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.

Acknowledgements

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. The exact solution to this problem scales exponentially with the number of electrons, making it computationally intractable for large or strongly-correlated systems on classical computers, which cannot efficiently store the increasingly large wavefunctions [1]. In contrast, gate-based quantum computing offers the potential to represent electronic wavefunctions with polynomially scaling resources using quantum algorithms for chemical simulations [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, significant advancements in quantum computing hardware and software have brought us closer to the ability to simulate molecular systems. Despite these advancements, we are still in the Noisy Intermediate Scale Quantum (NISQ) era [4], characterised by challenges such as poor qubit fidelity, low qubit connectivity, and limited coherence times. The NISQ era represents a transitional phase in quantum computing, where quantum devices are not yet error-corrected but can perform computations beyond the capabilities of classical computers. Overcoming the limitations of the NISQ era is crucial for realising the full potential of quantum computing in various fields, including chemistry and materials science.

In this thesis, we focus on the Unitary Product State (UPS) ansatz which we use to represent fermionic wavefunctions on a quantum computer [5] via the Variational Quantum Eigensolver (VQE) algorithm [6]. Specifically, we study the excitation

operators used to prepare UPS ansätze using the ZX calculus, a diagrammatic language for reasoning about quantum processes [7]. The VQE algorithm is 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, and then optimising the wavefunction parameters classically until the energy converges to the best approximation of 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] and Cowtan *et al* [10] on Pauli gadgets and Yordanov *et al* [11] and Kornell *et al* [12] on fermionic excitation operators. Our research focuses on two main questions: Can we use the ZX calculus to gain insights into the structure of ansätze within VQE algorithms for quantum chemistry? Secondly, in the context of NISQ devices, can we use these insights to develop better ansätze with reduced circuit depth and more efficient resource usage? By attempting to reduce circuit depth, we address a major source of error in NISQ devices – the noise of today's quantum hardware [10].

In Chapter 1, we introduce the theoretical foundation necessary for simulating molecules on quantum computers. In Chapter 2, we present the ZX calculus, its generators, and its rewrite rules. Chapter 3 introduces Pauli gadgets, the building blocks for fermionic ansätze. In Chapter 4, we develop a representation for controlled rotations using the ZX calculus, applying this to replicate the work of Yordanov *et al.* [11] and Kornell *et al.* [12] using the ZX calculus. In **Chapter 5**, we combine the concepts discussed in the previous chapters to study the excitation operators used in the UPS ansatz, integrating the research ideas presented in this thesis. Finally, in Chapter 6, we introduce the ZxFermion software package, which we developed to facilitate the study of circuits of Pauli gadgets, and demonstrate how it can be used to replicate our findings.

Contents

Contents

Chapter 1

Background

In this chapter, we 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 start by giving an overview of Electronic Structure Theory in Section 1.2. We then introduce the Fundamentals of Quantum Computation in Section 1.3. Finally, we introduce the Variational Quantum Eigensolver (VQE) algorithm in Section 1.4.

1.1   Context & Motivation

Using the ZX calculus, this thesis focuses on the diagrammatic representation of excitation operators used to account for the correlation in fermionic systems. The ZX calculus is diagrammatic language for reasoning about quantum processes [7] that has recently shown an increased usage in quantum computing and quantum simulation. The research represented in this thesis is conducted within the framework of the *Variational Quantum Eigensolver (VQE)*, a promising hybrid quantum-classical algorithm for achieving quantum advantage on *Noise Intermediate-Scale Quantum (NISQ)* devices [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. The *Unitary-Product State (UPS)* ansatz, derived from the *Unitary Coupled Cluster (UCC)* formulation of the wavefunction, allows us to implement parametrised representations of fermionic states on a quantum computer.

The VQE algorithm generates multiple distinct UPS ansätze, each yielding the same energy expectation value. Hence, while different sequences of unitary excitation operators are employed to rotate the reference state to a state approximating the true ground state, their identical energy expectation values suggests that they equivalently capture the correlation present in the ground state, implying that that it may be possible to demonstrate the equivalence between these UPS ansätze through algebraic manipulation.

In this context, the research presented in this thesis focuses on the diagrammatic representation of unitary excitation operators using the ZX calculus. 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. 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 UPS ansätze and the nature of correlations in molecular quantum systems. This broader understanding could potentially lead to more efficient and effective quantum simulation algorithms. Furthermore, due to the poor fidelity of today's quantum computers, reducing the depth of quantum circuits is crucial to minimise errors in simulations. By identifying a generalised structure for the excitation operators used in the UPS ansatz, we aim to uncover novel methods for optimising ansätze that represent fermionic wavefunctions.

This led us to the work by Yordanov *et al* [11] and Kornell *et al* [12], which show that excitation operators can be rewritten in terms of controlled rotations, as well as the work by Cowtan *et al.* [10], showing that commuting sets of Pauli gadgets can be diagonalised and optimally resynthesised. In this thesis, we demonstrate the correspondence between excitation operators and controlled rotations using the ZX calculus, potentially informing improved optimisation strategies.

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. Furthermore, the ZX calculus, despite its immense promise as a next-generation framework for studying quantum computing, has seen relatively low usage by quantum chemists. Therefore, we hope that this thesis, along with the tools developed herein, will help to lower the barrier to entry for future Master's and PhD students interested in quantum computing. By providing a solid theoretical foundation and practical insights, we aim to facilitate a smoother transition and foster greater interest in this rapidly developing field.

To summarise, we aim to identify a generalised representation of excitation operators within the ZX calculus to gain insights into the nature of correlations in molecular quantum systems. Then, using these insights, we seek to rationalise the expectation energy-equivalent outputs of VQE algorithms and discover more efficient implementations of UPS ansätze. Finally, we hope to lower the barrier to entry for quantum computing in the context of quantum chemistry.

1.2   Electronic Structure Theory

Electronic Structure Problem

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

The full molecular Hamiltonian describes all of the interactions within a system of N interacting electrons and M nuclei. 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. Following this, we obtain 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 1.1: Electronic molecular Hamiltonian in atomic units.

The first term corresponds to the kinetic energy of the N electrons in the system, the second term corresponds to the pairwise attractive Coulombic interactions between the N electrons and M nuclei and the final term corresponds to the repulsive Coulombic interactions between N electrons in the system. Throughout the remainder of this text, we 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 be calculated by solving the electronic Schrödinger equation and including a constant nuclear term.

Many-Electron Wavefunctions

The many-electron wavefunction, which describes all the electrons 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 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 [14].

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

Figure 1.2: Slater determinant representing an antisymmetrised N-electron wavefunction.

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 [14]. 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.

Second Quantisation

In second quantisation, both observables and states are represented by creation and annihilation operators. Unlike the standard formulation of quantum mechanics, these operators inherently incorporate Bose or Fermi statistics, eliminating the need to track symmetrised or antisymmetrised products of single-particle wavefunctions. Put differently, the antisymmetry of an electronic wavefunction follows from the algebra of these operators, which greatly simplifyies the discussion of many identical interacting fermions [15], [16].

The Fock space is a linear abstract vector space spanned by N orthonormal occupation number vectors, each representing a single Slater determinant [15]. 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 representation of fermionic states 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.

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

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 [15].

Operators in second quantisation are constructed from the creation and annihilation operators $a_j^\dagger$ and a_j, where the subscripts i and j denote the spin orbital. $a_j^\dagger$ and a_j are one another's Hermitian adjoints, and are not self-adjoint [15]. 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. This phase factor is automatically kept track of by the creation and annhilation operators' anticommutation relations [15].

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

Figure 1.3: Anticommutation relations of the creation and annhilation operators.

The Hamiltonian in second quantisation can be expressed as follows, 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.

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

Figure 1.4: Hamiltonian in second quantisation.

These matrix elements are then computed classically, allowing us to simulate only the inherently quantum aspects of the problem on a quantum computer.

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

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

1.3   Fundamentals of Quantum Computating

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

Introduction to Qubits

Classical computation encodes information using binary strings formed from the 0 and 1 computational basis. 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 Z eigenbasis, corresponding to the $|0\rangle$ and $|1\rangle$ states, forms the standard computational basis for encoding information on a quantum computer.
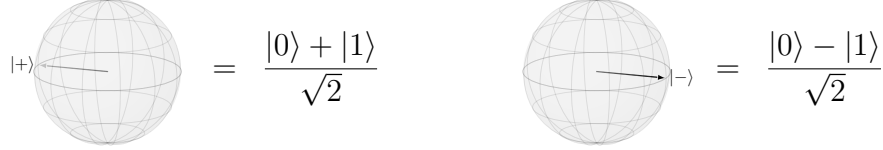
Figure 1.5: Z eigenbasis.

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 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, allowing us to represent $|\psi\rangle$ using only three real numbers. Furthermore, since $|\psi\rangle$ is normalised, we can describe it using two real phases as follows $|\psi\rangle = \cos\left(\frac{\theta}{2}\right) + e^{i\phi} \sin\left(\frac{\theta}{2}\right) |1\rangle$. This representation allows us to derive a three-dimensional representation for arbitrary qubit states known as the Bloch sphere. The Bloch sphere is a unit sphere in which opposite points correspond to mutually orthonal states. At one end of the Z axis, we have the $|0\rangle$ state, and at the other end, we have the $|1\rangle$ state (Figure 1.5).

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

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

Figure 1.6: X eigenbasis.

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 basis state, or the other. This result gives rise to the *no-cloning theorem*, which states that we cannot create indentical and independent copies of arbitrary qubit states, as this would first involve 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 1.7: 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, $U^{-1} = U^\dagger$ [17]. 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 1.8: 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 is a member of the Clifford group that 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.

 $= \dfrac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$
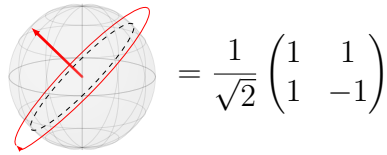
Figure 1.9: Bloch sphere representation of the Hadamard gate.

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.
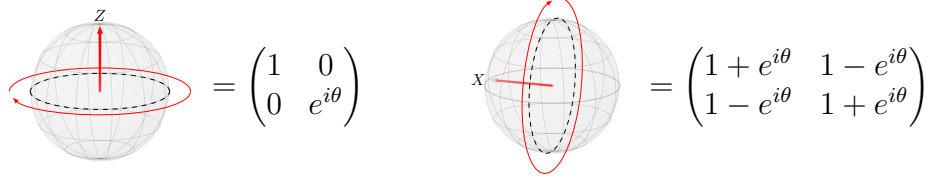
Figure 1.10: Bloch sphere representations of arbitrary Z and X rotations.

Multiple-Qubit Gates

The CNOT gate is a two-qubit member of the Clifford group which we can use to entangle qubits. Consider a two-qubit state of the form $|\alpha\rangle \otimes |\beta\rangle$. Taking the first qubit (α) to be the control, and the second qubit (β) to be the target, 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 CNOT gate acts on the two-qubit basis states (1.7) 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.*

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

Figure 1.11: 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, applying CNOT gate to the $|+0\rangle$ state, letting the first qubit be the control qubit ($|+\rangle$), yields the following Bell state.

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

1.4　The Variational Quantum Eigensolver

In this chapter, we introduce the *Variational Quantum Eigensolver (VQE)* algorithm, a particular class of variational quantum algorithms developed by Peruzzo and McClean *et al* [18] in 2014 to estimate the ground state energy of molecular systems. We then introduce the *Unitary Product State (UPS) ansatz* which we use to represent fermionic wavefunctions on quantum devices. Finally, we present the DISCO-VQE algorithm, a variant of the VQE algorithm that serves as the framework for the research presented in this thesis.

The VQE algorithm consists of a quantum subroutine, a *Parametrised Quantum Circuit (PQC)*, that implements some UPS ansatz, and a classical subroutine that makes use of the variational principle to classically optimise the 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 [3].

The VQE input state is some reference state, which in the case of the single-Slater determinant Hartree-Fock state, corresponds to a single vector in the Fock space. The output state is then some linear combination of vectors in the Fock space, that captures the correlation present in 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 ansatz. The quantum subroutine computes the energy expectation value of the ansatz via a quantum circuit consisting of the ansatz itself and the Hamiltonian for the system.
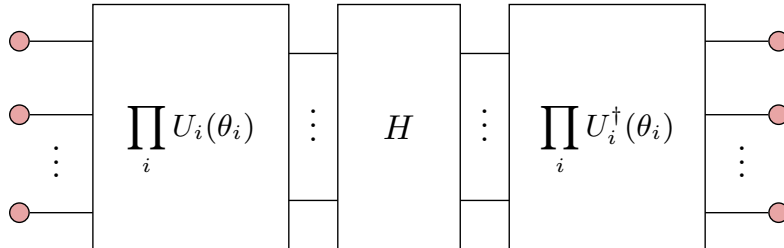
$$E(\boldsymbol{\theta}) = \langle 0 | \, U^{\dagger}(\boldsymbol{\theta}) H U(\boldsymbol{\theta}) \, | 0 \rangle$$

Unitary Product State Ansatz

As suggested by Peruzzo *et al* [18], the UCC formulation of a wavefunction can be efficiently implemented on a quantum device in terms of quantum gates. We refer to the UCC ansatz $|\psi_{\mathrm{UCC}}(\boldsymbol{\theta})\rangle$ as the application of the unitary excitation operator $U_{\mathrm{UCC}}(\boldsymbol{\theta})$ to some reference state, usually a single-Slater determinant Hartree-Fock state $|\psi_0\rangle$ obtained via the self-consistent field method.

$$|\psi_{\mathrm{UCC}}(\boldsymbol{\theta})\rangle = U_{\mathrm{UCC}}(\boldsymbol{\theta}) |\psi_0\rangle = e^{\hat{T}(\boldsymbol{\theta}) - \hat{T}^\dagger(\boldsymbol{\theta})} |\psi_0\rangle$$

The operator $\hat{T}(\boldsymbol{\theta})$ is a linear combination of fermionic second-quantised 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 formulation of the UCC ansatz that includes only single and double excitations has been shown by Evangelista *et al* to exactly parametrise any state [19]. However, since the terms of the unitary operator $U_{\mathrm{UCC}}(\boldsymbol{\theta})$ do not commute, it cannot be directly implemented on a quantum computer. Instead, by invoking the Trotter formula to approximate the unitary with a single Trotter step, we define the *Unitary Product State (UPS)* ansatz $|\psi(\boldsymbol{\theta})\rangle$ as the following product of k parametrised unitary excitation operators [5].

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

Figure 1.12: Parametrised k-UPS 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 refer to fermionic excitation operators as the the parametrised exponentials of these anti-Hermitian operators, $U_m(\theta_m)$. Since these fermionic excitation

operators are derived from second-quantised operators, they preserve the fermionic anti-symmetry and particle number of the reference state [5]. The operators are then mapped to quantum gates using the Jordan-Wigner transformation, introduced in Section 5.1, before being implemented on a quantum computer.

In one sense, by invoking the Trotter formula to approximate $U_{\text{UCC}}(\boldsymbol{\theta})$, the UPS operator abandons its connection to many body theory. Hence instead of representing correlational processes, the UPS operator should be viewed as some unitary rotation that parametrically explores the Hilbert state of possible states. Nevertheless, as shown by Burton *et al* [5], the *Singles–Doubles* formulation of the UPS ansatz, that includes only single and double excitations, can represent any vector in the Hilbert space, provided that we combine enough suitably-ordered fermionic excitation operators. In this way, the UPS ansatz is said to achieve *universality*.

DISCO-VQE

For the purposes of this thesis, we focus on a variant of the VQE algorithm known as the *Discretely and Continuously Optimised Variational Quantum Eigensolver (DISCO-VQE)* developed by Burton *et al* [5]. This algorithm finds approximate solutions for the ground state of a molecular system by both (*discretely*) optimising the sequence of fermionic excitation operators in the ansatz and (*continuously*) optimising their parameters.

The DISCO-VQE algorithm selects excitation operators from a pool of one-body and two-body excitation operators to implement the UPS ansatz discussed in the previous section. The excitation operators in the pool are chosen to preserve the spin quantum numbers $\langle \hat{S} \rangle$ and $\langle \hat{S}^2 \rangle$. Specifically, we consider only one-body and *paired* two-body excitation operators that conserve the spin symmetry of the initial state. Paired two-body excitation operators represent the excitation of a pair of electrons from the same spatial orbital to another spatial orbital and have the advantage of being implemented by a constant number of quantum gates. In contrast, unpaired two-body excitation operators require a number of quantum gates that grows linearly with the number of orbitals [5].

Chapter 2

ZX Calculus

In this chapter, we introduce the ZX calculus, a diagrammatic language for reasoning about quantum processes first introduced by Coecke *et al* [7]. The ZX calculus is described as *universal* in that it can represent any linear map (any complex matrix of dimension $2^n \times 2^m$). Therefore, any equation involving linear maps that is derivable in multilinear algebra can also be derived in the ZX calculus through its rewrite rules [20]. It is also considered *sound*, meaning that the diagrammatic rewrite rules preserve the underlying semantics of the linear map represented by a given ZX diagram [21].

In this thesis, we use the scalar-free ZX calculus, meaning that the rewrite rules presented in this chapter 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 later efficiently compute the scalar factor [21].

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 [21]. 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.
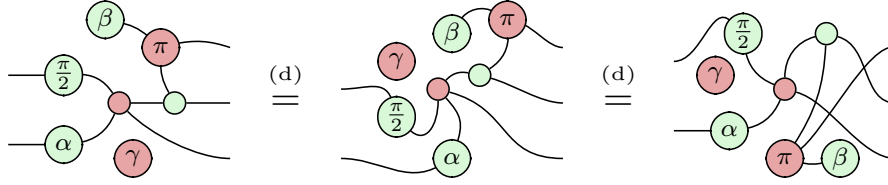
Figure 2.1: Three equivalent ZX diagrams (*only connectivity matters*).

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 (1.5). A Z Spider with n inputs and m outputs represents the following linear map.

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

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

X Spiders (red), are defined with respect to the X eigenbasis (1.6).

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

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

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

$$\bigcirc\!\!- \ = \ |+\rangle + |-\rangle = \sqrt{2}\,|0\rangle \qquad \overset{\pi}{\bigcirc}\!\!- \ = \ |+\rangle - |-\rangle = \sqrt{2}\,|1\rangle$$

Figure 2.4: $|0\rangle$ eigenstate   **Figure 2.5:** $|1\rangle$ eigenstate

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

$$\bigcirc\!\!- = |0\rangle + |1\rangle = \sqrt{2}\,|+\rangle \qquad \overset{\pi}{\bigcirc}\!\!- = |0\rangle - |1\rangle = \sqrt{2}\,|-\rangle$$

Figure 2.6: $|+\rangle$ eigenstate   **Figure 2.7:** $|-\rangle$ eigenstate

Arbitrary single-qubit rotations (1.10) in the Z basis are represented by a Z Spider with a single input and a single output. Similarly, we have that the corresponding rotation in the X basis is represented by an X spider. We can view these as rotations of the Bloch sphere.

$$-\!\overset{\alpha}{\bigcirc}\!\!- \quad |0\rangle\langle 0| + e^{i\alpha}|1\rangle\langle 1| = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\alpha} \end{pmatrix} \rightarrow$$

$$-\!\overset{\alpha}{\bigcirc}\!\!- \quad |+\rangle\langle +| + e^{i\alpha}|-\rangle\langle -| = \frac{1}{2}\begin{pmatrix} 1 + e^{i\alpha} & 1 - e^{i\alpha} \\ 1 - e^{i\alpha} & 1 + e^{i\alpha} \end{pmatrix} \rightarrow$$

We can recover the Pauli Z and Pauli X matrices (1.8) by setting $\alpha = \pi$.

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

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

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

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.

$$-\!\overset{\alpha}{\bigcirc}\!\!-\!\overset{\beta}{\bigcirc}\!\!-\!\overset{\gamma}{\bigcirc}\!\!- = \frac{1}{2}\begin{pmatrix} 1 & 0 \\ 0 & e^{i\gamma} \end{pmatrix}\begin{pmatrix} 1 + e^{i\beta} & 1 - e^{i\beta} \\ 1 - e^{i\beta} & 1 + e^{i\beta} \end{pmatrix}\begin{pmatrix} 1 & 0 \\ 0 & e^{i\alpha} \end{pmatrix}$$

Figure 2.9: Sequential composition corresponding to the matrix product.

In contrast, by composing spiders in parallel, we obtain the tensor product. Note that the order of operation should be interpreted from top to bottom.

$$
\begin{array}{c} \alpha \\ \beta \end{array} = \frac{1}{2} \begin{pmatrix} 1 & 0 \\ 0 & e^{i\alpha} \end{pmatrix} \otimes \begin{pmatrix} 1 + e^{i\beta} & 1 - e^{i\beta} \\ 1 - e^{i\beta} & 1 + e^{i\beta} \end{pmatrix}
$$

Figure 2.10: Parallel composition corresponding to the tensor product.

Hadamard Generator

Up to a global phase, an arbitrary single qubit gate U can be viewed as a rotation of the Bloch sphere about some axis. Therefore, we can decompose U using Euler angles to represent it as three successive rotations [21].

$$
\boxed{U} \quad = \quad \alpha \text{---} \beta \text{---} \gamma
$$

The Hadamard gate (1.9) 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 into three successive rotations by choosing $\alpha = \beta = \gamma = \frac{\pi}{2}$.

$$
\blacksquare \quad = \quad \frac{\pi}{2} \text{---} \frac{\pi}{2} \text{---} \frac{\pi}{2} \quad = \quad \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}
$$

Recall that the Hadamard gate is the member of the Clifford group that 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.

$$
\cdots \, \alpha \, \cdots \quad = \quad \cdots \, \alpha \, \cdots
$$

CNOT Gate

The CNOT gate (1.11) is represented in the ZX calculus 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.
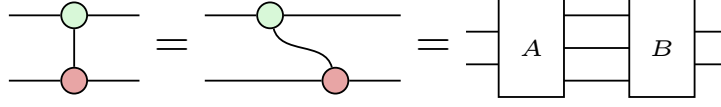
We can calculate matrix A, consisting of a single-input and two-output Z Spider ($4{\times}2$ matrix) and an empty wire (identity matrix), by taking the tensor product.

$$\text{[diagram } A\text{]} = \text{[diagram]} = \begin{pmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 1 \end{pmatrix} \otimes \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

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

$$\text{[diagram } B\text{]} = \text{[diagram]} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \otimes \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix}$$

We can then calculate the CNOT matrix by taking the matrix product of matrix A and matrix B as follows (up to a global scalar factor of $1/\sqrt{2}$).

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

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

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

CZ Gate

The CZ gate applies the Pauli Z gate to the target qubit *iff* the control qubit is in the $|1\rangle$ state. It is derived by conjugating the CNOT target with Hadamards.
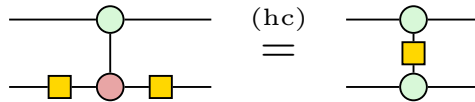
Figure 2.11: The CZ gate in the ZX calculus.

2.2   Rewrite Rules

Notation – *We 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 [21].
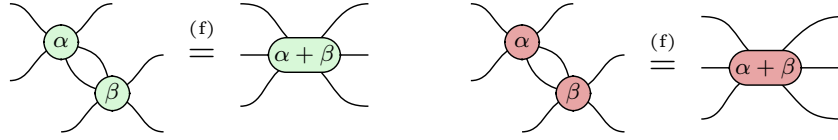
Figure 2.12: Spider fusion rule for Z spiders (left) and X spiders (right).

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.
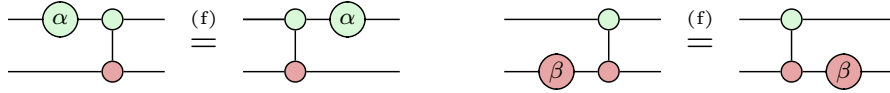
Identity Rule

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.
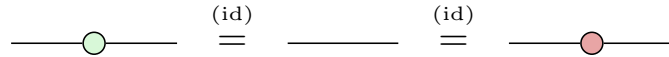
Figure 2.13: Identity rule.

Combining this with the spider fusion rule (2.12), 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 (2.5, 2.7) by assigning a phase of π multiplied by a Boolean variable a (0 or 1) to an X or Z spider, respectively [21].

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

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

The π *copy rule* (c) states that when a Pauli Z or Pauli X gate is pushed through a spider of the opposite colour, it is copied on all other legs and negates the spider's phase. A similar *state copy rule* (c) applies to the Z and X eigenstates.
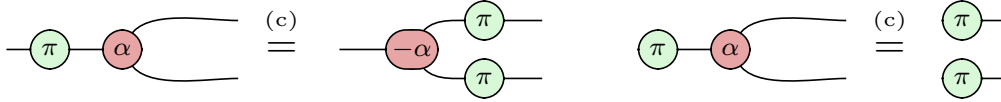
Figure 2.14: π copy (left) and state copy (right) rules for Pauli Z gate and Z eigenstates.

Hadamard Rules

Using that the Hadamard gate is both unitary and Hermitian, we define the *Hadamard self-inverse rule* (hi) as follows.
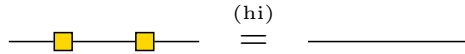
Figure 2.15: Hadamard self-inverse rule.

Recalling that the Hadamard generator changes the colour of a spider and is self-inverse, we define the *Hadamard commutation rule* (hc).
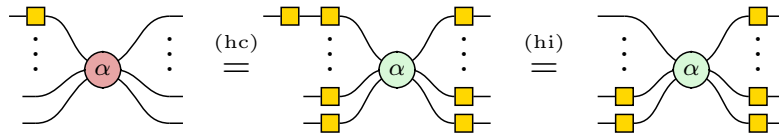
Figure 2.16: Hadamard commutation rule.

Bialgebra Rule

Using the spider fusion rule (2.12), 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 (2.14), 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 2.17: XOR gate (left) and COPY gate (right) with respect to the Z eigenstates.

The natural commutation relation of the classical XOR and COPY gates, implies the *bialgebra rule* (*ba*). By successively applying the two-input/two-output case, we can generalise to any number of inputs and outputs.

Figure 2.18: The bialgebra rule.

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 define the *Hopf rule* (*hpf*).

Figure 2.19: The Hopf rule.

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 $\pi \to -\pi = \pi$).

We now introduce 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* [22], in the ZX calculus.

Figure 2.20: Representation of single-qubit Clifford gates in the ZX calculus.

Since the Y basis, unlike the Z and X bases, is *not* Hermitian [3], swapping the inputs and outputs of a Y *Spider* does *not* yield its transpose. Instead, we define rotations in the Y basis by conjugating with the Clifford gates described above.

Figure 2.21: Conjugation of generators in the Z and X bases into the Y basis.

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 2.22: Single-qubit Clifford commutation relations.

Chapter 3

Pauli Gadgets

Pauli gadgets are a class of unitary transformations that can be used to naturally represent the excitation operators used in UPS ansätze (Section 1.4). In this chapter, we introduce Pauli gadgets and their representation in the ZX calculus. We then develop a method for deriving a set of commutation relations that describes the interaction of Pauli gadgets with the Pauli and the Clifford gates. In Chapter 5, we use Pauli gadgets to construct one-body and paired two-body excitation operators before using the commutation relations derived here in a number of derivations.
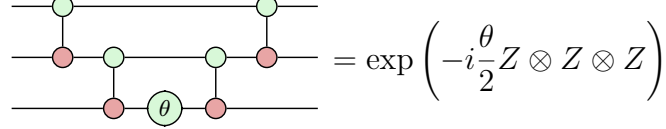
A Pauli string P is defined as a tensor product of Pauli matrices $\{I, X, Y, Z\}^{\otimes n}$, where n is the number of qubits in the system, and each Pauli gate acts on a distinct qubit. Thus $Z \otimes X$ represents the Pauli Z and X gates acting on the first and second qubits respectively. *Stone's Theorem* [23] states that a strongly-continuous one parameter unitary group $U(\theta) = \exp\left(-i\frac{\theta}{2}H\right)$ is generated by the Hermitian operator H. Since Pauli strings are Hermitian, we can define Pauli gadgets as the one parameter unitary groups generated by the Pauli strings.

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

Figure 3.1: Two examples of Pauli gadgets.
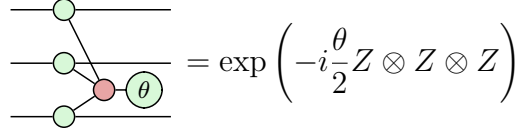
3.1   Phase Gadgets

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 nested between two ladders of CNOT gates.

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

Phase gadgets correspond to unitary maps which are diagonal in the Z computational basis [10]. 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.

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

Using the identity (2.13), spider fusion (2.12) and bialgebra (2.18) rules, we are able to derive its representation in the ZX calculus.
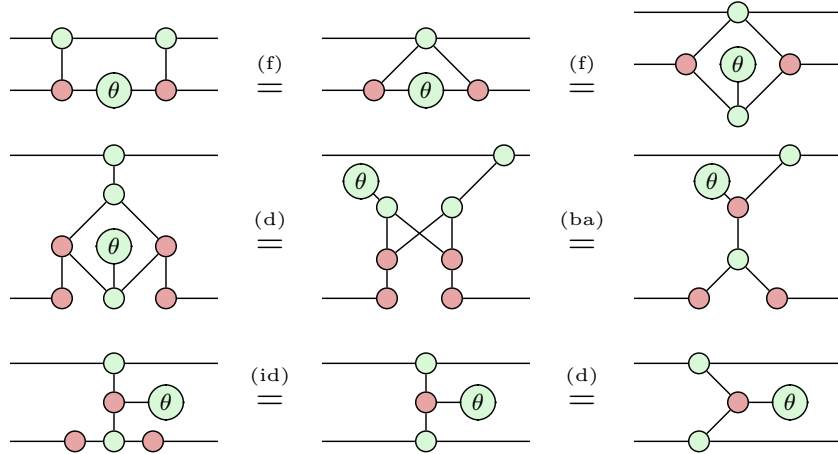
Figure 3.2: Phase gadget result.

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

3. Pauli Gadgets
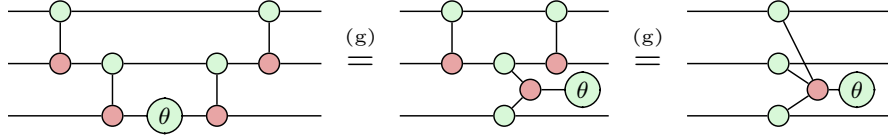
Phase gadgets can be interpreted as first copying each input in the Z basis (2.17), computing the parity of the state by taking the XOR (2.17), then multiplying the state by $\exp\left(-i\frac{\theta}{2}\right)$ or $\exp\left(i\frac{\theta}{2}\right)$ depending on its parity [3].

$$
\bigoplus_i q_i = \quad \mathrm{diag}\left\{\begin{pmatrix} 1 \\ e^{i\theta} \\ e^{i\theta} \\ 1 \\ e^{i\theta} \\ 1 \\ 1 \\ e^{i\theta} \end{pmatrix} \cdot e^{-i\frac{\theta}{2}}\right\}
$$

Phase Gadget Identity

We can show that phase gadgets with an angle $\theta = 0$ are equivalent to identity using the state copy (2.14), spider fusion (2.12) and identity (2.13) rules.
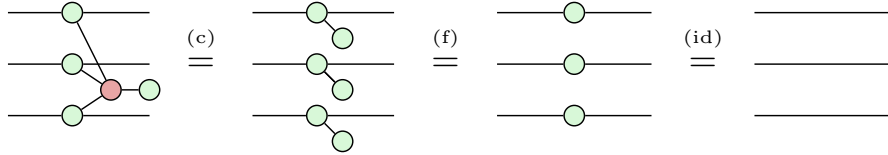
Phase Gadget Fusion

We can show that phase gadgets with the same distribution of legs fuse together and their phases add using the spider fusion (2.12) and bialgebra (2.18) rules.

Phase Gadget Commutation

Phase gadgets can be shown to commute using the spider fusion rule (2.12).
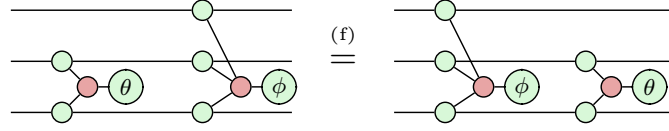

Phase Gadget Decomposition

Using the phase gadget result described above (3.2), we can show that a two-legged phase gadget can be decomposed in the following two ways.
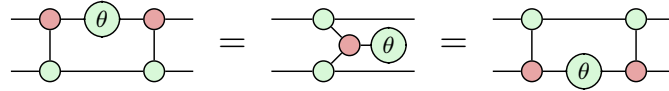
**Figure 3.3:** Phase gadget decomposition result.

Balanced Tree Representation

By recursively applying this decomposition, we can demonstrate that a phase gadget in the balanced tree representation can achieve a circuit depth of $2\log_2(n)+1$, rather than $2n-1$ as in the ladder representation, where n is the number of qubits [24].
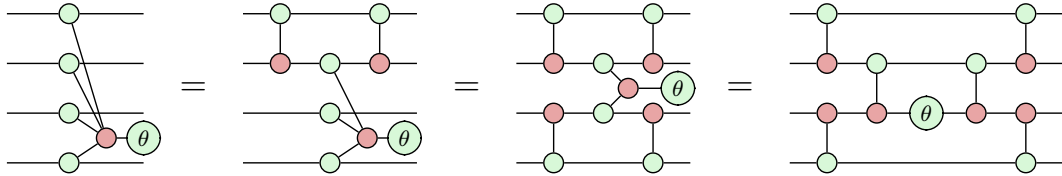
Figure 3.4: Phase gadget in the balanced tree representation.

Single-Legged Phase Gadgets

Single-legged phase gadgets simply correspond to Z rotations. Therefore, phase gadgets can be thought of as the many-qubit generalisation of Z rotations.
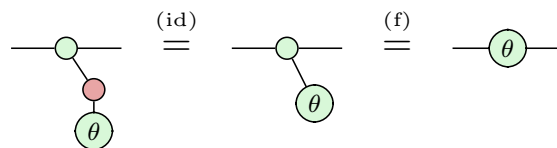
Figure 3.5: Single-legged phase gadget as a Z rotation.

3.2   Pauli Gadgets

Pauli gadgets are defined as the one parameter unitary groups of Pauli strings in the set $\{I, X, Y, Z\}^{\otimes n}$. Recognising that Pauli gadgets are simply phase gadgets associated with a change of basis, we can construct Pauli gadgets by conjugating the legs of a phase gadget with the appropriate Clifford gates, as described in Section 2.3. Whilst phase gadgets alone cannot alter the distribution of the observed state, Pauli gadgets can [3].

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

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 3.6: 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 3.7: 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 refer to a set of Pauli gadgets as diagonal when it consists only of phase gadgets [10]. Such sets of phase gadgets are known as *phase polynomials* and are themselves diagonal in the computational Z basis [24]. Phase polynomial synthesis refers to finding a quantum circuit that optimally implements some phase polynomial, for which there are several well-known algorithms [25], [26], [27].
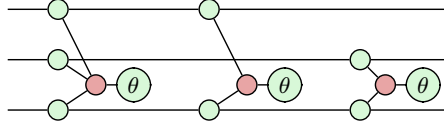
Figure 3.8: A phase polynomial corresponds to a circuit of phase gadgets.

As in Cowtan *et al* [10], 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 rewrite the circuit as some phase polynomial that can later be synthesised in some optimal way.
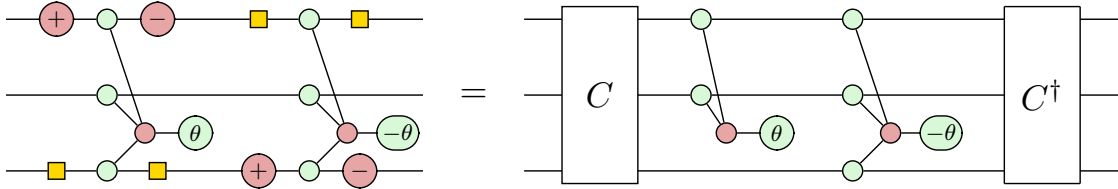

**Figure 3.9:** Diagonalisation of a pair of commuting Pauli gadgets by C.

As we will see in Chapter 5, the excitation operators used in VQE algorithms consist of commuting sets of Pauli gadgets. Therefore, the quantum circuits implementing such excitation operators can be diagonalised with some Clifford subcircuit C. As stated in Cowtan *et al* [10], 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.
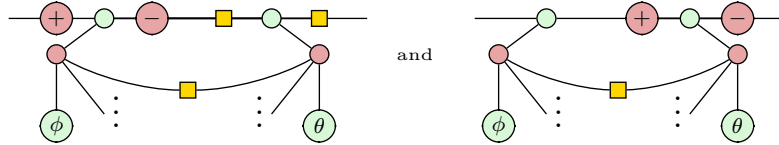
3.4 Commutation Relations

A pair of Pauli gadgets commute when their Hamiltonians commute. 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.
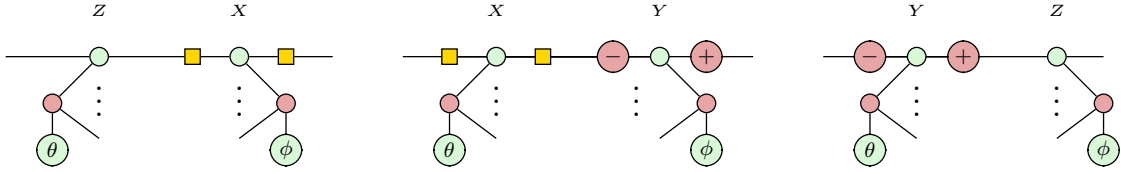

Starting with the mismatching Z/X pair and using the bialgebra (2.18), Hadamard commutation (2.16) and spider fusion (2.12) rules, we can show that commuting the gadgets' legs introduces a Hadamard between the bodies of the gadgets [3].
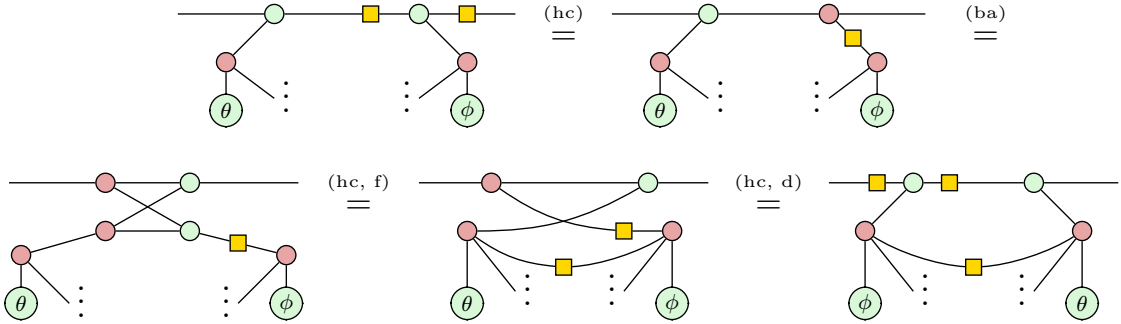
Similarly, for the X/Y and Y/Z mismatching pairs, we have the following.

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 (2.19) to remove the wires between them [3].
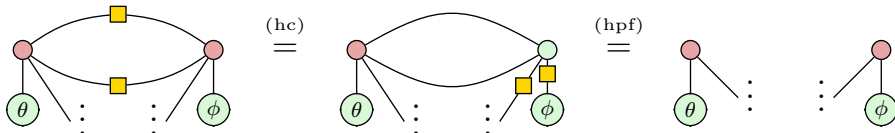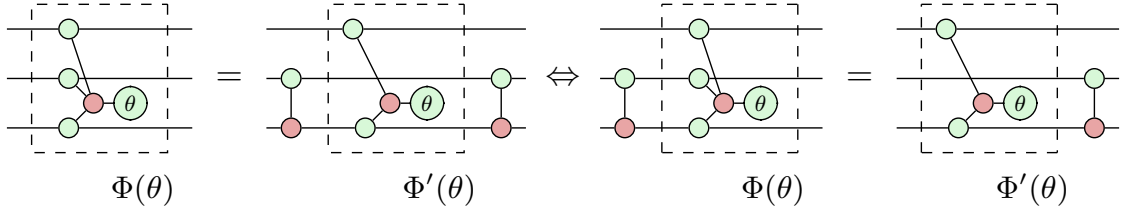
Clifford Commutation Relations

We 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 $C = \text{CNOT}_{1,2}$, we can interpret the conjugation of a *phase gadget* diagrammatically (diagram on left) as the phase gadget decomposition result (3.3). 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 (1.11) and CZ (2.11) 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 $\text{CNOT}_{0,1}$ gate commutes through the $\exp\left[-i\frac{\theta}{2}\left(Z \otimes Z\right)\right]$ gadget to give the $\exp\left[-i\frac{\theta}{2}\left(I \otimes Z\right)\right]$ gadget. We first push the bottom Pauli Z gate through the
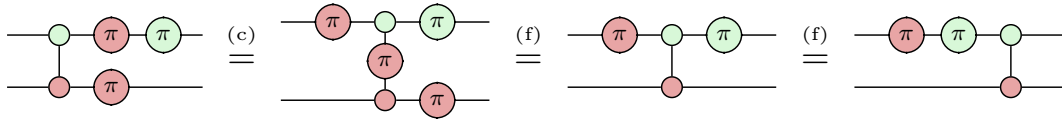
3. Pauli Gadgets

CNOT target using the π copy rule (2.14), then, we push the top Pauli Z gate through the CNOT control using the spider fusion rule (2.12), cancelling one of the copied Pauli Z gates in the process.
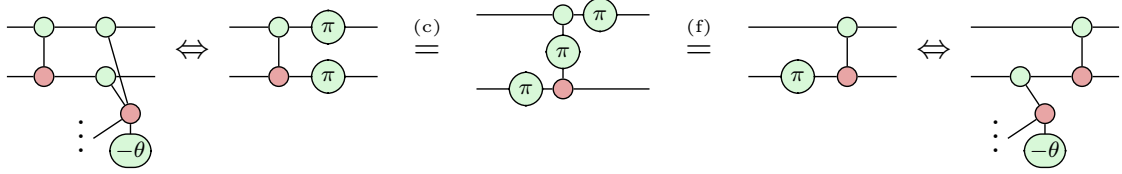

The Pauli Y gate is obtained by conjugating the Pauli Z gate with Clifford gates (2.21). Using the π copy (2.14) and spider fusion (2.12) rules, we can show that $Y = XZ$ up to a global phase of i and $Y = ZX$ up to a global phase of $-i$.
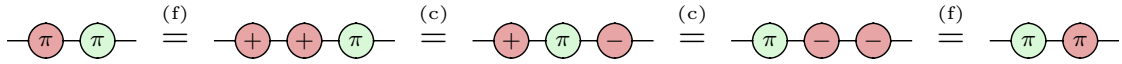
In the example below, one of the definitions of the Pauli Y gate above to derive how the CNOT gate interacts with the $\exp\left[-i\frac{\theta}{2}(Y \otimes X)\right]$ Pauli gadget by identifying how it gate interacts with the $Y \otimes X$ Pauli string.


Similarly, we can identify how single-qubit gates interact with Pauli gadgets by identifying how they interact with the Pauli gates. In the example below, we use the fact that the above definitions of the Pauli Y gate differ by a factor of -1 to show that commuting a Hadamard through a Y leg flips the gadget's phase.
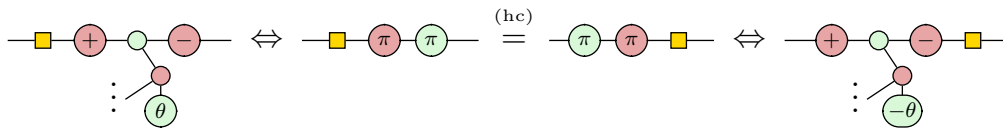

Using the rules described in this section, we were able to identify how Pauli gadgets interact with the Clifford and Pauli gates, in agreement with [3] and [24]. We have summarised these commutation relations below. In Chapter 5, we demonstrate how these commutation relations can be used to prove several important results.
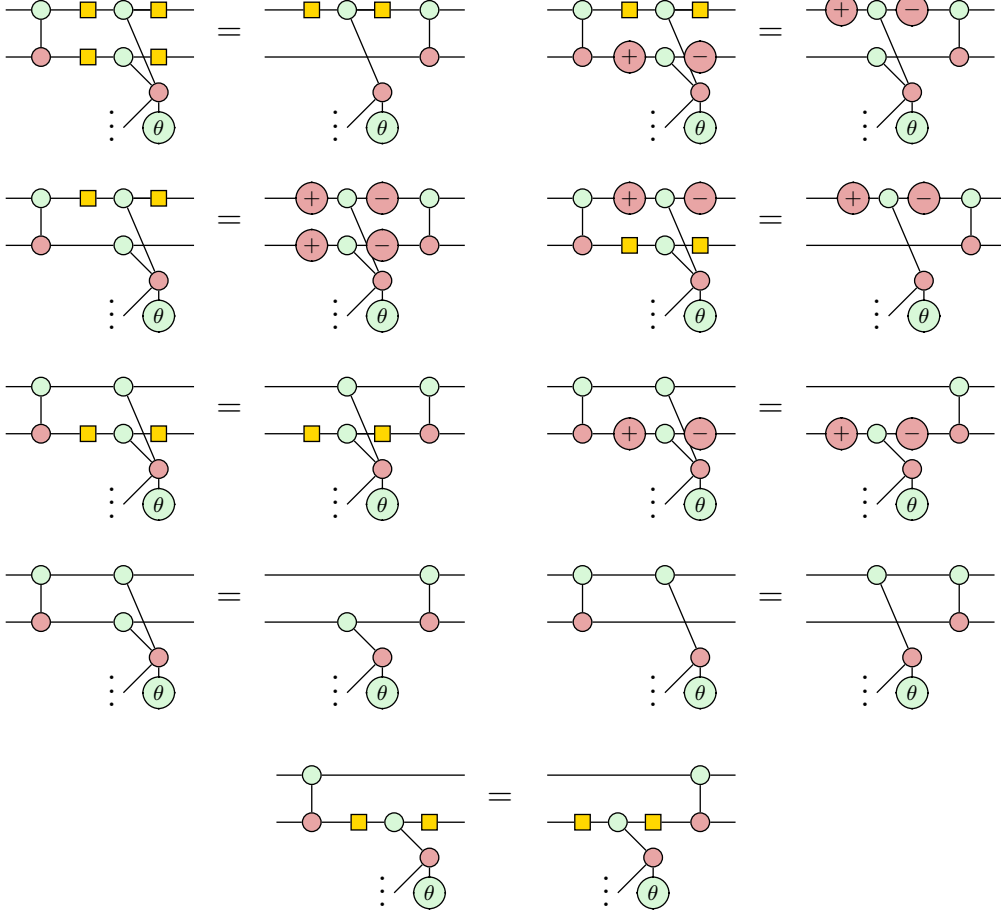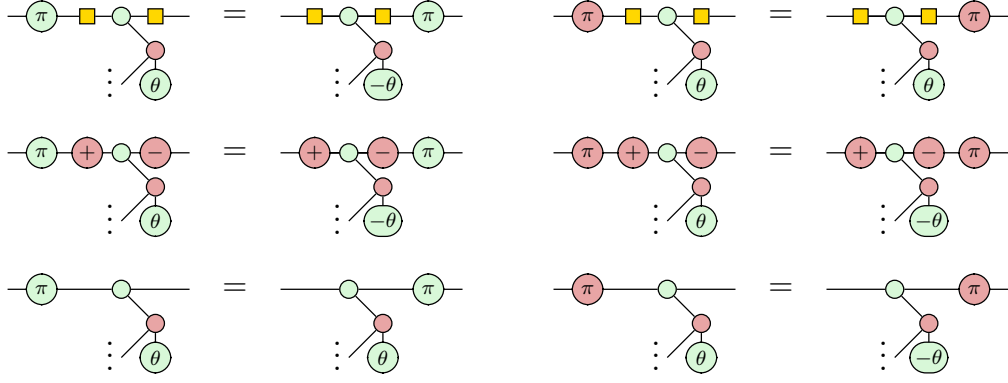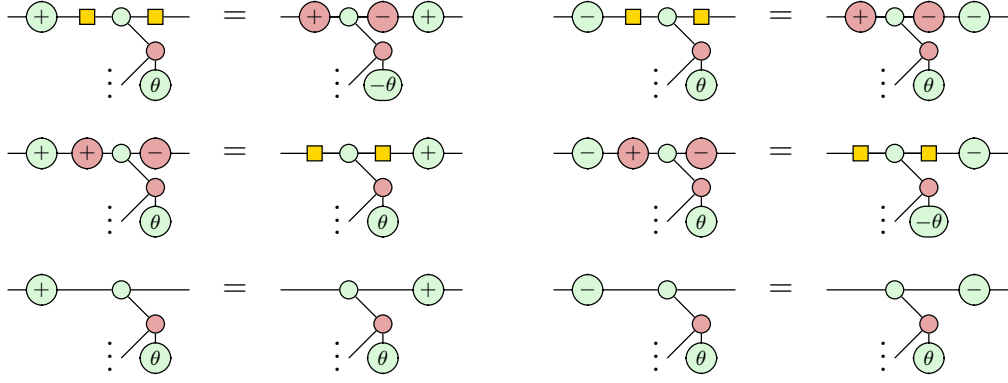
Figure 3.10: CNOT commutation relations excluding repetitions.
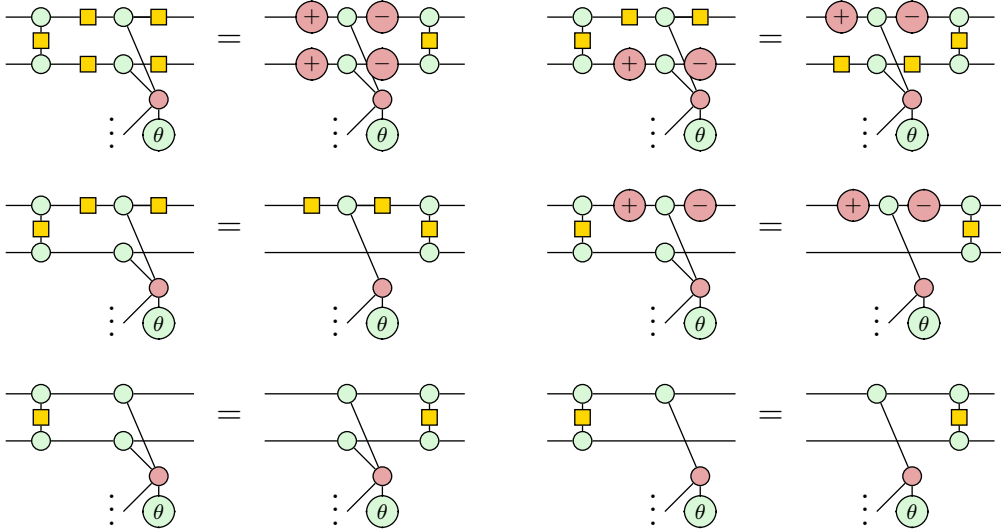
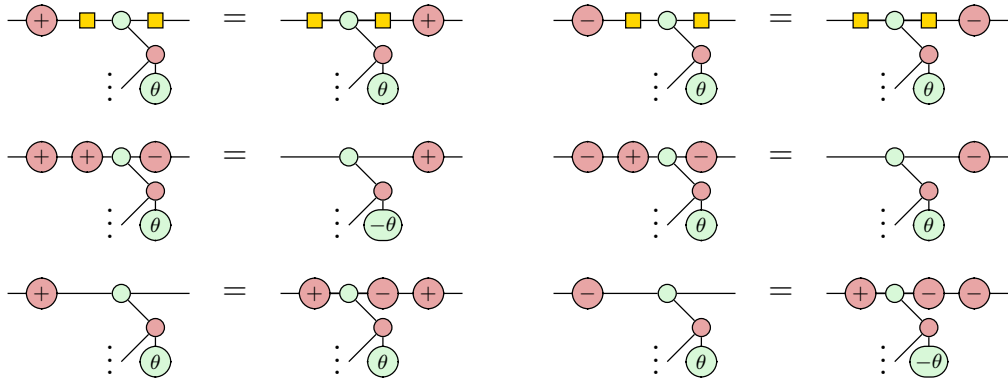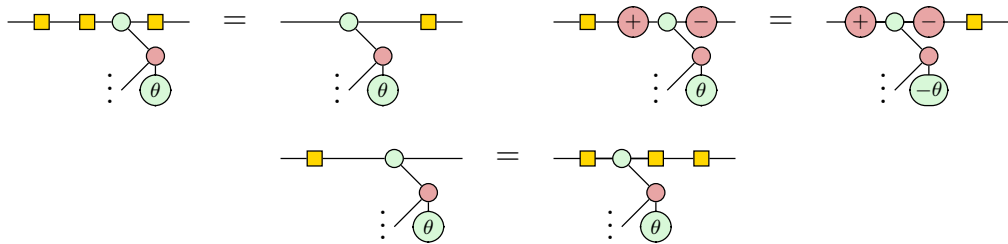**Figure 3.11:** CZ commutation relations excluding repetitions.

Figure 3.12: Pauli commutation relations.

Figure 3.13: Clifford Z commutation relations.

Figure 3.14: Clifford X commutation relations.

Figure 3.15: Hadamard commutation relations.

Chapter 4

Controlled Rotations

In our attempt to identify a generalised representation for excitation operators using the ZX calculus, we came across the work done by Yordanov *et al*, which shows that excitation operators can be rewritten in terms of controlled rotations by through conjugation by some subcircuit. Our interpretation of this result is that controlled rotations can be used to account for the Pauli antisymmetry of fermionic systems by conditionally 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 begin by discussing the well-established representation of singly-controlled rotations the ZX calculus. We then demonstrate how these singly-controlled rotations can be used to construct higher-order controlled rotations, developing a representation for them 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.
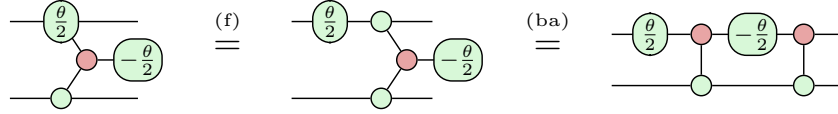
4.1   Singly-Controlled-Rotations

We can express a singly-controlled Z rotation gate $\mathrm{CR}_Z(\theta)$, controlled by the second qubit, as two phase gadgets. This aligns with our expectations since the $\mathrm{CR}_Z(\theta)$ gate is diagonal in the computational Z basis [3].
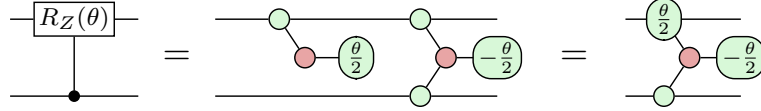

Figure 4.1: Representation of the $\mathrm{CR}_Z(\theta)$ gate in the ZX calculus.

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 (2.12) and bialgebra (2.18) 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 (2.12), state copy (2.14) and identity (2.13) rules,
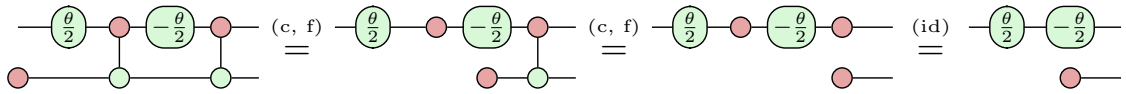
As expected, the resulting diagram fuses to give identity. Conversely, when the control qubit is in the $|1\rangle$ state, we obtain a Z rotation by θ.
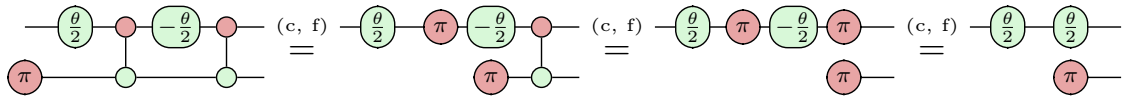
Using the the phase gadget fusion rule (3.1) and the phase gadget decomposition result (3.3), we can decompose the $\mathrm{CR}_Z(\theta)$ gate into four equivalent circuits.
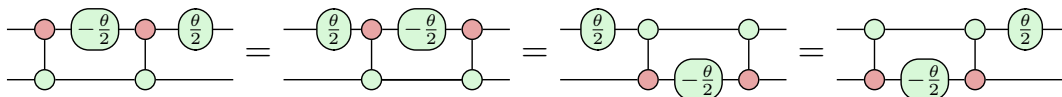
4.2   Higher-Order Controlled Rotations
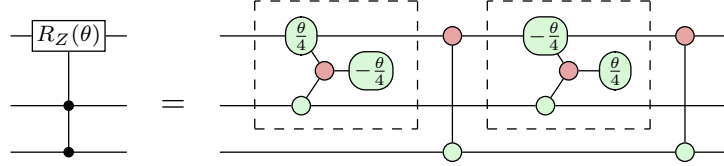
We can implement higher order controlled rotations by nesting singly-controlled rotations in CNOT gates [11]. For instance, we implement a doubly-controlled Z rotation using two singly-controlled Z rotations with opposite phases as follows.


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 CNOT gates are adjacent and cancel each another (right diagram).
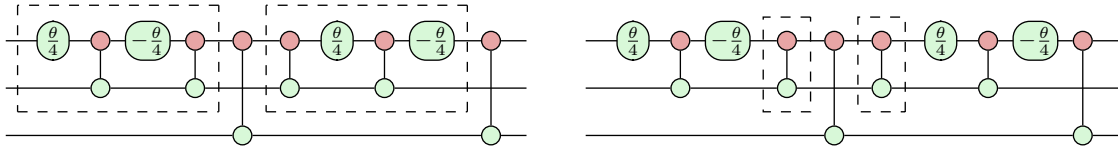
Figure 4.2: $\mathrm{CCR}_Z(\theta)$ circuit decomposition (left). Cancelling CNOT gate pair (right).

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 (3.5) and commuting the CNOT gates through them using the commutation rules in Figure 3.10, we can cancel the CNOT gates and obtain the following phase polynomial.
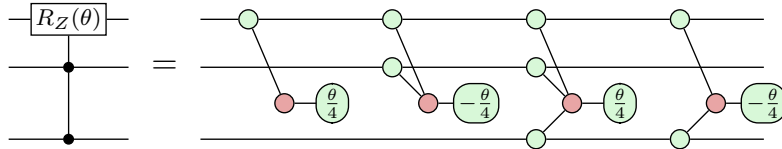
Figure 4.3: Representation of the $\mathrm{CCR}_Z(\theta)$ gate in the ZX calculus.

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.
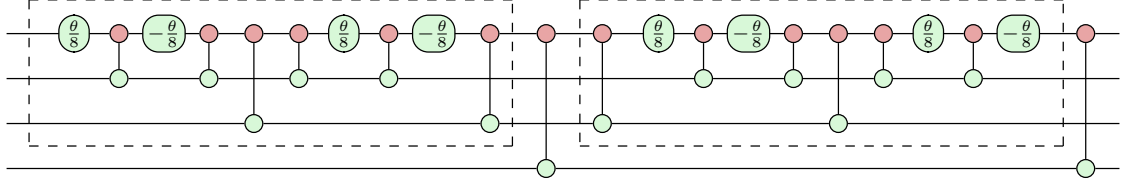
Figure 4.4: $\mathrm{CCCR}_Z(\theta)$ gate.

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

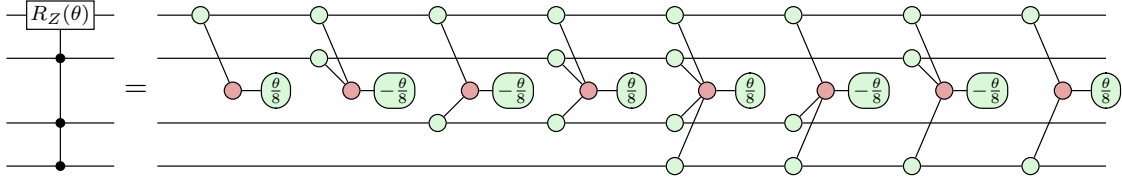**Figure 4.5:** Representation of the $\mathrm{CCCR}_Z(\theta)$ gate in the ZX calculus.

Controlled Rotations in Different Bases

Starting with a controlled Z rotation and surrounding the target qubit with Hadamard gates (1.9), we obtain the corresponding controlled X rotation.
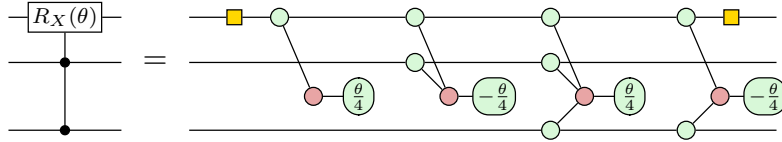
Figure 4.6: Representation of the $\mathrm{CCR}_X(\theta)$ gate in the ZX calculus.

Similarly, by surrounding the target qubit of a controlled Z rotation with the Clifford gates described in Section 2.3, we obtain a controlled Y rotation.
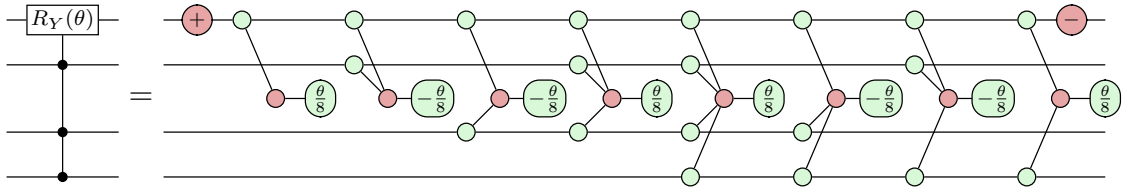
Figure 4.7: Representation of the $\mathrm{CCCR}_Y(\theta)$ gate in the ZX calculus.

Chapter 5

Excitation Operators

In this chapter, we use the concepts discussed in the previous chapters to study one-body and paired two-body excitation operators using the ZX calculus. In Section 5.1, we discuss the represention of the fermionic excitation operators in terms of Pauli gadgets as well as their specific circuit implementations. Then, in Section 5.2, we discuss the several UPS ansätze generated by the DISCO-VQE algorithm, using the ZX calculus to attempt to rationalise their form. Finally, in Section 5.3, we draw on the work done by Yordanov *et al* [11] and Kornell *et al* [12], to demonstrate how fermionic excitation operators can be expressed in terms of the controlled rotations discussed in Chapter 4.
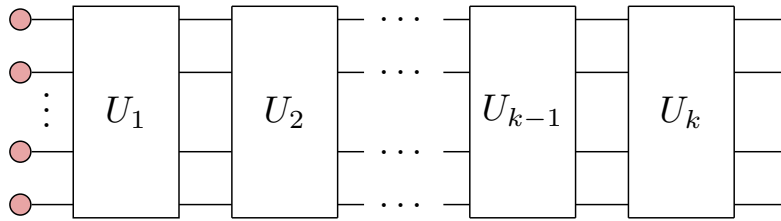
Figure 5.1: Example circuit implementation of the UPS ansatz.

5.1 Implementing Excitation Operators

In Section 1.4, we introduced the UPS ansatz, which can represent any state using a sufficiently large sequence of k parametrised one-body and paired two-body excitation operators.

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

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-qubit system, we do this by taking the following linear combinations of the Pauli gates.

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

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-qubit 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 5.2: The Jordan-Wigner transformation.

This mapping is known as the Jordan-Wigner transformation [28] 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 directly encodes the occupation number of each spin orbital in the fermionic state: $|f_{n-1}, f_{n-2}, \ldots f_1, f_0\rangle \rightarrow |q_{n-1}, q_{n-2}, \ldots q_1, q_0\rangle$.

After applying the Jordan-Wigner transformation to the anti-Hermitian seecond-quantised operator $a_q^{\dagger}a_p - a_p^{\dagger}a_q$ and finding its corresponding unitary operator $U_p^q(\theta)$, we obtain the following expression 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 5.3: One-body excitaiton operator expressed in terms of the Pauli matrices.

Similarly, by applying the Jordan-Wigner transformation to the two-body excitation operator derived from $a_r^\dagger a_s^\dagger a_q a_p - a_p^\dagger a_q^\dagger a_s a_r$, we obtain the following.

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

Figure 5.4: Two-body excitation operator expressed in terms of the Pauli matrices.

The expression for the one-body excitation operator in Figure 5.3 can then be rewritten as the following product of commuting exponential terms.

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

Recognising these exponential terms as the one parameter unitary groups of Pauli strings, we see that they each correspond to a Pauli gadget, as we discussed in Section 3.2. Then, by sequentially composing these Pauli gadgets (taking their matrix product) we obtain the following diagram in the ZX calculus.
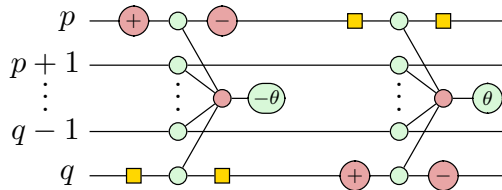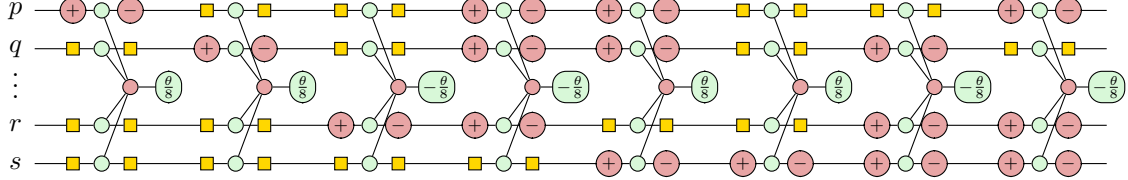

Figure 5.5: Representation of a one-body excitation operator in the ZX calculus.

Recall that, in the context of the DISCO-VQE algorithm, introduced in Section 1.4, we are only interested in paired two-body excitation operators. In contrast to their

5. Excitation Operators

general counterparts, the CNOT overhead associated with the parity computation is eliminated in paired two-body operators [5], such that they can be expressed as the following *four-legged* Pauli gadgets, each having a constant CNOT depth.

Figure 5.6: Representation of a paired two-body excitation operator in the ZX calculus.

At this point, it is worth acknowledging the expressive power of the ZX calculus. Had we been using traditional circuit notation to represent the one-body excitation operator, we would have needed to choose a specific circuit implementation. Quantum computing literature usually discusses this operator in the ladder representation, which gives no indication of the many possible equivalent implementations.
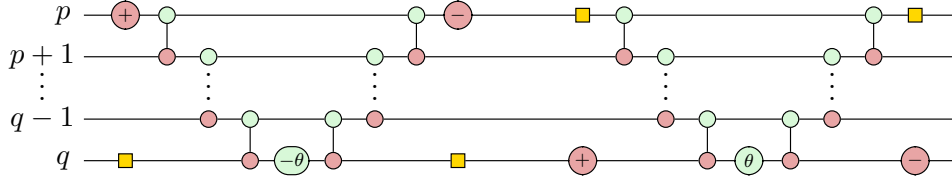
The commonly-used ladder representation isn't even the most efficient circuit implementation of the Pauli gadgets. Recalling the balanced tree representation (3.4), we can implement operator with a depth of $2\log_2(n)+1$ rather than $2n-1$.
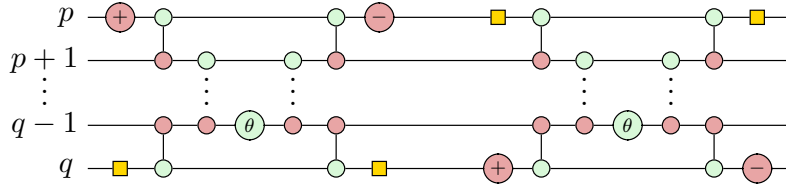
Furthermore, the set of rules that we introduced in Chapter 3 allow us to describe the interactions of Pauli gadgets using the ZX calculus. For instance, we can easily verify that the Pauli gadgets comprising fermionic excitation operators mutually commute with each other by realising that each pair of Pauli gadgets in these operators possesses two mismatching pairs of legs, as discussed in Section 3.4.

5.2 Commuting Excitation Operators

The DISCO-VQE algorithm, introduced in Section 1.4, generates multiple UPS ansätze, each yielding the same energy expectation value but corresponding to a unique sequence of fermionic excitation operators. This indicates that the ansätze equivalently capture the correlation present in the molecular system of interest, implying potential redundancies in their representation. In this section, we discuss how to address this problem, aiming to rationalise the redundancies observed among the energy-equivalent ansätze. To achieve this goal, we use the Pauli gadget commutation rules outlined in Section 3.4 to identify several commuting excitation operators within the operator pool employed by the DISCO-VQE algorithm.

Recall from Section 1.4 that the UPS ansatz allows us to parametrically explore the Hilbert space of possible states in search of a good approximation of the ground state. Provided we combine enough suitably-ordered fermionic excitation operators, the UPS ansatz can represent any vector in Hilbert space. One immediate question that arises from this is: what is the smallest finite sequence of m fermionic excitation operators $\prod_i U_i(\theta_i)$, each parametrised by some angle θ_i, that can equivalently traverse the Hilbert space as some other finite sequence of n fermionic operators $\prod_j U_j(\theta_i)$? By answering this question, we anticipate being able to demonstrate the equivalence between some of the energy-equivalent ansätze by replacing one sequence of excitation operators in the ansatz with another sequence.

We begin by considering the fermionic excitation operators in the operator pool employed by the DISCO-VQE algorithm. We begin by noting that for a given ansatz, the ordering of qubits is defined such that spin-up and spin-down orbitals, which share the same spatial wavefunction, are adjacent [5], and that the fermionic excitation operators in the operator pool are chosen to conserve the spin symmetry of the initial state, as we discuss in Section 1.4. Next, recall that in second quantisation, operators representing observables are constructed from the creation and annihilation operators presented in Section 1.2. For instance, recalling the expression for the second-quantised Hamiltonian in Figure 1.4, we see that the

5. Excitation Operators

Hamiltonians of any two systems with the same number of spin orbitals differ only by the values of their matrix elements, h_{ij} and h_{ijkl}. Similarly, we find that the same fermionic excitation operators can be used to construct ansätze for any molecular system, provided it has the same number of spin orbitals.

Recall that the Pauli gadgets comprising the one-body and paired two-body excitation operators mutually commute with each other by identifying that every pair of gadgets in these operators possesses two mismatching pairs of legs, as discussed in Section 5.1. We then define *trivially commuting* operators as commuting operators, that act on a disjoint set of qubits, and *non-trivially commuting* operators as commuting operators that act on an intersecting set of qubits. Using this approach, we were able to successfully identify six previously-unknown pairs of non-trivially commuting excitation operators for linear H_4. Since the same fermionic excitation operators can be used to construct ansätze for any molecular system with the same number of spin orbitals, it follows that the operator commutation relations that we derived for linear H_4 are also true for square H_4 and trapezium H_4, as is indeed the case. More generally, these commutation relations apply to the fermionic excitation operators describing any fermionic system comprised of eight spin orbitals.

In contrast to traditional VQE algorithms, which optimise ansatz parameters continuously, the DISCO-VQE algorithm also incorporates discrete optimisation techniques. Specifically, it optimises the sequence of excitation operators through cyclic permutations, mutations, and pair swaps [5]. In this context, we propose a modification to the DISCO-VQE algorithm to account for the additional non-trivially commuting excitation operators identified. We anticipate that by doing so, we might minimise redundancies among the expectation energy-equivalent ansätze, thereby decreasing the number of genuinely unique ansätze produced by the algorithm. We expect that this smaller set of ansätze may provide valuable insights into the correlations within the molecular system under study and potentially inform the development of more efficient optimisation algorithms and strategies.

5.3   Excitations as Controlled Rotations

Recall that our objective is to establish a generalised representation for excitation operators using the ZX calculus. To achieve this, we draw upon the work of Yordanov *et al* [11] and Kornell *et al* [12] and replicate their findings within the ZX calculus framework, using the commutation relations derived in Section 3.4 and the representations of higher-order controlled rotations developed in Chapter 4.

Fermionic excitation operators implemented as quantum circuits exhibit significant circuit depths due to the numerous CNOT gates required to compute the parity of the fermionic state. Both Yordanov *et al* and Kornell *et al* propose that by rewriting the excitation operators in terms of controlled rotations, through conjugation with some subcircuit, and subsequently selecting a more efficient implementation for the controlled rotation, the circuit depth can be substantially decreased.

Figure 5.7: Conjugating $U(\theta)$ by subcircuit S to reveal a controlled rotation $\mathrm{CR}(\theta)$.

To illustrate this, we begin by demonstrating the outcome of conjugating a minimal one-body excitation operator with CNOT gates using the ZX calculus.

Figure 5.8: Quantum circuit (left) and in Pauli gadget form (right).

By inserting two adjacent CNOT gates, where $p = $ control and $q = $ target, and using the CNOT commutation relations in Figure 3.10, we reveal two Pauli gadgets.

5. Excitation Operators

Recognising the single-legged Pauli gadget as a Y rotation and fusing it with the two-legged Pauli gadget, yields the $\mathrm{CR}_Y(\theta)$ gate introduced in Section 4.1.
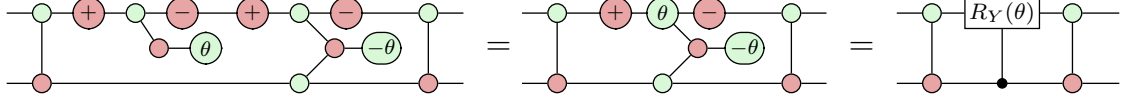

Had we instead chosen $p = $ target and $q = $ control, we would have obtained the following singly-controlled Y rotation, now controlled by the other qubit.
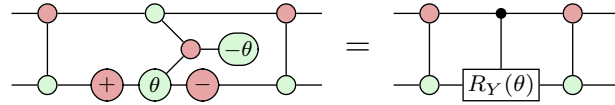

In the general case, we begin by conjugating with CNOT gates and a CZ gate (2.11) using the commutation relations in Figures 3.10 and 3.11.
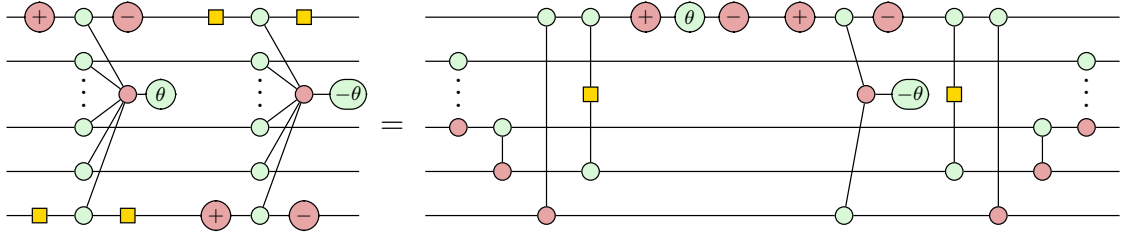
As before, by fusing the resulting single-qubit Y rotation with the adjacent Pauli gadget, we obtain the $\mathrm{CR}_Y(\theta)$ gate. Therefore, we have succesfully replicated the result in Yordanov *et al* using the ZX calculus.
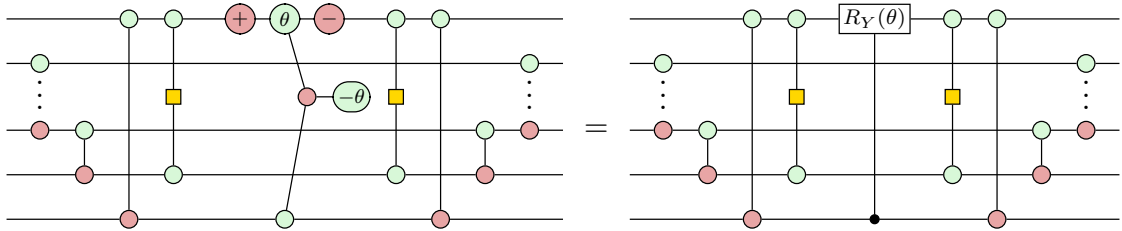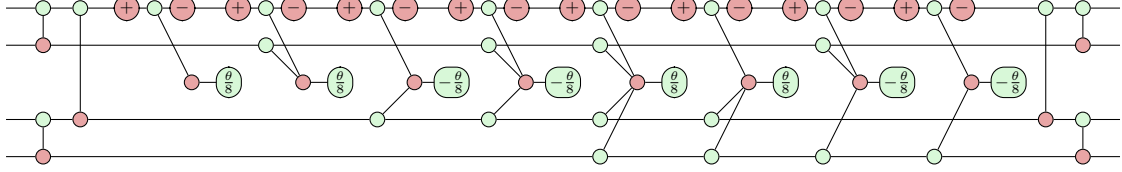

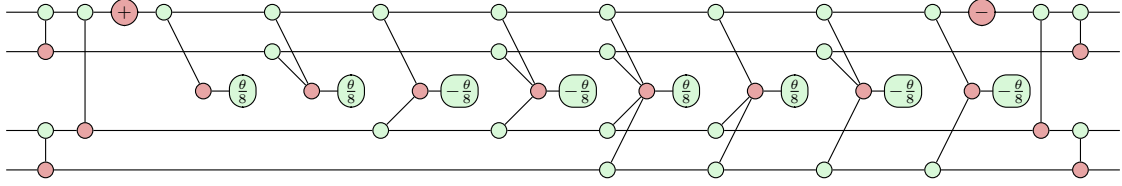Let us now consider the following paired two-body excitation operator.
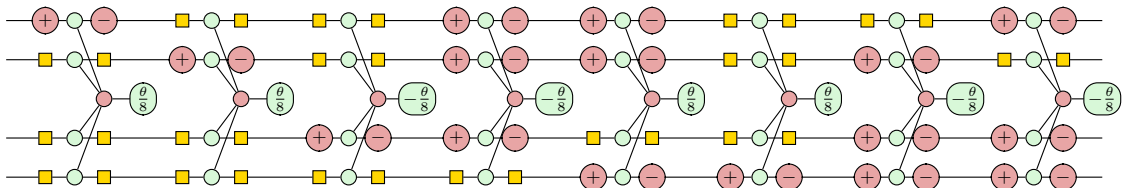
5. Excitation Operators

We begin by conjugating the circuit with three CNOT gates, using Figure 3.10.

Then, by cancelling the adjacent Clifford gates, we reveal a phase polynomial.

Whilst the phase gadgets we obtain have the correct distribution of legs, their phases do not match those of the triply-controlled rotation shown in Figure 4.5. We correct this by conjugating the second and fourth qubits with Pauli X gates, using the commutation relations in Figure 3.12, then fuse them with the CNOT targets. Note that the controls conjugated by Pauli X gates are now controlled by the $|0\rangle$ state, rather than the $|1\rangle$ state, as denoted by the white control nodes.
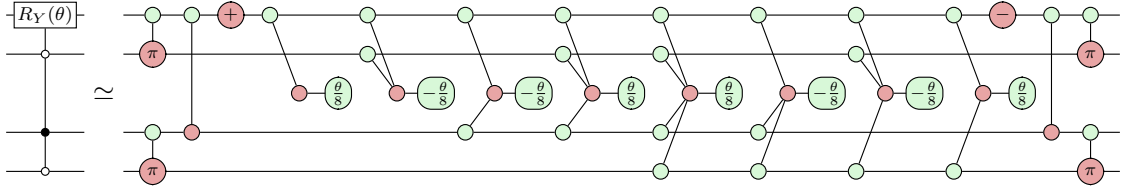
Finally, recognising that the target (top) qubit is in the Y basis, we have revealed the $\mathrm{CCCR}_Y(\theta)$ gate and, therefore, succesfully replicated the result in Yordanov *et al.* Note that above, we use '$\simeq$' since we haven't included the conjugation by the Clifford subcircuit in the left side of the equality. We have, however, implicitly included the Pauli X gates by using white control nodes on the left.

Chapter 6

ZxFermion Software

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

Motivated by the need for an accessible tool to explore research ideas related to circuits of Pauli gadgets, we decided to build the ZxFermion Python package for the visualisation and manipulation of circuits of Pauli gadgets. It is built on top of the PyZX `BaseGraph` API [29] and the Stim `Tableau` class [30].

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 within Jupyter notebook environments, enabling users to generate interactive ZX diagrams directly in the output cell. ZxFermion offers an accessible tool for studying the interactions of Pauli gadgets in the context of VQE.

Using ZxFermion, we can replicate all of the commutation relations described in 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 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 $\exp\left[-i\frac{\pi}{4}\left(Y \otimes Z \otimes X\right)\right]$ gadget as follows.

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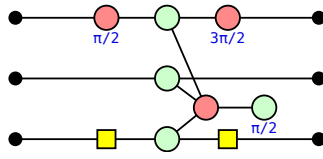

By setting `as_gadget=False`, we expand the gadget in quantum circuit notation.

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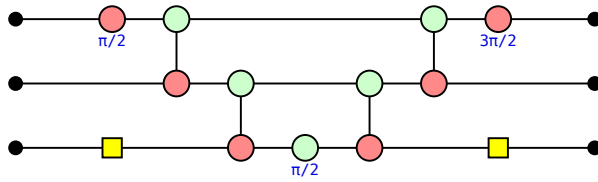

We can construct a circuit of Pauli gadgets using the `GadgetCircuit` class, providing it with an ordered list of `Gadget()` objects. For instance, we implement a one-body excitation operator below, as discussed in Chapter 5.

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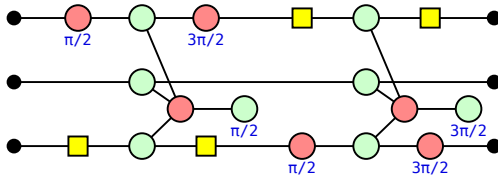

We now introduce the classes implementing the standard quantum gates. As we will see in Section 6.2, ZxFermion encodes the logic describing the interaction of

6. ZxFermion Software

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()
```
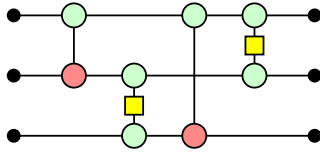

The Pauli Z and Pauli X gates are implemented by the `z` and `x` classes respectively. We specify the target qubit using the `qubit` parameter (defaults to `qubit=0`).

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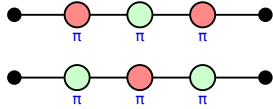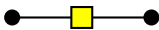

Similarly, the single-qubit Clifford gates are implemented by the `ZPlus`, `ZMinus`, `XPlus` and `XMinus` classes. Below we implement the $Y \otimes Y$ Pauli string (Section 2.3).

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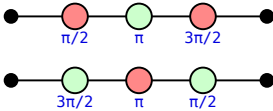

Finally, we have the Hadamard gate, which is implemented by the `H` class.

```
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)
```
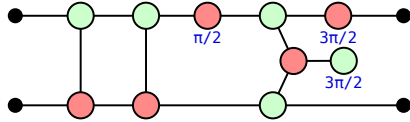

If no insertion positions are specified, the specified quantum gate and its adjoint are inserted at the start and at the end of the circuit respectively. The relevant commutation relations developed in Section 3.4 are then applied as required.

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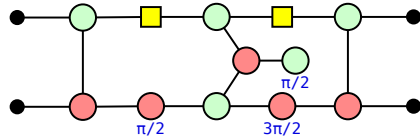

The `apply()` method is not limited to CNOT gates – we could have instead chosen any of the quantum gates mentioned in the previous section. The commutation logic uses Stim's `Tableau` class to construct the required Clifford tableau ensuring that the correct transformation is applied. Below we conjugate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

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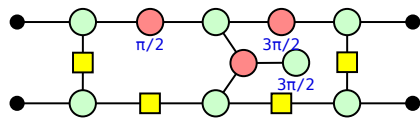
6. ZxFermion Software

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 5.8.

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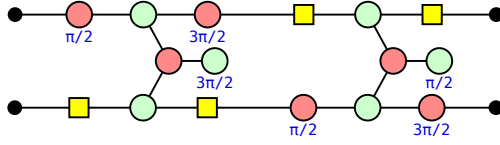

It is then simply a matter of conjugating with CNOT gates to replicate the derivation revealing a singly-controlled Y rotation, as shown in Section 5.3.

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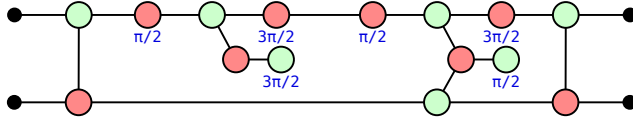

We can replicate the derivation that reveals a singly-controlled Y rotation from a one-body excitation operator shown in Section 5.3. We begin by instantiating the `GadgetCircuit()` object with the following lines of code.

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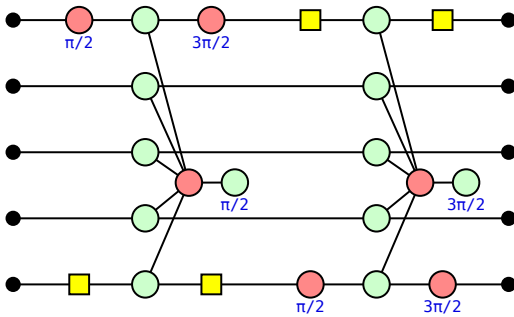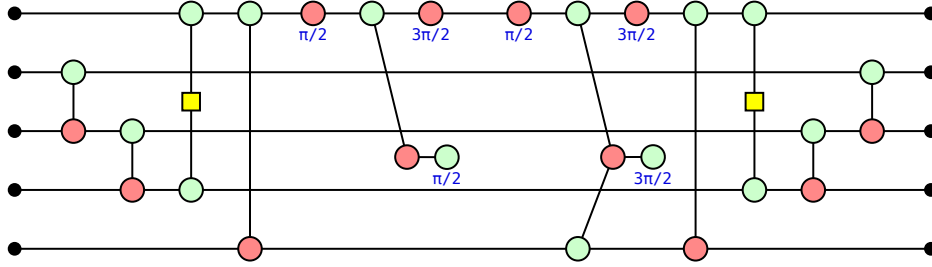

Then, using the following lines of code, we eliminate the Z legs responsible for calculating the parity of the fermionic state, and reveal the two Pauli gadgets, that

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 derivation in Kornell *et al* [12]. 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()
```
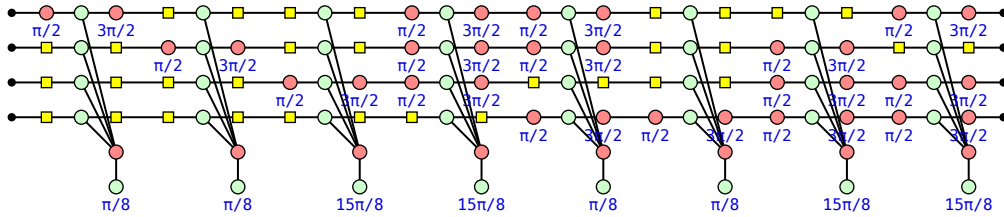

Then, using the following lines of code, we reveal a triply-controlled Y rotation.

```
circuit.apply(CX(0, 3), draw=False)
circuit.apply(CX(0, 2), draw=False)
circuit.apply(CX(0, 1), draw=True)
```
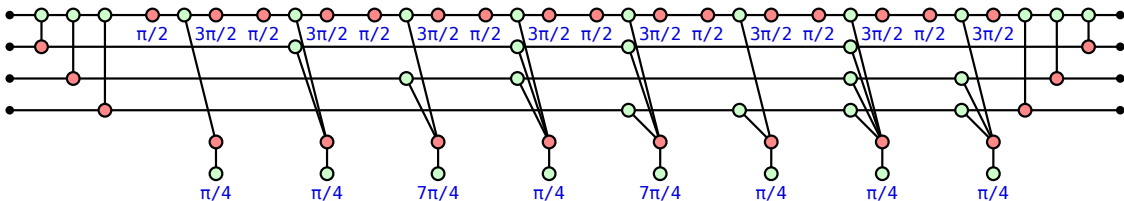
Chapter 7

Conclusion

7.1   Summary

7.2   Future Work

Bibliography

[1] Szalay, P. G., Müller, T., Gidofalvi, G., Lischka, H. & Shepard, R. Multi-configuration self-consistent field and multireference configuration interaction methods and applications. *Chemical Reviews* **112**, 108–181 (2011).

[2] Kassal, I., Whitfield, J. D., Perdomo-Ortiz, A., Yung, M.-H. & Aspuru-Guzik, A. Simulating chemistry using quantum computers. *Annual Review of Physical Chemistry* **62**, 185–207 (2011).

[3] Yeung, R. Diagrammatic design and study of ansätze for quantum machine learning (2020). 2011.11073.

[4] Preskill, J. Quantum computing in the nisq era and beyond. *Quantum* **2**, 79 (2018).

[5] 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).

[6] Wecker, D., Hastings, M. B. & Troyer, M. Progress towards practical quantum variational algorithms. *Physical Review A* **92**, 042303 (2015).

[7] Coecke, B. & Duncan, R. Interacting quantum observables: categorical algebra and diagrammatics. *New Journal of Physics* **13**, 043016 (2011).

[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] Cowtan, A., Simmons, W. & Duncan, R. A generic compilation strategy for the unitary coupled cluster ansatz (2020). 2007.10515.

[11] 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).

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

[13] Cerezo, M. *et al.* Variational quantum algorithms. *Nature Reviews Physics 3, 625-644 (2021)* **3**, 625–644 (2020). 2012.09265.

Bibliography

[14] 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`.

[15] Helgaker, T., Jørgensen, P. & Olsen, J. *Molecular Electronic-Structure Theory* (Wiley, 2000).

[16] Fetter, A. L., Walecka, J. D. & Kadanoff, L. P. *Quantum Theory of Many Particle Systems*, vol. 25 (AIP Publishing, 1972).

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

[18] Peruzzo, A. *et al.* A variational eigenvalue solver on a photonic quantum processor. *Nature Communications* **5** (2014).

[19] 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`.

[20] Poór, B. *et al.* Completeness for arbitrary finite dimensions of zxw-calculus, a unifying calculus (2023). `2302.12135`.

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

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

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

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

[25] Amy, M., Maslov, D., Mosca, M. & Roetteler, M. A meet-in-the-middle algorithm for fast synthesis of depth-optimal quantum circuits. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* **32**, 818–830 (2013).

[26] Amy, M., Maslov, D. & Mosca, M. Polynomial-time t-depth optimization of clifford+t circuits via matroid partitioning. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* **33**, 1476–1489 (2014).

[27] Nam, Y., Ross, N. J., Su, Y., Childs, A. M. & Maslov, D. Automated optimization of large quantum circuits with continuous parameters. *npj Quantum Information* **4** (2018).

[28] 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`.

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

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