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

Thank you Thomas Cervoni for your constant motivation and support.
Thank you David Tew and Stefano Gogioso for your patient supervision.
Thank you Razin Shaikh, Boldizsár Poór, Richie Yeung and Harny Wang for always finding the time to answer my questions.
Thank you to my friends and family for supporting me during this unconvential Master's.

Summary

A central challenge in computational quantum chemistry is the accurate simulation of fermionic systems. At the heart of these calculations lies the need to solve the Schrödinger equation to determine the many-electron wavefunction. An exact solution to this problem scales exponentially with the number of electrons. Classical computers struggle to store the increasingly large wavefunctions making this problem computationally intractable in many cases. In contrast, gate-based quantum computing presents a promising solution, offering the potential to represent electronic wavefunctions with polynomially scaling resources [1]. In other words, quantum computers are a natural tool of choice for simulating processes that are inherently quantum [2].

In the last two decades many advancements in quantum computing have been made in both hardware and software bringing us closer to being able to simulate molecular systems. Despite these advancements, we remain in the so-called Noisy Intermediate Scale Quantum (NISQ) era, 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 are still capable of performing computations beyond the reach of classical computers. Overcoming the limitations of the NISQ era is crucial for realising the full potential of quantum computing in various fields, including quantum chemistry and materials science.

The Variational Quantum Eigensolver (VQE) algorithm is a method used to estimate the ground state energy of a molecular Hamiltonian by preparing a trial wavefunction,

calculating its energy, and optimising the wavefunction parameters classically until the energy converges to the best approximation for the ground state energy [3]. 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 [4].

This thesis extends methods developed by Richie Yeung [2] for the preparation and analysis of parametrised quantum circuits, and applies them to ansätze representing fermionic wavefunctions. We are concerned with two main questions on this theme. Firstly, can we use the ZX calculus [cite] to gain insights into the structure of the unitary product ansatz in the context of variational algorithms for quantum chemistry? Secondly, in the context of NISQ devices, can we use these insights to build better ansätze with reduced circuit depth and more efficient resources?

Contents

Chapter 1

Background

In this chapter, we will discuss the framework that we use to simulate fermionic systems on a quantum computer, as well as the notation that we will use throughout the text. Starting with Quantum Computation REF(quantum-computation) and Electronic Structure Theory REF(electronic-structure-theory), we will build up to unitary coupled cluster theory and the Variational Quantum Eigensolver REF(vqe).

Fermionic states can generally be represented on a quantum computer in the occupation number representation (section REF(second-quantisation)). That is, the state of each qubit is taken to represent the occupancy of each spin orbital. By representing the fermionic creation and annhilation operators in terms of qubit operators in a way that preserves the fermionic anticommutation relations, we can express the molecular Hamiltonian in terms of qubit operations.

1.1 Quantum Computation

Introduction to Qubits

In contrast to classical computation, where bits form the basis for encoding information, quantum computation makes use of quantum bits (qubits). There are many physical implementations of qubits, however, for the purposes of this thesis, it will suffice to think of them as purely mathematical objects.

Qubits can exist as superpositions of the computational basis: the $|0\rangle$ and $|1\rangle$ states. These states are orthonormal vectors in a two-dimensional complex Hilbert space $\mathbb{C}^2$. We can depict these states on a Bloch sphere as in figure 1.1. Note that the Bloch space does not represent the complex Hilbert space itself.

More generally, we can choose any pair of orthonormal states to form our computational basis. On the bloch sphere, this corresponds to any two vectors pointing in opposite directions. One such computational basis if formed by the $|+\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$ and $|-\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$ states.

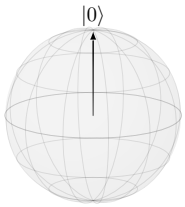


Figure 1.1: $|0\rangle$ basis state

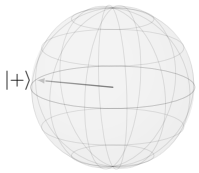


Figure 1.2: $|+\rangle$ basis state

Any qubit $|\psi\rangle$ can be represented as complex linear combination of the chosen basis, provided that the qubit state vector is normalised.

$$|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle \qquad |\alpha|^2 + |\beta|^2 = 1 \qquad \alpha, \beta \in \mathbb{C}$$

Multiple Qubit States

Suppose we have n qubits. By taking the Kronecker product, we can construct 2^n computational basis states.

$$|00\ldots00\rangle = |0\rangle_n \otimes |0\rangle_{n-1} \otimes \cdots \otimes |0\rangle_1 \otimes |0\rangle_0$$
$$\cdots$$
$$|11\ldots11\rangle = |1\rangle_n \otimes |1\rangle_{n-1} \otimes \cdots \otimes |1\rangle_1 \otimes |1\rangle_0$$

Figure 1.3: 2^n computational basis states.

It follows then that any complex linear combination of the computational basis states is also a valid qubit state.

$$|\psi\rangle = \alpha_{00\ldots00}\,|00\ldots00\rangle + \alpha_{00\ldots01}\,|00\ldots01\rangle + \cdots + \alpha_{11\ldots11}\,|11\ldots11\rangle$$

Whilst the Bloch sphere representation of a single qubit is incredibly useful, there is no easy generalisation of the Bloch sphere for multiple qubit states [5].

Quantum Gates

1.2 Electronic Structure Theory

Electronic Structure Problem

References: [6]

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

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

Figure 1.4: Full molecular Hamiltonian in atomic units, where Z_i is the charge of nucleus i and M_i is its mass relative to the mass of an electron.

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

We are able to simplify the problem to an electronic one using the Born-Oppenheimer approximation. Motivated by the large difference in mass of electrons and nuclei, we can approximate the nuclei as stationary on the timescale of electronic motion such that the electronic wavefunction depends only parametrically on the nuclear coordinates. The full molecular wavefunction can then be expressed as an adiabatic separation as below.

$$\Phi_{\text{total}} = \psi_{\text{elec}}(\{r\};\{R\})\,\psi_{\text{nuc}}(\{R\})$$

Within this approximation, the nuclear kinetic energy term can be neglected and the nuclear repulsive term is considered to be constant. Since constants in

eigenvalue equations have no effect on the eigenfunctions and simply add to the resulting eigenvalue, we will omit this too. The resulting equation is the electronic Hamiltonian for N electrons.

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

Figure 1.5: Electronic molecular Hamiltonian in atomic units.

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

$$E_{\text{total}} = E_{\text{elec}} + \sum_{i=1}^{M}\sum_{j>i}^{M} \frac{Z_i Z_j}{|R_i - R_j|}$$

Many-Electron Wavefunctions

References: [6]

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

A spatial molecular orbital is defined as a one-particle function of the position vector, spanning the whole molecule. The spatial orbitals form an orthonormal set $\{\psi_i(\mathbf{r})\}$, which if complete can be used to expand any arbitrary single-particle molecular wavefunction, that is, an arbitrary single-particle function of the position vector. In practice, only a finite set of such orbitals is available to us, spanning only

a subspace of the complete space. Hence, wavefunctions expanded using this finite set are described as being 'exact' only within the subspace that they span.

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

$$\psi_{1,2}(\mathbf{x_1}, \mathbf{x_2}) = \phi_i(\mathbf{x_1})\phi_j(\mathbf{x_2}) \qquad \psi_{2,1}(\mathbf{x_2}, \mathbf{x_1}) = \phi_i(\mathbf{x_2})\phi_j(\mathbf{x_1})$$
$$\Psi_{1,2}(\mathbf{x_1}, \mathbf{x_2}) = \frac{1}{\sqrt{2}}\left[\psi_{1,2}(\mathbf{x_1}, \mathbf{x_2}) - \psi_{2,1}(\mathbf{x_2}, \mathbf{x_1})\right]$$

Figure 1.6: Symmetric Hartree products $\psi_{1,2}(\mathbf{x_1}$ and $\mathbf{x_2})$, $\psi_{2,1}(\mathbf{x_2}, \mathbf{x_1})$, and their antisymmetric linear combination $\Psi_{1,2}(\mathbf{x_1}, \mathbf{x_2})$.

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.

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

Figure 1.7: Slater determinant representing an antisymmetrised N-electron wavefunction.

Since exchanging any two rows or columns of a determinant changes its sign, Slater determinants satisfy the Pauli principle by definition. Slater determinants

constructed from orthonormal spin orbitals are themselves normalised and N electron Slater determinants constructed from different orthonormal spin orbitals are orthogonal to one another [6].

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

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 [6]. 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.

$$\left[-\frac{1}{2}\nabla^2 - \sum_{A=1}^{M}\frac{Z_A}{r_{i_A}} + \boldsymbol{\nu}^{\mathrm{HF}}(i)\right]\phi_i(\mathbf{x}_i) = \varepsilon\phi_i(\mathbf{x}_i)$$

Figure 1.8: Hartree-Fock equation.

For an N electron system, and given a set of $2K$ Hartree-Fock spin orbitals, where $2K > N$, there exist many different single Slater determinants. The Hartree-Fock groundstate being one of these. The remainder are excited Slater determinants, recalling that all of these must be orthogonal to one-another. By treating the Hartree-Fock ground state as a reference state, we can describe the excited states relative to the reference state, as single, double, ..., N-tuple excited states [6].

Second Quantisation

References: [7], [8]

1. Background

In second quantisation, both observables and states (by acting on the vacuum state) are represented by operators, namely the creation and annhilation operators [7]. In contrast to the standard formulation of quantum mechanics, operators in second quantisation incorporate the relevant Bose or Fermi statistics each time they act on a state, circumventing the need to keep track of symmetrised or antisymmetrised products of single-particle wavefunctions [8]. Put differently, the antisymmetry of an electronic wavefunction simply follows from the algebra of the creation and annhilation operators [7], which greatly simplifies the discussion of systems of many identical interacting fermions [8].

The Fock space is a linear abstract vector space spanned by N orthonormal occupation number vectors [7], each representing a single Slater determinant. Hence, given a basis of N spin orbitals we can construct 2^N single Slater determinants, each corresponding to a single occupation number vector in the full Fock space.

The occupation number vector for fermionic systems is succinctly denoted in Dirac notation as below, where the occupation number f_j is 1 if spin orbital j is occupied, and 0 if spin orbital j is unnoccupied.

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

Whilst there is a one-to-one mapping between Slater determinants with canonically ordered spin orbitals and the occupation number vectors in the Fock space, it is important to distinguish between the two since, unlike the Slater determinants, the occupation number vectors have no spatial structure and are simply vectors in an abstract vector space. [7].

Creation and Annhilation Operators

References: [7]

Operators in second quantisation are constructed from the creation and annhilation operators $a_j^\dagger$ and a_j, where the subscripts i and j denote the spin orbital. $a_j^\dagger$ and a_j are one another's Hermitian adjoints, and are not self-adjoint [7].

Taking the excitation of an electron from spin orbital 0 to spin orbital 1 as an example, we can construct the following excitation operator.

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

show ladder operators acting on opposite states

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

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

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

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

Figure 1.9: Anti-commutation relations of fermionic creation and annhilation operators.

The phase factor required for the second quantised representation to be consistent with the first quantised representation is automatically kept track of by the anticommutation relations of the creation and annhilation operators [7].

Hamiltonian in Second Quantisation

The Hamiltonian in second quantisation is constructed from creation and annhilation operators as follows.

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

Where the one-body matrix element h_{ij} corresponds to the kinetic energy of an electron and its interaction energy with the nuclei.

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

1. Background

The two-body matrix element h_{ijkl} corresponds to the repulsive interaction between electrons i and j.

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

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

1.3 Variational Quantum Eigensolver

References: [9], [3], [10]

Fermion-Qubit Encodings

References: [11]

In order to represent the fermionic wavefunction on a quantum computer, we must first create a mapping betweent the fermionic state vector and the qubit state vector. There are a number of fermion-qubit encodings used today including the Jordan-Wigner transformation, Bravyi-Kitaev transformation and the Parity mapping. Whilst the Bravyi-Kitaev transformation and Parity mappings can more efficiently encode the fermionic wavefunction in some ways, we will only consider the Jordan-Wigner transformation for the remainder of this text as its simplicity provides us with a more intuitive picture of the computation.

The form of the occupation number representation vector and the qubit statevector suggests the following identification between electronic states and qubit states.

$$|f_{n-1}\dots f_0\rangle \quad \rightarrow \quad |q_{n-1}\dots q_0\rangle$$

That is, we allow each qubit to store the occupation number of a given spin-orbital. Hence, in order to actually simulate a Hamiltonian we must map the fermionic creation and annhilation operators onto qubit operators, and these operators must behave in the same way as their fermionic analogues.

$$\hat{Q}^+|0\rangle = |1\rangle \qquad \hat{Q}^+|1\rangle = 0 \qquad \hat{Q}|1\rangle = |0\rangle \qquad \hat{Q}|0\rangle = 0$$

The qubit operators must also preserve the fermionic anti-commutation relations in order to satisfy the Pauli antisymmetry requirement.

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

One such qubit encoding is known as the Jordan-Wigner transformation. It expresses the fermionic creation and annhilation operators as a linear combination of the

Pauli matrices.

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

When dealing with **multiple-qubits**, we must also account for the occupation parity of the qubits preceding the target qubit j.

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

We do this by introducing a string of Pauli Z operators that computes the parity of the qubits preceding the target qubit.

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

Where $\prod$ is the tensor product.

A more compact notation is,

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

Where $Z_i^{\rightarrow}$ is the parity operator with eigenvalues ± 1, and ensures the correct phase is added to the qubit state vector.

$$Z_i^{\rightarrow} = Z_i \otimes Z_{i-1} \otimes \cdots \otimes Z_0$$

For instance, the creation operator $a_3^{\dagger}$ maps to the following Pauli string,

$$\hat{a}_3^{\dagger} = \frac{1}{2}(X_3 - iY_3)\otimes Z_2 \otimes Z_1 \otimes Z_0$$
$$\hat{a}_3^{\dagger} = \frac{1}{2}(X_3 \otimes Z_2 \otimes Z_1 \otimes Z_0) - \frac{1}{2}i(Y_3 \otimes Z_2 \otimes Z_1 \otimes Z_0)$$

Usually we drop the subscript specifying the orbital acted on.

Unitary Coupled Cluster

Whilst unitary coupled cluster theory was proposed in X, interest in its applications has been minimal due to the inability of classical computers to efficiently evaluate its equations. As suggested by Peruzzo et al [12], the UCC formulation of a

wavefunction can be efficiently implemented on a quantum computer using quantum gates.

Unitary coupled cluster theory allows us to represent an arbitrary state $|\psi\rangle$ using the following exponential ansatz.

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

Where $|\psi_0\rangle$ is a single reference Slater determinant, usually the Hartree-Fock groundstate obtained via the self-consistent field method. The exponential operator $U(\theta) = e^{\hat{T}(\theta)-\hat{T}^{\dagger}(\theta)}$ is unitary since its exponent $\hat{T}(\theta) - \hat{T}^{\dagger}(\theta)$ is anti-Hermitian.

The excitation operators are given by

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

Where i, j indexes occupied spin orbitals and a, b indexes virtual, or unoccupied, spin orbitals. The resulting unitary operator $U(\theta)$ cannot be directly implemented on a quantum computer since the terms of the excitation operator do not commute. Instead, we must invoke the Trotter formula to approximate the unitary.

$$e^{A+B} = (e^{A}\, e^{B})^{\rho}$$

Taking a single Trotter step $\rho = 1$, we obtained the disentangled UCC

$$U_{t1}(\theta) = \prod_{m} e^{\theta_m(\tau_m - \tau_m^{\dagger})} \tag{1.1}$$

Where m indexes all possible excitations. It has been shown in [13] that the disentangled UCC can exactly parametrise any state.

"In practice, the excitations are truncated to only include single and double excitations. This UCCSD ansatz has been popular in the VQE literature and is often the benchmark for more cost effective methods." [10].

1. Background

"Only UCCSD excitation operators which conserve spin were used for ansatz construction. Ansatze generated in this manner preserve the spin symmetry of the spin reference" [10]

—

In this chapter we introduce we introduce Unitary Coupled Cluster theory which will serve as the basis for the Variational Quantum Eigensolver algorithm introduced in chapter XXX. In particular, we are interested in developing the Unitary-Product State which will serve as a variational ansatz.

One notable advantage of the VQE algorithm is its ability to be run on any quantum architecture, moreover, we can leverage architecture requirements when designing the variational ansatz. [3].

"Even in the event that some error correction is required to exceed current computational capabilities, this same robustness may translate to requiring minimal error cor- rection resources when compared with other algorithms." [3]

We begin by...

Within the traditional coupled-cluster framework, the ground electronic state is prepared by applying the CC operator to a reference state (usually Hartree-Fock).

$$|\psi\rangle = e^{\hat{T}} |\phi_0\rangle$$

Where $\hat{T}$ is the cluster excitation operator.

Quantum gates, however, must be unitary operators, so instead, we work within the UCC framework.

$$|\psi\rangle = e^{\hat{T}} |\phi_0\rangle$$

Where $\hat{T}$ is now an **anti-Hermitian** operator, and $e^{\hat{T}}$ is unitary.

1. Background

In general, we can prepare exact electronic states by applying a sequence of k parametrised unitary operators to our reference state.

$$|\psi\rangle = \prod_{i}^{k} U_i(\theta_i) \, |\phi_0\rangle$$

Where $U_i(\theta_i)$ is a parametrised unitary operator

The parameters θ_i are then optimised to find the ground state energy.

General fermionic single and double excitation operators are defined as,

$$a_q^\dagger a_p \text{ and } a_r^\dagger a_s^\dagger a_q a_p$$

Exciting one electron from p to q, and two electrons from p, q to r, s respectively.

Taking a linear combination of these, we obtain **anti-Hermitian** fermionic single and double excitation operators.

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

Such that upon exponentiating, we obtain **unitary** operators.

$$U_p^q = e^{\hat{\kappa}_p^q} \qquad U_{pq}^{rs} = e^{\hat{\kappa}_{pq}^{rs}}$$

Variational Quantum Eigensolver

Recalling the Jordan-Wigner encoding for the creation and annhilation operators,

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

1. Background

The anti-Hermitian fermionic single and double excitation operators κ_p^q and κ_{pq}^{rs}

$$F_p^q = \frac{i}{2}(Y_pX_q - X_pY_q)\prod_{k=p+1}^{q-1} Z_k$$

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

Multiplying by θ and exponentiating yields the parametrised unitary qubit operators,

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

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

To summarise, we constructed anti-Hermitian single and double excitation operators from a linear combination of fermionic excitation operators,

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

We then mapped these to qubit operators using the Jordan-Wigner transformation,

$$\hat{\kappa}_p^q \xrightarrow{\text{JW}} F_p^q \qquad\qquad \hat{\kappa}_{pq}^{rs} \xrightarrow{\text{JW}} F_{pq}^{rs}$$

And finally, we exponentiated to yield the parametrised unitary qubit operators.

$$U_p^q(\theta) = e^{\theta_p^q F_p^q} \qquad U_{pq}^{rs}(\theta) = e^{\theta_{pq}^{rs} F_{pq}^{rs}}$$

Let's look again at the parametrised single-body unitary operator,

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

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

The first exponential term can be implemented by the following phase gadget.

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

Left CNOT ladder construction calculates the parity of the qubit state, and applies a rotation in the Z basis if the parity is odd.

Whilst the second exponential term can be implemented by the phase gadget.

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

Together, they constitute the single-body unitary excitation operator $U_p^q(\theta)$

By defining the ordering of spin-orbitals such that adjacent spin-orbitals share the same spatial orbital, adjacent single-body operators commute.

$$\left[\hat{\kappa}_p^q, \hat{\kappa}_{p+1}^{q+1}\right] = 0$$

The same is therefore true for the resulting qubit operators,

$$\left[F_p^q, F_{p+1}^{q+1}\right] = 0$$
$$p, q \in \text{even} \qquad p+1, q+1 \in \text{odd}$$

This allows us to define the parametrised unitary qubit operators in terms of spin-adapted excitation operators.

$$U_p^q(\theta) = \exp\left[\theta\left(F_p^q + F_{p+1}^{q+1}\right)\right]$$

In other words, since F_p^q and F_{p+1}^{q+1} commute, we can think of them as a single operator with a single parameter.

Chapter 2

ZX Calculus

The ZX calculus is a diagrammatic language for reasoning about quantum processes LALALA

We will then introduce the rewrite rules... that come equipped with the ZX calculus. These rules extend the ZX calculus from notation into a language.

Notation - flow of time goes from left to right - global scalar factor

Note that in this text, we will interpret the flow of time from left to right. Whilst we obtain the correct states, we obtain the wrong scalar factor. For the remainder of this thesis, we will ignore global non-zero scalar factors. Hence, equal signs should be interpreted as 'equal up to a global phase'.

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 [14]. 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, provided that the order of inputs and outputs is preserved.

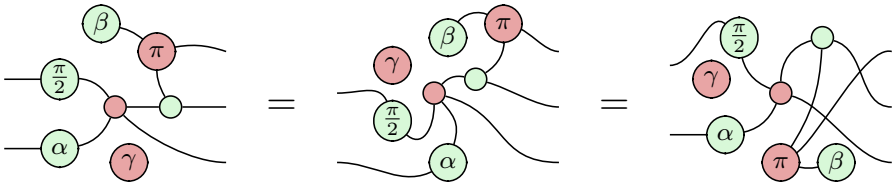
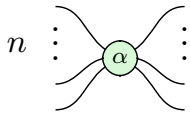


Figure 2.1: Three equivalent ZX diagrams (*only connectivity matters*).

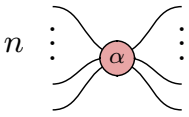
Z Spiders (green) are defined with respect to the Z eigenbasis ($|0\rangle$ and $|1\rangle$) such that a Z Spider with n inputs and m outputs represents the following linear map.



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

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

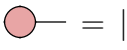
X Spiders (red), are defined with respect to the X eigenbasis ($|+\rangle$ and $|-\rangle$).



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

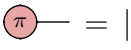
Figure 2.3: Interpretation of an X Spider as a linear map.

We can represent the Z eigenstates ($|0\rangle$ and $|1\rangle$) using an X spider, or the X eigenstates ($|+\rangle$ and $|-\rangle$) using a Z spider, with a phase of either 0 or π.



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

Figure 2.4: $|0\rangle$ eigenstate



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

Figure 2.5: $|1\rangle$ eigenstate

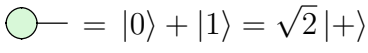


Figure 2.6: $|+\rangle$ eigenstate

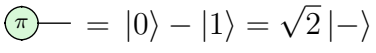
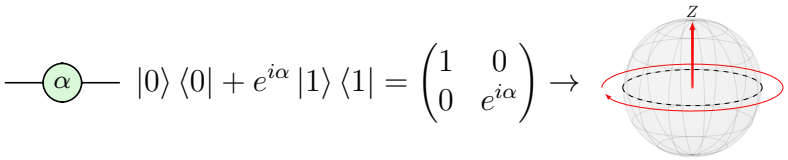
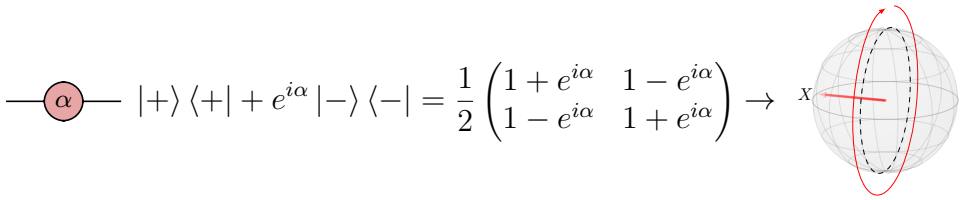


Figure 2.7: $|-\rangle$ eigenstate

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





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

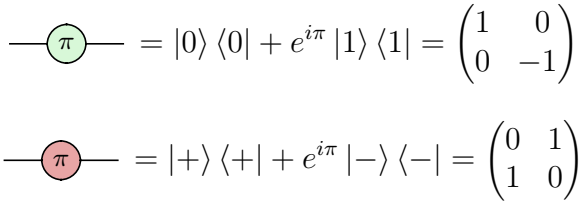
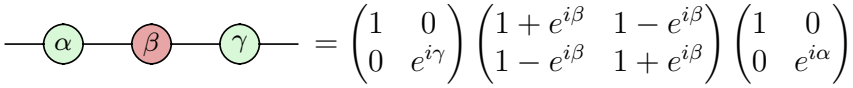


Figure 2.8: Pauli Z and X gates in the ZX calculus.

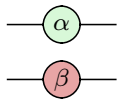
Composition

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



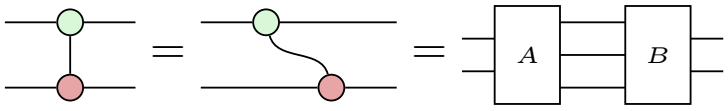
Alternatively, we could have chosen to compose the spiders in parallel, resulting in

the tensor product.

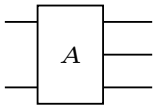


$$= \begin{pmatrix} 1 & 0 \\ 0 & e^{i\alpha} \end{pmatrix} \otimes \begin{pmatrix} 1+e^{i\beta} & 1-e^{i\beta} \\ 1-e^{i\beta} & 1+e^{i\beta} \end{pmatrix}$$

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

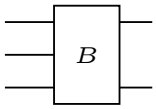


We can calculate matrix A, consisting of a single-input and two-output Z Spider (4×2 matrix) and an empty wire (identity matrix), by taking the tensor product.



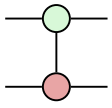
$$= \quad = \begin{pmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 1 \end{pmatrix} \otimes \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

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



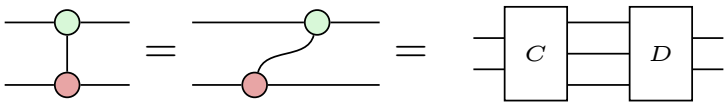
$$= \quad = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \otimes \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix}$$

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

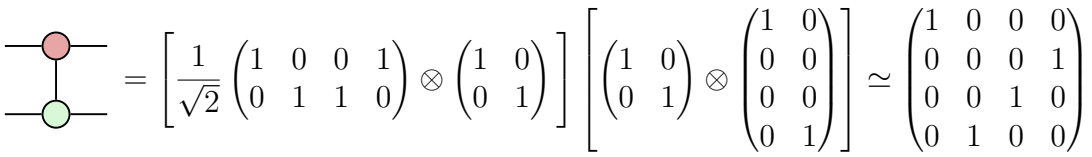


$$= \left[\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \otimes \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix} \right] \left[\begin{pmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 1 \end{pmatrix} \otimes \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \right] \simeq \begin{pmatrix}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

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



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



Hadamard Generator

All quantum gates are unitary transformations. Therefore, up to a global phase, an arbitrary single qubit rotation U can be viewed as a rotation of the Bloch sphere about some axis. We can decompose the unitary U using Euler angles to represent the rotation as three successive rotations [14].

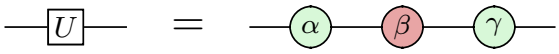


Figure 2.9: Arbitrary single-qubit rotation.

Recall that the Hadamard gate H switches between the $|0\rangle/|1\rangle$ and $|+\rangle/|-\rangle$ bases. That is, it corresponds to a rotation of the Bloch sphere π radians about the line bisecting the Z and X axes.

There are many equivalent ways of decomposing the Hadamard gate H using Euler angles. By choosing $\alpha = \beta = \gamma = \frac{\pi}{2}$, we obtain H up to a global phase of $\exp(-i\pi/4)$. See Appendix 6.1 for other definitions.

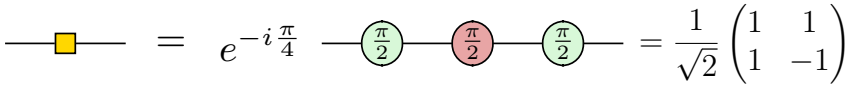


Figure 2.10: Definition of the Hadamard generator.

2.2 Rewrite Rules

Spider Fusion

The most fundamental rule of the ZX calculus is the *spider fusion* rule [14]. It states that two spiders connected by one or more wires fuse if they are the same colour. It is the generalisation of adding the phases of successive rotations of the Bloch sphere. Since we interpret the phases α and β as $e^{i\alpha}$ and $e^{i\beta}$, it follows that the phase $\alpha + \beta$ is modulo 2π.

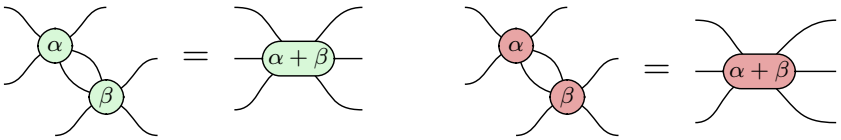


Figure 2.11: Spider fusion rule for Z spiders (left) and X spiders (right).

We can use this rule to identify commutation relations such as Z rotations commuting through CNOT controls, and X rotations, through CNOT targets.

Identity Removal

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

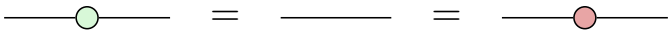
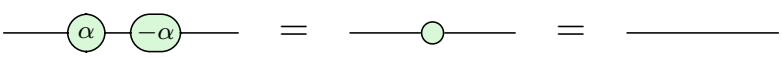


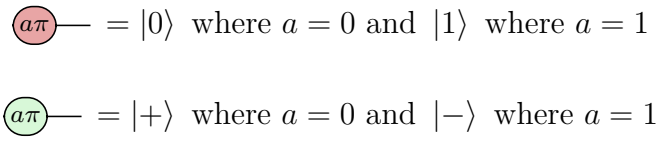
Figure 2.12: Identity removal rule.

Combining this with the spider fusion rule (2.11), we see that two successive rotations with opposite phases is equivalent to an empty wire.



State Copy and π Copy Rules

We can represent the Z and X eigenstates setting the phases of Z or X spiders by some boolean variable a (0 or 1) multiplied by π [14].



The π *copy* rule 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 flips the spider's phase. The *state copy* rule is a similar rule that applies to the Z and X eigenstates.

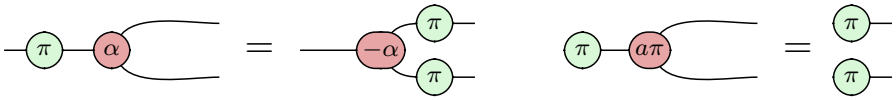


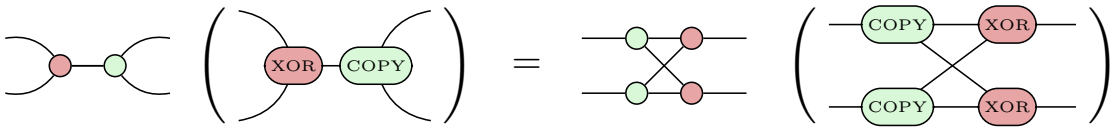
Figure 2.13: π copy (left) and state copy (right) rules for Pauli Z gate and Z eigenstates.

Bialgebra Rule

Using the spider fusion rule (2.11), 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 (2.13), 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 2.14: X spider as a XOR gate (left) and Z spider as a COPY gate (right).

Hence, using the natural commutation relation of the classical XOR and COPY gates, we define the *bialgebra* rule. We encourage the reader to verify this relation.



Chapter 3

Pauli Gadgets

Pauli gadgets form the building blocks for ansätze for quantum chemical simulations. We will see in chapter XXX how they can be used to construct excitation operators to ultimately calculate electronic correlation in a molecular system.

3.1 Phase Gadgets

Missing: matrix representation

Phase gadgets are a special type of Pauli gadget formed from Pauli strings consisting of the Pauli I and Z matrices. A Pauli string P is defined as a tensor product of Pauli matrices $P \in \{I, Y, Z, X\}^{\otimes n}$, where n is the number of qubits in the system. each Pauli operator acts on a distinct qubit. Note that since the Pauli matrices are Hermitian, so are Pauli strings. Thus $Z \otimes Y \otimes X = ZYX$ means apply Pauli Z, Y and X to qubits 0, 1 and 2 respectively.

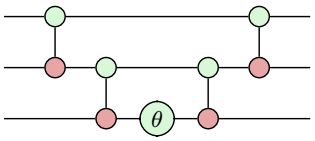
Stone's Theorem [15] states that a strongly continuous one parameter unitary group $U(\theta)$ is generated by a Hermitian operator, P.

$$U(\theta) = \exp\left(i\frac{\theta}{2}P\right) \quad \text{where} \quad e^X = 1 + X + \frac{1}{2}X^2 + \frac{1}{6}X^3 + \dots$$

There is therefore a one-to-one correspondence between Hermitian operators and one parameter unitary groups [2]. The time evolution of a quantum mechanical system, described by the Hamiltonian H, is defined by the one parameter unitary group e^{itH}, whilst arbitrary rotation gates in the Z, X and Y bases are described by the one parameter unitary groups of the Pauli matrices Z, X and Y.

$$R_Z(\theta) = \exp\left(i\frac{\theta}{2}Z\right) \qquad R_Y(\theta) = \exp\left(i\frac{\theta}{2}Y\right) \qquad R_X(\theta) = \exp\left(i\frac{\theta}{2}X\right)$$

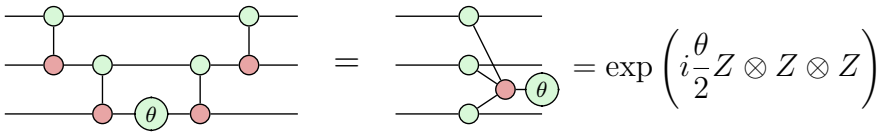
Phase gadgets are defined as the one parameter unitary groups of Pauli strings consisting of the I and Z matrices, $P \in \{I, Z\}^{\otimes n}$. They correspond to quantum circuits made up of a Z rotation wedged between two CNOT layers.



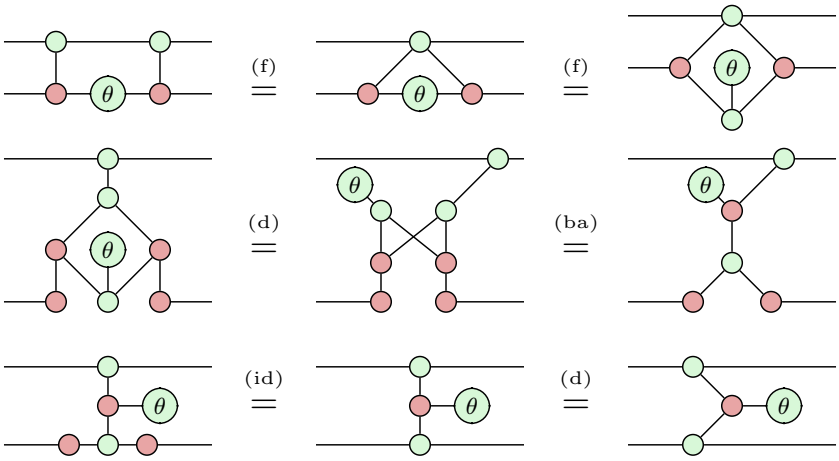
$$= \exp\left(i\frac{\theta}{2}Z \otimes Z \otimes Z\right)$$

The first CNOT layer can be thought of as computing the parity of the input state by entangling the qubits. The Z rotation then rotates the entangled state by $e^{i\theta/2}$ or $e^{-i\theta/2}$, depending on the parity of the state, and the final CNOT layer

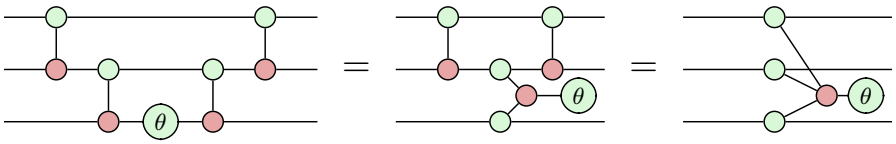
uncomputes the parity. Phase gadgets necessarily correspond to diagonal unitary matrices in the Z basis since they apply a global phase to a given state without changing the distribution of the observed state [2]. This diagonal action suggests that a symmetric ZX diagram exists for phase gadgets, as is indeed the case.



By deforming d our phase gadget in quantum circuit notation and using the identity id (2.12), spider fusion f (2.11) and bialgebra ba (2.2) rules, we are able to show the correspondence with its form in the ZX calculus.



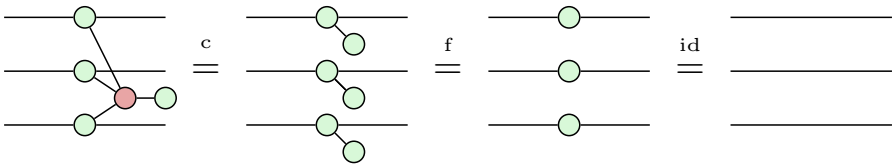
It is then a simple matter of recursively applying this proof to phase gadgets in quantum circuit notation to generalise to arbitrary arity.



Not only is this representation intuitively self-transpose, but it comes equipped with various rules describing its interaction with other phase gadgets and quantum gates.

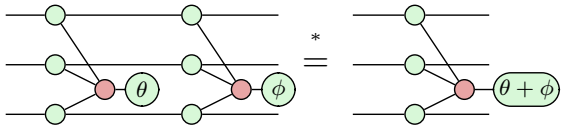
Phase Gadget Identity Rule

Phase gadgets with an angle $\theta = 0$ can be shown to be equivalent to identity using the state copy (2.13), spider fusion (2.11) and identity removal (2.12) rules.



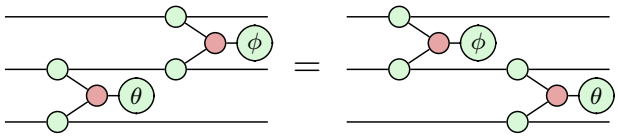
Phase Gadget Fusion Rule

Any two adjacent phase gadgets formed from the same Pauli string fuse and their phases add. This is achieved using the spider fusion rule (2.11) and the bialgebra rule (2.2). See Appendix 6.1 for the intermediate steps marked (*).



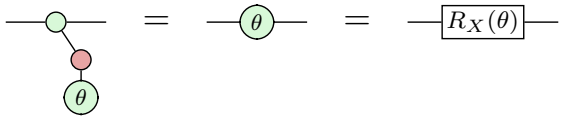
Phase Gadget Commutation Rule

Any two adjacent phase gadgets commute, as can be shown using the spider fusion rule 2.11 (fuse then unfuse the legs 2.11).



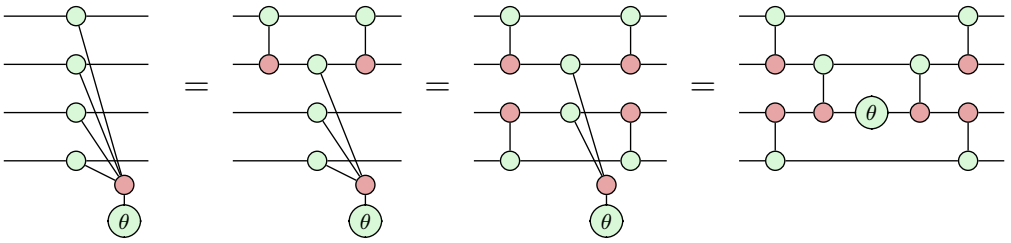
Single-Legged Phase Gadgets

Single-legged phase gadgets are equivalent to Z rotations (see Appendix 6.1).



Phase Gadget Decomposition

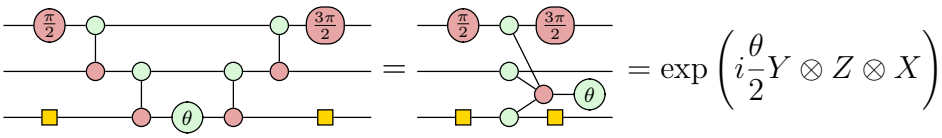
There are many equivalent ways of decomposing a phase gadget into quantum circuit notation using the bialgebra rule (2.2). More generally, we can show that it is possible to decompose a phase gadget such that it his a circuit depth of $\log_2(n)$ instead of n, where n is the number of qubits.



Clearly, it is then advantageous to manipulate circuits of phase gadgets using the ZX calculus, and to extract the corresponding quantum circuit only after the ZX diagram has been optimised.

3.2 Pauli Gadgets

Pauli gadgets are defined as the one parameter unitary groups of Pauli strings consisting of all four Pauli matrices, $P \in \{I, Z, X, Y\}$. They are essentially phase gadgets associated with an additional change of basis.



Whilst phase gadgets alone cannot change the distribution of the observed state, Pauli gadgets, which are associated with a change of basis, can [2]. We will later see how Pauli gadgets form the building blocks in ansätze used for quantum chemical simulations. Similarly to phase gadgets, Pauli gadgets come equipped with a set of rules describing their interactions with other gadgets and quantum gates.

Rules for Pauli Gadgets

Adjacent Pauli gadgets with *matching legs* fuse and their phases add (cancel adjacent change of basis layers, then fuse phase gadgets 3.1). Adjacent Pauli gadgets with *no mismatching legs* commut (cancel adjacent change of basis layers, then use the spider fusion rule 2.11).

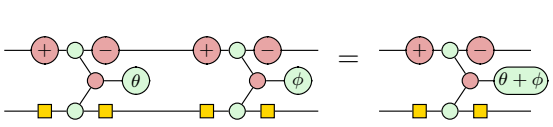


Figure 3.1: Fusion Rule

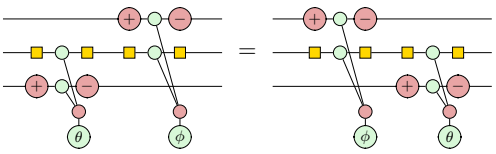
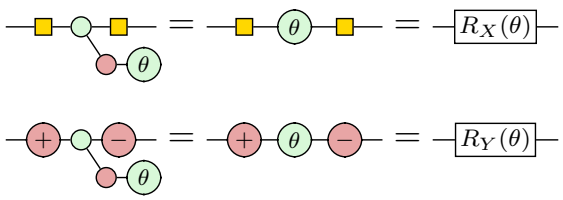


Figure 3.2: Commutation Rule

Single-Legged Pauli Gadgets

Finally, we can show that single-legged Pauli gadgets are equivalent to a rotation in the corresponding basis using the single-legged phase gadget rule (3.1).


θ
=
θ
=
$R_X(\theta)$
+
−
θ
=
+
θ
−
=
$R_Y(\theta)$

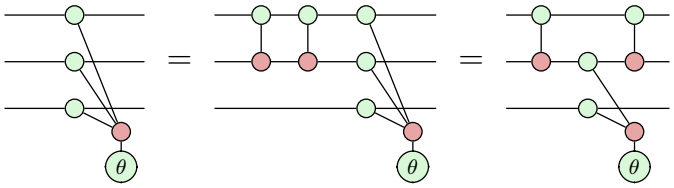
Chapter 4

Commutation Relations

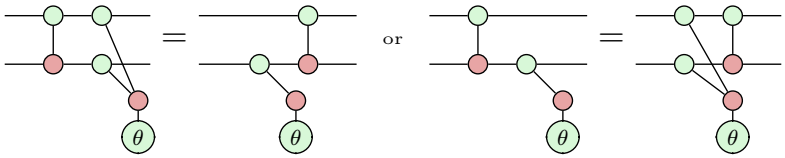
In this chapter, we will develop a set of rules to describe the interaction of Pauli gadgets with other Pauli gadgets, CNOT gates and Clifford gates. We will refer to these rules as *commutation relations*. Note, we are not referring to the quantum mechanical definition of commutation, but rather, what happens to two ZX diagrams when pushed through one another.

4.1 CNOT Commutation

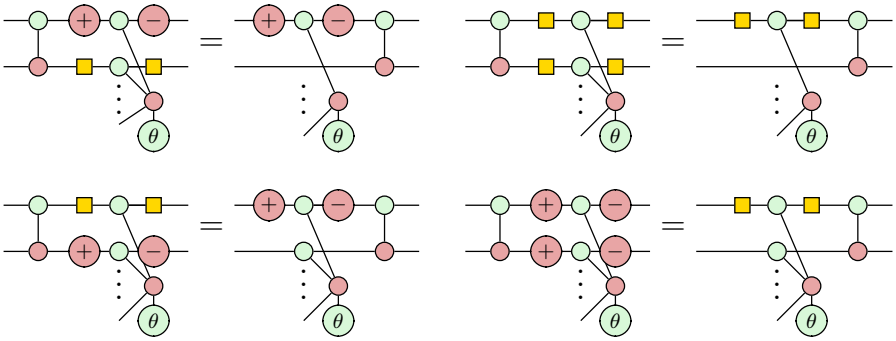
When a CNOT gate is *pushed* through a Pauli gadget, it modifies its legs. That is, the resulting gadget is defined by a new Pauli string. We have already encountered this when decomposing phase gadgets (3.1) using the bialgebra rule (2.2).



Above, we have inserted two adjacent CNOTs (self-inverse), then pushed one of them through the phase gadget. We can say that a CNOT gate 'commutes' through a $\exp[i\frac{\theta}{2}(Z \otimes Z)]$ gadget yielding a $\exp[i\frac{\theta}{2}(I \otimes Z)]$ gadget.



The ZX calculus can be used to identify how CNOT gates commute through any Pauli gadget. Note that the following examples are not exhaustive – there are 16 possible permutations with repetition of $\{I, X, Y, Z\}$ taken two at a time.



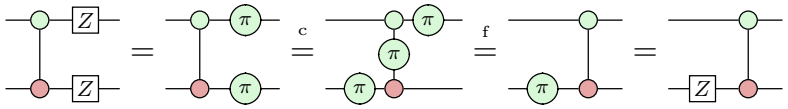
In practice, it may be tedious to use the bialgebra and other rules to identify each commutation relation. There exists, however, a simple trick for identifying these relations. The CNOT gate belongs to the Clifford group C. That is, the set of transformations that normalise the Pauli group. For instance, conjugating the Pauli

X gate with the Hadamard H (where $H \in C$), yields the Z gate.

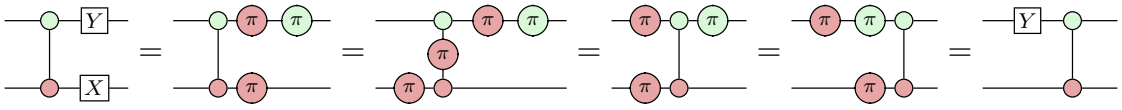
$$Z = HXH$$

Recall that Pauli gadgets are defined as the one parameter unitary groups of a given Pauli string P, where $P \in \{I, Z, X, Y\}^{\otimes n}$. 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 conjugated Pauli string (see Appendix 6.1). In other words, if we can determine the behaviour of a pair of Paulis with the CNOT gate, we can know the behaviour of the corresponding gadget.

Let us first derive the phase gadget decomposition commutation relation (3.1). We first express two Pauli Z gates as Z rotations. We then push the bottom Pauli through the CNOT target (red X spider) using the π copy rule (2.13). We can then push the top Pauli through the CNOT control using the spider fusion rule (2.11) to cancel one of the copied Pauli Z gates obtaining the same relation as before.



Up to a global phase of $-i$ (CHECK), the Pauli Y gate can be expressed as a Pauli X gate followed by a Pauli Z gate. We will use this to identify how the CNOT gate interacts with a $\exp[i\frac{\theta}{2}(X \otimes Y)]$ gadget.



Using this method, we are able to derive all CNOT commutation relations (see Appendix 2 for the complete set of CNOT commutation relations).

Chapter 5

ZxFermion Software

ZxFermion (visit github.com/aymannel/zxfermion for documentation) is a Python package that I wrote for the manipulation and visualisation of circuits of Pauli gadgets. It is built on top of the PyZX `BaseGraph` API [16] and Stim [17]. The motivation for building this package came from the need for a user-friendly tool to explore research ideas related to circuits of Pauli gadgets.

Whilst there are existing software solutions like PauliOpt, which focus on circuit simplification using architecture-aware synthesis algorithms [18], ZxFermion provides a more acccessible alternative, as well as offering tools for studying the interaction of Pauli gadgets with Clifford and Pauli gates using Stim's `Tableau` class.

ZxFermion adheres to strong software engineering principles and has undergone thorough testing, ensuring the reliability and ease of use for users implementing research ideas. It is designed to integrate with Jupyter notebook environments, enabling users to visualise interactive ZX diagrams directly in the output cell.

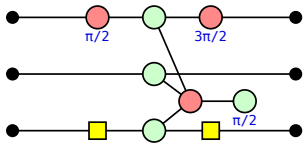
All of the proofs developed in the following chapter can be replicated using ZxFermion, showcasing a noteworthy acceleration in research pace. We anticipate that both chemists and computer scientists exploring quantum computing within the VQE framework will find this software tool advantageous.

ZxFermion is entirely my work. This chapter only introduces the relevant features of the software. Please see the full documentation for other features.

5.1 Creating Gadgets and Circuits

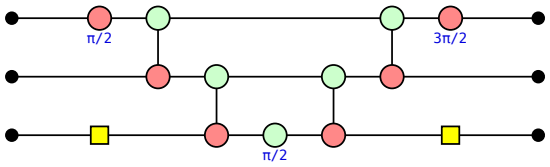
In this section, we will introduce the `Gadget` class, which we use to represent Pauli gadgets. We provide a Pauli string and a phase to instantiate a `Gadget()` object.

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



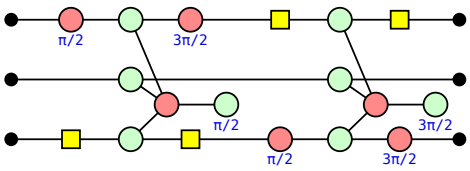
By setting the `as_gadget` option to `False`, we can view the gadget in its expanded form, that is, in quantum circuit notation.

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



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

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

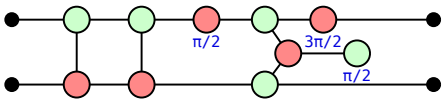


ZxFermion also implements standard quantum gates via the `CX`, `CZ`, `X`, `Z`, `XPlus`, `XMinus`, `ZPlus` and `ZMinus` classes. As we will see in the next section, we have implemented the logic describing the interaction of Pauli gadgets with these gates.

5.2 Manipulating Circuits

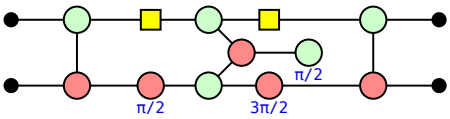
The `GadgetCircuit` class comes equipped with the `apply()` method that allows us to insert a quantum gate, and its Hermitian conjugate, into the circuit. This operation preserves the matrix corresponding to a given cricuit. The method takes the quantum gate to apply as its first parameter, whilst the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

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



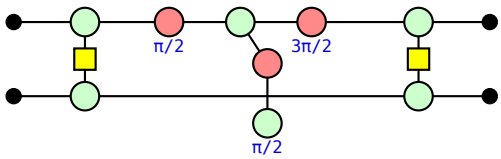
If no specific positions are specified, the insertion defaults to placing one quantum gate at the start, and its Hermitian conjugate at the end, of the circuit. The `GadgetCircuit` class then manages the relevant commutation relations, ensuring the expected gadget outcome. Hence, we below, we observe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

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



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

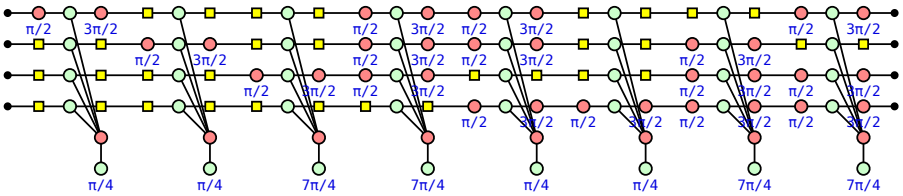
```
circuit.apply(CZ(0, 1), draw=True)
```



We can also use the `GadgetCircuit` class to manipulate circuits consisting of multiple gadgets simultaneously. In the following circuit, we have instantiate a paired double excitation operator as a `GadgetCircuit` object.

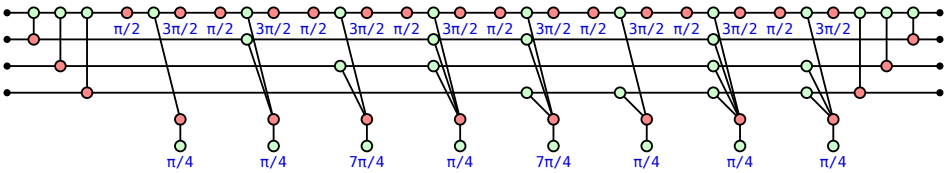
```
gadgets = [
    Gadget('YXXX', phase=1/4),  Gadget('XYXX', phase=1/4),
    Gadget('XXYX', phase=-1/4), Gadget('YYYX', phase=-1/4),
    Gadget('YYXY', phase=1/4),  Gadget('XXXY', phase=1/4),
    Gadget('XYYY', phase=-1/4), Gadget('YXYY', phase=-1/4)
]

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



Then, by conjugating the circuit with CNOTs, each with a different target qubit, we observe the following quantum circuit.

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



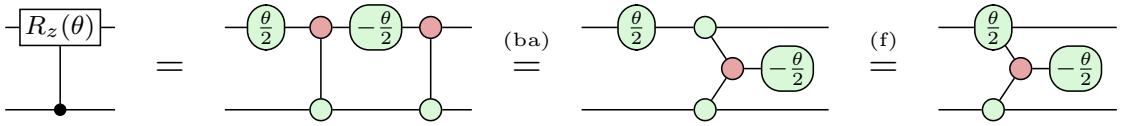
Chapter 6

Controlled Rotations

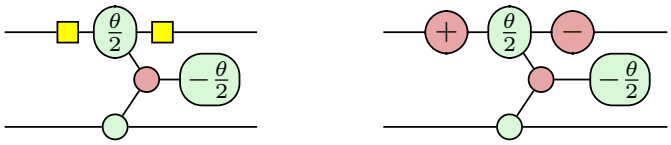
6.1 Controlled-Rotations

Singly-Controlled Rotations

Singly-controlled Z rotation.



Singly-controlled X and Y rotations obtained by conjugating the control qubit.



Doubly-Controlled Rotations

hello world

Triply-Controlled Rotations

hello world hello world

Appendices

Hadamard

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

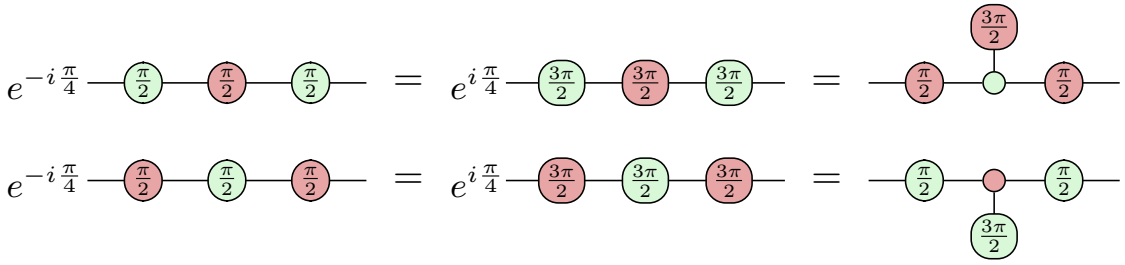
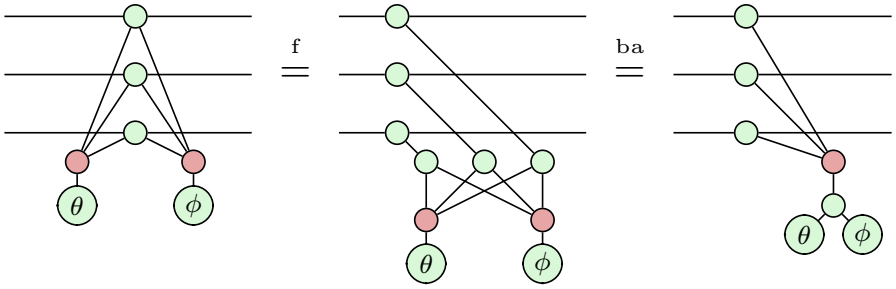


Figure 1: Equivalent definitions of the Hadamard generator.

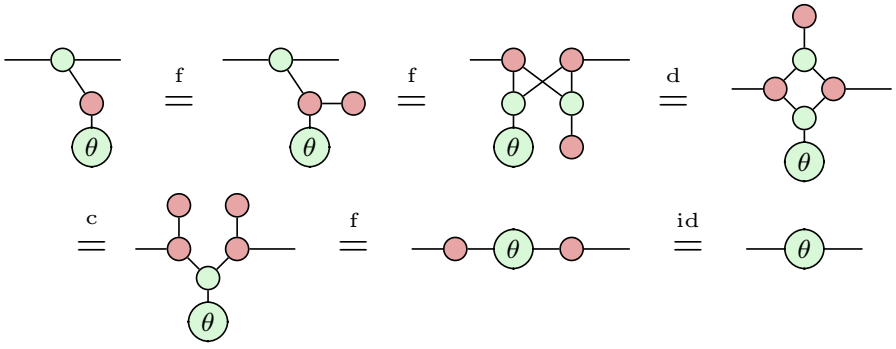
Phase Gadgets

We can show how two adjacent phase gadgets fuse using the spider fusion (2.11) and bialgebra (2.2) rules as follows.



Single-Legged Phase Gadgets

We can show that single-legged phase gadgets are equivalent to Z rotations using the bialgebra (2.2), spider fusion (2.11), state copy (2.13) and identity (2.12) rules as follows.



Clifford Conjugation Stuff

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

CNOT Commutation Relations

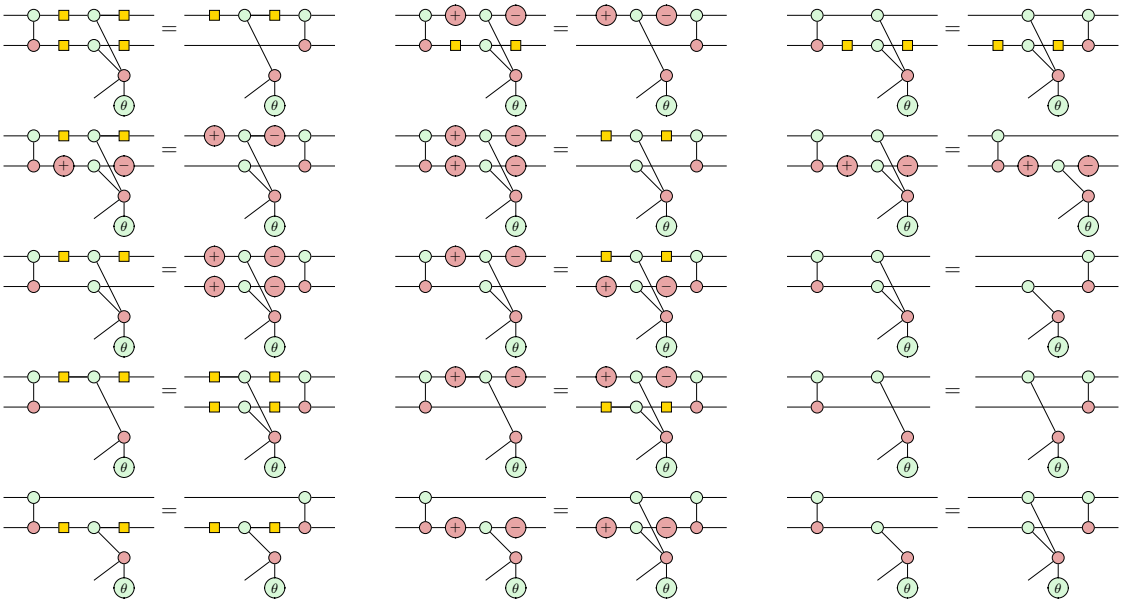


Figure 2: Complete set of CNOT commutation relations.

Bibliography

[1] 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).

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

[3] 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).

[4] Kirby, W. M. & Love, P. J. Variational quantum eigensolvers for sparse hamiltonians. *Phys. Rev. Lett. 127, 110503 (2021)* **127**, 110503 (2020). 2012.07171.

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

[6] 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.

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

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

[9] Anand, A. *et al.* A quantum computing view on unitary coupled cluster theory. *Chemical Society Reviews* **51**, 1659–1684 (2021). 2109.15176.

[10] Chan, H. H. S., Fitzpatrick, N., Segarra-Martí, J., Bearpark, M. J. & Tew, D. P. Molecular excited state calculations with adaptive wavefunctions on a quantum eigensolver emulation: reducing circuit depth and separating spin states. *Physical Chemistry Chemical Physics* **23**, 26438–26450 (2021).

[11] 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.

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

[13] 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.

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

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

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

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

[18] Gogioso, S. & Yeung, R. Annealing optimisation of mixed zx phase circuits. *Electronic Proceedings in Theoretical Computer Science* **394**, 415–431 (2023).