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. Since an exact solution to this problem scales exponentially with the number of electrons, and classical computers have no means by which to efficiently store the increasingly large wavefunctions, this problem becomes 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 it 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 Electronic Structure Theory 4

1.2 Fundamentals of Quantum Computating 9

1.3 The Variational Quantum Eigensolver 13

2 ZX Calculus **16**

2.1 Generators 17

2.2 Rewrite Rules 21

2.3 Clifford Conjugation 24

3 Pauli Gadgets **25**

3.1 Phase Gadgets 26

3.2 Pauli Gadgets 29

3.3 Phase Polynomials 30

3.4 Commutation Relations 31

4 Controlled Rotations **36**

4.1 Singly Controlled-Rotations 37

4.2 Higher Order Controlled-Rotations 38

5 Excitation Operators **40**

5.1 Implementing Excitation Operators 41

5.2 Excitation Operators as Pauli Gadgets 44

5.3 Commuting Excitation Operators 45
5.4 Excitations as Controlled Rotations 46

6 ZxFermion Software **49**
6.1 Creating Gadgets and Circuits 50
6.2 Manipulating Circuits 52

7 Conclusion **55**
7.1 Summary 55
7.2 Future Work 55

Appendices

Bibliography **58**

Chapter 1

Background

In this chapter, we will begin by discussing the context and motivation for the research conducted in this thesis, then develop the theoretical foundation required to simulate fermionic systems on quantum computers. We will start by giving an overview of Electronic Structure Theory in Section 1.1, then we will introduce the Fundamentals of Quantum Computation in Section 1.2 and finally, the Variational Quantum Eigensolver algorithm in Section 1.3.

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 are able to parametrically explore the Hilbert space of possible quantum states [8].

The Discretely and Continuously Optimised 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 optimising ansätze that represent fermionic wavefunctions.

Additionally, 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.

1.1 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

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.

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, . . . , N-tuple excited states [17].

Second Quantisation

In second quantisation, both observables and states (by acting on the vacuum state) are represented by operators, namely the creation and annhilation 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 annhilation 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 unnoccupied.

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

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 annhilation 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

construct the following excitation operator.

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

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

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

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

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

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 annhilation operators [18].

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

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

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.

$$h_{ij} = \int_{-\infty}^{\infty} \psi_{i(x_1)}^* \left(-\frac{1}{2}\nabla^2 + \hat{V}_{(x_1)} \right) \psi_{j(x_1)} \, d^3x_1$$

$$h_{ijkl} = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \psi_{i(x_1)}^* \psi_{j(x_2)}^* \left(\frac{1}{|x_1 - x_2|} \right) \psi_{k(x_2)} \psi_{l(x_1)} \, d^3x_1 d^3x_2$$

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.

1.2 Fundamentals of Quantum Computating

The central concept that allows gate-based quantum computation to represrent exponentially scaling states with polynomially scaling quantum resources is its ability to encode and manipulate superpositions of states. In this section, we will provide an introduction to qubits and basic quantum gates.

Introduction to Qubits

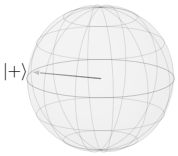
Classical computation encodes information using binary strings formed from the computational basis states 0 and 1. Thus, given n classical bits, we can encode 2^n binary strings. In contrast, information on a quantum computer is encoded using two quantum states, corresponding to vectors in a two-dimensional complex Hilert space $\mathbb{C}$. The $|0\rangle$ and $|1\rangle$ states, known as the Z computational basis, form the standard computational basis for encoding information on a quantum computer.

Figure 1.4: Z computational basis states.

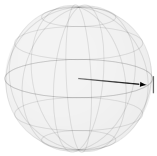
An arbitrary qubit state $|\psi\rangle$ can be described as a complex linear combination of the computational basis states, $|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle$, provided that the qubit state is normalised. In other words we require two complex numbers, or four real numbers, to describe an arbitrary qubit state. Since only the relative phase between the basis states is directly measureable, there is a redundancy in this description that allows us to represent an arbitrary qubit state using only three real numbers.

By taking advantage of this redundancy, we can derive a three-dimensional representation of an arbitrary qubit state, known as the Bloch sphere. Note that opposite points on the Bloch sphere correspond to mutually orthonal states, as in Figure 1.4 in which we have represented the Z computational basis states.

We could have chosen to form our computational basis with any pair of orthonormal states. For instance, we define the X basis states as follows.



$$= \frac{|0\rangle + |1\rangle}{\sqrt{2}}$$



$$= \frac{|0\rangle - |1\rangle}{\sqrt{2}}$$

Figure 1.5: Computational X basis states.

In theory, a qubit can exist in an infinite number of states, however, upon measuring with respect to a particular basis, the qubit state collapses into one computational basis state, or the other. This result is known more formally as the *no-cloning theorem*, which states that we cannot create indentical and independent copies of an arbitrary qubit state, as this would involve first measuring that state.

Multiple-Qubit States

Let us now consider systems consisting of multiple qubits. Similarly to how n classical bits give rise to 2^n binary strings, we have that, n qubits give rise to 2^n basis states. These basis states are formed by taking the Kronecker product. For instance, a two-qubit system gives rise to the four following basis states.

$$|00\rangle = |0\rangle \otimes |0\rangle \qquad |01\rangle = |0\rangle \otimes |1\rangle \qquad |10\rangle = |1\rangle \otimes |0\rangle \qquad |11\rangle = |1\rangle \otimes |1\rangle$$

Figure 1.6: The four basis states associated with a two-qubit system.

An arbitrary n-qubit *state vector*, describing the state of the entire system, can be formed by taking a complex linear combination of the 2^n basis states. Therefore, in order to fully describe an arbitrary n-qubit state vector, we need to specify 2^n complex coefficients. In the case of a two-qubit system, we have the following.

$$|\psi\rangle = \alpha\,|00\rangle + \beta\,|01\rangle + \gamma\,|10\rangle + \delta\,|11\rangle$$

$$\text{where} \quad \alpha,\, \beta,\, \gamma,\, \delta \in \mathbb{C}$$

Introduction to Quantum Gates

By definition, quantum gates correspond to unitary transformations, $U^{-1} = U^{\dagger}$ [20]. In other words, any quantum gate corresponds to a square unitary matrix that conserves the complex inner product of the state it acts on. Consequently, quantum gates can be viewed as rotations of the qubit state vector in Hilbert space.

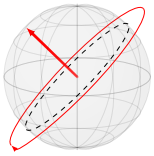
Let us now introduce the most fundamental quantum gates: the Pauli gates. The Pauli gates are described by the 2×2 Pauli matrices and rotate an arbitrary qubit state by π radians about their respective axes.

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

Figure 1.7: The Pauli matrices corresponding to the Pauli gates.

The Pauli X gate is the quantum analogue of the classical NOT gate in that it maps $|0\rangle \leftrightarrow |1\rangle$. Importantly, the Pauli X differs from its classical counterpart in that it can act on any arbitrary qubit state. Similarly, we have that the Pauli Z gate maps $|+\rangle \leftrightarrow |-\rangle$, but can also act on any arbitrary qubit state.

The Hadamard gate interconverts between the Z and X bases and maps $|0\rangle \leftrightarrow |+\rangle$ and $|1\rangle \leftrightarrow |-\rangle$. Therefore, the Hadamard gate can be viewed as a rotation of the Bloch sphere π radians about the line bisecting the z and x axes.

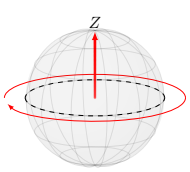


$$= \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$$

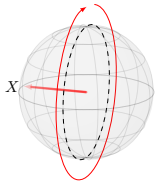
Figure 1.8: Bloch sphere representation of the Hadamard gate.

Since the Pauli gates and the Hadamard gate each correspond to unitary and Hermitian matrices, it follows that they are also self-inverse. Consequently, successively applying any of these gates is equivalent to applying the identity matrix I.

The $R_Z(\theta)$ and $R_X(\theta)$ rotation gates correspond to rotations of the Bloch sphere by some angle θ about the Z and X axes respectively.



$$= \begin{pmatrix} 1 & 0 \\ 0 & e^{i\theta} \end{pmatrix}$$



$$= \begin{pmatrix} 1+e^{i\theta} & 1-e^{i\theta} \\ 1-e^{i\theta} & 1+e^{i\theta} \end{pmatrix}$$

Figure 1.9: Bloch sphere representations of arbitrary Z and X rotations.

Multiple-Qubit Gates

The most fundamental multi-qubit gate is the two-qubit CNOT gate, which is used to entangle two qubits. Consider a two-qubit state of the form $|\alpha\rangle \otimes |\beta\rangle$. Taking α to be the control qubit, and β to be the target qubit, we define the CNOT gate as the gate that maps $|\alpha\rangle \otimes |\beta\rangle \to |\alpha\rangle \otimes |\alpha \oplus \beta\rangle$. That is, the CNOT gate applies the Pauli X gate to the target qubit *iff* the control qubit is in the $|1\rangle$ state. The same CNOT gate acts on the two-qubit basis states (Figure 1.6) as follows.

$$|00\rangle \to |00\rangle \qquad |01\rangle \to |01\rangle \qquad |10\rangle \to |11\rangle \qquad |11\rangle \to |10\rangle$$

In this way the CNOT gate is the quantum generalisation of the classical XOR gate. We define the two matrices corresponding to the two CNOT gates, with the first qubit being the control and the second qubit being the target, and *vice versa*.

$$\mathrm{CNOT}^{c=0}_{t=1} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \qquad \mathrm{CNOT}^{c=1}_{t=0} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}$$

Figure 1.10: Matrix definitions of the CNOT gate.

The CNOT gate can be used to entangle two qubits when the control qubit exists in a superposition of states. For instance, starting with the $|+0\rangle$ state, and letting the first qubit be the control qubit ($|+\rangle$), we have the following.

$$\mathrm{CNOT}\,|+0\rangle = \frac{|00\rangle + |11\rangle}{\sqrt{2}}$$

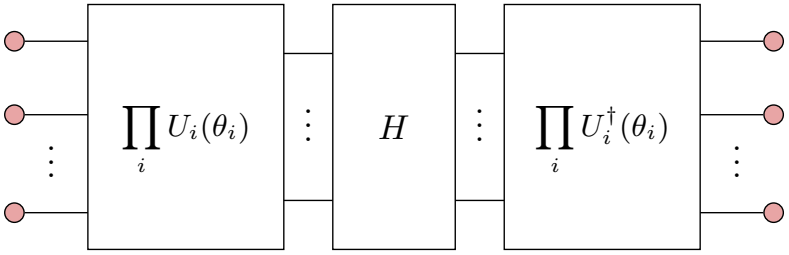
1.3 The Variational Quantum Eigensolver

In this chapter, we introduce the *Variational Quantum Eigensolver (VQE)* algorithm, a particular class of variational quantum algorithms used to estimate the ground state energy of molecular systems. The research presented in this thesis is conducted within this framework. As providing an overview of the VQE algorithm, we will introduce the fundamentals of quantum computing and the *Unitary Coupled Cluster (UCC) ansatz* used to represent fermionic wavefunctions on quantum devices.

The VQE algorithm consists of a quantum subroutine, a *Parametrised Quantum Circuit (PQC)*, that implements some UCC ansatz, and a classical subroutine that classically optimises the UCC ansatz until it converges to the best approximation of the true ground state. PQCs are similar to classical neural networks in concept, but by definition, correspond to $2^n \times 2^n$ unitary maps, where n is the number of qubits, and hence have the same number of inputs as outputs [3].

The input state for the PQC is the reference state that the UCC operator $U(\boldsymbol{\theta})$ acts on, which 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 PQC is then an entangled state, that is, some linear combination of vectors in the Fock space, that captures the correlation present in true ground state of the molecular system of interest.

Upon measuring the PQC output state, 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 via a quantum circuit consisting of the PQC and the Hamiltonian for the system.



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

For the purposes of this thesis, we are interested in a variant of the VQE algorithm developed by Burton *et al* [15] known as the *Discretely and Continuously Optimised Variational Quantum Eigensolver (DISCO-VQE)*. We implement a fermionic state on a quantum device as a sequence of quantum gates representing unitary excitation operators that act on some reference state. The DISCO-VQE algorithm then finds approximate solutions to the fermionic ground state by both (*discretely*) optimising the sequence of fermionic excitation operators, chosen from a finite operator pool, and (*continuously*) optimising their parameters. This allows us to parametrically explore the Hilbert space of possible quantum states until we find a good approximation of the true fermionic ground state [14]. The DISCO-VQE algorithm, therefore, discovers accurate representations of the fermionic UCC wavefunction using a minimal number of parameters [15].

Unitary Coupled Cluster Ansatz

As suggested by Peruzzo *et al* [13], the UCC formulation of a wavefunction can be efficiently implemented on a quantum device in terms of quantum gates, whose implementation we refer to as the UCC ansatz $|\psi(\boldsymbol{\theta})\rangle$. In other words, the UCC ansatz corresponds to some unitary excitation operator $U(\boldsymbol{\theta})$ that acts on a reference state, usually a single reference Slater determinant Hartree-Fock state $|\psi_0\rangle$ obtained via the self-consistent field method.

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

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 operator $\hat{T}(\boldsymbol{\theta}) - \hat{T}^\dagger(\boldsymbol{\theta})$ 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.

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

Figure 1.11: Parametrised k-UCC ansatz.

Where m indexes all possible excitations and $\tau_m - \tau_m^\dagger$ corresponds to anti-Hermitian fermionic excitation operators in second quantisation. For the remainder of this thesis, we will refer to fermionic excitation operators $U_m(\theta_m)$ as the the exponentials of these anti-Hermitian operators. As these fermionic excitation operators derive from second-quantised operators, they preserve the fermionic anti-symmetry and particle number of the reference state.

The operators are then mapped to quantum gates (see Chapter 5) before being implemented on a quantum device. It has been shown by Evangelista *et al* [21] that the UCC ansatz can exactly parametrise any state. Within the DISCO-VQE framework, we consider only generalised *single* (one-body) and *double* (two-body) fermionic excitation (operators), which as shown by Evangelista *et al* [21] achieves universality provided that we combine enough suitably-ordered one-body and two-body excitation operators [15]. In other words, the UCCSD ansatz is sufficient to represent any vector in the Hilbert space.

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

Figure 1.12: Truncated operator containing only *single* and *double* excitations.

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 its basic generators 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 we 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].

Remark – *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.

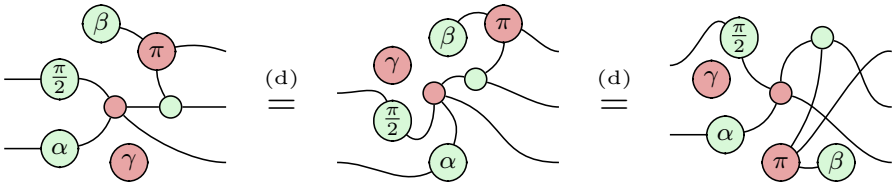
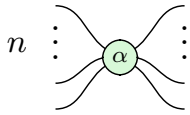


Figure 2.1: Three equivalent ZX diagrams (*only connectivity matters*).

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.*

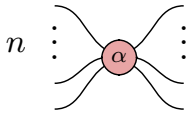
Z Spiders (green) are defined with respect to the Z eigenbasis, $|0\rangle$ and $|1\rangle$, such that a Z Spider with n inputs and m outputs represents the following linear map.



$$= |0\rangle^{\otimes m} \langle 0|^{\otimes n} + e^{i\alpha} |1\rangle^{\otimes m} \langle 1|^{\otimes n}$$

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

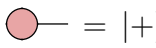
X Spiders (red), are defined with respect to the X eigenbasis, $|+\rangle$ and $|-\rangle$.



$$= |+\rangle^{\otimes m} \langle +|^{\otimes n} + e^{i\alpha} |-\rangle^{\otimes m} \langle -|^{\otimes n}$$

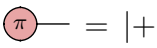
Figure 2.3: Interpretation of an X Spider as a linear map.

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



$$= |+\rangle + |-\rangle = \sqrt{2}\,|0\rangle$$

Figure 2.4: $|0\rangle$ eigenstate



$$= |+\rangle - |-\rangle = \sqrt{2}\,|1\rangle$$

Figure 2.5: $|1\rangle$ eigenstate

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

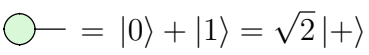


Figure 2.6: $|+\rangle$ eigenstate

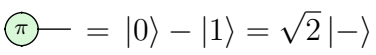
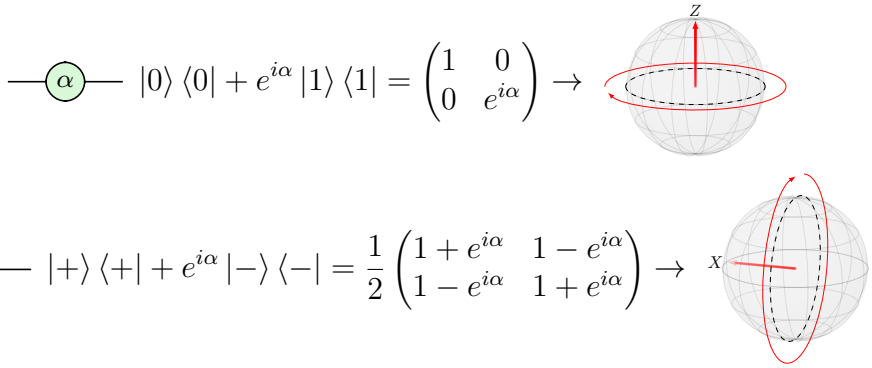


Figure 2.7: $|-\rangle$ eigenstate

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.



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

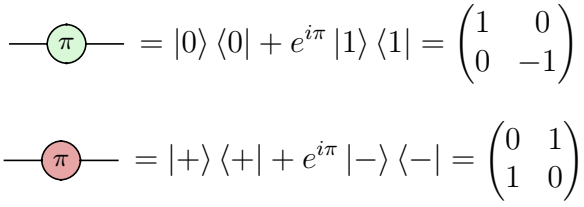
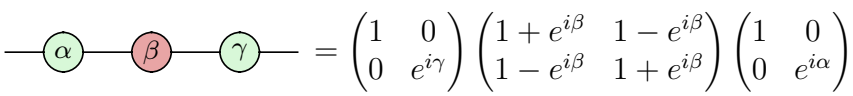


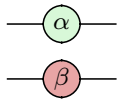
Figure 2.8: Pauli Z and X gates in the ZX calculus.

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.

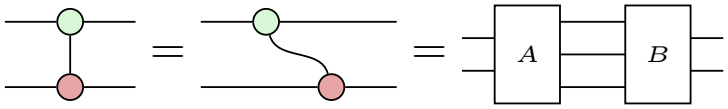


Alternatively, we could have chosen to compose the spiders in parallel, resulting in the tensor product.

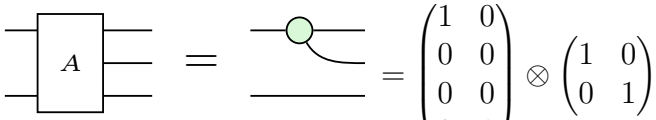


$$= \begin{pmatrix} 1 & 0 \\ 0 & e^{i\alpha} \end{pmatrix} \otimes \begin{pmatrix} 1+e^{i\beta} & 1-e^{i\beta} \\ 1-e^{i\beta} & 1+e^{i\beta} \end{pmatrix}$$

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.

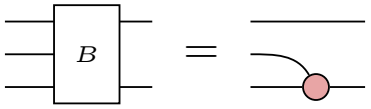


We can calculate matrix A, consisting of a single-input and two-output Z Spider (4×2 matrix) and an empty wire (identity matrix), by taking the tensor product.



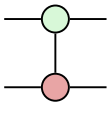
$$= \begin{pmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 1 \end{pmatrix} \otimes \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

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



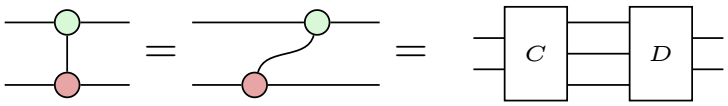
$$= \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \otimes \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix}$$

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

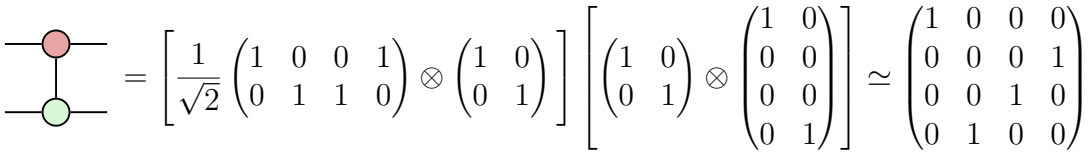


$$= \left[\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \otimes \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix} \right] \left[\begin{pmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 1 \end{pmatrix} \otimes \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \right] \simeq \begin{pmatrix}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

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



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



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].

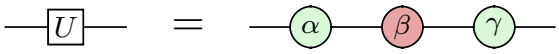
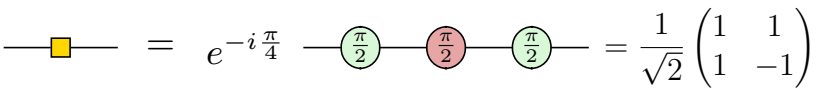
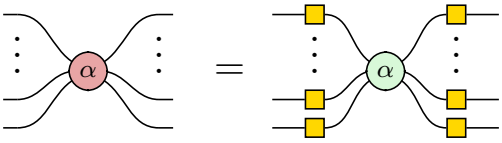


Figure 2.9: Arbitrary single-qubit rotation.

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 $\exp\left(-i\frac{\pi}{4}\right)$, it can be decomposed using Euler angles by choosing $\alpha = \beta = \gamma = \frac{\pi}{2}$.



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].

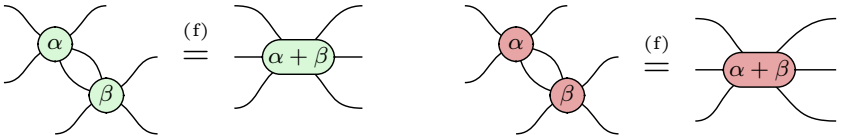


Figure 2.10: Spider fusion rule for Z spiders (left) and X spiders (right).

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.

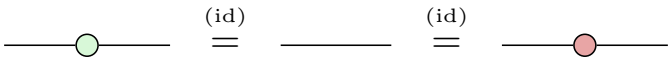
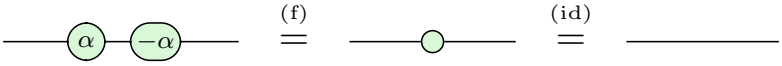


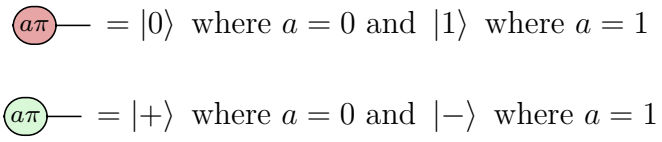
Figure 2.11: Identity removal rule.

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].



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.

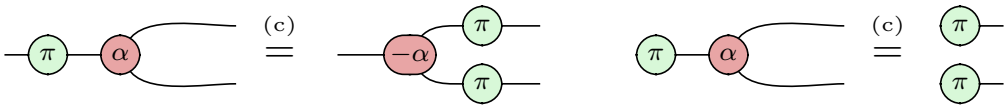


Figure 2.12: π copy (left) and state copy (right) rules for Pauli Z gate and Z eigenstates.

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.

Hadamard Rules

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

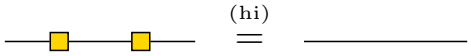


Figure 2.13: Hadamard self-inverse rule.

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

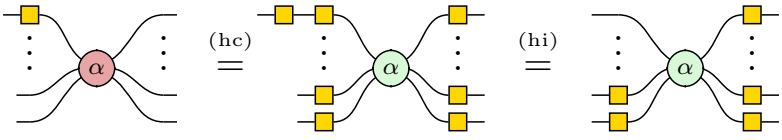


Figure 2.14: Hadamard commutation rule.

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.

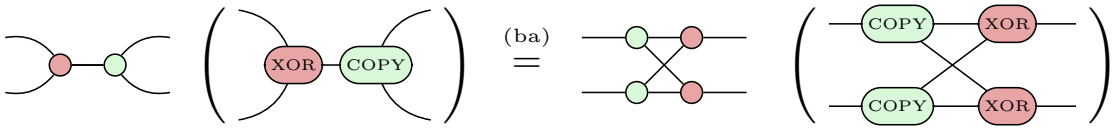


Figure 2.16: The bialgebra rule (and its classical motivation).

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*).

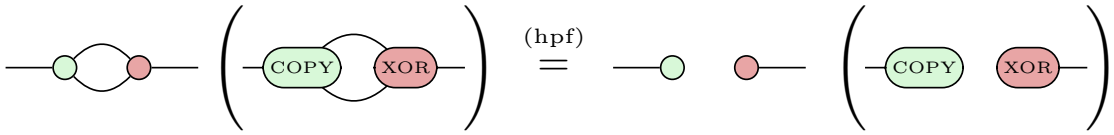
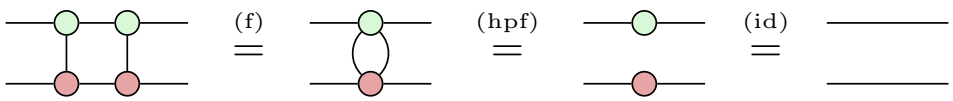


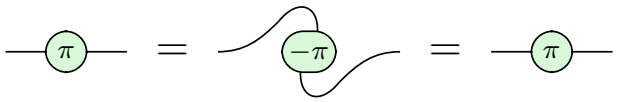
Figure 2.17: The Hopf rule (and its classical motivation).

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.



2.3 Clifford Conjugation

The ZX calculus is a diagrammatic language that makes use of the complementary observables in the Z and X bases. Since both of these bases are Hermitian, we can arbitrarily deform their wires as we see fit. For instance, it is easy to show that the Pauli Z gate (Z spider with phase π) is Hermitian, $Z = Z^\dagger$, by finding its transpose (converting its inputs into outputs and *vice verse*), then taking its complex conjugate (negating its phase $\pi \to -\pi = \pi$).



The Y basis, unlike the Z and X bases, is *not Hermitian*. Therefore, converting the inputs of a Y *Spider* into outputs and *vice verse* does *not* yield its transpose. Instead, we define rotations in the Y basis by conjugating Z and X spiders [3].

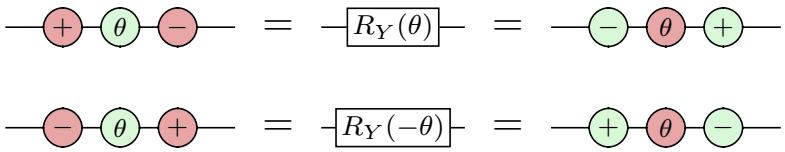


Figure 2.18: Conjugation of generators in the Z and X bases into the Y basis.

The single-qubit Clifford gates, $R_Z\left(\frac{\pi}{2}\right)$, $R_Z\left(\frac{3\pi}{2}\right)$, $R_X\left(\frac{\pi}{2}\right)$ and $R_X\left(\frac{\pi}{2}\right)$, as defined in Grier *et al* [23], are represented as follows in the ZX calculus.

Figure 2.19: Definition of the single-qubit Clifford gates.

Using the fact that the Clifford gates are unitary $C^{-1} = C^\dagger$, we expand our definition of the Y rotation to obtain the following commutation relations [3].

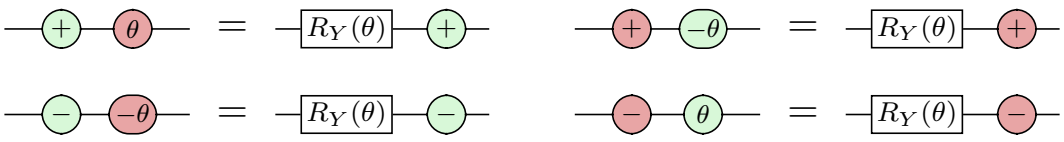


Figure 2.20: Single-qubit Clifford commutation relations.

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.

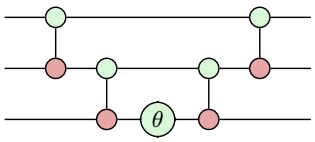
A Pauli string P is defined as a tensor product of Pauli matrices $P \in \{I, X, Y, Z\}^{\otimes n}$, where n is the number of qubits in the system. Each Pauli gate acts on a distinct qubit. Thus $Z \otimes X$ represents the Pauli Z and X gates acting on the first and second qubits respectively.

Stone's Theorem [24] 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.

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

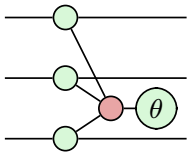
3.1 Phase Gadgets

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]. Using the identity (2.11), spider fusion (2.10) and bialgebra (2.16) rules, we are able to derive its representation in the ZX calculus.

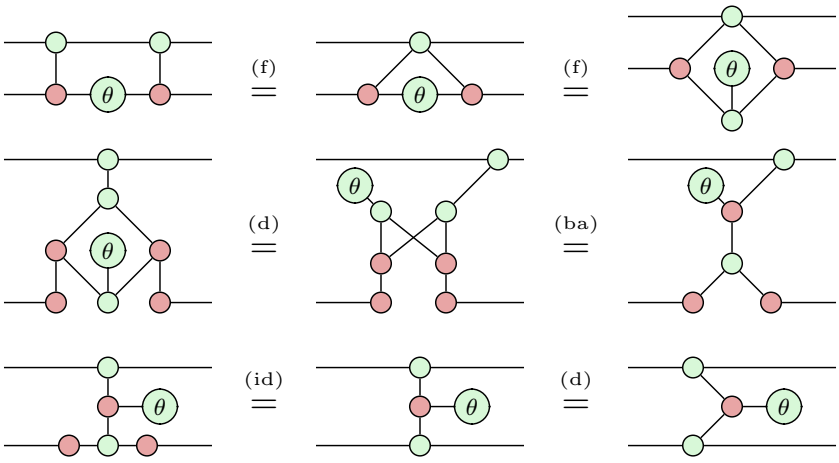
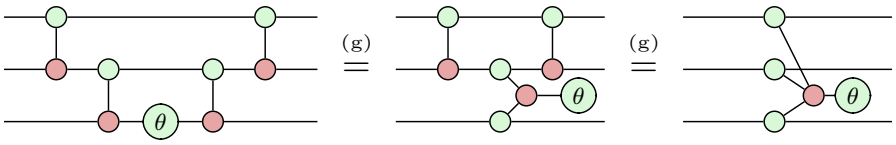


Figure 3.1: Phase gadget result.

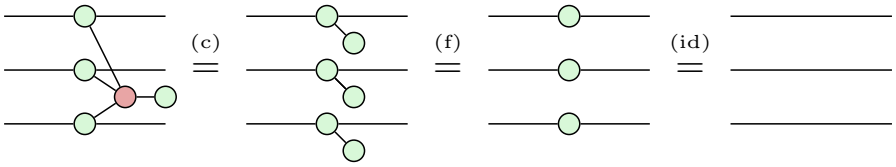
It is then a simple matter of recursively applying this result to phase gadgets in quantum circuit notation to generalise to arbitrary arity.



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

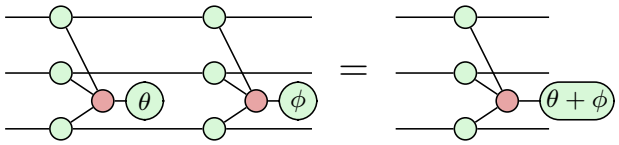
Phase Gadget Identity

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.



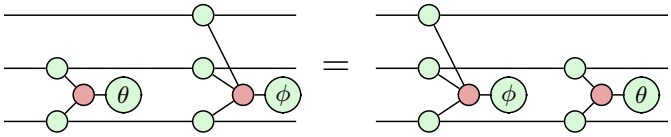
Phase Gadget Fusion

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).



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.

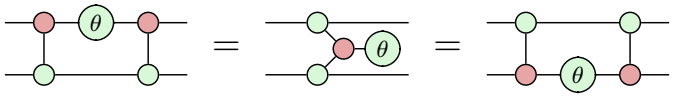


Figure 3.2: Phase gadget decomposition result.

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 [25].

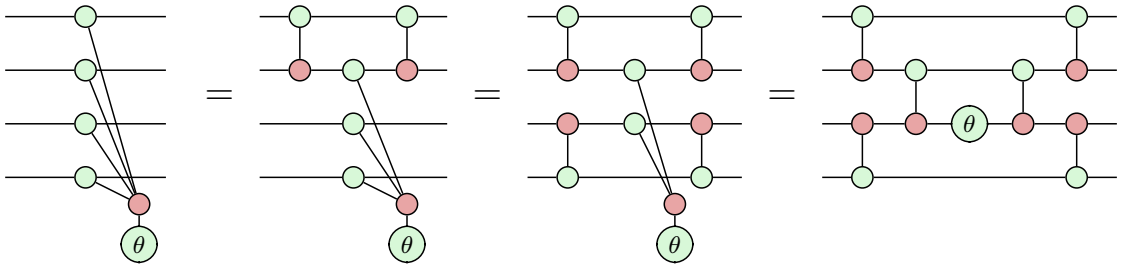


Figure 3.3: Balanced tree phase gadget decomposition.

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.

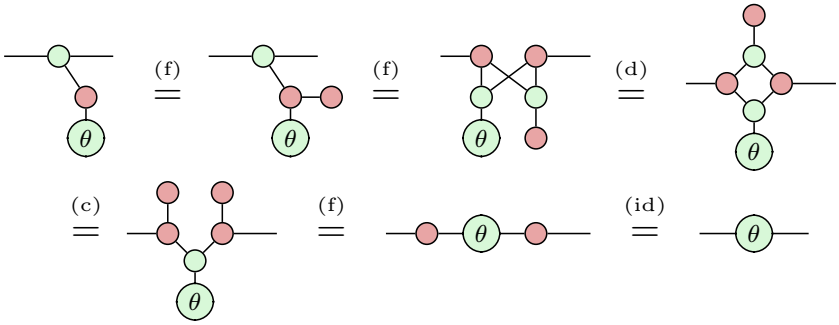
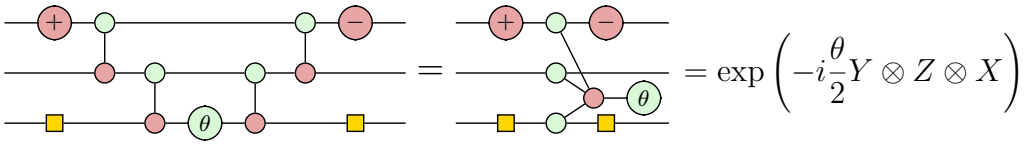


Figure 3.4: Single-legged phase gadget as a Z rotation.

3.2 Pauli Gadgets

Pauli gadgets are defined as the one parameter unitary groups of Pauli strings in the set $\{I, X, Y, Z\}^{\otimes n}$. By performing a Clifford conjugation (2.3) on the legs of a phase gadget, we obtain the corresponding Pauli gadget. That is, Pauli gadgets are phase gadgets associated with a change of basis. Hence, whilst phase gadgets alone cannot alter the distribution of the observed state, Pauli gadgets can [3].



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.

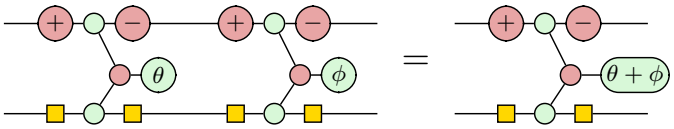


Figure 3.5: Pauli gadget fusion rule.

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

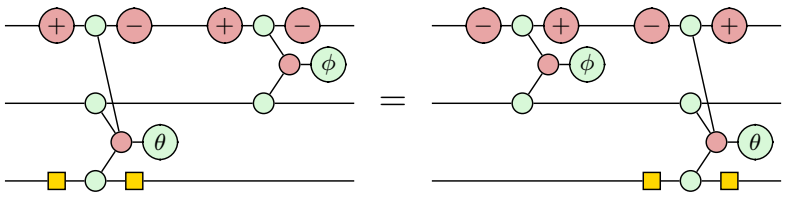
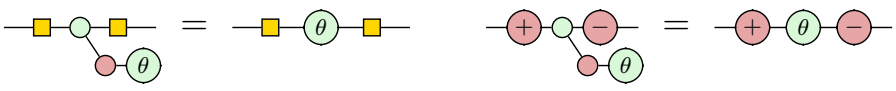


Figure 3.6: Pauli gadget commutation rule.

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



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 [25]. 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 [26], [27], [28], [29].

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.

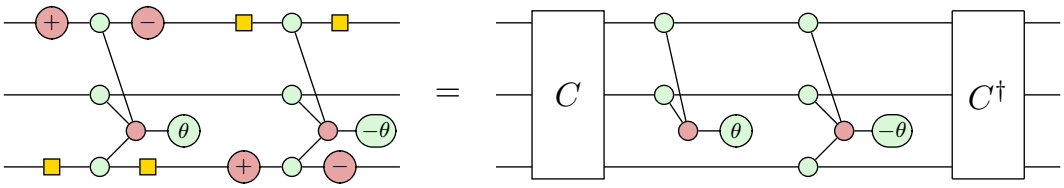
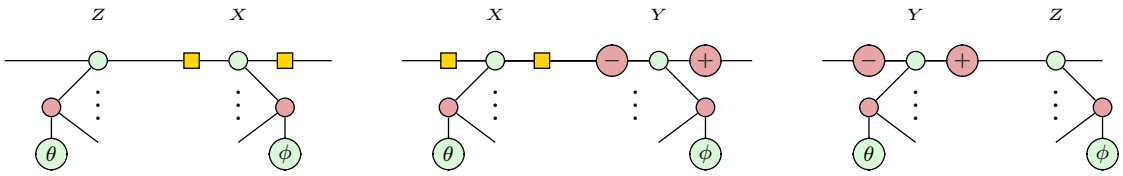


Figure 3.7: Diagonalisation of a pair of commuting Pauli gadgets by C.

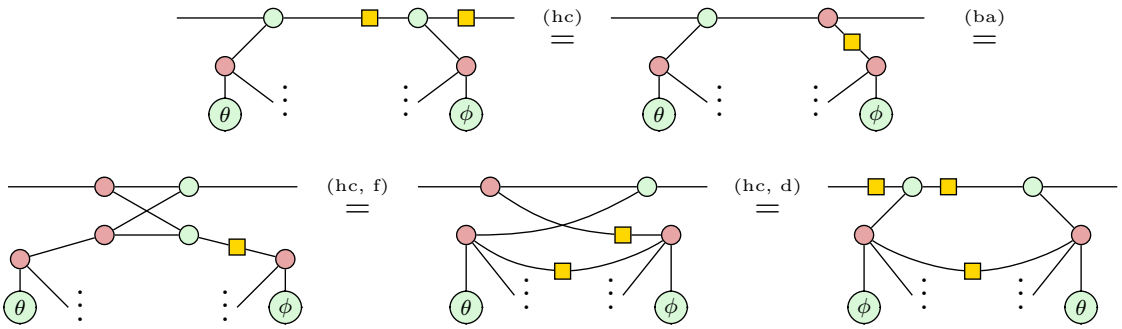
As we will see in Chapter 5, the excitation operators used in VQE algoriths consist of commuting sets of Pauli gadgets. Therefore, the quantum circuits implementing such excitation operators can be diagonalised with some Clifford subcircuit C. As stated in Cowtan *et al* [11], whilst diagonalisation 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

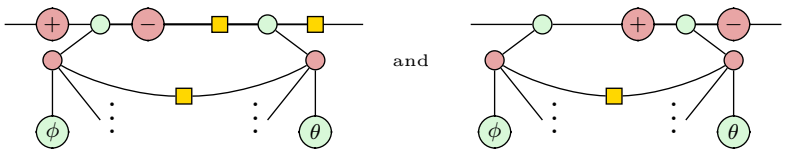
A pair of Pauli gadgets commute when their Hamiltonians commute [3]. That is, a pair of Pauli gadgets commute when the Pauli strings that they are defined by also commute. Diagrammatically, we have that a pair of Pauli gadgets commute when they *mismatch on an even number of legs* [3]. Let us demonstrate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



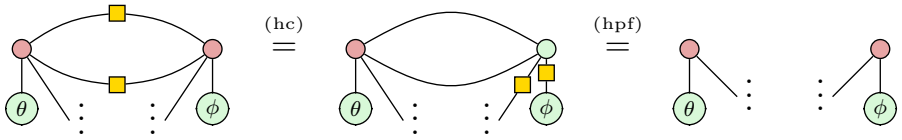
Starting with the mismatching Z/X pair and using the bialgebra (2.16), Hadamard commutation (2.14) and spider fusion (2.10) rules, we can show that commuting the gadgets' legs introduces a Hadamard between the bodies of the gadgets. [3].



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.

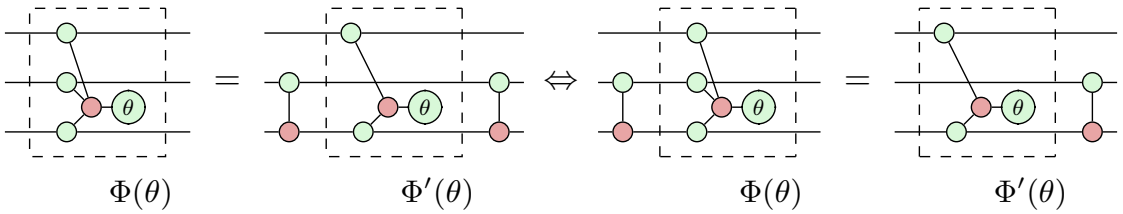


Clifford Commutation Relations

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 $\Phi'(\theta) = C^\dagger \Phi(\theta) C$.

Taking $C = \mathrm{CNOT}_{1,2}$, we can interpret the conjugation of a *phase gadget* diagrammatically (diagram on left) as the phase gadget decomposition result (3.2). Recalling that the members of the Clifford group are unitary transformations, $C^{-1} = C^\dagger$, we define *commutation relation* as $\Phi(\theta)C = C\,\Phi'(\theta)$ (diagram on right).

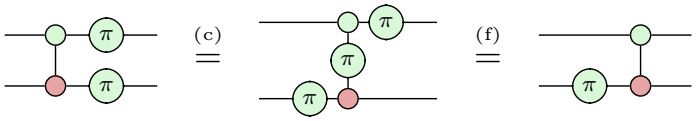


Since the CNOT and CZ gates act on two qubits ($n = 2$), we can form 16 unique Pauli gadgets from the set of Pauli strings $\{I, X, Y, Z\}^{\otimes 2}$. Consequently, we must derive 16 commutation relations to fully describe the interaction of Pauli gadgets with the CNOT and CZ gates.

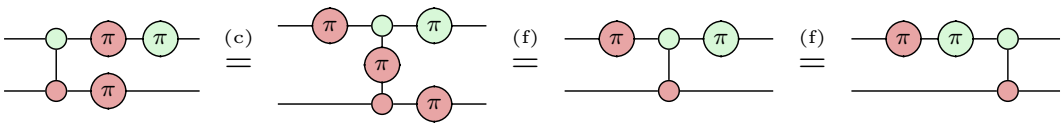
Whilst it is possible to derive each of these commutation relations directly, it can be shown, through the relevant Taylor expansion, that conjugating a Pauli gadget is equivalent to finding the one parameter unitary group of the corresponding conjugated Pauli string. In other words, identifying how 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 $\mathrm{CNOT}_{0,1}$ 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

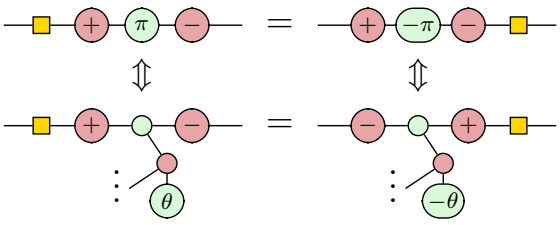
CNOT target using the π copy rule (2.12), then, we push the top Pauli Z gate through the CNOT control using the spider fusion rule (2.10), cancelling one of the copied Pauli Z gates in the process.



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.



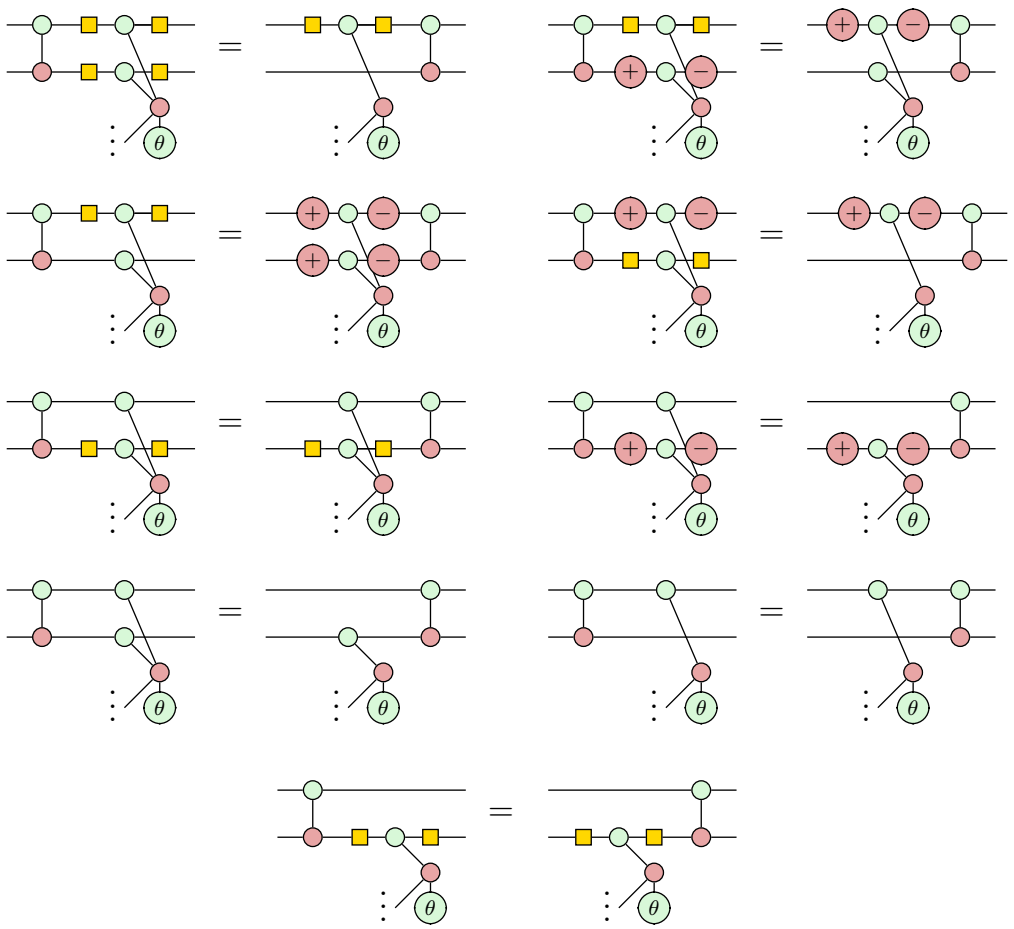


Figure 3.8: CNOT commutation relations excluding repetitions.

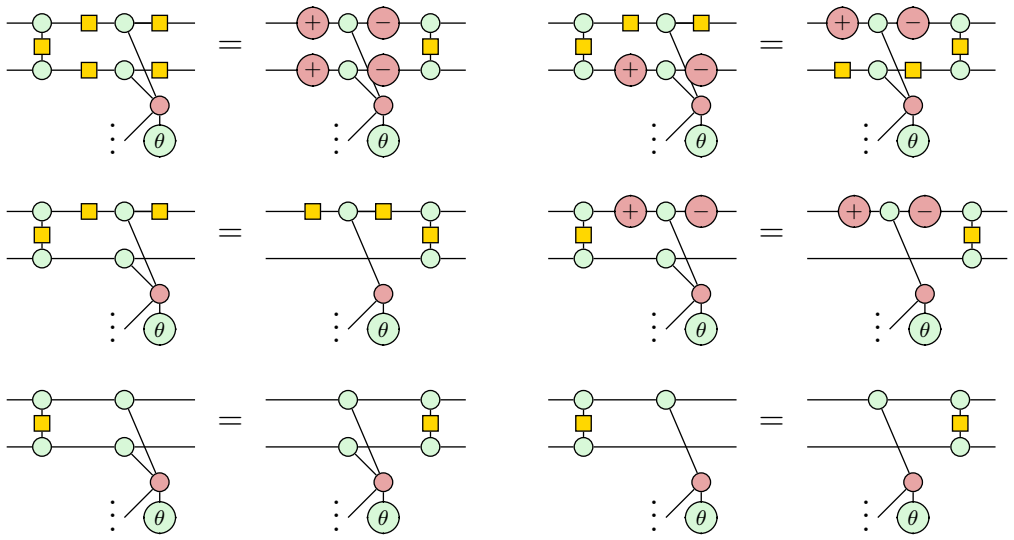


Figure 3.9: CZ commutation relations excluding repetitions.

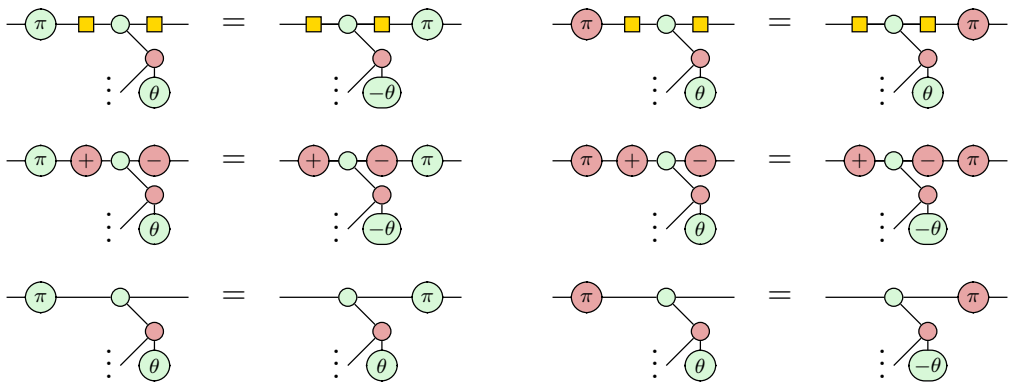


Figure 3.10: Pauli commutation relations.

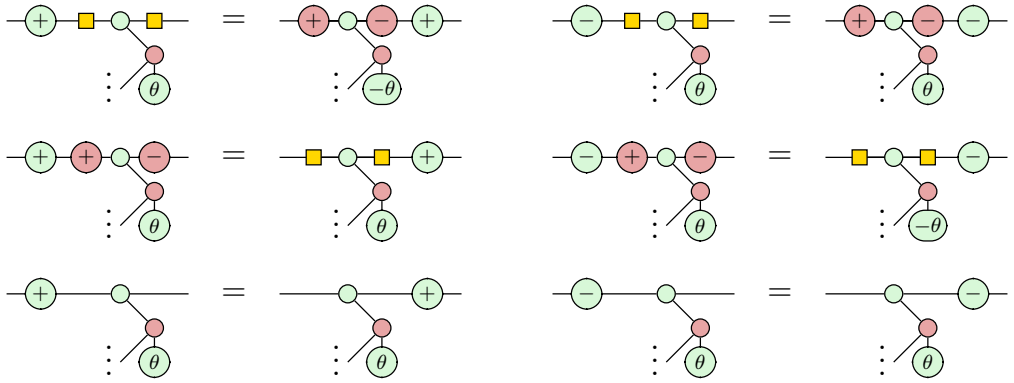


Figure 3.11: Clifford Z commutation relations.

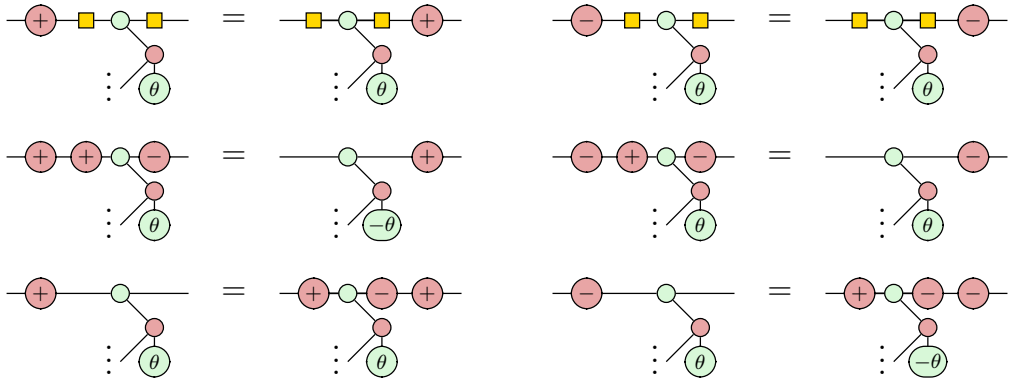


Figure 3.12: Clifford X commutation relations.

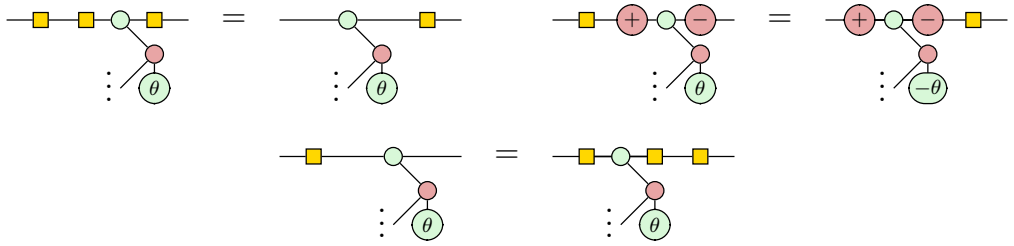


Figure 3.13: Hadamard commutation relations.

Chapter 4

Controlled Rotations

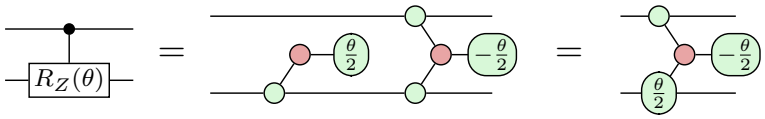
Controlled rotations play an important part in UCC ansätze representing fermionic systems. They can be used to account for the fermionic antisymmetry observed in fermionic systems by applying a phase (rotation) depending on the parity of the state. In other words, the rotation is *controlled* by the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

In this chapter, we develop a representation for higher-order controlled rotations in terms of phase polynomials, aiming to replicate the results described by Yordanov *et al* [10]. We begin by discussing the well-established representation of singly-controlled rotations the ZX calculus. Next, we demonstrate how these can be used to construct higher-order controlled rotations and show how they can be expressed in terms of phase polynomials.

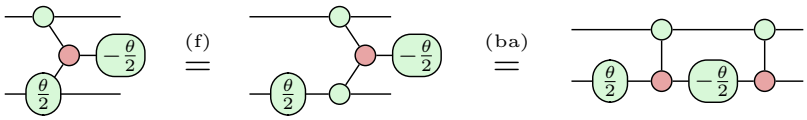
This approach was undertaken independently and the conclusions drawn in this section constitute a substantial portion of the research conducted in this thesis, requiring the use of the Clifford commutation relations developed in Section 3.4. In Chapter 5, we then use the representations of higher-order controlled rotations developed in this chapter to demonstrate the correspondence between excitation operators and controlled rotations.

4.1 Singly Controlled-Rotations

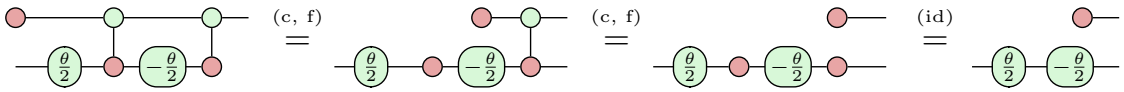
Starting with the singly-controlled Z rotation gate $\mathrm{CR}_Z(\theta)$, we will see that it can be expressed as a combination of phase gadgets, since it corresponds to a diagonal matrix in the computational Z basis [3].



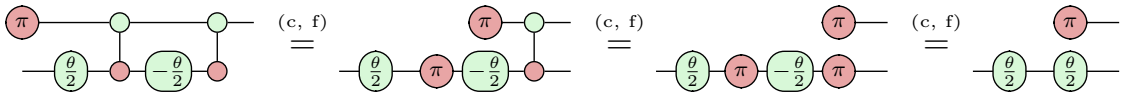
The $\mathrm{CR}_Z(\theta)$ gate applies a rotation to the target qubit if the control qubit is in the $|1\rangle$ state, and applies no rotation, if the control qubit is in the $|0\rangle$ state. This behaviour generalises to superpositions of states. Using the spider fusion (2.10) and bialgebra (2.16)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



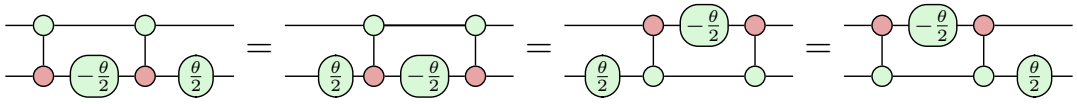
To verify that this circuit does indeed correspond to a controlled rotation in the Z basis, let us observe what happens when the control qubit is in the $|0\rangle$ state. Using the spider fusion (2.10), state copy (2.12) and identity (2.11) rules,



As expected, the resulting diagram fuses to give identity. Conversely, when the control qubit is in the $|1\rangle$ state, we obtain a Z rotation by θ.

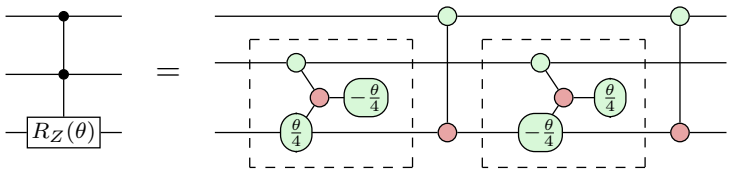


Using the the phase gadget fusion rule (3.1) and the phase gadget decomposition result (3.2), we can decompose the $\mathrm{CR}_Z(\theta)$ gate into four equivalent circuits.



4.2 Higher Order Controlled-Rotations

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



We can minimise the depth of the circuit implementing the $\mathrm{CCR}_Z(\theta)$ gate by choosing specific decompositions of the $\mathrm{CR}_Z(\theta)$ gate (left diagram) such that one pair of $\mathrm{CNOT}_{1,2}$ are adjacent and cancel one-another (right diagram).

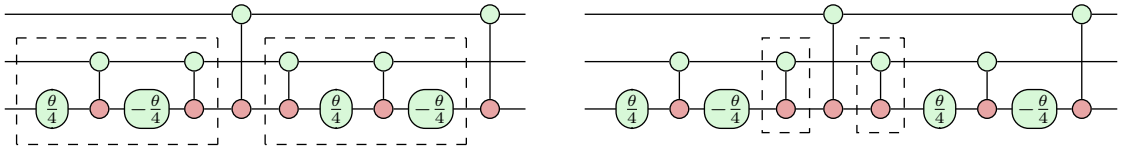
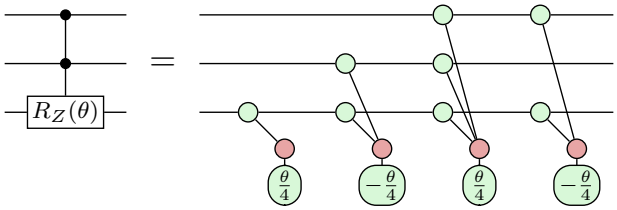
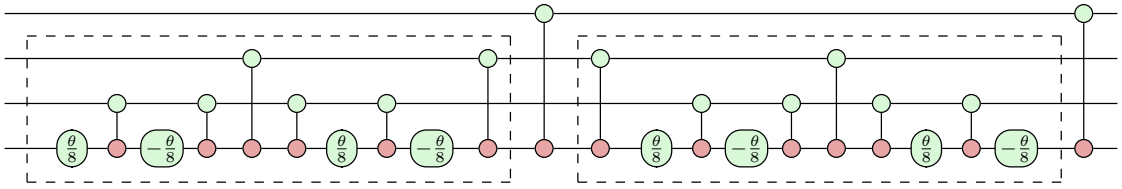


Figure 4.1: $\mathrm{CCR}_Z(\theta)$ circuit decomposition (left). Cancelling CNOT gate pair (right).

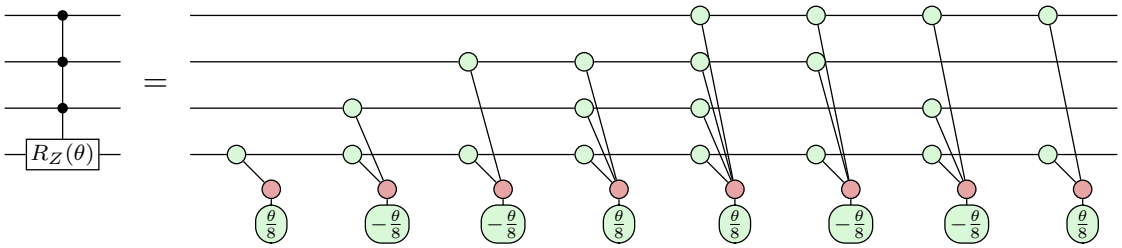
For the purposes of this thesis, we are interested in the form of controlled rotations in terms of Pauli gadgets. By expressing all Z rotations as phase gadgets (3.4) and commuting all CNOT gates through them such that the CNOT gates cancel, we obtain a phase polynomial consisting of four commuting phase gadgets.



More generally, we can build controlled rotations of arbitrary arity by recursively nesting controled rotations as above. Hence, as with the doubly-controlled rotation, we can construct triply-controlled rotations by nesting two doubly-controlled rotations. Below is one specific circuit implementation of a triply-controlled Z rotation that maximises the number of cancelling CNOT gates.



As before, we can show that the $\mathrm{CCCR}_Z(\theta)$ gate corresponds to a phase polynomial consisting of eight commuting phase gadgets.



Controlled Rotations in Different Bases

By conjugating the target qubit with the correct Clifford gate (2.3), we obtain a controlled rotation in the desired basis. Consider the following examples.

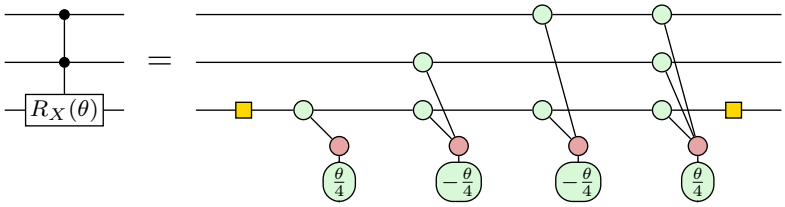


Figure 4.2: $\mathrm{CCR}_X(\theta)$ gate obtained by conjugating the $\mathrm{CCR}_Z(\theta)$ gate.

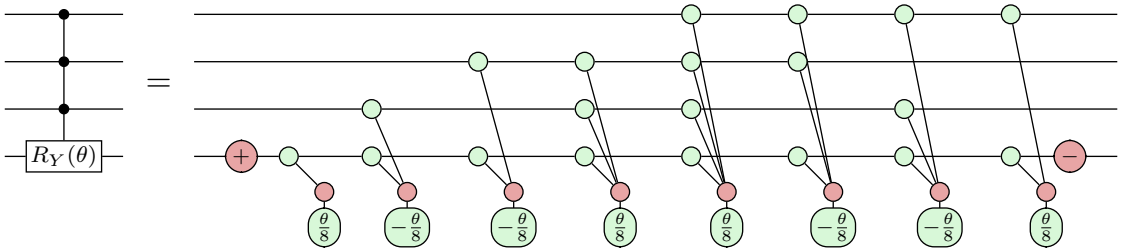


Figure 4.3: $\mathrm{CCCR}_Y(\theta)$ gate obtained by conjugating the $\mathrm{CCCR}_Z(\theta)$ gate.

Chapter 5

Excitation Operators

In this chapter, we will discuss how the excitation operators used to construct UCC ansätze can be implemented on a quantum computer. We will then use the ZX calculus to represent these excitation operators as Pauli gadgets and discuss a number of circuit optimisation strategies in the context of the DISCO-VQE algorithm. We will demonstrate how these excitation operators can be expressed in terms of controlled rotations (Chapter 4), as well as discuss their representation as phase polynomials following diagonalisation by some Clifford subcircuit.

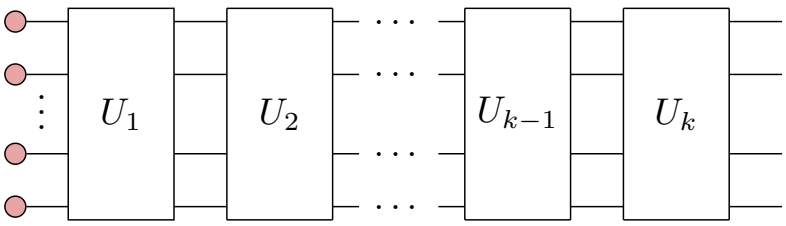


Figure 5.1: Example UCC ansatz approximating some fermionic ground state.

As stated in Cowtan *et al* [11], circuit optimisation amounts to pattern replacement, that is, recognising a subcircuit of a specific form and replacing it with an equivalent circuit that uses fewer quantum resources. Hence, by identifying the macroscopic structures representing these excitation operators, we facilitate the manipulation of large-scale structures in the quantum circuit, as well as identify the means by which they interact with one another.

5.1 Implementing Excitation Operators

In Section 1.3, we showed that any fermionic state can be represented by a finite sequence of k parametrised one-body and two-body excitation operators acting on a single-Slater determinant reference state. Hence, after invoking the Trotter formula using one time step, we obtained the following equation for the UCC ansatz.

$$U(\boldsymbol{\theta}) = \prod_{m=1}^{k} U_m(\theta_m) \qquad U_m(\theta_m) = e^{\theta_m(\tau_m - \tau_m^\dagger)}$$

Figure 5.2: UCC ansatz (sequence of k one-body and two-body excitation operators).

In order to implement the fermionic excitation operators $U_m(\theta_m)$ on a quantum computer, we must first represent the second-quantised creation and annhilation operators in terms of quantum gates. For a single-state system, we do this by taking the following linear combinations of the Pauli gates.

$$\hat{a}^\dagger = |1\rangle\langle 0| = \frac{1}{2}(X - iY) \qquad \hat{a} = |0\rangle\langle 1| = \frac{1}{2}(X + iY)$$

Figure 5.3: Jordan-Wigner transformation for a single-state system.

Recalling that the creation and annhilation operators must preserve the fermionic anti-symmetry imposed by the Pauli principle, we modify these equations to account for the parity of the spin orbitals preceding the target spin orbital j when dealing with many-state systems. We do this by introducing a string of Pauli Z operators to compute the parity of the spin orbitals preceding spin orbital j.

$$\hat{a}_j^+ = \frac{1}{2}(X - iY)\bigotimes_{k=1}^{j-1} Z_k \qquad \hat{a}_j = \frac{1}{2}(X + iY)\bigotimes_{k=1}^{j-1} Z_k$$

Figure 5.4: Jordan-Wigner transformation for a many-state system.

This mapping is known as the Jordan-Wigner transformation [30] and has the advantage of preserving a direct mapping between fermionic states in the occupation

number representation and the qubit state vector. In other words, each qubit in our quantum circuit now encodes the occupation number of each spin orbital in our fermionic state $|f_{n-1}\dots f_0\rangle \;\rightarrow\; |q_{n-1}\dots q_0\rangle$.

Let us take the unitary operator $U_p^q(\theta)$ to be a one-body excitation operator derived from the anti-Hermitian second-quantised operator $a_q^\dagger a_p - a_p^\dagger a_q$ and acting on qubits p and q. After applying the Jordan-Wigner transformation to the second-quantised operator and exponentiating it, we obtain the following one-body excitation operator in terms of the Pauli matrices.

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

Figure 5.5: One-body excitaiton operator expressed in terms of the Pauli matrices.

By applying the Jordan-Wigner transformation to the two-body excitation operator $U_{pq}^{rs}(\theta)$, derived from the second-quantised operator $a_r^\dagger a_s^\dagger a_q a_p - a_p^\dagger a_q^\dagger a_s a_r$, we obtain the following two-body excitation operator in terms of the Pauli matrices.

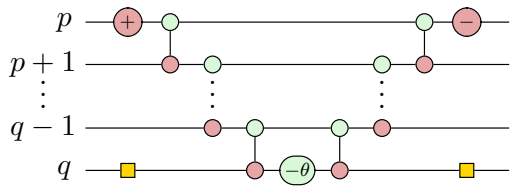
$$\begin{aligned} U_{pq}^{rs}(\theta) = \exp\Bigg(i\frac{\theta}{8}(&X_pX_qY_sX_r + Y_pX_qY_sY_r + X_pY_qY_sY_r + X_pX_qX_sY_r - \\ &Y_pX_qX_sX_r - X_pY_qX_sX_r - Y_pY_qY_sX_r - Y_pY_qX_sY_r)\prod_{k=p+1}^{q-1} Z_k \prod_{l=r+1}^{s-1} Z_l\Bigg) \end{aligned}$$

Figure 5.6: Two-body excitation operator expressed in terms of the Pauli matrices.

Now that we have expressed the one-body and two-body excitation operators in terms of quantum gates, we will discuss their specific implementations on a quantum computer. Starting with the one-body excitation operator, we can show that it can be expressed as the following product of commuting exponential terms.

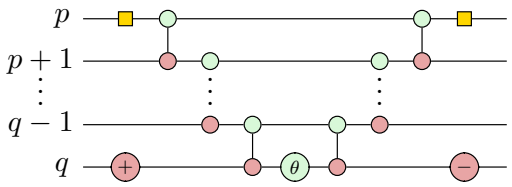
$$U_p^q(\theta) = \left(\exp\left[i\frac{\theta}{2}Y_pX_q\prod_{k=p+1}^{q-1} Z_k\right]\right)\left(\exp\left[-i\frac{\theta}{2}X_pY_q\prod_{k=p+1}^{q-1} Z_k\right]\right)$$

The first exponential term is implemented by the following quantum circuit.



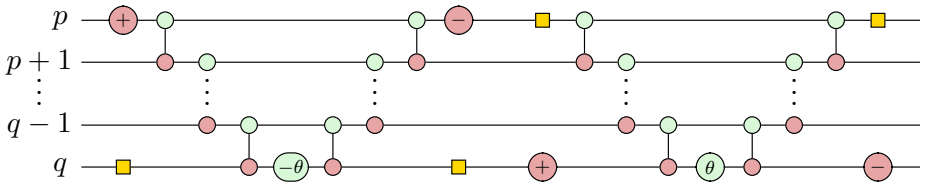
$$= \left(\exp \left[i \frac{\theta}{2} Y_p X_q \prod_{k=p+1}^{q-1} Z_k \right] \right)$$

And the second exponential term, as the following quantum circuit.

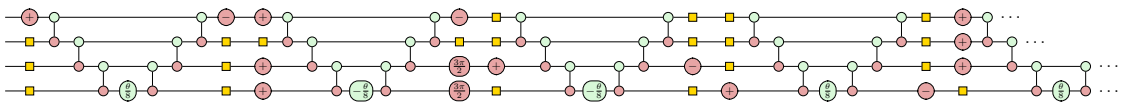


$$= \left(\exp \left[-i \frac{\theta}{2} X_p Y_q \prod_{k=p+1}^{q-1} Z_k \right] \right)$$

By sequentially composing these circuits, that is, taking their matrix product, we have implemented the one-body excitation operator acting on qubits p and q.



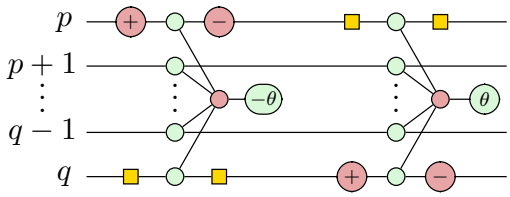
The CNOT ladder constructions arise from exponentiating the string of Pauli Z gates, and serve to compute the parity of the spin orbitals between p and q, ensuring that the correct phase is applied to the fermionic state, and therefore accounting for the fermionic anti-symmetry imposed by the Pauli principle. The two-body excitation operator can similarly be factorised into eight commuting exponential terms, and is implemented as eight subcircuits.



Although essential for accurately simulating fermionic systems, these CNOT ladder constructions are extremely resource-intensive and contribute significantly to the circuit depth. The poor fidelity of today's quantum (NISQ) devices results in an accumulation of error as circuit depth increases. Consequently, we are interested in optimising these excitation operators with respect to circuit depth.

5.2 Excitation Operators as Pauli Gadgets

Using the phase gadget result (3.1), we can show that the quantum circuit implementing a one-body excitation operator corresponds to two commuting Pauli gadgets the ZX calculus.



One immediate advantage of representing the one-body excitation operator in this form is that we can easily show that these Pauli gadgets do indeed commute by recognising that they have two mismatching pairs of legs (see Section 3.4). Another is that, from this intermediate form, we can resynthesise the quantum circuit using the balanced tree decomposition result (3.3), yielding a circuit depth of $2\log_2(n)$ rather than $2(n-1)$.

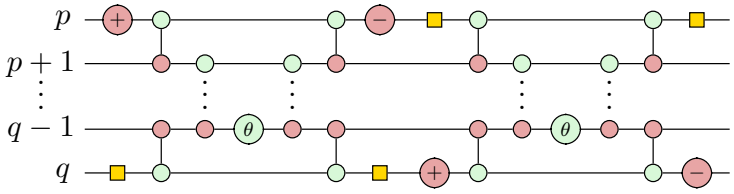
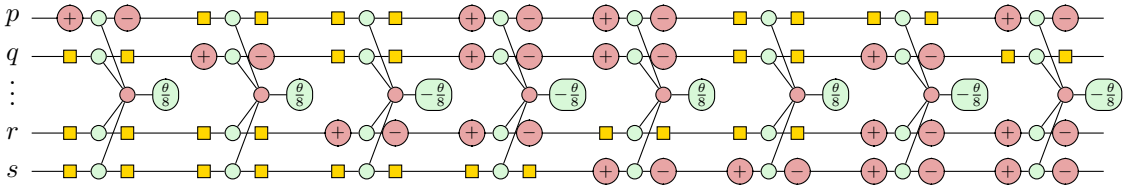


Figure 5.7: Balanced tree decomposition of a one-body excitation operator.

Similarly, we can show that the two-body excitation operator corresponds to eight commuting Pauli gadgets, each of which can be optimally resynthesised using the balanced tree decomposition result (3.3).



5.3 Commuting Excitation Operators

The DISCO-VQE algorithm generates multiple UCC ansätze, each yielding the same energy expectation value, but corresponding to a unique sequence of excitation operators. The identical energy expectation values of these UCC ansätze implies that they equivalently capture the correlation present in the molecular system of interest, suggesting that it may be possible to demonstrate an equivalence between them through algebraic manipulation. Motivated by this result, in this section we use the commutation rules developed in Section 3.4 to identify which excitation operators in the DISCO-VQE operator pool commute, and which do not.

We define *trivially commuting* excitation operators as commuting operators that act on a disjoint set of qubits, and *non-trivially commuting* excitation operators as commuting operators which act on an intersecting set of qubits. By expressing the excitation operators in the DISCO-VQE operator pool in terms of Pauli gadgets, we were able to identify several previously unknown non-trivially commuting pairs of excitation operators. We hope that by using these commutation relations, we might be able to demonstrate an equivalence between the UCC ansätze generated by the DISCO-VQE algorithm. Whilst we did not have sufficient time to attempt this, we leave it to future researchers to attempt. In the following graph, A – F represent one-body excitation operators and G – L represent two-body excitation operators. We have then drawn lines between commuting pairs.

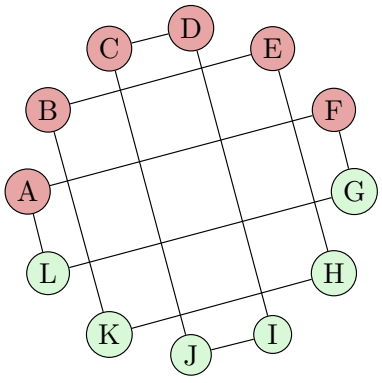


Figure 5.8: Graph identifying commuting operators used in the DISCO-VQE algorithm.

5.4 Excitations as Controlled Rotations

We began this research with the goal of identifying a generalised structure for the excitation operators used in the DISCO-VQE algorithm, in the ZX calculus. We anticipated that by doing so, we might discover novel ways of optimising UCC ansätze by developing a representation for these excitation operators that is independent of specific architectural constraints, as well as identifying rules describing their interactions.

This led us to the work done by Yordanov *et al* [10] and Kornell *et al* [31], which show that it is possible to reveal an underlying controlled rotation within each excitation operator by conjugating it with some subcircuit S. Hence, in this chapter, we replicate the results of Yordanov and Kornell in the ZX calculus using the results that we developed in previous chapters.

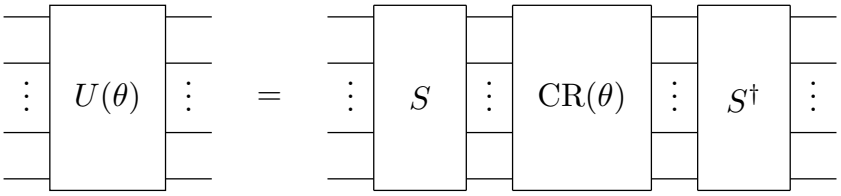


Figure 5.9: Revealling the controlled rotation $\mathrm{CR}(\theta)$ from the excitation operator $U(\theta)$.

Since there exist several efficient implementations of controlled rotations [32], [33], by rewriting the UCC ansatz in terms of controlled rotations, then substituting with a more efficient implementation, we can drastically reduce the circuit depth.

We begin by demonstrating the result in Yordanov *et al* by using a minimal one-body excitation operator, implemented by the following quantum circuit.

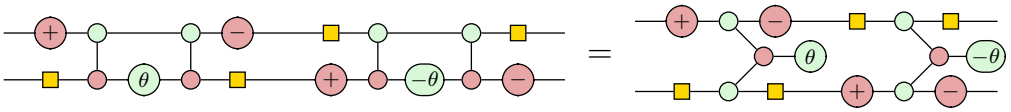
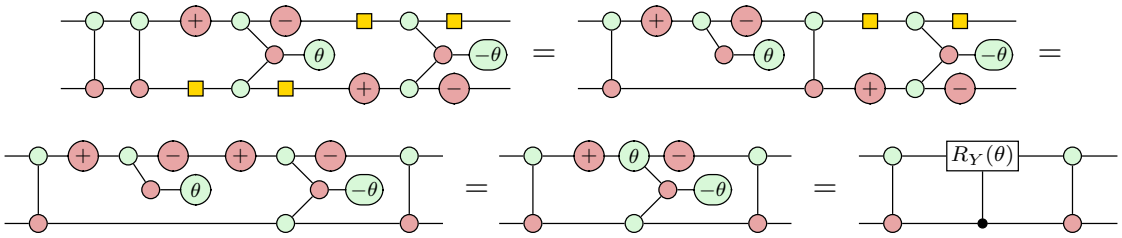


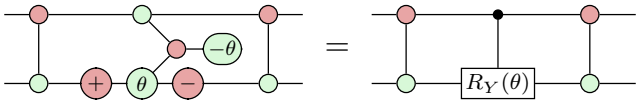
Figure 5.10: Quantum circuit (left) and in Pauli gadget form (right).

We can show that by inserting two adjacent $\mathrm{CNOT}_{0,1}$ gates (self-inverse) into the circuit and using the CNOT commutation rules derived in Chapter 3, the one-body

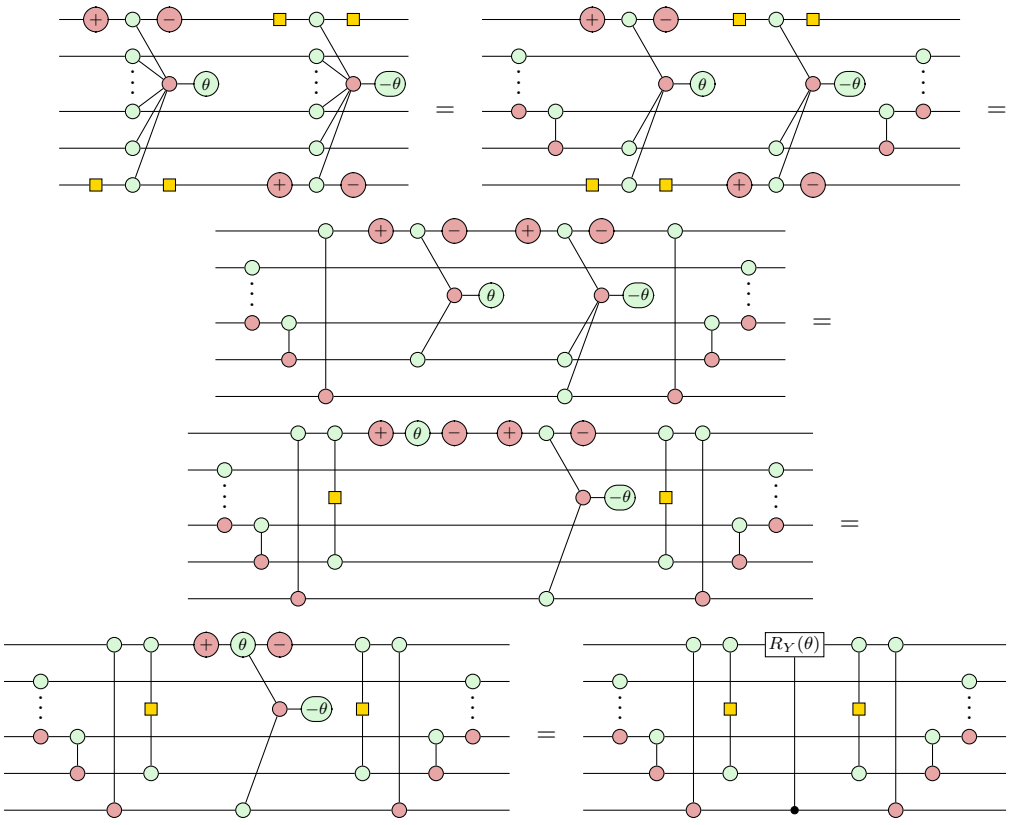
excitation operator can be expressed in terms of a singly-controlled Y rotation.



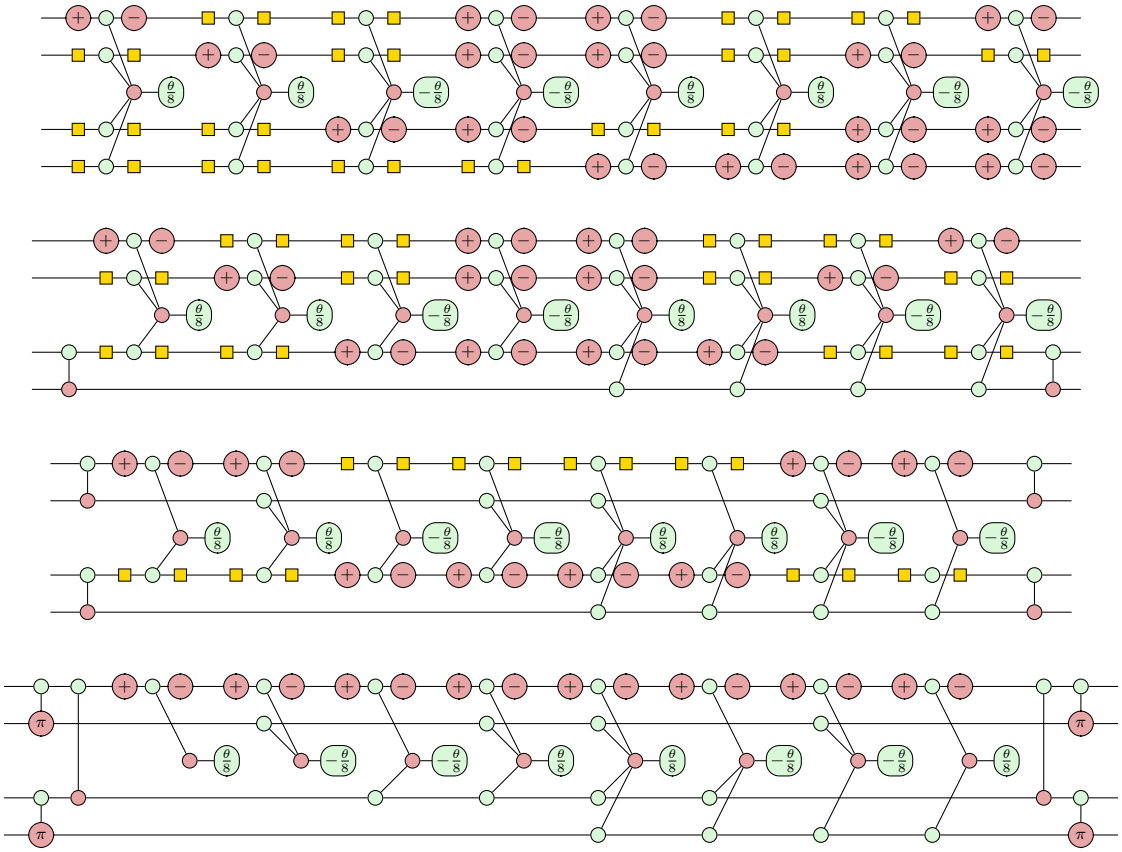
Had we instead chosen to insert two $\mathrm{CNOT}_{1,0}$ gates, we would have obtained the following controlled rotation, with the control on qubit 1.



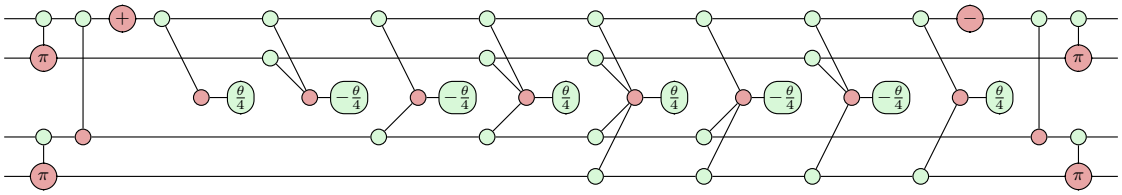
Let us now look at the general one-body excitation operator discussed in Section 5.2. We begin by eliminating the Z legs responsible for calculating the parity of the fermionic state, then, by expressing the single-legged Pauli gadget as a single-qubit rotation in the Y basis and fusing it with the adjacent Pauli gadget, we obtain the following controlled Y rotation.



Now let us consider the two-body excitation operator acting on qubits p, q, r and s. By conjugating the circuit with three CNOT gates as well as two Pauli X gates, we obtain the following.



Then, by cancelling adjacent Clifford gates, we obtain the a phase polynomial conjugated by Cliffords. Recognising this result as the triply-controlled rotation derived in Chapter 4, we have demonstrated the result in Yordanov *et al.*



Chapter 6

ZxFermion Software

Visit **github.com/aymannel/zxfermion** for the complete documentation.

Motivated by the need for an accessible tool to explore research ideas related to circuits of Pauli gadgets, we built the ZxFermion Python package for the visualisation and manipulation of circuits of Pauli gadgets. It is built on top of the PyZX `BaseGraph` API [34] and the Stim `Tableau` class [35].

ZxFermion provides classes to represent Pauli gadgets and common quantum gates and encodes the commutation relations developed in Section 3.4. It is designed to integrate with Jupyter notebook environments, enabling users to generate interactive ZX diagrams directly in the output cell. Thus, ZxFermion offers an accessible tool for studying the interaction of Pauli gadgets in the context of VQE algorithms.

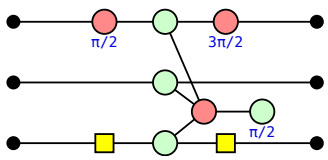
Using ZxFermion, we replicated all the commutation relations described in Section 3.4 and Section 3.4, as well as the proofs discussed in Chapter 5, showcasing a noteworthy acceleration in research pace. We anticipate that both chemists and computer scientists exploring quantum computing within the VQE framework will find this software tool advantageous.

Remark – *The ZxFermion package has also undergone thorough testing, ensuring its reliability and ease of use.*

6.1 Creating Gadgets and Circuits

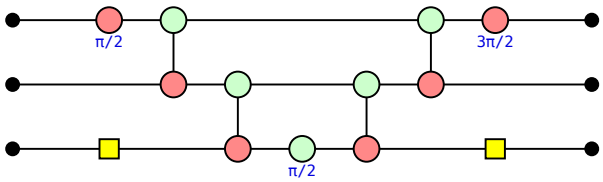
We begin by introducing the `Gadget` class, which we use to represent Pauli gadgets. The `Gadget` class takes a Pauli string and a phase as inputs. For instance, we instantiate the exp $\left[-i\frac{1}{4}\left(Y\otimes Z\otimes X\right)\right]$ gadget as follows.

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



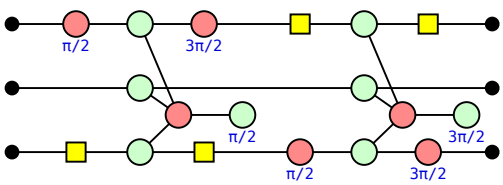
By setting `as_gadget=False`, we expand the gadget in quantum circuit notation.

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



We can construct a circuit of Pauli gadgets using the `GadgetCircuit` class, providing it with an ordered list of `Gadget()` objects. For instance, we implement a one-body excitation operator below, as discussed in Chapter 5.

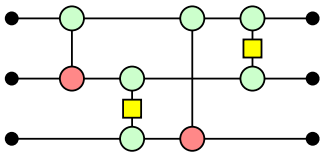
```
gadget1 = Gadget('YZX', phase=1/2)
gadget2 = Gadget('XZY', phase=3/2)
circuit = GadgetCircuit([gadget1, gadget2])
circuit.draw()
```



We will now introduce the classes implementing the standard quantum gates. As we will see in Section 6.2, ZxFermion encodes the logic describing the interaction of

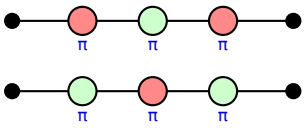
the these gates with Pauli gadgets, allowing us to replicate the results in Chapter 5. The CNOT and CZ gates are implemented by the `CX` and `CZ` classes respectively. We can specify the control and target qubits using the `control` and `target` parameters. When not specified, these parameters default to `control=0` and `target=1`.

```
gates = [CX(), CZ(1, 2), CX(0, 2), CZ(0, 1)]
circuit = GadgetCircuit(gates)
circuit.draw()
```



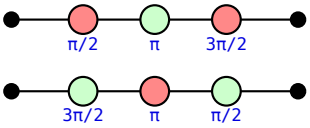
The Pauli Z and Pauli X gates are implemented by the `Z` and `X` classes respectively. We specify the target qubit using the `qubit` parameter (defaults to `qubit=0`).

```
gates = [X(), Z(0), X(0), Z(1), X(1), Z(1)]
circuit = GadgetCircuit(gates)
circuit.draw(stack=True)
```



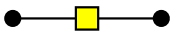
Similarly, the single-qubit Clifford gates are implemented by the `ZPlus`, `ZMinus`, `XPlus` and `XMinus` classes. Below we implement the $Y \otimes Y$ Pauli string (Section 2.3).

```
gates = [XPlus(), Z(0), XMinus(0), ZMinus(1), X(1), ZPlus(1)]
circuit = GadgetCircuit(gates)
circuit.draw(stack=True)
```



Finally, we have the Hadamard gate, which is implemented by the `H` class.

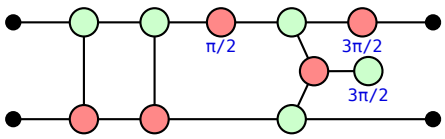
```
circuit = GadgetCircuit([H()])
circuit.draw()
```



6.2 Manipulating Circuits

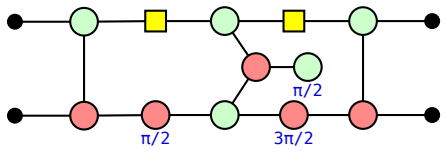
The `GadgetCircuit` class offers the ability to manipulate circuits containing Pauli gadgets via the `apply()` method. This method takes a quantum gate object as its input and inserts it, along with its adjoint, into the circuit. The `start` and `end` parameters allow us to specify the insertion positions.

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



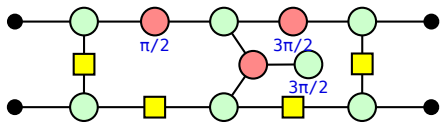
If no insertion positions are specified, the specified quantum gate and its adjoint are inserted at the start and at the end of the circuit respectively. The relevant commutation relations developed in Section 3.4 are then applied as required.

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



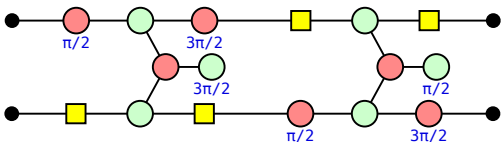
The `apply()` method is not limited to CNOT gates – we could have instead chosen any of the quantum gates mentioned in the previous section. The commutation logic uses Stim's `Tableau` class to construct the required Clifford tableau ensuring that the correct transformation is applied. Below we conjugate with the CZ gate.

```
gadget = Gadget('XY', phase=1/2)
circuit = GadgetCircuit([gadget])
circuit.apply(CZ(0, 1), draw=True)
```



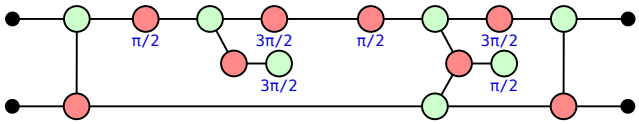
The `GadgetCircuit` class offers the ability to manipulate circuits containing multiple gadgets simultaneously. Below, we instantiate the minimal one-body excitation operator discussed in Figure 5.10.

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



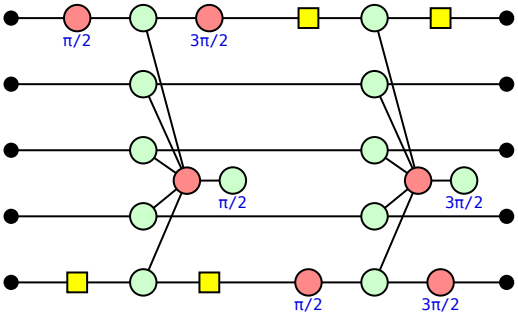
It is then simply a matter of conjugating with CNOT gates to replicate the derivation revealing a singly-controlled Y rotation, as shown in Section 5.4.

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



We can replicate the derivation that reveals a singly-controlled Y rotation from a one-body excitation operator shown in Section 5.4. We begin by instantiating the `GadgetCircuit()` object with the following lines of code.

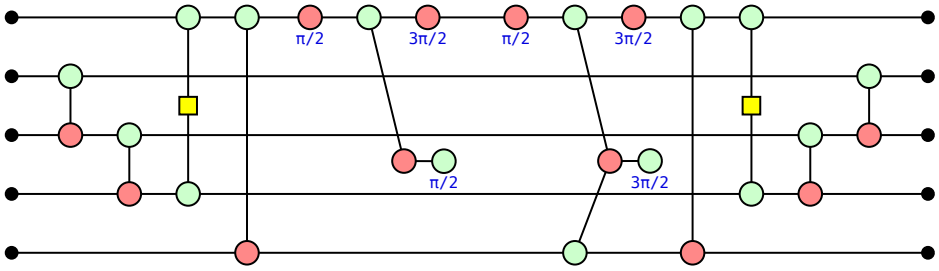
```
gadgets = [Gadget('YZZZX', phase=1/2), Gadget('XZZZY', phase=-1/2)]
circuit = GadgetCircuit(gadgets)
circuit.draw(as_gadgets=True)
```



Then, using the following lines of code, we eliminate the Z legs responsible for calculating the parity of the fermionic state, and reveal the two Pauli gadgets, that

together, correspond to a singly-controlled Y rotation.

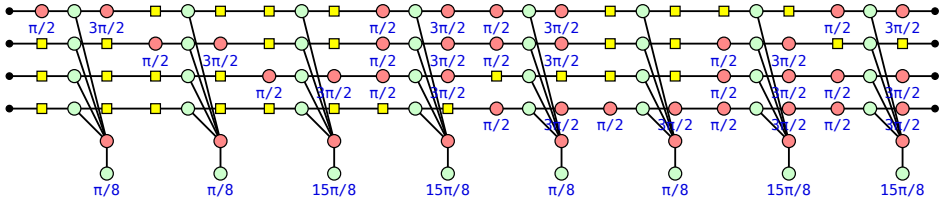
```
circuit.apply(CX(1, 2), start=0, end=2)
circuit.apply(CX(2, 3), start=1, end=-1)
circuit.apply(CZ(3, 0), start=2, end=-2)
circuit.apply(CX(0, 4), start=3, end=-3)
circuit.draw()
```



Similarly, we can replicate the corresponding derivation for the two-body excitation operator. We begin by instantiating the `GadgetCircuit()` object as follows.

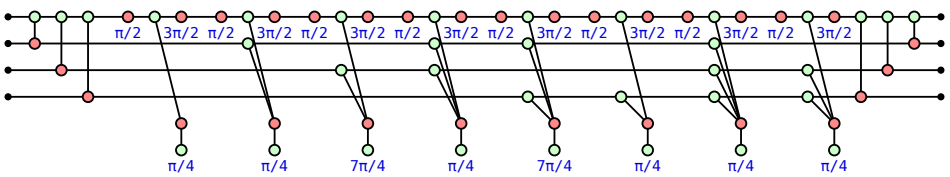
```
gadgets = [
    Gadget('YXXX', phase=1/8),  Gadget('XYXX', phase=1/8),
    Gadget('XXYX', phase=-1/8), Gadget('YYYX', phase=-1/8),
    Gadget('YYXY', phase=1/8),  Gadget('XXXY', phase=1/8),
    Gadget('XYYY', phase=-1/8), Gadget('YXYY', phase=-1/8)
]

circuit = GadgetCircuit(gadgets)
circuit.draw()
```



Then, using the following lines of code, we diagonalise the diagram, revealing the eight Pauli gadgets that togther constitute the triply-controlled Y rotation.

```
circuit.apply(CX(0, 3), draw=False)
circuit.apply(CX(0, 2), draw=False)
circuit.apply(CX(0, 1), draw=True)
```



Chapter 7

Conclusion

7.1 Summary

7.2 Future Work

- diagonalise commuting pairs of operators

Appendices

Hadamard

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

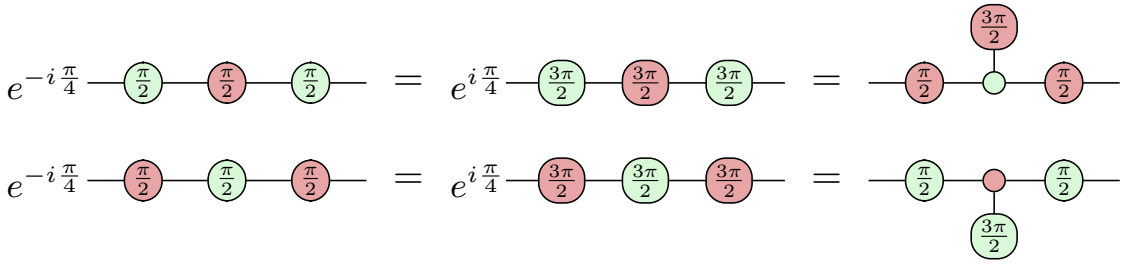
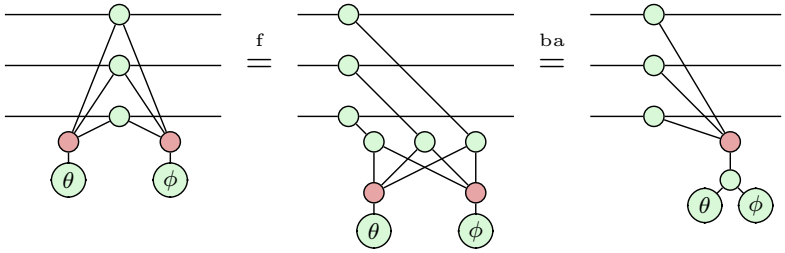


Figure 1: Equivalent definitions of the Hadamard generator.

Phase Gadgets

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



Bibliography

[1] Szalay, P. G., Müller, T., Gidofalvi, G., Lischka, H. & Shepard, R. Multiconfiguration 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).

[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] Nielsen, M. A. & Chuang, I. L. *Quantum Computation and Quantum Information: 10th Anniversary Edition* (Cambridge University Press, 2012).

[21] 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`.

[22] van de Wetering, J. Zx-calculus for the working quantum computer scientist (2020). `2012.13966`.

[23] Grier, D. & Schaeffer, L. The classification of clifford gates over qubits. *Quantum 6, 734 (2022)* **6**, 734 (2016). `1603.03999`.

[24] Stone, M. H. On one-parameter unitary groups in hilbert space. *The Annals of Mathematics* **33**, 643 (1932).

[25] Cowtan, A., Dilkes, S., Duncan, R., Simmons, W. & Sivarajah, S. Phase gadget synthesis for shallow circuits (2019). `1906.01734`.

[26] 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).

[27] 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).

[28] 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).

[29] Maslov, D. & Roetteler, M. Shorter stabilizer circuits via bruhat decomposition and quantum circuit transformations. *IEEE Transactions on Information Theory* **64**, 4729–4738 (2018).

[30] Seeley, J. T., Richard, M. J. & Love, P. J. The bravyi-kitaev transformation for quantum computation of electronic structure. *The Journal of Chemical Physics* **137** (2012). 1208.5986.

[31] Kornell, A. & Selinger, P. Some improvements to product formula circuits for hamiltonian simulation (2023). 2310.12256.

[32] Zomorodi-Moghadam, M. & Navi, K. Rotation-based design and synthesis of quantum circuits. *Journal of Circuits, Systems and Computers* **25**, 1650152 (2016).

[33] Ye, M.-Y., Zhang, Y.-S. & Guo, G.-C. Efficient implementation of controlled rotations by using entanglement. *Physical Review A* **73**, 032337 (2006).

[34] Kissinger, A. & van de Wetering, J. Pyzx: Large scale automated diagrammatic reasoning. *Electronic Proceedings in Theoretical Computer Science* **318**, 229–241 (2020).

[35] Gidney, C. Stim: a fast stabilizer circuit simulator. *Quantum* **5**, 497 (2021).