebra rule (2.16). See Appendix 7.2 for the intermediate steps.



Phase Gadget Commutation

Phase gadgets can be shown to commute using the spider fusion rule (2.10).



3. Pauli Gadgets

Phase Gadget Decomposition

Using the bialgebra rule (2.16), we can show that a two-legged phase gadget can be decomposed in the following two ways.

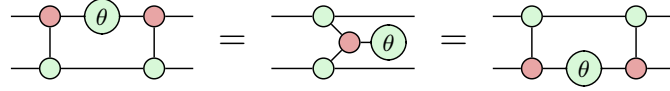
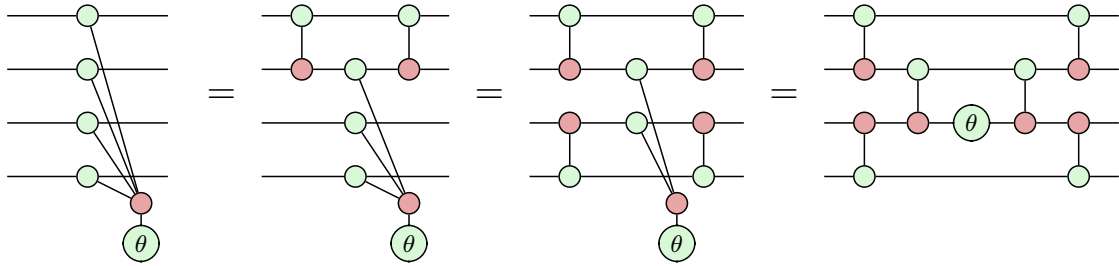


Figure 3.1: Phase gadget decomposition result.

By recursively applying this decomposition result, we can show that it is possible to decompose a phase gadget such that it has a circuit depth of $2 \log_2(n)$, in the balanced tree representation, instead of $2(n-1)$ in the CNOT ladder decomposition, where n is the number of qubits [24].



Phase gadgets can be thought of as the many-qubit generalisation of a rotation in the Z basis [3]. For instance, using the bialgebra (2.16), spider fusion (2.10), state copy (2.12) and identity (2.11) rules, we can demonstrate the correspondence between single-legged phase gadgets and Z rotations.

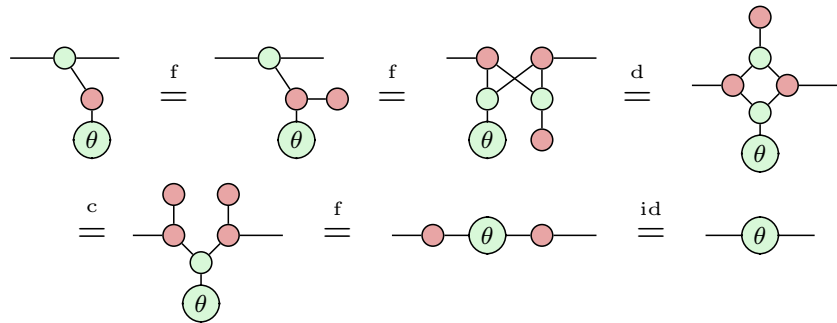


Figure 3.2: Single-legged phase gadget as a Z rotation.

3.2 Pauli Gadgets

Pauli gadgets are defined as the one parameter unitary groups of Pauli strings consisting of all four Pauli matrices, $P \in \{I, X, Y, Z\}^{\otimes n}$ [3]. They are phase gadgets associated with a change of basis.

$$\text{Circuit with Pauli string and phase gadget} = \text{Circuit with single Pauli gadget} = \exp\left(-i\frac{\theta}{2}Y \otimes Z \otimes X\right)$$

Whilst phase gadgets alone cannot change the distribution of the observed state, Pauli gadgets are able to do so [3]. In chapter 5 we will see how Pauli gadgets form the building blocks in ansätze used for quantum chemical simulations.

Pauli gadgets come equipped with a similar set of rules to phase gadgets that describe their interactions with other gadgets. For instance, adjacent Pauli gadgets with *matching legs* fuse, and their phases add modulo 2π .

$$\text{Two adjacent Pauli gadgets} = \text{Fused Pauli gadget with phase } \theta + \phi$$

Figure 3.3: Pauli gadget fusion rule.

Similar to the phase gadget commutation rule (3.1), we have that adjacent Pauli gadgets with *no mismatching legs* commute.

$$\text{Circuit 1} = \text{Circuit 2}$$

Figure 3.4: Pauli gadget commutation rule.

Single-legged Pauli gadgets correspond to rotations in their respective basis.

$$\text{Single-legged Pauli gadget} = \text{Rotation gate}$$

3.3 Phase Polynomials

Recalling that phase gadgets are diagonal in the computational Z basis, we call a set of Pauli gadgets diagonal when it consists only of phase gadgets [11]. Such sets of phase gadgets are known as *phase polynomials* and are themselves diagonal in the computational Z basis [24]. Phase polynomial synthesis then refers to finding a quantum circuit that optimally implements some phase polynomial. There are several well-known phase polynomial synthesis algorithms [25], [26], [27], [28].

As in Cowtan *et al* [11], a set of Pauli gadgets S can be simultaneously diagonalised by a Clifford subcircuit C when all of the Pauli gadgets in the set commute. That is, by conjugating a set of commuting Pauli gadgets, we can re-express the circuit as some phase polynomial that can later be synthesised in some optimal way.

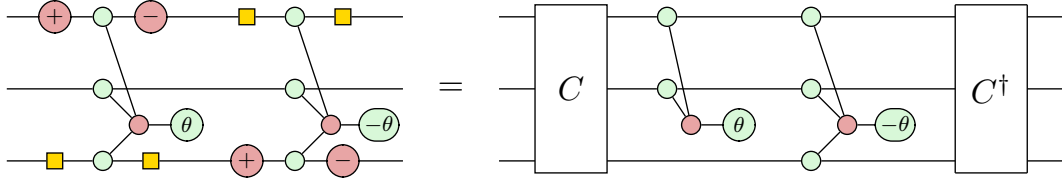
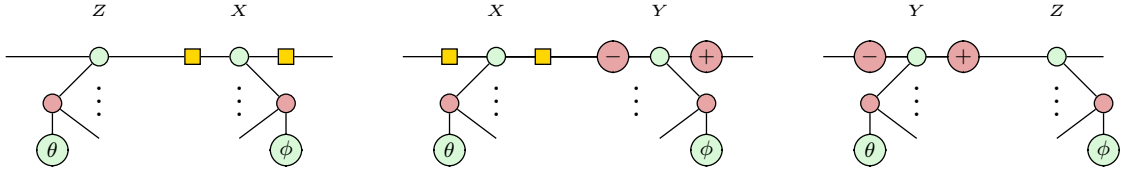


Figure 3.5: Diagonalisation of a pair of commuting Pauli gadgets by C .

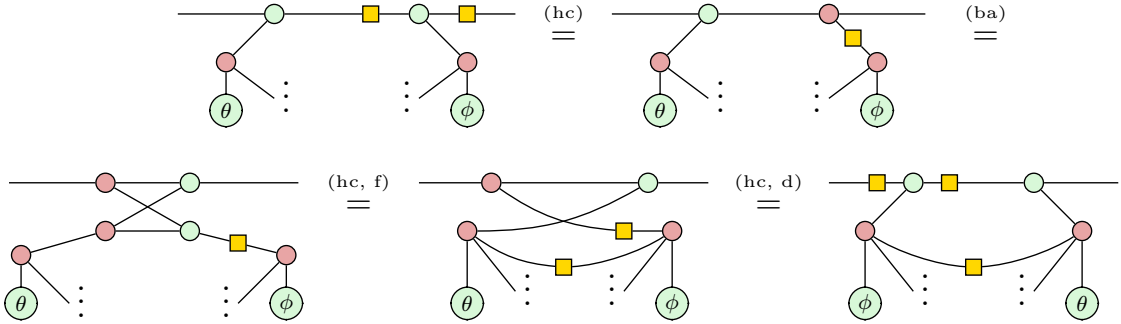
As we will see in Chapter 5, the excitation operators used in VQE algorithms consist of commuting sets of Pauli gadgets. Therefore, the quantum circuits implementing such excitation operators can be diagonalised with some Clifford subcircuit C . As stated in Cowtan *et al* [11], whilst diagonalisation may incur gate overhead, in practice, the reduction in circuit depth arising from synthesising the resulting phase polynomial usually more than makes up for the overhead.

3.4 Commutation Relations

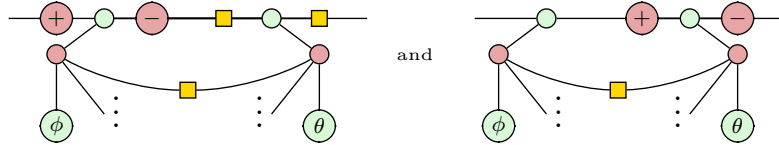
A pair of Pauli gadgets commute when their Hamiltonians commute [3]. That is, a pair of Pauli gadgets commute when the Pauli strings that they are defined by also commute. Diagrammatically, we have that a pair of Pauli gadgets commute when they *mismatch on an even number of legs* [3]. Let us demonstrate this by first considering the three possible mismatching pairs of Pauli gadget legs.



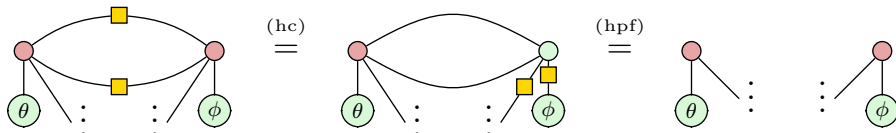
Starting with the mismatching Z/X pair and using the bialgebra (2.16), Hadamard commutation (2.14) and spider fusion (2.10) rules, we can show that commuting the gadgets' legs introduces a Hadamard between the bodies of the gadgets. [3].



Similarly, for the X/Y and Y/Z mismatching pairs, we have the following.



Therefore, we can show that two Pauli gadgets with an even number of mismatching legs commute by first commuting all of their legs, then using the Hopf rule (2.17) to remove the wires between them.



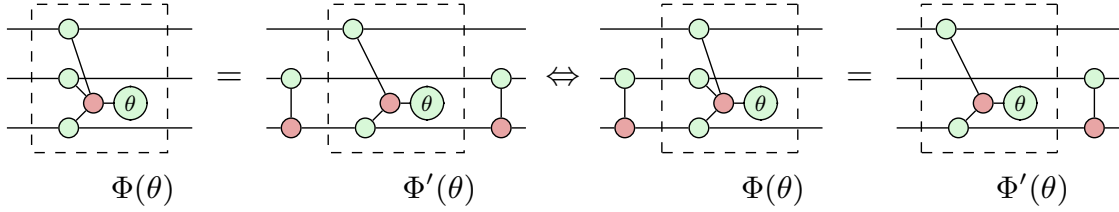
3. Pauli Gadgets

Clifford Commutation Relations

We will now develop a set of *commutation relations* describing the interaction of Pauli gadgets with members of the Clifford group. Diagrammatically, we interpret this as ‘what happens when a Clifford gate is pushed through a Pauli gadget’.

By definition, conjugating a Pauli string by a member of the Clifford group, $C^\dagger P C$, is closed in the set of Pauli strings, $\{I, X, Y, Z\}^{\otimes n}$, where n is the number of qubits that the Pauli string acts on. Similarly, conjugating a Pauli gadget $\Phi(\theta)$ by a member of the Clifford group always yields another Pauli gadget $\Phi'(\theta) = C^\dagger \Phi(\theta) C$.

We can interpret the conjugation of *phase gadgets* by CNOT gates diagrammatically as the phase gadget decomposition result (see diagram on the left). Recalling that the members of the Clifford group are unitary transformations, $C^{-1} = C^\dagger$, we can define *commutation relation* as $\Phi(\theta)C = C\Phi'(\theta)$ (see diagram on the right).



Since the CNOT gate acts on two qubits ($n = 2$), we can form 16 unique Pauli gadgets from the set of Pauli strings $\{I, X, Y, Z\}^{\otimes 2}$. Consequently, we must derive 16 commutation relations to fully describe the interaction of Pauli gadgets with the CNOT gate. Similarly, there are 16 commutation relations describing the interaction of Pauli gadgets with the CZ gate.

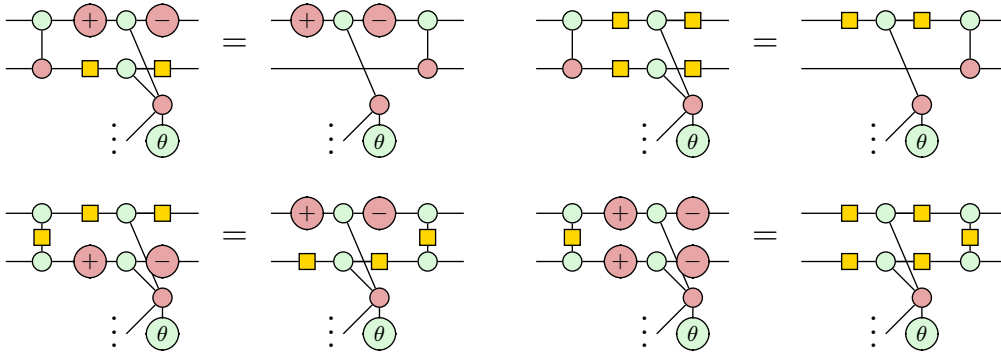
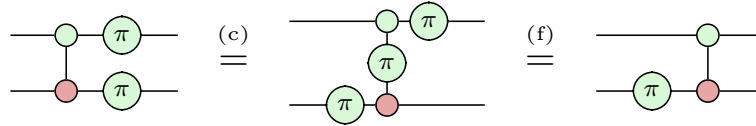


Figure 3.6: Example commutation relations. See Appendix 3.4 for the complete set.

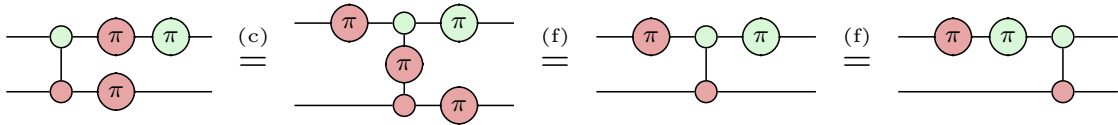
3. Pauli Gadgets

Whilst it is possible to derive each of these commutation relations directly, it can be shown, through the relevant Taylor expansion, that conjugating a Pauli gadget is equivalent to finding the one parameter unitary group of the corresponding conjugated Pauli string. In other words, identifying how a Pauli string interacts with Clifford gates tells us how the corresponding Pauli gadget behaves.

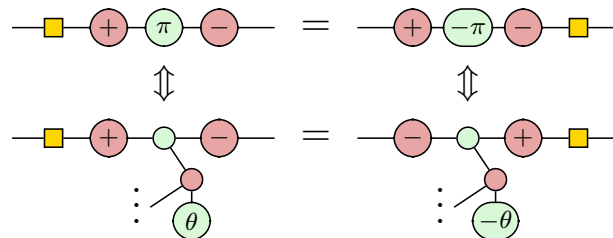
Let us illustrate this with an example. Using the $Z \otimes Z$ Pauli string, we show that the $\text{CNOT}_{0,1}$ gate commutes through the $\exp\left[-i\frac{\theta}{2}(Z \otimes Z)\right]$ gadget to give the $\exp\left[-i\frac{\theta}{2}(I \otimes Z)\right]$ gadget. We first push the bottom Pauli Z gate through the CNOT target using the π copy rule (2.12), then, we push the top Pauli Z gate through the CNOT control using the spider fusion rule (2.10), cancelling one of the copied Pauli Z gates in the process.



Recall that up to a global phase of $-i$, the Pauli Y gate can be expressed as a Pauli X gate followed by a Pauli Z gate (2.18). We can, therefore, derive how the $\text{CNOT}_{0,1}$ gate interacts with the $\exp\left[-i\frac{\theta}{2}(Y \otimes X)\right]$ Pauli gadget by identifying how the $\text{CNOT}_{0,1}$ gate interacts with the $Y \otimes X$ Pauli string.



Similarly, identifying how single-qubit Clifford gates interact with Pauli gadgets amounts to identifying how they interact with the Pauli gates. Importantly, if the phase of the Pauli gate is flipped, we must also flip the phase of the corresponding Pauli gadget.



Chapter 4

Controlled Rotations

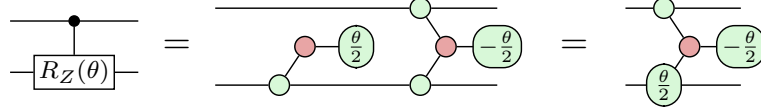
Controlled rotations play an important part in UCC ansätze representing fermionic systems. They can be used to account for the fermionic antisymmetry observed in fermionic systems by applying a phase (rotation) depending on the parity of the state. In other words, the rotation is *controlled* by the parity of the state.

In this chapter, we will first discuss the representation of singly-controlled rotations in the ZX calculus. We will then demonstrate how these can be used to construct higher order controlled rotations. We will later use the results derived in this chapter to demonstrate the correspondence between excitation operators and controlled rotations in Chapter 5.

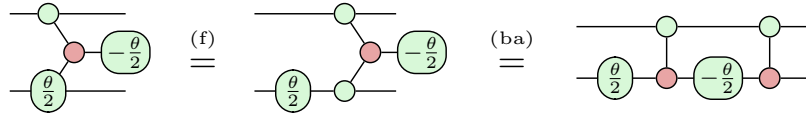
4. Controlled Rotations

4.1 Singly Controlled-Rotations

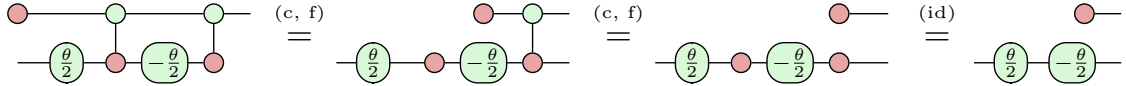
Starting with the singly-controlled Z rotation gate CR_Z , we will see that it can be expressed as a combination of phase gadgets, since it corresponds to a diagonal matrix in the computational Z basis [3].



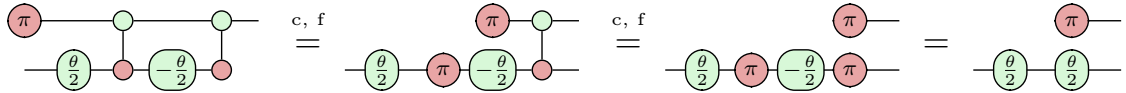
The CR_Z gate applies a rotation to the target qubit if the control qubit is in the $|1\rangle$ state, and applies no rotation, if the control qubit is in the $|0\rangle$ state. This behaviour generalises to superpositions of states. Using the spider fusion (2.10) and bialgebra (2.16) rules, we obtain the CR_Z gate in quantum circuit notation as follows.



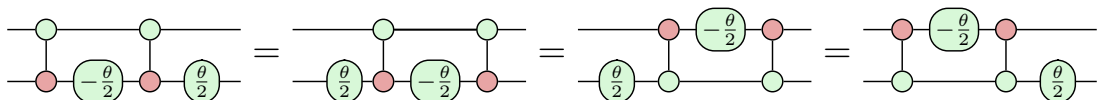
To verify that this circuit does indeed correspond to a controlled rotation in the Z basis, let us observe what happens when the control qubit is $|0\rangle$. Using the spider fusion (2.10), state copy (2.12) and identity (2.11) rules, we see that,



As expected, fusing the Z rotations in the resulting diagram yields identity. Conversely, when the control qubit is $|1\rangle$, we obtain a Z rotation by θ .



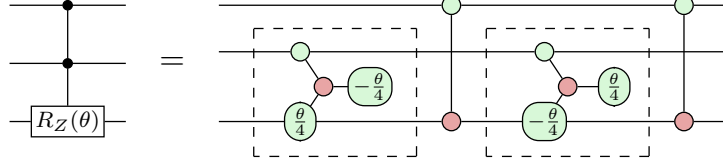
Using that the Z rotation commutes through the phase gadget (2.10), and the phase gadget decomposition result (3.1), we can decompose the CR_Z gate into four equivalent quantum circuits.



4. Controlled Rotations

4.2 Higher Order Controlled-Rotations

We can implement higher order controlled rotations by nesting singly-controlled rotations in CNOT gates as in Yordanov *et al* [10]. We implement a doubly-controlled Z rotation using two singly-controlled Z rotations with opposite phases.



We can minimise the depth of the circuit implementing the CCR_Z gate by choosing specific decompositions of the CR_Z gate (see diagram on the left) such that one pair of $\text{CNOT}_{1,2}$ are adjacent and cancel one-another (see diagram on the right).

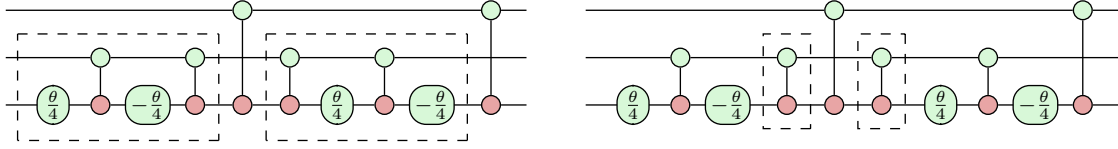
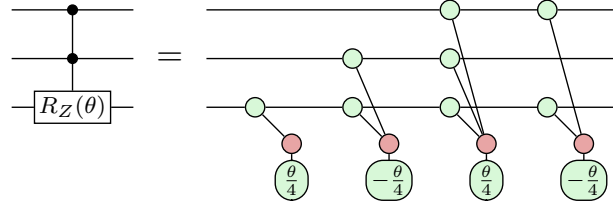


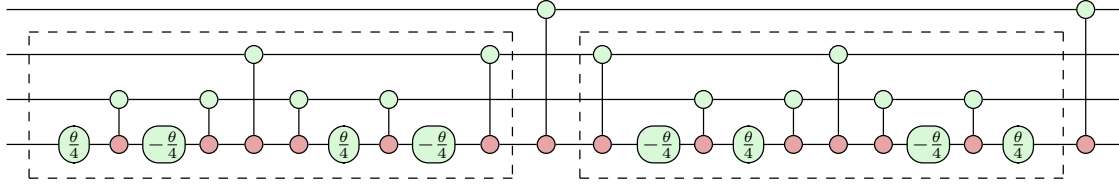
Figure 4.1: CCR_Z circuit decomposition (left). Cancelling CNOT gate pair (right).

However, for the purposes of this thesis, we are interested in the form of controlled rotations in terms of Pauli gadgets. By expressing all Z rotations as phase gadgets (3.2) and commuting all CNOT gates through them such that the CNOT gates cancel, we obtain a phase polynomial consisting of four commuting phase gadgets.

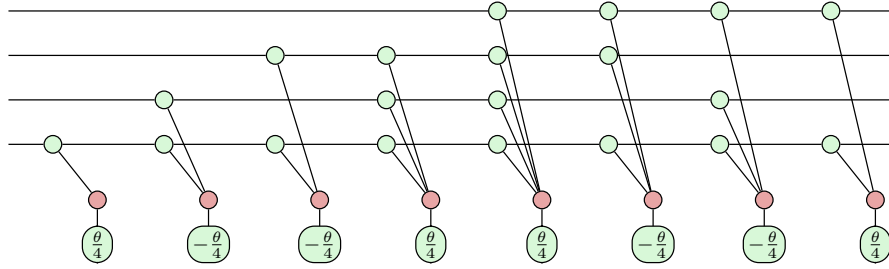


We can build controlled rotations to arbitrary arity by recursively applying the method of nesting controlled rotations described above. Hence, as with the doubly-controlled rotation, we can construct triply-controlled rotations by nesting doubly-controlled rotations. Below is one specific implementation of a triply-controlled Z rotation, with the largest number of adjacent CNOT gates.

4. Controlled Rotations



Using the same rules as before, we can show that a triply-controlled Z rotation corresponds to a phase polynomial consisting of eight commuting phase gadgets.



Chapter 5

Excitation Operators

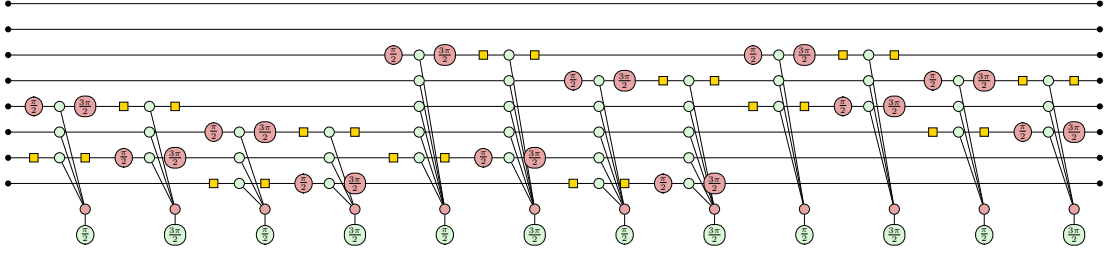
In this chapter, we will use the theory that we have developed in Chapter 3 on Pauli gadgets and their commutation relations, and in Chapter 4 on controlled rotations, to study the excitation operators used to construct Unitary-Product State ansätze.

- introduce maths for both singles, doubles and paired first
- show example UCC ansatz in full
- describe as set of commuting Pauli gadgets. show commutation relations
- show commutation relations between different excitation operators
- introduce conjugation by some clifford to yield phase polynomial and suggest optimisation from [11]
- introduce controlled rotation stuff in new section
- **optimisations section** – show balanced tree representation stuff "Circuit optimisation is typically carried out by pattern replacement: recognising a subcircuit of specific form and replacing it with an equivalent" [24]
- "In principle, local rewriting of gate sequences is sufficient for any circuit optimisation³. However, in practice, good results often require manipulation of large-scale structures in the quantum circuit. Phase gadgets are one such macroscopic structure that is easy to identify within circuits, easy to synthesise back into a circuit, and have a useful algebra of interactions with one another."

5. Excitation Operators

[24]

- "Further, in the balanced tree form more of the CX gates are “exposed” to the rest of the circuit, and could potentially be eliminated by a later optimisation pass" [24]



5. Excitation Operators

5.1 One Body Excitation Operators

Recall that a single fermionic excitation, from spin orbital p to spin orbital q , can be expressed as $a_q^\dagger a_p$ in second quantisation.

By taking a linear combination of the excitation and de-excitations acting on qubits p and q , we obtain the following anti-Hermitian fermionic one-body excitation operator.

$$\hat{\kappa}_p^q = a_q^\dagger a_p - a_p^\dagger a_q$$

Then, recalling the Jordan-Wigner transformation for the creation and annihilation operators,

$$\hat{a}_j^\dagger = \frac{1}{2}(X - iY) \otimes Z_{j-1}^\rightarrow \quad \hat{a}_j = \frac{1}{2}(X + iY) \otimes Z_{j-1}^\rightarrow$$

We can show that the anti-Hermitian fermionic one-body excitation operator can be expressed in terms of quantum gates as follows.

$$F_p^q = \frac{i}{2}(Y_p X_q - X_p Y_q) \prod_{k=p+1}^{q-1} Z_k$$

By multiplying by θ and exponentiating, we obtain the corresponding unitary *exponential one-body excitation operator*.

$$U_p^q(\theta) = \exp \left(i \frac{\theta}{2} (Y_p X_q - X_p Y_q) \prod_{k=p+1}^{q-1} Z_k \right)$$

From here onwards, we will simply refer to the unitary exponential one-body excitation operator as the *one-body excitation operator*.

We can express this one-body excitation operator as the following commuting exponential terms.

$$U_p^q(\theta) = \exp \left(i \frac{\theta}{2} (Y_p X_q - X_p Y_q) \prod_{k=p+1}^{q-1} Z_k \right)$$

$$U_p^q(\theta) = \left(\exp \left[i \frac{\theta}{2} Y_p X_q \prod_{k=p+1}^{q-1} Z_k \right] \right) \left(\exp \left[-i \frac{\theta}{2} X_p Y_q \prod_{k=p+1}^{q-1} Z_k \right] \right)$$

The first exponential term can be implemented by the following quantum circuit.

5. Excitation Operators

$$= \left(\exp \left[i \frac{\theta}{2} Y_p X_q \prod_{k=p+1}^{q-1} Z_k \right] \right)$$

The second exponential term can be implemented by the following circuit.

$$= \left(\exp \left[-i \frac{\theta}{2} X_p Y_q \prod_{k=p+1}^{q-1} Z_k \right] \right)$$

The left CNOT ladder can be thought of as calculating the parity of the fermionic state, whilst the right CNOT ladder construction uncomputes the parity. A phase of θ is then applied depending on the parity of the state. By sequentially composing (2.1) these circuits, that is, taking their matrix product, we have implemented the one-body excitation operator between qubits p and q . Expressing this operator as two Pauli gadgets, we obtain the following.

$$= \exp \left(i \frac{\theta}{2} (Y_p X_q - X_p Y_q) \prod_{k=p+1}^{q-1} Z_k \right)$$

One-Body Excitation Operators as Controlled Rotations

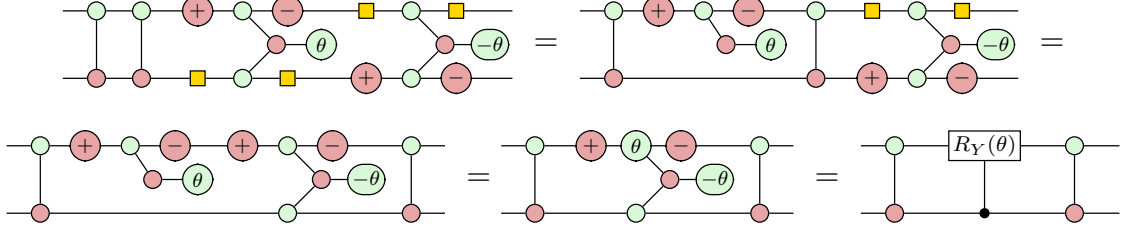
In this section, we will extend the work done by Yordanov *et al* [10], looking at how excitation operators can be expressed in terms of a controlled rotation. To the knowledge of the author, this work has not yet been done in the ZX calculus. Taking $U_0^1(\theta)$, $p = 0$ and $q = 1$ as an example, we have the following.

$$= \exp \left(i \frac{\theta}{2} (Y_0 X_1 - X_0 Y_1) \right)$$

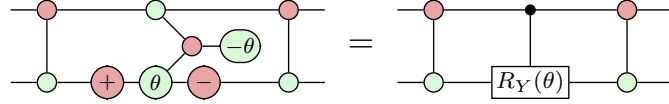
By inserting two adjacent $\text{CNOT}_{0,1}$ gates (self-inverse) into the circuit and using

5. Excitation Operators

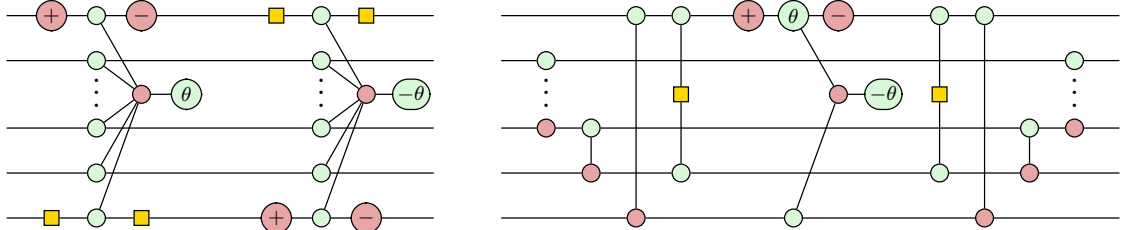
the CNOT commutation rules derived in Chapter 3, we are able to show that the one-body excitation operator can be expressed in terms of a singly-controlled rotation in the Y basis.



Had we instead chosen to insert two $\text{CNOT}_{1,0}$ gates, we would have obtained the following controlled rotation, with the control on qubit 1.



Let us now look at the general case. Yordanov *et al* [10] show that a one-body excitation operator can be expressed as a controlled rotation in the Y basis. See Appendix 7.2 for the intermediate steps.



5. Excitation Operators

5.2 Two Body Excitation Operators

A double fermionic excitation, from spin orbitals p and q to spin orbitals r and s , can be expressed as $a_r^\dagger a_s^\dagger a_q a_p$ in second quantisation.

By taking a linear combination of the excitation and de-excitation operators acting on qubits p, q, r and s , we obtain the following anti-Hermitian fermionic two-body excitation operator.

$$\hat{\kappa}_{pq}^{rs} = a_r^\dagger a_s^\dagger a_q a_p - a_p^\dagger a_q^\dagger a_s a_r$$

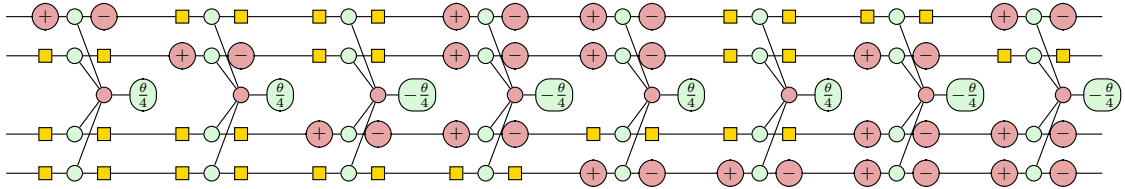
Which after applying the Jordan-Wigner transformation can be expressed in terms of the Pauli matrices as follows.

$$F_{pq}^{rs} = \frac{i}{8} (X_p X_q Y_s X_r + Y_p X_q Y_s Y_r + X_p Y_q Y_s Y_r + X_p X_q X_s Y_r - \\ Y_p X_q X_s X_r - X_p Y_q X_s X_r - Y_p Y_q Y_s X_r - Y_p Y_q X_s Y_r) \prod_{k=p+1}^{q-1} Z_k \prod_{l=r+1}^{s-1} Z_l$$

Multiplying by θ and exponentiating, we obtain the following unitary exponential two-body excitation operator, which we will refer to simply as a *two-body excitation operator*.

$$U_{pq}^{rs}(\theta) = \exp \left(i \frac{\theta}{8} (X_p X_q Y_s X_r + \dots - Y_p Y_q Y_s X_r - Y_p Y_q X_s Y_r) \prod_{k=p+1}^{q-1} Z_k \prod_{l=r+1}^{s-1} Z_l \right)$$

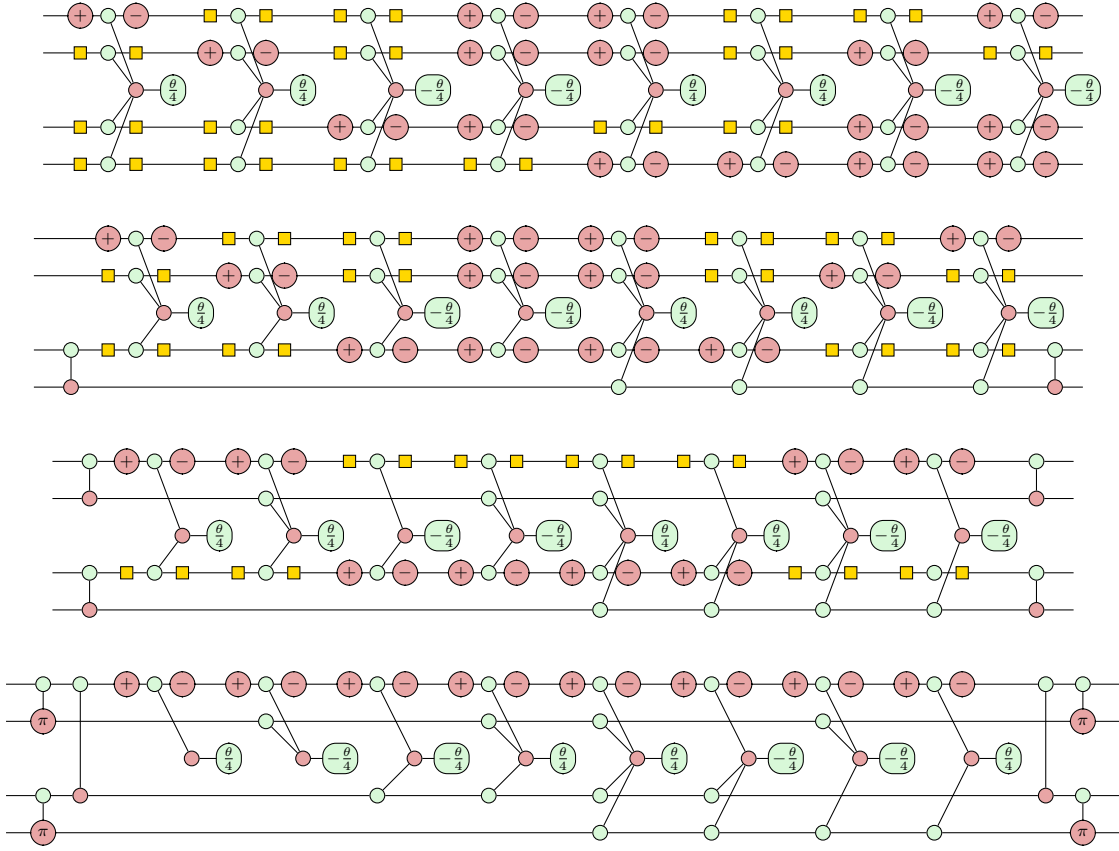
The full two-body excitation operator can be split into eight commuting exponential terms, implemented by eight Pauli gadgets. Taking $p = 0, q = 1, r = 2$ and $s = 3$ as an example, we obtain the following in Pauli gadget form.



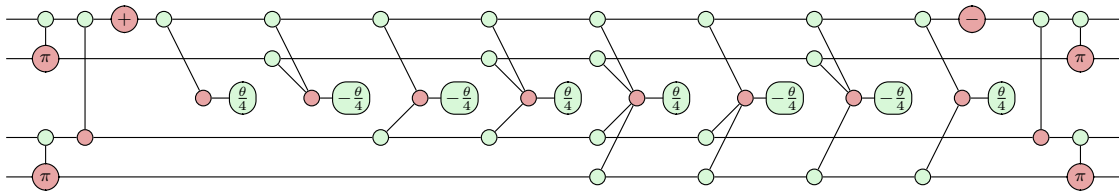
Two-Body Excitation Operators as Controlled Rotations

Using the commutation relations derived in Chapter 3, we can see that by conjugating the circuit with CNOT gates we obtain the following phase polynomial.

5. Excitation Operators



By cancelling adjacent change of basis gates, we can show that a two-body excitation operator corresponds to a triply-controlled rotation in the Y basis (recalling the result for triply controlled rotations in Chapter 4), as in Yordanov *et al.*



Chapter 6

ZxFermion Software

ZxFermion (visit github.com/aymannel/zxfermion for documentation) is a Python package that I wrote for the manipulation and visualisation of circuits of Pauli gadgets. It is built on top of the PyZX `BaseGraph` API [29] and Stim [30]. The motivation for building this package came from the need for a user-friendly tool to explore research ideas related to circuits of Pauli gadgets.

Whilst there are existing software solutions like PauliOpt, which focus on circuit simplification using architecture-aware synthesis algorithms [31], ZxFermion provides a more accessible alternative, as well as offering tools for studying the interaction of Pauli gadgets with Clifford and Pauli gates using Stim’s `Tableau` class.

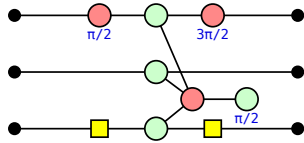
ZxFermion is designed to integrate with Jupyter notebook environments, enabling users to visualise interactive ZX diagrams directly in the output cell. The package has also undergone thorough testing, ensuring its reliability and ease of use.

All of the proofs so far derived can be replicated using ZxFermion, showcasing a noteworthy acceleration in research pace. We anticipate that both chemists and computer scientists exploring quantum computing within the VQE framework will find this software tool advantageous.

6.1 Creating Gadgets and Circuits

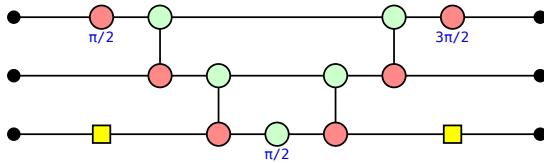
In this section, we will introduce the `Gadget` class, which we use to represent Pauli gadgets. We provide a Pauli string and a phase to instantiate a `Gadget()` object.

```
gadget = Gadget('YZX', phase=1/2)
gadget.draw()
```



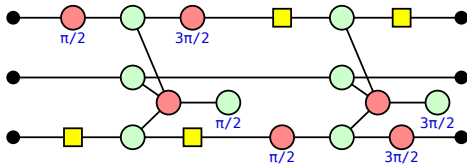
By setting the `as_gadget` option to `False`, we can view the gadget in its expanded form, that is, in quantum circuit notation.

```
gadget = Gadget('YZX', phase=1/2, as_gadget=False)
gadget.draw()
```



We can construct a circuit of Pauli gadgets using the `GadgetCircuit` class. The underlying data structure for this class is simply an ordered list.

```
gadget1 = Gadget('YZX', phase=1/2)
gadget2 = Gadget('XZY', phase=3/2)
circuit = GadgetCircuit([gadget1, gadget2])
circuit.draw()
```

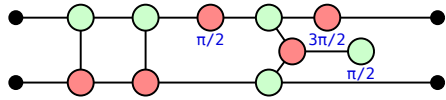


ZxFermion also implements standard quantum gates via the `CX`, `CZ`, `X`, `Z`, `XPlus`, `XMinus`, `ZPlus` and `ZMinus` classes. As we will see in the next section, we have implemented the logic describing the interaction of Pauli gadgets with these gates.

6.2 Manipulating Circuits

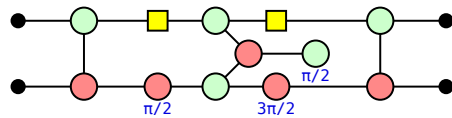
The `GadgetCircuit` class comes equipped with the `apply()` method that allows us to insert a quantum gate, and its Hermitian conjugate, into the circuit. This operation preserves the matrix corresponding to a given circuit. The method takes the quantum gate to apply as its first parameter, whilst the `start` and `end` parameters designate the insertion positions.

```
gadget = Gadget('YZ', phase=1/2)
circuit = GadgetCircuit([gadget])
circuit.apply(CX(0, 1), start=0, end=0, draw=True)
```



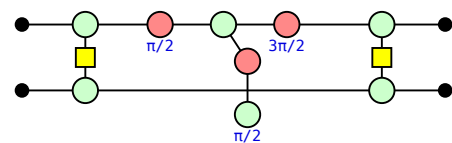
If no specific positions are specified, the insertion defaults to placing one quantum gate at the start, and its Hermitian conjugate at the end, of the circuit. The `GadgetCircuit` class then manages the relevant commutation relations, ensuring the expected gadget outcome. Hence, we below, we observe the gadget that results upon pushing a CNOT gate through the `Gadget("YZ", phase=1/2)` object.

```
circuit.apply(CX(0, 1), draw=True)
```



We could have chosen any Pauli gadget by simply instantiating the relevant `Gadget()` object. Alternatively, we could have chosen to insert any Hermitian conjugate pair of gates. The commutation logic implements Stim's `Tableau` class to identify the behaviour of a Pauli string with a given Clifford or Pauli gate.

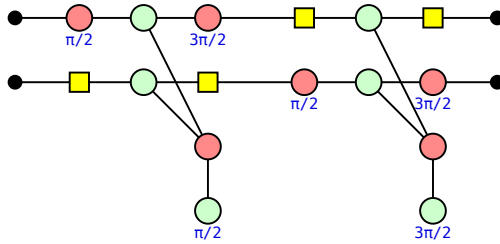
```
circuit.apply(CZ(0, 1), draw=True)
```



6. ZxFermion Software

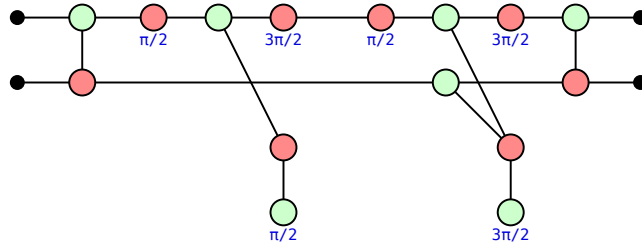
The `GadgetCircuit` class offers the ability to manipulate circuits containing multiple gadgets simultaneously. Below, we instantiate the parametrised exponential of the single excitation operator, $a_0^\dagger a_1 - a_1^\dagger a_0$, in terms of quantum gates.

```
gadget1 = Gadget('YX', phase=1/2)
gadget2 = Gadget('XY', phase=3/2)
circuit = GadgetCircuit([gadget1, gadget2])
circuit.draw()
```



Conjugating the circuit with CNOTs reveals two Pauli gadgets, that combined, represent a controlled rotation in the Y basis. We have therefore faithfully replicated the result outlined in Yordanov *et al* [10].

```
cliff = CX(0, 1)
circuit.apply(cliff, draw=True)
```



We consider it a significant achievement that with just a few lines of code, a user can now faithfully replicate derivations that took us many hours.

GIVE SOME MORE EXAMPLES IN COMPACT FORM INCLUDING CONJUGATING BY XPLUS ETC

Chapter 7

Conclusion

7.1 Summary

7.2 Future Work

Appendices

Hadamard

Below are several equivalent definitions of the Hadamard generator. Note that the two rightmost definitions do not require any scalar correction.

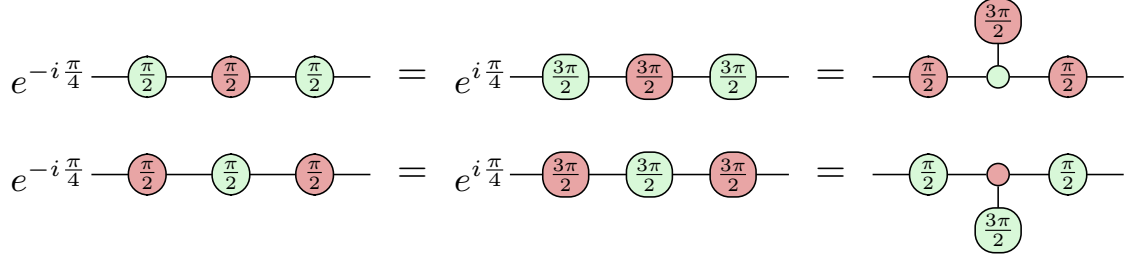
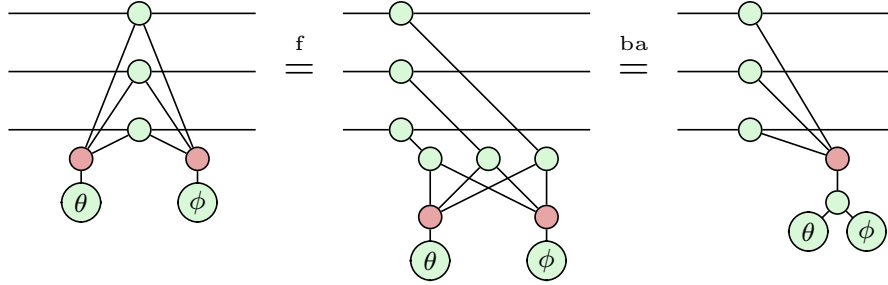


Figure 1: Equivalent definitions of the Hadamard generator.

Phase Gadgets

We can show how two adjacent phase gadgets fuse using the spider fusion (2.10) and bialgebra (2.16) rules as follows.



Clifford Conjugation Stuff

$$\begin{aligned}
 C e^P C^\dagger &= C \sum_{n=0}^{\infty} \left(\frac{P^n}{n!} \right) C^\dagger \\
 C P^n C^\dagger &= \sum_{n=0}^{\infty} \frac{C P^n C^\dagger}{n!} \\
 C P^n C^\dagger &= \sum_{n=0}^{\infty} \frac{(C P C^\dagger)^n}{n!} \\
 C P^n C^\dagger &= (C P C^\dagger)^n
 \end{aligned}$$

CNOT Commutation Relations

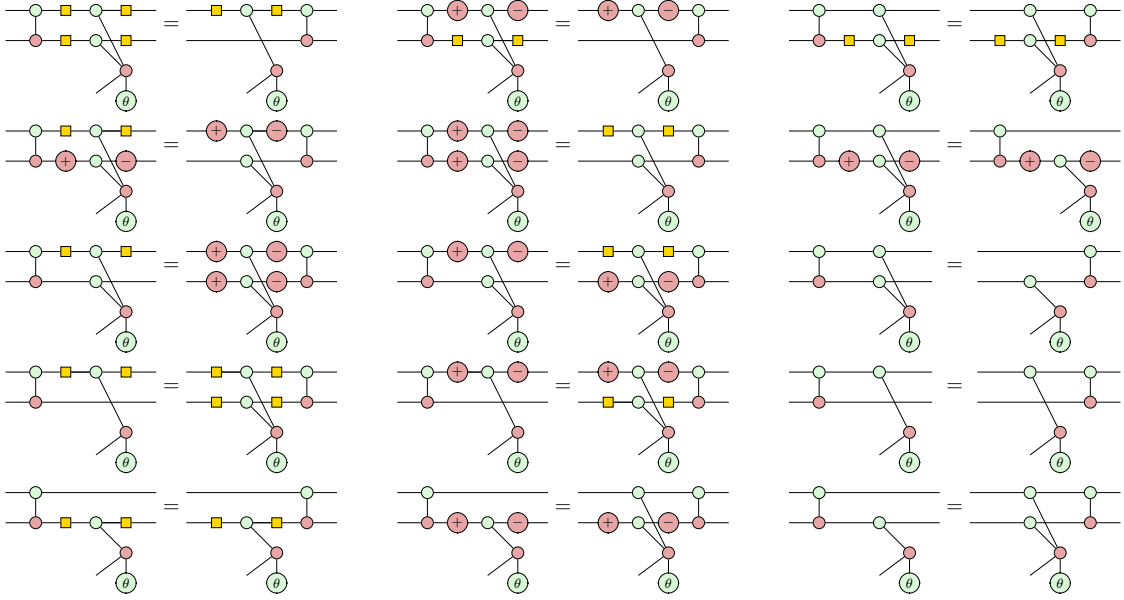
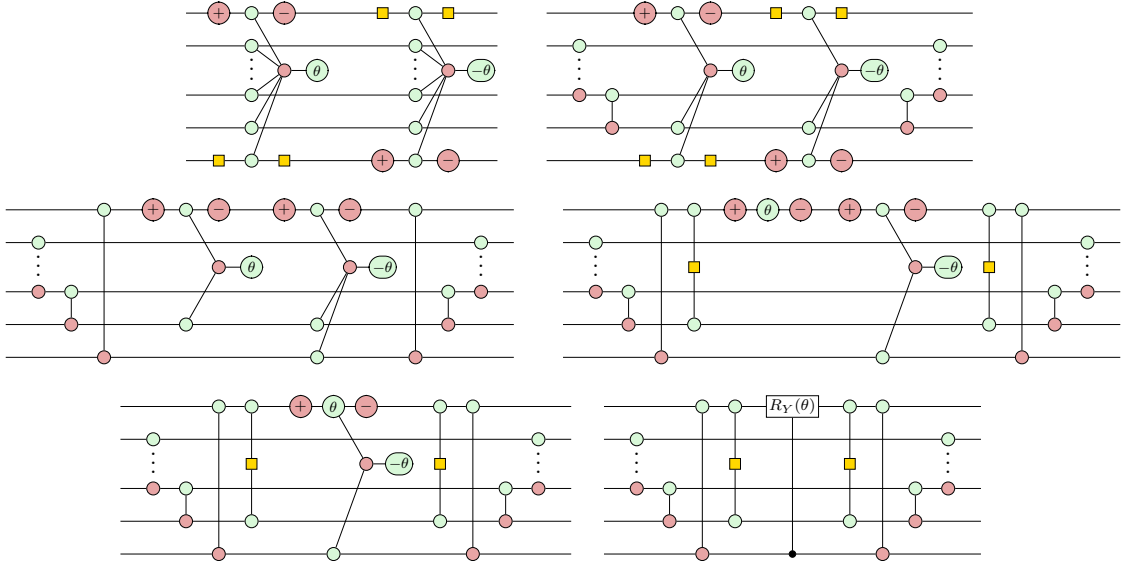


Figure 2: Complete set of CNOT commutation relations.

General One-Body Excitation Operator



Bibliography

- [1] Szalay, P. G., Müller, T., Gidofalvi, G., Lischka, H. & Shepard, R. Multi-configuration self-consistent field and multireference configuration interaction methods and applications. *Chemical Reviews* **112**, 108–181 (2011).
- [2] Kassal, I., Whitfield, J. D., Perdomo-Ortiz, A., Yung, M.-H. & Aspuru-Guzik, A. Simulating chemistry using quantum computers. *Annual Review of Physical Chemistry* **62**, 185–207 (2011).
- [3] Yeung, R. Diagrammatic design and study of ansätze for quantum machine learning (2020). 2011.11073.
- [4] Preskill, J. Quantum computing in the nisq era and beyond. *Quantum* **2**, 79 (2018).
- [5] Poulin, D. *et al.* The trotter step size required for accurate quantum simulation of quantum chemistry. *QIC 15*, 361 (2015) (2014). 1406.4920.
- [6] Romero, J. *et al.* Strategies for quantum computing molecular energies using the unitary coupled cluster ansatz. *Quantum Science and Technology* **4**, 014008 (2018).
- [7] Wecker, D., Hastings, M. B. & Troyer, M. Progress towards practical quantum variational algorithms. *Physical Review A* **92**, 042303 (2015).
- [8] McClean, J. R., Romero, J., Babbush, R. & Aspuru-Guzik, A. The theory of variational hybrid quantum-classical algorithms. *New Journal of Physics* **18**, 023023 (2016).
- [9] Kirby, W. M. & Love, P. J. Variational quantum eigensolvers for sparse hamiltonians. *Phys. Rev. Lett.* *127*, 110503 (2021) **127**, 110503 (2020). 2012.07171.
- [10] Yordanov, Y. S., Arvidsson-Shukur, D. R. M. & Barnes, C. H. W. Efficient quantum circuits for quantum computational chemistry. *Physical Review A* **102**, 062612 (2020).
- [11] Cowtan, A., Simmons, W. & Duncan, R. A generic compilation strategy for the unitary coupled cluster ansatz (2020). 2007.10515.
- [12] Cerezo, M. *et al.* Variational quantum algorithms. *Nature Reviews Physics* *3*, 625–644 (2021) **3**, 625–644 (2020). 2012.09265.
- [13] Peruzzo, A. *et al.* A variational eigenvalue solver on a photonic quantum processor. *Nature Communications* **5** (2014).

Bibliography

- [14] Taube, A. G. & Bartlett, R. J. New perspectives on unitary coupled-cluster theory. *International Journal of Quantum Chemistry* **106**, 3393–3401 (2006).
- [15] Burton, H. G. A., Marti-Dafcik, D., Tew, D. P. & Wales, D. J. Exact electronic states with shallow quantum circuits from global optimisation. *npj Quantum Information* **9** (2023).
- [16] Coecke, B. & Duncan, R. Interacting quantum observables: categorical algebra and diagrammatics. *New Journal of Physics* **13**, 043016 (2011).
- [17] Szabó, A. v. & Ostlund, N. S. *Modern quantum chemistry : introduction to advanced electronic structure theory* (Mineola (N.Y.) : Dover publications, 1996). URL <http://lib.ugent.be/catalog/rug01:000906565>.
- [18] Helgaker, T., Jørgensen, P. & Olsen, J. *Molecular Electronic-Structure Theory* (Wiley, 2000).
- [19] Fetter, A. L., Walecka, J. D. & Kadanoff, L. P. *Quantum Theory of Many Particle Systems*, vol. 25 (AIP Publishing, 1972).
- [20] Evangelista, F. A., Chan, G. K.-L. & Scuseria, G. E. Exact parameterization of fermionic wave functions via unitary coupled cluster theory. *The Journal of Chemical Physics* **151** (2019). 1910.10130.
- [21] Chan, H. H. S., Fitzpatrick, N., Segarra-Martí, J., Bearpark, M. J. & Tew, D. P. Molecular excited state calculations with adaptive wavefunctions on a quantum eigensolver emulation: reducing circuit depth and separating spin states. *Physical Chemistry Chemical Physics* **23**, 26438–26450 (2021).
- [22] van de Wetering, J. Zx-calculus for the working quantum computer scientist (2020). 2012.13966.
- [23] Stone, M. H. On one-parameter unitary groups in hilbert space. *The Annals of Mathematics* **33**, 643 (1932).
- [24] Cowtan, A., Dilkes, S., Duncan, R., Simmons, W. & Sivarajah, S. Phase gadget synthesis for shallow circuits (2019). 1906.01734.
- [25] Amy, M., Maslov, D., Mosca, M. & Roetteler, M. A meet-in-the-middle algorithm for fast synthesis of depth-optimal quantum circuits. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* **32**, 818–830 (2013).
- [26] Amy, M., Maslov, D. & Mosca, M. Polynomial-time t-depth optimization of clifford+t circuits via matroid partitioning. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* **33**, 1476–1489 (2014).
- [27] Nam, Y., Ross, N. J., Su, Y., Childs, A. M. & Maslov, D. Automated optimization of large quantum circuits with continuous parameters. *npj Quantum Information* **4** (2018).
- [28] Maslov, D. & Roetteler, M. Shorter stabilizer circuits via bruhat decomposition and quantum circuit transformations. *IEEE Transactions on Information Theory* **64**, 4729–4738 (2018).
- [29] Kissinger, A. & van de Wetering, J. Pyzx: Large scale automated diagrammatic reasoning. *Electronic Proceedings in Theoretical Computer Science* **318**, 229–241 (2020).

Bibliography

- [30] Gidney, C. Stim: a fast stabilizer circuit simulator. *Quantum* **5**, 497 (2021).
- [31] Gogioso, S. & Yeung, R. Annealing optimisation of mixed zx phase circuits. *Electronic Proceedings in Theoretical Computer Science* **394**, 415–431 (2023).