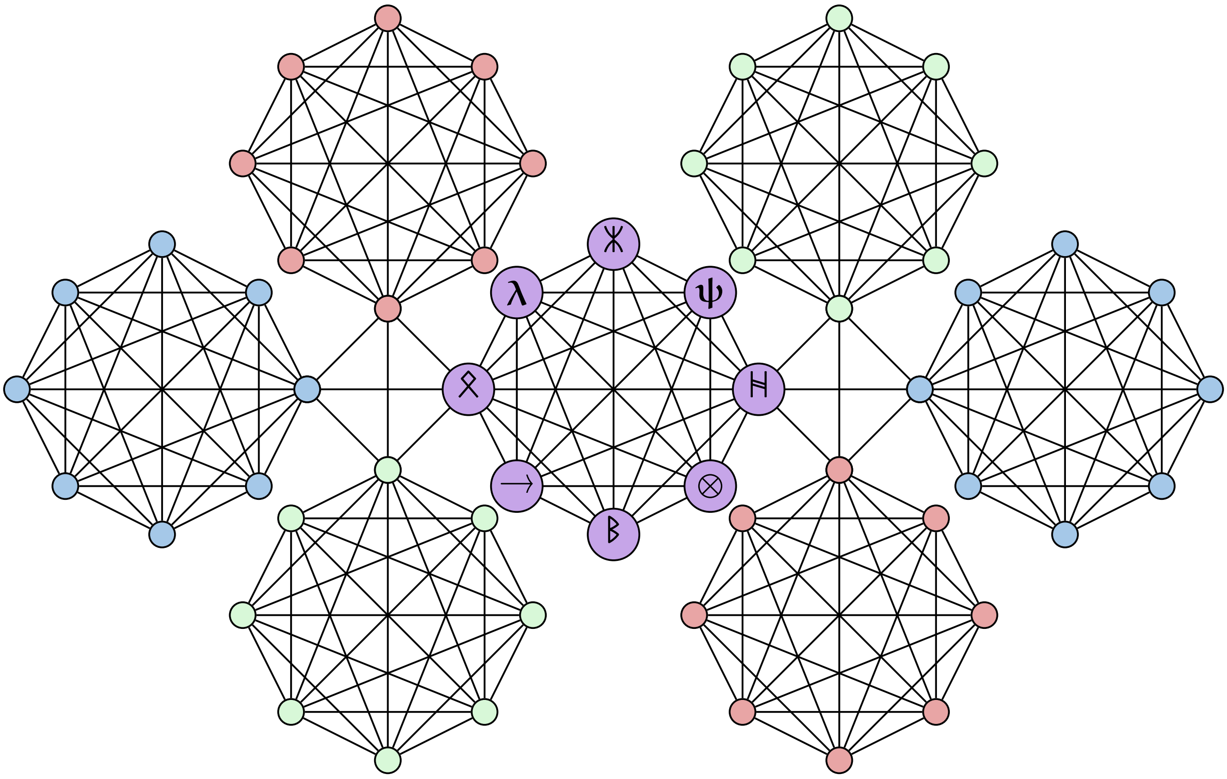


The Zeta Calculus

A λ-calculus for quantum theories

Master's Thesis in Physics & Computer Science

NICKLAS BOTÖ
FABIAN FORSLUND

Department of Computer Science and Engineering
Chalmers University of Technology
University of Gothenburg
Gothenburg, Sweden 2023

Master's thesis 2023

The Zeta Calculus

A λ-calculus for quantum theories

NICKLAS BOTÖ ⊗ FABIAN FORSLUND

Department of Computer Science and Engineering
Chalmers University of Technology
University of Gothenburg
Gothenburg, Sweden 2023

The Zeta Calculus
A λ-calculus for quantum theories
NICKLAS BOTÖ and FABIAN FORSLUND

Supervisor: Robin Adams, Computer Science and Engineering
Examiner: Devdatt Dabhushi, Computer Science and Engineering

Master's Thesis 2023

Department of Computer Science and Engineering
Division of Computing Science
Logic and Types Unit

Chalmers University of Technology and University of Gothenburg
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: A representation of the complex Hopf fibration

Typeset in LaTeX
Gothenburg, Sweden 2023

The Zeta Calculus
A λ-calculus for quantum theories
NICKLAS BOTÖ and FABIAN FORSLUND
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg

Abstract

We propose an extension of the λ-calculus that models the behaviour of observables in quantum theory, *the ζ-calculus*. We give the definition of this language in terms of symmetric monoidal categories, a category which has sufficient structure to capture finite-dimensional Hilbert spaces. The addition of abstractions which introduce a variable with reference to some observable makes it possible to both duplicate and discard this variable freely, while possibly producing some effect such as entanglement. The general definition is then applied to concrete cases, the first one producing a functional quantum programming language. This language is shown to provide novel features and programming techniques, and is able to be compiled to current quantum computers. The second case applies the theory to fermionic quantum field theory, demonstrating that the generality of the theory is able to capture more than quantum computation, creating a model which we call *spacetime computation*. We identify that these models of the theory lie on a hierarchy of models, on which we conjecture the existence of *orders of computation*, ranging from the classical λ-calculus up to two orders above that of quantum computation. Lastly, we try to connect the scientific background of the thesis, quantum mechanics, and discuss what the philosophical implications of a certain interpretation, the relational interpretation of quantum mechanics, are on the philosophical framework of materialism. With this we present both a formal system at the intersection of theoretical computer science and physics, and philosophical motivations underlying the field as a whole.

Keywords: Lambda calculus, formal languages, type theory, category theory, quantum computation, quantum mechanics, dialectical materialism.

Acknowledgements

Detta arbete har genomförts under stress och oroliga tider. Akademins stela och envisa manér hjälper sällan då. Trots detta finns det alltid människor som vill hjälpa. I denna mån vill jag självklart tacka min handledare Robin, som visade mig att jag kunde bidra till forskning. Jag vill även tacka mina kära vänner, som varit mest värdefulla för mig. Däribland kamrat Fabian, vars fysikaliska tankesätt frustrerar och utvecklar mig.

Nicklas Botö, Göteborg, 2023

Jag har ingen att tacka, förutom Gud.

Fabian Forslund, Göteborg, 2023

Contents

List of Figures

List of Tables

1

Introduction

Beauty crowds me till I die,
Beauty, mercy have on me!
But if I expire today,
Let it be in sight of thee.

E. Dickinson

Welcome to the thesis. We introduce an extension to the classical λ-calculus for computation over a set of observable structures in monoidal categories. We call this extension the ζ-calculus. We extend the internal type theory for symmetric monoidal categories [1], a λ-calculus with a type system based on the multiplicative fragment of intuitionistic linear logic, to a non-linear language by introducing *abstraction bases* that allow for both contraction and weakening. Abstraction bases correspond to observable structures which are †-special commutative Frobenius algebras in †-symmetric monoidal categories. [2]

We present the language as an extension of the classical λ-calculus with a non-linear type system whose semantics is interpreted through monoidal string diagrams. It is defined with respect to a general symmetric monoidal category, whose concretisation in the category of finite-dimensional Hilbert spaces will also be covered enabling a novel paradigm of quantum programming in terms of abstractions denoting rotations of quantum states. Regarding quantum computation, the language implicitly embraces two fundamental aspects of quantum theory: complementarity and entanglement where (i) complementarity takes effect in the interaction between two abstraction bases and (ii) the non-linearity of the language enables implicit entanglement of variables.

Hopefully, the meaning behind this quite difficult language will become more clear in later chapters. In section 1.1 we present the background necessary mainly for chapter 2 and to a small extent to chapter 3. In chapter 2 we provide the full description of the language, presenting its syntax, type system and semantics. Chapter 3 covers the concretisation of the language presenting specific models for quantum computation as well as *spacetime computation* and introduce the notion of *orders of computation*. We digress slightly and discuss the philosophical backbone of this work in chapter 4. Finally, in chapter 5 we discuss the implications of the previous chapters as well as the work the two authors leave for the future.

1.1 Background

In this section, we will introduce the necessary background for coming to grips with the remainder of the thesis. The work mainly concerns the introduction of a language for monoidal categories making use of observable structures for allowing contraction and weakening in a category usually reserved for linear languages. While the primary intention is to introduce the language for a general monoidal category, the primary application of the language is presented as one for describing quantum processes. With this in mind, we will begin this section by describing a number of notions from quantum computation that we find applicable. Then we move on to the section describing the background in computer science that is necessary for the construction of the language. As the language is introduced for monoidal categories we will spend the next part giving a description of the categorical environment we will be working in. Finally, we bring the reader up to speed with the ZX-calculus, whose description of the interaction of observable structures provided much inspiration for the authors in starting this work.

1.1.1 Quantum computation

Here we will cover some of the prerequisites necessary for the understanding of quantum computation as we see it. Using the categorical description of quantum theory that is employed here throughout, it is not necessary to include some of the typical constructions used in describing computation – quantum circuits in particular – and it is further the view of the authors that this type of description of quantum computation is more limiting in its scope than it is helpful, if at that. We begin with describing what a Hilbert space is and then give the typical description of quantum states and effects as vectors in a Hilbert space that are acted upon by observables.

1.1.1.1 Hilbert spaces

A Hilbert space be seen as a generalisation of regular Euclidean space with some additional structure.[3] A vector in a Hilbert space can be of any arbitrary dimension – even infinite-dimensional. Dealing with these types of infinite-dimensional vector spaces is difficult and we will restrict ourselves here to only finite-dimensional Hilbert spaces. There is a notion of inner product, written $\langle u, v \rangle$ over elements $u, v \in \mathcal{H}$. Using this inner product, the notion of the length of an element, called the norm, can also be written as $\|v\| = \sqrt{\langle v, v \rangle}$. The inner product is

1. Positive definite, meaning that $\|v\|^2 \geq 0$ for all $v \in \mathcal{H}$.

2. Complete, meaning that any Cauchy sequence of elements of $\mathcal{H}$ converges.

An infinite series of elements $v_m, \ldots, v_n$ in $\mathcal{H}$ is a Cauchy sequence if there is some positive real number ε and an integer N for $m, n \leq N$ such that $\|v_m - v_n\| < \varepsilon$.

Formally, a Hilbert space is a space V defined over a field $\mathbb{K}$, along with the inner product, an operation $(+) : V \times V \to V$ for adding elements and an operation $(*) : \mathbb{K} \times V \to V$ for scaling elements, satisfying certain axioms. Any Hilbert space $\mathcal{H}$ over $\mathbb{K}$ is then isomorphic to $\mathbb{K}^n$ meaning that elements of $\mathbb{H}$ can always be written as n-tuples in $\mathbb{K}$, after some particular vector basis is chosen.

We can define a Hilbert space over any number system with appropriate structure. Of particular note are the four normed division algebras: the reals $\mathbb{R}$, complex numbers $\mathbb{C}$, quaternions $\mathbb{H}$ and octonions $\mathbb{O}$.[1] In quantum theory, this field is usually taken to be the complex numbers $\mathbb{C}$. As such, we will henceforth consider n-dimensional Hilbert spaces isomorphic to the complex numbers $\mathcal{H} \cong \mathbb{C}^n$. As this is the topic mainly covered here, we will restrict ourselves to this case for now. This brings about another property of the inner product: it is Hermitian, meaning that $\langle u, v \rangle = \langle v, u \rangle^*$, where $(-)^*$ is complex conjugation.

Vectors in a complex Hilbert space that differ only by a complex scalar $c \in \mathbb{C}$ are said to correspond to the same physical state.[5] Thus we find a set of equivalence classes of non-zero vectors $v, w \in \mathcal{H}$ (1.1).

$$v \sim w \quad \text{iff} \quad v = c\,w \quad \text{for some } c \in \mathbb{C} \tag{1.1}$$

where the vectors for which this relation applies are said to belong to the same *ray*. The reason for this distinction is quite fundamental. For a quantum state represented as a vector v of some dimension d in a Hilbert space $\mathcal{H}$, one usually makes the presupposition (one of many) for this state to be called *physically realisable* that all d entries of v, when normed and squared, add up to 1. I.e. v is normalised. This is equivalent to the statement that the probabilities of the different measurement outcomes of this quantum state add up to 1. Thus, for vectors belonging to the same ray one wishes to consider these in their normalised state which is any vector in this ray modulo the above equivalence relation. The Hilbert space modulo this equivalence relation is usually called the projective Hilbert space.[6] Note that normalised vectors in a ray can still differ by a unit complex number, that is a term of the form $e^{i\theta}$ where $|e^{i\theta}| = 1$ for $\theta \in [0, 2\pi)$. I.e. if v is a normalised vector in a ray then $\{e^{i\theta}v : 0 \leq \alpha < 2\pi\}$ is the set of all normalised vectors in that ray.

1.1.1.2 Quantum states, effects, and everything in between

The physical state of a quantum system is represented by a state vector $|\alpha\rangle$, called a *ket*[2] belonging to a Hilbert space $\mathcal{H}$. This vector is taken to encode all the relevant degrees of freedom of that system, meaning that any information that might be asked about the system in question could be ascertained from that state vector.[5]

State vectors that are of particular interest are those that are in an *eigenstate* of some observable A, represented by a Hermitian (self-adjoint, with real eigenval-

[1] See Solèr's theorem. A discussion can be found in [4].

[2] This funny notation is due to Dirac, called the bra-ket notation. It is introduced by the man himself in [7].

ues) operator. The operator itself is usually represented by a complex matrix of dimension n, i.e. of $\mathbb{C}^{n\times n}$. Examples of typical operators are the position operator x giving the position of a particle or the spin operator S_z giving the spin along the z-axis of a spin-$\frac{1}{2}$ particle. We will return to the different spin observables later. The eigenvectors of our observable A are said to span the space of possible states for $|\alpha\rangle$ to occupy. These eigenvectors (when normalised) form an orthonormal set of *basis* vectors for the particular state space $\mathcal{H}$ considered by that observable. State vectors $|\alpha\rangle$ that are of particular interest are those that can be written as a linear combination of the eigenvectors $|a_i\rangle$ of an observable A (1.2).

$$|\alpha\rangle = \sum_i c_i \, |a_i\rangle \tag{1.2}$$

where the sum is taken over the set of eigenvectors and c_i are *amplitudes* associated with each eigenvector, calculated by projection of the state vector upon each eigenvector, written as $c_i = \langle a_i|\alpha\rangle$ so that the expansion can in full be written (1.3).

$$|\alpha\rangle = \sum_i |a_i\rangle\langle a_i|\alpha\rangle \tag{1.3}$$

Here we have introduced the *bra* vector $\langle a_i|$, also called an effect. This is a vector that is said to live in the *dual* bra-space to the regular ket-space (which is just $\mathcal{H}$). Dual in the sense that, for each ket $|\alpha\rangle$ there exists a bra $\langle\alpha|$. If they are both normalised then their inner product will equal unity, i.e. $\langle\alpha|\alpha\rangle = 1$. Also for each set of basis vectors $\{|a_i\rangle\}$ in the ket space there exists a corresponding set of basis vectors $\{\langle a_j|\}$ for the bra space so that any inner product between these is written in terms of the Kronecker delta, i.e. $\langle a_j|a_i\rangle = \delta_{ij}$. An observable A can now be written in terms of its eigenvectors $\{|a_i\rangle\}$ and eigenvalues $\{\lambda_i\}$ (1.4).

$$A := \sum_i \lambda_i \, |a_i\rangle\langle a_i| \tag{1.4}$$

We will now consider the example of the spin-observable above. The measurable quantities of the spin direction of a spin-$\frac{1}{2}$ particle, as measured along the spatial axes x, y and z are represented by operators S_x, S_y and S_z. One of the first signs of behaviour particular to the quantum nature of particles was the discovery that particle spin is *quantised*. This was discovered in the famous Stern-Gerlach experiment performed in the early 1920s and is covered in detail in [5]. What quantised in this case means is that the basis vectors for a spin-observable take on a discrete distribution. In particular, the two so-called spin-up and spin-down states. The two spin-states for each spatial axis are represented by the eigenvectors of the spin operator matrices, defined by $S_i = \frac{\hbar}{2}\sigma_i$ for $i = \{x, y, z\}$ and σ_i the

Pauli-matrices (1.5).

$$\sigma_x := \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \qquad \sigma_y := \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix} \qquad \sigma_z := \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \tag{1.5}$$

With corresponding eigenvectors

$$|+\rangle := \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 1 \end{pmatrix} \qquad |-\rangle := \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ -1 \end{pmatrix} \tag{1.6}$$

$$|i^{+}\rangle := \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ i \end{pmatrix} \qquad |i^{-}\rangle := \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ -i \end{pmatrix} \tag{1.7}$$

$$|0\rangle := \begin{pmatrix} 1 \\ 0 \end{pmatrix} \qquad |1\rangle := \begin{pmatrix} 0 \\ 1 \end{pmatrix} \tag{1.8}$$

These eigenvectors represent the basis vectors of the state space defined by the spin observables. In quantum computation, where usually only the spin of quantum states is considered, these play a fundamental role. The two spin-states $\{|0\rangle, |1\rangle\}$ are usually taken as the so-called *computational basis* on which further computation is carried out. The basic unit of information in quantum computation is the qubit. The spin-observable is one way of implementing qubit but any two-level quantum system can be used. This naturally inherits the particular properties and interactions of quantum states. A particular example of this is that as opposed to classical bits which only exist in either state of 0 or 1, a qubit can occupy any superposition of the basis states. That is, the state of a qubit can be written as any linear combination of the basis vectors like

$$|\psi\rangle := \alpha |0\rangle + \beta |1\rangle \tag{1.9}$$

for $\alpha, \beta \in \mathbb{C}$, only that $\| |\psi\rangle \|^2 = |\alpha|^2 + |\beta|^2 = 1$. This has some interesting geometric interpretations which will be explored further in section 3.1.

The state of a single qubit is usually represented in terms of its projective Hilbert space, which in this case is the Bloch sphere. The state is represented as a point on this unit two-sphere S^2, with the three spin-observables corresponding to the three spatial axes of the space the Bloch sphere sits in.

This concludes our introductory remarks on the theory of quantum physics relevant for this thesis. While most of what is needed to fully comprehend this theory, if possible, has been left out, this is covered in greater detail in various textbooks on the subject, such as [5]. We will return later to a categorical description of quantum physics and computation as described in categorical quantum mechanics and the ZX-calculus.

1.1.2 The λ-calculus

The λ-calculus is a formal language and model of computation introduced by Alonzo Church in the 1930s. The syntax of the language, the set Λ of λ-terms, is defined inductively in figure 1.1. An introduction to the theory can be found in [8], while [9] is a more extensive resource.

$$\frac{}{x \in \Lambda}\ \textsc{var} \qquad \frac{M \in \Lambda}{\lambda x M \in \Lambda}\ \textsc{abs} \qquad \frac{M \in \Lambda \qquad N \in \Lambda}{MN \in \Lambda}\ \textsc{app}$$

Figure 1.1: The syntax of the λ-calculus.

The core of the language is the λ-abstraction, a term on the form $\lambda x M$, introduced by the rule (abs). Intuitively this represents a function that accepts a variable x and returns the term M, where x may or may not appear. An application, introduced by the (app) rule, represents a function being applied to an argument. The (var) rule introduces variables, which are assumed to range over a countably infinite set of names.

$$\begin{aligned} \mathbf{fv} &\in \Lambda \to \mathcal{P}(\mathrm{Var}) \\ \mathbf{fv}(x) &:= \{x\} \\ \mathbf{fv}(\lambda x M) &:= \mathbf{fv}(M) \setminus \{x\} \\ \mathbf{fv}(MN) &:= \mathbf{fv}(M) \cup \mathbf{fv}(N) \end{aligned} \tag{1.10}$$

We differentiate variables that are introduced by a λ-abstraction, from those which appear free. A *bound* variable x appears in a λ-term on the form $\lambda x M$, whereas a *free* variable appears without a λ-abstraction binding it. The meta-function **fv** returning the set of free variables in a term is defined in equation (1.10). Let the set of λ-terms which contain the set of free variables X be defined as $\Lambda^X := \{\, M \in \Lambda \mid \mathbf{fv}(M) = X \,\}$. The set of *closed* λ-terms, which contain no free variables, is then denoted $\Lambda^\emptyset$.

We also define the operation of *substitution*, denoted $M[x := N]$. This represents the replacing of every free occurrence of x in M with the term N. Its full definition for terms $N, P, Q \in \Lambda$ is in (1.11).

$$\begin{aligned} x[x := N] &\equiv N \\ y[x := N] &\equiv y \\ (\lambda x P)[x := N] &\equiv \lambda x P \\ (\lambda y P)[x := N] &\equiv \lambda y P[x := N] \qquad y \notin \mathbf{fv}(N) \\ (PQ)[x := N] &\equiv P[x := N] Q[x := N] \end{aligned} \tag{1.11}$$

Here, the notation $M \equiv N$ denotes syntactic equality, meaning α-equivalence, that we can make these terms exactly equal with a change of bound variable names.

1.1.2.1 Notions of reduction

Moving on from the definition of the syntax we have the subject of reduction. A *notion of reduction* is simply a relation on the set of λ-terms which fulfils rules in figure 1.2, called a compatible relation. Compatibility of a relation ensures that it holds for all subterms that appear in some expression.

$$\frac{(M, M') \in R}{(\lambda x M, \lambda x M') \in R} \qquad \frac{(M, M') \in R}{(MN, M'N) \in R} \qquad \frac{(N, N') \in R}{(MN, MN') \in R}$$

Figure 1.2: A compatible relation $R \in \Lambda \times \Lambda$.

Moreover, we have the following notational convention. A compatible relation $(M, N) \in R$:

- is a *one-step reduction relation* written $M \rightarrow_R N$.
- is a *reduction relation* written $M \twoheadrightarrow_R N$ iff it is also reflexive and transitive.
- is a *congruence relation* written $M \equiv_R N$ iff it is also an equivalence relation.

The two most common notions of reduction are called β-reduction and η-reduction. The relation of β-reduction is one of substitution.

$$\beta \quad := \quad \{\, ((\lambda x M)N, M[x := N]) \mid M, N \in \Lambda \,\} \tag{1.12}$$

It states that when a λ-abstraction $\lambda x M$ is applied to some term N, we substitute the variable x for N in M. This is simply the application of a function in the λ-calculus.

The relation of η-reduction represents extensional equality, the statement that two functions are equal if and only if they are equal in all arguments.

$$\eta \quad := \quad \{\, (\lambda x M x, M) \mid M, N \in \Lambda \text{ and } x \notin \mathbf{fv}(M) \,\} \tag{1.13}$$

It has one additional requirement of x not appearing in the term M. When this requirement is fulfilled the term $\lambda x M x$ represents a function that takes an argument and passes it to M, and thus it has the same meaning as M. For both of these cases, the actual relation is the least compatible relation which includes them.

1.1.3 Type theory

Here we will present a short exposition of another concept central to our thesis, type theory. Type theory defines and studies type systems, sets of formal rules that assign a *type* to every *term* in some language. When presenting a term M of some type A we write $M : A$, a simple example of which would be $0 : \mathbb{N}$. When presenting a set of rules for a type system, we usually define them as judgements dependent on a *context*. A context is a set of typed variables, from which we can draw conclusions in a judgement. For a context of typed variables Γ we write a typing judgement as $\Gamma \vdash M : A$. A typing rule is presented as a number of premises, judgements that we know to hold, and a conclusion we can draw from them. For example, if we are defining a set of typing rules for natural numbers we might have rules on the form (1.14).

$$\frac{\Gamma \vdash M : \mathbb{N} \qquad \Gamma \vdash N : \mathbb{N}}{\Gamma \vdash M + N : \mathbb{N}}\;\textsc{add} \qquad\qquad \frac{x : A \in \Gamma}{\Gamma \vdash x : A}\;\textsc{var} \tag{1.14}$$

The first rule states that, if we can deduce that both M and N are natural numbers, then their sum would also be a natural number. The second rule is an axiom (a rule without premises) stating simply that if a variable has a type in the context, we can deduce that it has that type. Using these rules, in the context $x : \mathbb{N}$ we have a valid type derivation (1.15).

$$\frac{\dfrac{x : \mathbb{N} \in \{x : \mathbb{N}\}}{x : \mathbb{N} \vdash x : \mathbb{N}}\;\textsc{var} \qquad \dfrac{x : \mathbb{N} \in \{x : \mathbb{N}\}}{x : \mathbb{N} \vdash x : \mathbb{N}}\;\textsc{var}}{x : \mathbb{N} \vdash x + x : \mathbb{N}}\;\textsc{add} \tag{1.15}$$

This derivation proves that, if we have some natural number x, the sum of x with itself is also a natural number.

Underlying the usual presentation of typing rules are rules which do not refer to any specific construct of a language, called *structural rules*. These rules present properties of the type system more generally. The usual structural rules that are employed are weakening, contraction, and exchange, presented in figure 1.3.

Weakening states that, if we can derive some judgement in a context Γ, we are also free to derive it with some superfluous variable. In a sense, contraction states the converse of weakening, that if we derive some judgement using two instances of a variable of the same type, we can derive it with one variable of that type instead. Finally, exchange states that the order of variables in the context of some judgement does not matter, and we can derive the same judgement with any ordering of variables. With the preliminaries of type theory defined we can move on to a specific type system for the λ-calculus. These rules are implicit in the pre-

$$\frac{\Gamma \vdash M : B}{\Gamma, x : A \vdash M : B}\ \text{W} \qquad \frac{\Gamma, x_1 : A, x_2 : A \vdash M : B}{\Gamma, x : A \vdash M[x_1 := x, x_2 := x] : B}\ \text{C}$$

$$\frac{\Gamma, x : A, y : B, \Delta \vdash M : C}{\Gamma, y : B, x : A, \Delta \vdash M : C}\ \text{X}$$

Figure 1.3: Structural rules.

sentation of typing rules we have presented so far, the context Γ being copied (by contraction) to each of the premises in the (ADD) rule, and the remaining variables in Γ, other than x, being discarded (by weakening) in the (VAR) rule.

1.1.3.1 The simply typed λ-calculus

One of the first type systems that is studied when getting familiar with type theory is the *simply typed λ-calculus* [9, Appendix A]. In this language we assign a type to terms of the untyped λ-calculus we outlined earlier. First we define the set of types as *Types* $A, B \ ::= \ C \mid A \to B$, where C is some set of predefined types. We need not worry about this set here, usually, it entails some set of constants of the language, for example the set of integers. We will instead focus on the second constructor, that of function types. We present the set of typing rules for the simply typed λ-calculus in figure 1.4.

$$\frac{x : A \in \Gamma}{\Gamma \vdash x : A}\ \text{VAR} \qquad \frac{\Gamma, x : A \vdash M : B}{\Gamma \vdash \lambda x M : A \to B}\ \to_i \qquad \frac{\Gamma \vdash M : A \to B \qquad \Gamma \vdash N : A}{\Gamma \vdash MN : B}\ \to_e$$

Figure 1.4: The typing rules of the simply typed λ-calculus.

Each of the typing rules corresponds to a specific syntactic construct in the λ-calculus, that is we have rules for (VAR), (ABS), and (APP) of figure 1.1 respectively. The first rule, as the variable rule before, states that if a typed variable appears in the context, we can deduce that the variable has that type. The second rule introduces the function type (wherefore we name it $\to_i$). It states that if we can deduce that, given that the variable x has the type A, that M has the type B, then the λ-abstraction $\lambda x M$ is a function that accepts an argument of type A and produces a term of type B. The final rule eliminates the function type, by applying a function to an argument. It states that, given a function M that accepts terms of type A and producing terms of type B, if we have some term N of type A, we can apply M to N and obtain a term of type B.

1.1.3.2 Linear type theory

We will now revisit the structural rules presented before, and specifically what happens when we omit contraction and weakening. The reason for omitting these rules comes from linear logic, a logic which is *resource aware*. What this means for us is that we want to be aware of exactly how many times a variable can be used in a typing derivation. If weakening is allowed we are free to discard variables that are left unused. Conversely, if contraction is allowed we are free to copy variables. To see this concretely we revisit (1.15). In this derivation the variable x is used in both of the premises of the (ADD) rule, which is allowed due to contraction. Thus, to ensure that variables are used *exactly once*, we have to omit both weakening and contraction from the structural rules. A type system of this form, called a *linear* type system was first introduced for the λ-calculus by Philip Wadler [10] for this exact purpose. We present the subset of the typing rules relevant to the syntax of figure 1.1, in figure 1.5.

$$\frac{}{x : A \vdash x : A}\ \textsc{var} \qquad \frac{\Gamma, x : A \vdash M : B}{\Gamma \vdash \lambda x M : A \multimap B}\ \multimap_i \qquad \frac{\Gamma \vdash M : A \multimap B \qquad \Delta \vdash N : A}{\Gamma, \Delta \vdash MN : B}\ \multimap_e$$

Figure 1.5: Typing rules for the linear λ-calculus.

Note the main difference here, compared to the rules presented in figure 1.4, that the setup of the contexts is different. For the (VAR) rule we require the context to consist of exactly the variable to be typed. In the ($\multimap_e$) rule we have two separate contexts. With the structural rules of weakening and contraction omitted, we can no longer write the typing rules in the same way as before, duplicating and discarding contexts freely. This set of rules then guarantees that only the judgements which use variables exactly once are derivable.

1.1.4 Category theory

We introduce here a subject that is central to the paradigm of quantum theory this thesis utilises, category theory. A comprehensive review of the topic from a perspective useful to this thesis can be found in [11, 12]. Category theory (unsurprisingly) studies categories, collections of things (called objects) and transformations between things (called morphisms).

Definition 1.1.1 (Category)**.** A category $\mathcal{C}$ is defined by:

- A collection of objects $\mathrm{Obj}(\mathcal{C})$.
- A collection of morphisms $\mathrm{Hom}(\mathcal{C})$. For each pair of objects $A, B \in \mathrm{Obj}(\mathcal{C})$ there exists a collection of morphisms $\mathrm{Hom}_{\mathcal{C}}(A, B)$, from A to B. We sometimes write $f : A \to B$ for a morphism $f \in \mathrm{Hom}_{\mathcal{C}}(A, B)$.

- The binary operation $\circ : \mathrm{Hom}_{\mathcal{C}}(B, C) \times \mathrm{Hom}_{\mathcal{C}}(A, B) \to \mathrm{Hom}_{\mathcal{C}}(A, C)$, composition of morphisms, with the following properties:

    1. (*Associativity*) For any morphisms $f : A \to B$, $g : B \to C$, and $h : C \to D$:

$$h \circ (g \circ f) = (h \circ g) \circ f \tag{1.16}$$

    2. (*Identity*) For any object $A \in \mathrm{Obj}(\mathcal{C})$ there exists a morphism $1_A : A \to A$, such that for any morphism $f : A \to B$:

$$f \circ 1_A = f = 1_B \circ f \tag{1.17}$$

Throughout the thesis we will introduce a number of categories with added structure at an abstract level. When talking about these categories with reference to some mathematical structure, that is, objects are inhabited by an instance of this structure and morphism by maps between them, they are called *concrete categories*.

A simple example of a concrete category is **Set**, the category of sets and functions on sets. Then,

- $\mathrm{Obj}(\mathbf{Set})$ is the collection of all sets.
- For any sets $X, Y \in \mathrm{Obj}(\mathbf{Set})$, $\mathrm{Hom}_{\mathbf{Set}}(X, Y)$ are the functions between the sets.
- The composition of morphisms is the usual composition of functions.
- The identity morphism is the function $1(x) := x$.

At times, we will need to display collective properties of categorical constructs. A common tool for this is the commutative diagram. For example, let $\mathcal{C}$ be some category, $A, B, C, D \in \mathrm{Obj}(\mathcal{C})$, $f : A \to B$, $g : C \to D$, $f' : C \to D$, and $g' : B \to D$. Then we can display this as the commutative diagram:

$$\begin{tikzcd} A \arrow[r, "f"] \arrow[d, "g"'] & B \arrow[d, "g'"] \\ C \arrow[r, "f'"'] & D \end{tikzcd}$$

To say that a diagram *commutes* is saying that for any directed path with the same start and end point we obtain the same result; in this case $g' \circ f = f' \circ g$.

1.1.4.1 Functors and natural isomorphisms

Here we will introduce some concepts central to category theory which might seem, at first glance, somewhat unjustified. We will introduce the concepts on a formal level here, to serve as a reference later. The first concept is the *functor*.

Definition 1.1.2 (Functor)**.** For categories $\mathcal{C}$ and $\mathcal{D}$, a functor $F : \mathcal{C} \to \mathcal{D}$ is defined by:

- An object map $F : \mathrm{Obj}(\mathcal{C}) \to \mathrm{Obj}(\mathcal{D})$, which maps objects $A \in \mathrm{Obj}(\mathcal{C})$ to objects $F(A) \in \mathrm{Obj}(\mathcal{D})$.

- A morphism map $F : \mathrm{Hom}(\mathcal{C}) \to \mathrm{Hom}(\mathcal{D})$, which for any objects $A, B \in \mathrm{Obj}(\mathcal{C})$ maps morphisms $f \in \mathrm{Hom}_{\mathcal{C}}(A, B)$ to morphisms $F(f) \in \mathrm{Hom}_{\mathcal{D}}(F(A), F(B))$.

Such that it preserves composition and identities, that is:

1. For any morphisms $f \in \mathrm{Hom}_{\mathcal{C}}(A, B)$ and $g \in \mathrm{Hom}_{\mathcal{C}}(B, C)$:

$$F(g \circ f) = F(g) \circ F(f) \tag{1.18}$$

2. For any object $A \in \mathrm{Obj}(\mathcal{C})$:

$$F(1_A) = 1_{F(A)} \tag{1.19}$$

Functors can also accept many arguments, a bifunctor $F : \mathcal{C}_1 \times \mathcal{C}_2 \to \mathcal{D}$ for example. The functor requirements are required to hold for each argument of the bifunctor, together with the requirement that $F(1_{A_1}, 1_{A_2}) = 1_{F(A_1,A_2)}$ and that the following diagram commutes:

(1.20)
$$\begin{tikzcd}
F(A_1 \times A_2) \arrow[r, "F(1_{A_1},f_2)"] \arrow[d, "F(f_1,1_{A_2})"'] & F(A_1 \times B_2) \arrow[d, "F(f_1,1_{B_2})"] \\
F(B_1 \times A_2) \arrow[r, "F(1_{B_1},f_2)"'] & F(B_1 \times B_2)
\end{tikzcd}$$

for objects $A_n, B_n \in \mathrm{Obj}(\mathcal{C}_n)$ and morphisms $f_n \in \mathrm{Hom}_{\mathcal{C}_n}(A_n, B_n)$.

Put more simply, a functor is a structure-preserving map between categories. Extending this even further we have natural transformations. In this sense, a functor is a morphism between categories, and a natural transformation is a morphism between functors.

Definition 1.1.3 (Natural transformation)**.** For categories $\mathcal{C}$ and $\mathcal{D}$, and functors $F, G : \mathcal{C} \to \mathcal{D}$, a natural transformation $\xi : F \Rightarrow G$ is defined by:

- An assignment $\xi_A : F(A) \to G(A)$ for every object $A \in \mathrm{Obj}(\mathcal{C})$ such that for any morphism $f \in \mathrm{Hom}_{\mathcal{C}}(A, B)$ the following diagram commutes:

$$\begin{tikzcd}
F(A) \arrow[r, "F(f)"] \arrow[d, "\xi_A"'] & F(B) \arrow[d, "\xi_B"] \\
G(A) \arrow[r, "G(f)"'] & G(B)
\end{tikzcd} \tag{1.21}$$

In a diagram, a natural transformation $\xi : F \Rightarrow G$ on categories $\mathcal{C}$ and $\mathcal{D}$ is written:

$$\begin{tikzcd}
\mathcal{C} \arrow[r, bend left, "F"{name=U}] \arrow[r, bend right, "G"'{name=D}] & \mathcal{D} \arrow[from=U, to=D, Rightarrow, "\xi"']
\end{tikzcd}$$

Finally, we have natural isomorphisms which add the requirement of invertibility to a natural transformation.

Definition 1.1.4 (Natural isomorphism)**.** For categories $\mathcal{C}$ and $\mathcal{D}$, and functors $F, G : \mathcal{C} \to \mathcal{D}$, a natural isomorphism $\xi : F \simeq G$ is a natural transformation such that:

- There exists, for each assignment $\xi_A : F(A) \to G(A)$, an inverse morphism $\xi_A^{-1} : G(A) \to F(A)$ such that:

$$\xi_A^{-1} \circ \xi_A = 1_{F(A)} \qquad \text{and} \qquad \xi_A \circ \xi_A^{-1} = 1_{G(A)} \tag{1.22}$$

This concludes our summary of the general definitions of category theory. This field is rich and varied, and we have left out many things for the sake of brevity. In the remainder of the thesis, we will use categories, functors, and natural isomorphisms to define the important concepts in categorical quantum mechanics and to connect our work to them.

1.1.4.2 Monoidal categories

The central category studied in categorical quantum mechanics is the *monoidal category*. This is a category extended with a monoidal product. Monoidal referring to that this product has a notion of associativity and units. We begin by defining the bare monoidal category and step-wise add new structure to finally get a description of the specific category we consider henceforth: a †-compact symmetric monoidal category. The definitions of these categories come with requirements on the structure that they add. As these requirements are not of importance for the concepts we define in the thesis, we will refer to [11] for the definitions of these. We will continually make comparisons to the concrete category of finite-dimensional Hilbert spaces, **FdHilb**, to make these definitions somewhat more clear in the context of quantum theory.

Definition 1.1.5 (Monoidal category)**.** A monoidal category is a category $(\mathcal{C}, \otimes, I)$ with a bifunctor $\otimes : \mathcal{C} \times \mathcal{C} \to \mathcal{C}$ and a distinguished object I such that:

1. For any objects $A, B, C \in \mathrm{Obj}(\mathcal{C})$, there is a natural isomorphism $\alpha_{A,B,C} : (A \otimes B) \otimes C \simeq A \otimes (B \otimes C)$ called the *associator*.

2. For any object $A \in \mathrm{Obj}(\mathcal{C})$, there are two natural isomorphisms called the *left unitor* $\lambda_A : I \otimes A \simeq A$ and *right unitor* $\rho_A : A \otimes I \simeq A$.

In the context of quantum theory, we have that the monoidal product is the tensor product on Hilbert spaces, and the unit is the Hilbert space $\mathbb{C}$ of complex scalars. If we further add a notion of swaps performed on products we get the following definition.

Definition 1.1.6 (Symmetric monoidal category)**.** A symmetric monoidal category is a monoidal category $\mathcal{C}$, with the addition of a natural isomorphism $\sigma_{A,B} : A \otimes B \simeq B \otimes A$ such that:

1. For any objects $A, B \in \mathrm{Obj}(\mathcal{C})$, we have that $\sigma_{A,B} \circ \sigma_{B,A} = 1_{A \otimes B}$.

Looking again at the category of finite dimensional Hilbert spaces we have that the usual swap map $\sigma := |00\rangle\langle 00| + |01\rangle\langle 10| + |10\rangle\langle 01| + |11\rangle\langle 11|$ fulfils this condition, and thus is also a symmetric monoidal category (SMC).

Definition 1.1.7 (†-symmetric monoidal category)**.** A †-symmetric monoidal category is a symmetric monoidal category $\mathcal{C}$ with the addition of a functor $\dagger : \mathcal{C}^{op} \to \mathcal{C}$, where $\mathcal{C}^{op}$ is the category with all morphisms reversed, such that:

1. For any object $A \in \mathrm{Obj}(\mathcal{C})$, we have $A^\dagger = A$.

2. For any objects $A, B \in \mathrm{Obj}(\mathcal{C})$, and any morphism $f \in \mathrm{Hom}_{\mathcal{C}}(A, B)$ we have that $f^\dagger \in \mathrm{Hom}_{\mathcal{C}}(B, A)$, preserving the monoidal structure of morphisms:

$$(f \circ g)^\dagger = g^\dagger \circ f^\dagger \qquad (f \otimes g)^\dagger = f^\dagger \otimes g^\dagger \qquad 1_A^\dagger = 1_A \qquad f^{\dagger\dagger} = f \tag{1.23}$$

In **FdHilb**, the †-functor appears as the adjoint of maps between Hilbert spaces satisfying the above conditions, making **FdHilb** also a †-SMC. Vectors $\psi, \phi : \mathbb{C} \to \mathcal{H}$ also appear as morphisms, so that together with the adjoint one gets a categorical notion of the inner product of vectors in a Hilbert space. I.e.

$$\langle \phi | \psi \rangle = \phi^\dagger \circ \psi : \mathbb{C} \to \mathbb{C} \tag{1.24}$$

Vectors like those above of the form $I \to \mathcal{H}$ are referred to as states or points and those of the type $\mathcal{H} \to I$ are referred to as effects.

Definition 1.1.8 (†-compact category)**.** A †-compact category $\mathcal{C}$ is a †-symmetric monoidal category which for every object $A \in \mathrm{Obj}(\mathcal{C})$ includes:

1. A dual object $A^* \in \mathrm{Obj}(\mathcal{C})$
2. A pair of morphisms $\eta_A : I \to A \otimes A^*$ and $\eta_A^\dagger : A^* \otimes A \to I$, respectively called the unit and counit.

We found in 1.1.1.1 the notion of a bra-space, being the space dual to the usual Hilbert space of kets. Thus, for **FdHilb** we have for every object $\mathcal{H}$, a dual object $\mathcal{H}^*$ corresponding to a bra-space. Further, we can form the unit η as the maximally entangled quantum state $\frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$, with $\eta^\dagger$ formed using the adjoint of this expression. We thus find that all the conditions above are fulfilled, meaning that **FdHilb** is a †-compact category (fully, a †-compact symmetric monoidal category).

1.1.4.3 Internal language for an SMC

The internal logic for symmetric monoidal closed categories is the multiplicative fragment of intuitionistic linear logic [12]. Mackie et. al. produced a language with a type system corresponding to this logic in [1]. This language is a linear λ-calculus with a tuple construction, corresponding to the monoidal functor, and a unit, corresponding to the monoidal unit. The syntax of this language, the set Λ_{SMC}, is presented in figure 1.6.

$$\frac{}{\star \in \Lambda_{\mathrm{SMC}}} \qquad \frac{}{x \in \Lambda_{\mathrm{SMC}}} \qquad \frac{M \in \Lambda_{\mathrm{SMC}}}{\lambda x M \in \Lambda_{\mathrm{SMC}}} \qquad \frac{M \in \Lambda_{\mathrm{SMC}} \quad N \in \Lambda_{\mathrm{SMC}}}{MN \in \Lambda_{\mathrm{SMC}}}$$

$$\frac{M \in \Lambda_{\mathrm{SMC}} \quad N \in \Lambda_{\mathrm{SMC}}}{\langle M, N\rangle \in \Lambda_{\mathrm{SMC}}} \qquad \frac{M \in \Lambda_{\mathrm{SMC}} \quad N \in \Lambda_{\mathrm{SMC}}}{\text{let } \langle x, y\rangle = M \text{ in } N \in \Lambda_{\mathrm{SMC}}}$$

$$\frac{M \in \Lambda_{\mathrm{SMC}} \quad N \in \Lambda_{\mathrm{SMC}}}{M \mathbin{;} N \in \Lambda_{\mathrm{SMC}}}$$

Figure 1.6: The syntax of the internal language of SMCs.

This is an extension of the syntax presented in figure 1.1. Other than the tuple $\langle M, N\rangle$ and unit $\star$, we have their respective eliminations as let-expressions. We have chosen to present the let-elimination of the unit let $\star = M$ in N as $M \mathbin{;} N$, to reduce clutter. It is the expression that discards the unit M.

The following types are defined:

- For types A and B, $A \multimap B$ is the linear function type.
- For types A and B, $A \otimes B$ is the tensor product type.

- The unit type I, which is the unit of the tensor product type.

Concretely given by the grammar $A, B ::= I \mid A \otimes B \mid A \multimap B$. Contexts are defined in the same way as in the simply typed λ-calculus, $\Gamma, \Delta ::= \emptyset \mid \Gamma, x : A$.

The language is a linear one, so the structural rules of contraction and weakening are omitted, leaving only exchange. The type system extends on the one given for the linear λ-calculus in figure 1.5, adding rules for the additional constructs. These rules are presented in figure 1.7.

$$\frac{}{x : A \vdash x : A}\,\textsc{ax} \qquad \frac{\Gamma, x : A, y : B \vdash M : C}{\Gamma, y : B, x : A \vdash M : C}\,\textsc{ex} \qquad \frac{}{\vdash \star : I}\,I_i$$

$$\frac{\Gamma \vdash M : I \qquad \Delta \vdash N : A}{\Gamma, \Delta \vdash M \,⨾\, N : A}\,I_e \qquad \frac{\Gamma \vdash M : A \qquad \Delta \vdash N : B}{\Gamma, \Delta \vdash \langle M, N \rangle : A \otimes B}\,\otimes_i$$

$$\frac{\Gamma \vdash M : A \otimes B \qquad \Delta, x : A, y : B \vdash N : C}{\Gamma, \Delta \vdash \text{let } \langle x, y \rangle = M \text{ in } N : C}\,\otimes_e \qquad \frac{\Gamma, x : A \vdash M : B}{\Gamma \vdash \lambda x M : A \multimap B}\,\multimap_i$$

$$\frac{\Gamma \vdash M : A \multimap B \qquad \Delta \vdash N : A}{\Gamma, \Delta \vdash MN : B}\,\multimap_e$$

Figure 1.7: Typing rules for the internal language of SMCs.

The paper also provides the language with notions of reduction, both β-reduction and η-reduction as we discussed them earlier, on the added constructs. These reduction relations are presented in figure 1.8.

$$\begin{aligned} \star \,⨾\, M &\to_\beta M & M \,⨾\, \star &\to_\eta M \\ \text{let } \langle x, y \rangle = \langle M, N \rangle \text{ in } P &\to_\beta P[x := M, y := N] & \text{let } \langle x, y \rangle = M \text{ in } \langle x, y \rangle &\to_\eta M \\ (\lambda x M)N &\to_\beta M[x := N] & \lambda x\, Mx &\to_\eta M \end{aligned}$$

Figure 1.8: Notions of reduction on the internal language of SMCs.

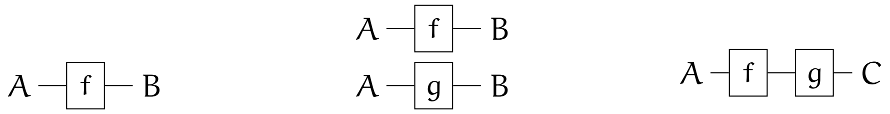
This language is called the internal language of SMCs since we can use it to describe the morphisms of the category[3]. This is done by constructing the category where the objects are the types of the language. A morphism $f : A \to B$ in the category is an equivalence class of pairs $(x : A, M : B)$ of terms M which get assigned by the type system the type B in the context of $x : A$, that is $x : A \vdash M : B$. A more detailed description of this process of constructing an SMC from the theory of Λ_{SMC} can be found in the original paper [1].

[3] The general definition of an internal language or internal logic is more involved than this, but this is the way in which it is relevant for the work here.

1.1.4.4 String diagrams

We will shortly consider the use of string diagrams in computer science and physics for describing processes. When linearly writing down a description of a process, the equations can easily become complicated and difficult to work with. Penrose [13] introduced a diagrammatic calculus for representing and manipulating abstract tensors, involving operations like composition, addition and contraction of indices, etc. He showed that the planar representation of large tensorial calculations gave a simpler pictorial interpretation.

A monoidal category $\mathcal{C}$ has a process $f : A \to B$, that is a morphism $f \in \mathrm{Hom}_{\mathcal{C}}(A, B)$ with input and output objects A and B, respectively, for $A, B \in \mathrm{Obj}(\mathcal{C})$. This morphism is represented in a string diagram as in figure 1.9a.



(a) A single morphism. **(b)** Parallel. **(c)** Sequential.

Figure 1.9: Representation of a single morphism, parallel and sequential composition of morphisms in a string diagram.

Two morphisms $f : A \to B$, $g : A \to B$ can be placed in parallel using the monoidal product as $f \otimes g : (A \to B) \otimes (A \to B)$, likewise $f \otimes g : (A \otimes A) \to (B \otimes B)$, represented in figure 1.9b. The sequential composition of two morphisms $f : A \to B$, $g : B \to C$, written as $g \circ f : A \to C$ can also be represented simply in a string diagram, shown in figure 1.9c.

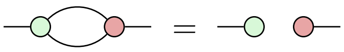
One important aspect of string diagrams, noted in [14], is that topologically equivalent diagrams denote the same morphism. That is, if one diagram can be arbitrarily deformed into another then they are equivalent. Thus we need only consider how the components of a diagram are connected, coined in the shorthand 'only topology matters'. [2]

We hope this short introduction to string diagrams has been enough for the interpretation of diagrams presented later in the thesis. The particular form of string diagrams that we employ is that used in the ZX-calculus introduced below, where we hope that greater intuition can be built around string diagrams than our introduction in this short section.

1.1.5 The ZX-calculus

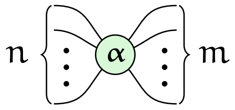
Moving on from general string diagrams, abstract category theory and quantum physics, we now wish to introduce a particular synthesis of these varied fields, the ZX-calculus [2]. This is the seminal language for describing quantum processes in a categorical setting and was a leading inspiration to both authors in the work presented here. The ZX-calculus is a graphical language for describing quantum processes, although much more than this. It provides distinct and intuitive means of reasoning about quantum processes through categorical string diagrams, where the usual abstract description of the manipulation of algebras of quantum observables, and their interactions are given a simple interpretation in the building blocks and rules of the calculus. We include an example of a particular ZX diagram equation (1.25).

(1.25)



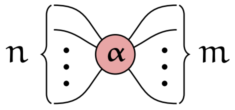
The example shows the diagrammatic building blocks of the ZX-calculus, red and green nodes (otherwise known as *spiders*), connected by a number of wires. As we are in the categorical setting, diagrams represent morphisms in a monoidal category and as the concrete category usually is taken to be **FdHilb**, these morphisms denote linear maps on finite-dimensional Hilbert spaces. The language centres around these red and green *spiders*, the general versions of which are shown in (1.26) and (1.27).

(1.26)



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

(1.27)



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

The colour of a spider denotes the *observable structure* from which it is constructed (this term will be formally defined below). As noted in section 1.1.1.2, every observable comes with an orthonormal set of basis vectors, and it is from these that the spiders are constructed. Specifically, the spiders here describe rotation of a state about the axes Z and X of the Bloch-sphere by a phase $\alpha \in [0, 2\pi)$. In figure 1.10 we have drawn these axes on the Bloch sphere, as well as the points which lie on them, and the rotations defined by them. The Z axis is drawn in green, and the X axis as red, like their spider counterparts.

A special node for the Hadamard gate is also defined, as a yellow box. This

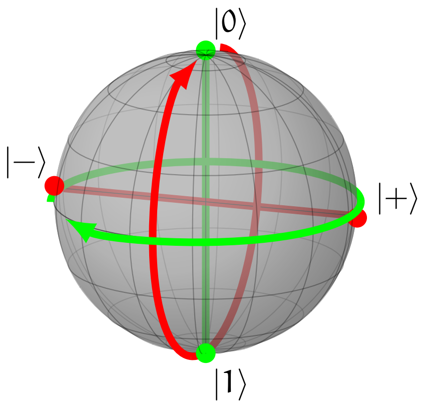
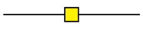
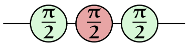


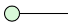
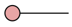
Figure 1.10: The Bloch-sphere with axes of rotation.

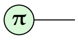
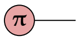
gate can be defined by decomposing it into angles of rotation through the already defined spiders, producing its definition in (1.28).

(1.28)  $:=$  $= |+\rangle\langle 0| + |-\rangle\langle 1| = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$

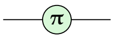
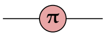
With this, we have all of the elements of the language defined. Comparing this description of quantum processes we have presented before, with the distinction of quantum states, effects, and ”everything in-between”, it becomes very clear in the ZX-calculus. A state is simply a diagram with no input wires, an effect has no outputs, and everything else is in-between. The basis vectors of each of the observables are recovered by states of the other spider, shown in (1.29).

(1.29)

 $= |0\rangle + |1\rangle = |+\rangle$  $= |+\rangle + |-\rangle = |0\rangle$

 $= |0\rangle - |1\rangle = |-\rangle$  $= |+\rangle - |-\rangle = |1\rangle$

The actual observables, as witnessed by the Pauli-Z and X matrices, are simply rotations about their respective axes by a π phase. As such, in the ZX-calculus, these are unary spiders with phase arguments of π, as shown in (1.30).

(1.30)  $= |0\rangle\langle 0| - |1\rangle\langle 1| = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$  $= |+\rangle\langle +| - |-\rangle\langle -| = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$

With the definitions of the elements of ZX-diagrams, we move on to the equational rules of the language. This ruleset originates from the algebraic construction of the two *complementary* observables, which we will cover further in section 1.1.5.1. These are defined over equality up to a global non-zero scalar. This means that, for ZX-diagrams $\mathcal{D}_1$ and $\mathcal{D}_2$, the statement $\mathcal{D}_1 = \mathcal{D}_2$ states that their respective interpretations can be made equal by some complex scalar $\lambda \in \mathbb{C}$, that is $[\![\mathcal{D}_1]\!] = \lambda[\![\mathcal{D}_2]\!]$, where $\lambda \neq 0$. We show the full set of rules of the equational theory of the ZX-calculus in figure 1.11.

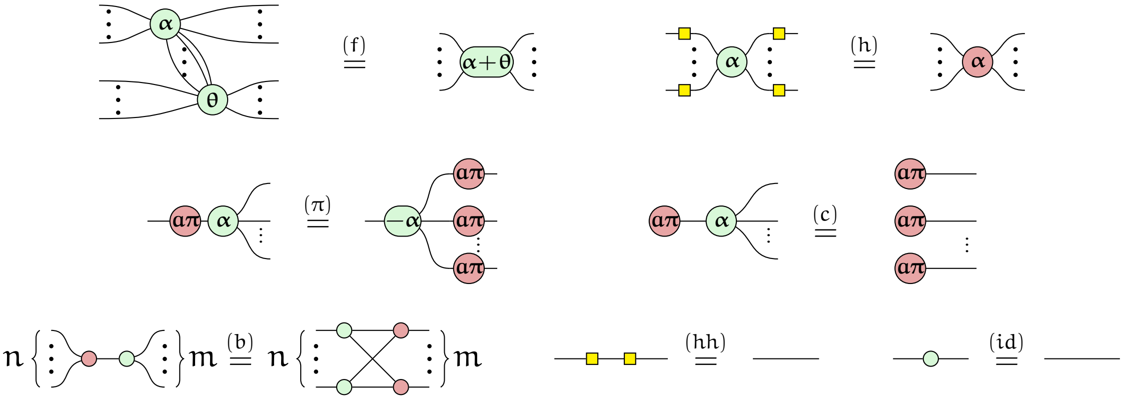


Figure 1.11: The equational theory of the ZX-calculus.

In this figure, all the rules hold with the colours of the spiders reversed, and two wires with vertical dots between them denoting that there can be zero or more of wires. Though all of these rules are of importance when using the ZX-calculus for simplifying quantum processes, we wish to highlight only a subset of them for reference in the remainder of the thesis.

The fusion rule (f) states that spiders that are connected by one or more wires can be *fused* together. This produces a single spider with the previous phases summed. The basis state copy rule (c) states that the basis states of one spider are perfectly copied by the dual spider, no matter the phase of the copying spider. The colour change rule (h) states a well-known fact about the Hadamard transform, that it switches the perspective between Z and X. What this means for the ZX-calculus is that when a spider of one colour is surrounded by Hadamard boxes, then we can simply remove the boxes and change the colour of the spider. Lastly, the identity removal rule (id) remarks that a unary spider (one input, one output) of any colour, with zero phase, is equal to the identity wire.

This is the extent to which we intend to cover the ZX-calculus as a language on its own in this thesis, though we will discuss our language in connection to it frequently. A beginner-friendly introduction to the subject can be found in [15], while the foundational paper on the subject [2] provides its full definition.

1.1.5.1 Observable structures

Having given a top-level description of the ZX-calculus, we will now move on to the categorical and algebraic machinery that lies at the core of its seeming simplicity. Here the central notion is that of *observable structures*. Its name suggests a correspondence with the quantum observables discussed previously and while this specific application will be encountered below, we move on with the general algebraic definition.

Before this, we need some background in abstract algebra, specifically the concept of *monoids*. A monoid is a triple $(A, \star \in A \times A \to A, e \in A)$ consisting of some set A, a binary operator $\star$, and a special element e. On the binary operation in this triple, we place the requirements of associativity and that e is its left and right unit, as shown for $a, b, c \in A$ in (1.31).

$$a{\star}(b{\star}c) \;=\; (a{\star}b){\star}c \qquad\qquad e{\star}a \;=\; a \;=\; a{\star}e \tag{1.31}$$

Further, for purposes which will become clear later, a *commutative monoid* is a monoid $(A, \star, e)$ such that for $a, b \in A$ we have $a{\star}b = b{\star}a$.

These concepts have a natural correspondence in the categorical setting, defining these concepts as internal to a monoidal category $\mathcal{C}$. Firstly, an internal commutative monoid is defined as the triple (A, m, e) with $A \in \mathrm{Obj}(\mathcal{C})$, $m : A \otimes A \to A$ called the multiplication and $e : I \to A$ its unit, placing similar restrictions as in the algebraic case, shown below.

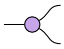
$$m \circ (m \otimes 1_A) \;=\; m \circ (1_A \otimes m) \tag{1.32}$$
$$m \circ (e \otimes 1_A) \circ \lambda_A^{-1} \;= 1_A \;=\; m \circ (1_A \otimes e) \circ \rho_A^{-1} \tag{1.33}$$
$$m \circ \sigma_{A,A} \;=\; m \tag{1.34}$$

We can also define an *internal cocommutative comonoid* here, which is the internal commutative monoid but with the arrows of its morphisms reversed. It is a triple (A, δ, ε), with A again an object, $\delta : A \to A \otimes A$ as the comultiplication, and $\varepsilon : A \to I$ as the counit, along with its restrictions.

$$(\delta \otimes 1_A) \circ \delta \;=\; (1_A \otimes \delta) \circ \delta \tag{1.35}$$
$$\lambda_A \circ (\varepsilon \otimes 1_A) \circ \delta \;= 1_A \;=\; \rho_A \circ (1_A \otimes \varepsilon) \circ \delta \tag{1.36}$$
$$\sigma_{A,A} \circ \delta \;=\; \delta \tag{1.37}$$

When working in a category equipped with the †-functor, we can identify the internal comonoid with the adjoint of the internal monoid such that $e = \varepsilon^\dagger$ and $m = \delta^\dagger$ and vice versa. With these definitions, we are ready to present observable structures.

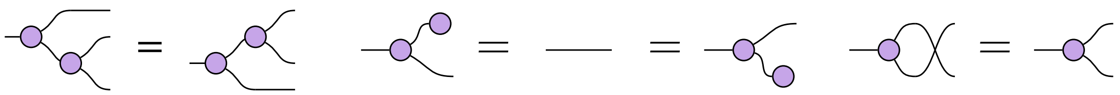
Definition 1.1.9 (Observable structure)**.** An observable structure is a triple:

$$\left(A\,,\ \delta = \right.$$  $$:\ A \to A \otimes A\,,\ \varepsilon =$$ $$\left.:\ A \to I\right)$$

With string diagram representations as above, such that it:

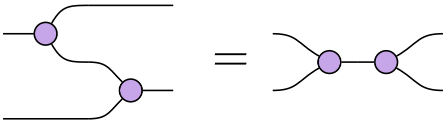
(i) is a cocommutative comonoid (equations (1.35)-(1.37)*)*

Coassociativity Counital elimination Cocommutativity

(1.38) 

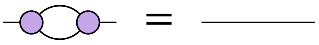
(ii) satisfies the Frobenius law, i.e. $(1_A \otimes \delta^\dagger) \circ (\delta \otimes 1_A) = \delta \circ \delta^\dagger$

Frobenius law

(1.39) 

(iii) is special, i.e. $\delta^\dagger \circ \delta = 1_A$

Speciality

(1.40) 

Where the graphical representation of the internal commutative monoid δ, is the one for the comonoid reflected horizontally. We can further define a new type of algebraic structure by combining the monoid and comonoid into one, defining the quintuple $(A, m : A \otimes A \to A, e : I \to A, \delta : A \to A \otimes A, \varepsilon : A \to I)$. Any structure of this form satisfying the Frobenius law (1.39) is called a *Frobenius algebra*. Thus, for the †-SMC of **FdHilb**, we can identify observable structures with †-special commutative Frobenius algebras. The final condition of speciality in the definition above corresponds exactly with the normalisation condition of the basis vectors of the observable structure. [16]

As noted in [16], observable structures in **FdHilb** correspond precisely to an orthonormal basis for a Hilbert space $\mathcal{H}$. We will attempt to explain this shortly. The comultiplication δ and the counit ε of an observable structure can be associated with operations that respectively uniformly copy and erase a set of orthogonal basis vectors. We show the action of these operators in equations (1.41) and (1.42), written in terms of a set of basis vectors $\{|\phi_i\rangle\} \in \mathcal{H}$.

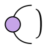
(1.41) $$\delta : A \to A \otimes A := |\phi_i\rangle \mapsto |\phi_i\rangle \otimes |\phi_i\rangle$$

(1.42) $$\varepsilon : A \to \mathbb{C} := |\phi_i\rangle \mapsto 1$$

Looking at the copying function δ. We can see that it encodes information about the basis associated with the observable structure, copying a state perfectly only when that state is a basis vector. For this reason, the comultiplication is said to uniquely capture the basis vectors of the observable structure to which it belongs. Thus in **FdHilb**, we can associate with every observable structure an orthogonal basis.

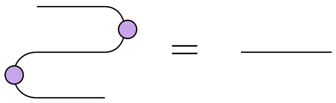
An observable structure induces what is called a compact structure. If permitting the †-operator, the dual pair can also be derived. We present its full definition, with this dual pair below.

Definition 1.1.10 (†-compact structure)**.** A †-compact structure is a tuple:

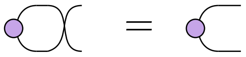
$$(A, \eta : I \to A \otimes A = $$  $$)$$

Such that:

(1.43) $$\lambda_A \circ (\eta^\dagger \otimes 1_A) \circ (1_A \otimes \eta) \circ \rho_A^\dagger = 1_A$$

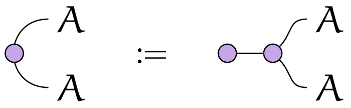


(1.44) $$\sigma_{A,A} \circ \eta = \eta$$



Being induced by the definition of its observable structure, the compact structure can be defined as in (1.45).

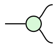
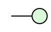
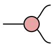
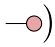
(1.45) $$\eta \quad := \quad \delta \circ \epsilon^\dagger$$



Note that this induced compact structure is internal, i.e. not the same as $\eta_A : I \to A^* \otimes A$ as introduced for †-compact categories. Though that their name and symbol are the same is not a coincidence. For the canonical example of the ZX-calculus, the induced †-compact structures of the Z and X observable structures both have the same interpretation in Hilbert space as η_A, given the same object A. They are also subject to the same requirements, the conditions being placed on the †-compact structure above being precisely the ones in a †-compact category.

So far, we have only considered the single observable structure. We showed that these coincided firstly with a complete orthonormal set of basis vectors in a Hilbert space, as well as coinciding with a special commutative †-Frobenius algebra [2]. We will now consider the interaction of multiple of these observable structures, covering complementarity of observable structures and a number of the consequences that follow from complementarity. We begin with defining what is meant by complementary observable structures.

Definition 1.1.11. (*Complementarity*) Two observable structures

(1.46) $\mathbf{Z} := \big(A,\ \delta_Z =$ $,\ \epsilon_Z =$ $\big)$ $\qquad$ $\mathbf{X} := \big(A,\ \delta_X =$ $,\ \epsilon_X =$ $\big)$

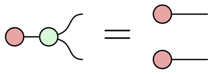
are complementary if there are morphisms

(i) $z : I \to A$ that is copied by $\mathbf{X}$ and unbiased for $\mathbf{Z}$, and

(ii) $x : I \to A$ that is copied by $\mathbf{Z}$ and unbiased for $\mathbf{X}$.

As mentioned in section 1.1.4.2, morphisms of the form of z, x generate points. In the context of observable structures, points are generated relative to these. The above definition thus tells us that $\mathbf{Z}$ and $\mathbf{X}$ are complementary if the unit $e_Z = \varepsilon_Z^\dagger$ of $\mathbf{Z}$ is copied by the comultiplication of $\mathbf{X}$ and vice versa (1.47); and further that the unit of $\mathbf{Z}$ is unbiased to $\mathbf{Z}$ and vice versa. We will however leave the discussion of unbiased points for later.

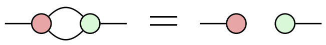
(1.47) $$\delta_Z \circ \varepsilon_X^\dagger = \varepsilon_X^\dagger \otimes \varepsilon_X^\dagger$$



Returning shortly to the concrete case of **FdHilb**. We saw previously that the comultiplication uniquely captures the basis vectors of its observable structure. That is, it produces two copies when acting on a basis vector. This is the concrete version of the main statement presented above. Two observable structures are thus complementary if the unit of the one generates a basis vector from the basis of the other.

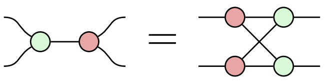
We wish to present two further algebraic relations that follow from complementarity. However, as these will not be as relevant for the remaining thesis, we will only mention them shortly. The first of these being the *Hopf law*, shown in (1.48). This relation further relies on coinciding compact structures, i.e, $\eta_Z = \eta_X$. For coinciding compact structures, it can be shown [2] that the complementarity of two observable structures follows from the Hopf law.

(1.48) $$\delta_Z^\dagger \circ \delta_X = \varepsilon_Z \circ \varepsilon_X^\dagger$$



A stronger form of complementarity [4] that applies to complementary observable structures is the *Bialgebra rule* shown in (1.49).

(1.49) $$\delta_X \circ \delta_Z^\dagger = (\delta_Z^\dagger \otimes \delta_Z^\dagger) \circ (1_A \otimes \sigma_{A,A} \otimes 1_A) \circ (\delta_X \otimes \delta_X)$$



[4] In the sense that a bialgebra of observables inherently satisfies the Hopf law. That is, any bialgebra of observable structures satisfying the bialgebra rule can be shown to satisfy the Hopf law.

1.1.5.2 Unbiased points and phase groups

We wish to introduce what together with the topics introduced above will lead to a full derivation of relevant parts of the ZX-calculus, unbiased points and the phase group. Both of these definitions come from the need to describe quantum computation in terms of Hilbert spaces, and as such, their definition in the categorical language may not seem entirely clear. We will try our best to refer back to Hilbert spaces where we can, to give the reader some intuition of where these concepts stem from.

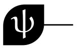
Before moving on, we hope to provide some intuition regarding unbiasedness in the concrete case. For any vector basis $\{|\phi_i\rangle\}$, a vector given as a linear combination of these such as $|\psi\rangle := \sum_i c_i |\phi_i\rangle$, for some set of scalars $\{c_i\}$ is unbiased relative to this basis if

(1.50) $$|\langle v_i|\psi\rangle| = |\langle v_j|\psi\rangle|$$

for all i, j. This projection of ψ onto the different basis vectors can be interpreted as a post-selection. From this, (1.50) tells us that the amplitudes of ψ are the same relative to every basis vector of that basis. It is unbiased relative to that basis. Physically this means that every outcome of a measurement of this state in that observable is equally likely.

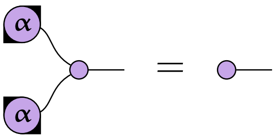
Returning to the general case of an observable structure (A, δ, ε) in a †-SMC $\mathcal{C}$. We refer to the set of morphisms in $\mathrm{Hom}_{\mathcal{C}}(I, A)$ as the set of points (corresponding to states in the concretisation) of this category. Graphically, we denote an arbitrary point $\psi : I \to A$ as a black node with edges, as in (1.51).

(1.51)



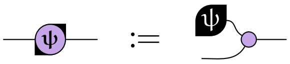
These edges denote that this point may be sensitive to conjugation, that ψ^* may not necessarily equal ψ. Extending this notation to unbiasedness, we denote an unbiased point α relative to some observable structure as a node with edges, of the same colour as the observable structure. To define unbiasedness in the categorical language we have the following. A point α is unbiased relative to an observable structure (A, δ, ε) if $\delta^\dagger \circ (\alpha^* \otimes \alpha) = \varepsilon^\dagger$, up to normalisation. Here, $(-)^*$ represents conjugation in the sense of definition 6.17 in [2]. The statement is shown as a diagram in (1.52).

(1.52)



Now to define the phase group. The intuition of this construction is most clear when we refer back to the ZX-calculus, where the phase groups of each of the observables are the rotation operators around their axes on the Bloch-sphere. Here, we wish to define this concept in an abstract categorical setting. We begin by defining a multiplication on points $\Lambda(\psi)$ in (1.53).

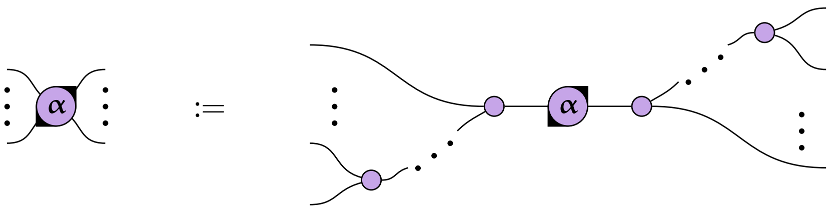
$$\Lambda(\psi) := \delta^{\dagger} \circ (\psi \otimes 1_A) \tag{1.53}$$



This is then the multiplication of some observable structure, with some point ψ. The relationship between this construction and unbiased points is found in proposition 7.18 of [2], that a point α is unbiased to some observable structure if the multiplication on points $\Lambda(\alpha)$ of the same observable structure is unitary. From this, we get the fact that the multiplication on unbiased points of some observable structure is always unitary, and this is the phase group.

With these definitions, the observable structures, unbiased points, and the phase group, we can finally recover the spider of the ZX-calculus. We show its definition in (1.54).

(1.54)



Simply put, a spider is constructed from its observable structure by a number of stacked multiplications as inputs, followed by a phase group with some unbiased point, and a number of stacked comultiplications on its output. The specific conditions on observable structures outlined earlier each implies important properties of the spider, which we will not cover here. These properties, together with rules from complementarity, and rules specific to the Hadamard transform, make up the elements and rules of the ZX-calculus. Much has been left out here, as we only cover what's necessary for the comprehension of the thesis, but it is safe to say tf G(; B) a A Γ f a terhat the brilliant simplicity of the ZX-calculus is backed by an incredible formal categorical machinery. We hope that we have made it some justice.

2

Theory

At dawn my lover comes to me
And tells me of her dreams
With no attempts to shovel the glimpse
Into the ditch of what each one means
At times I think there are no words
But these to tell what's true
And there are no truths outside the Gates of Eden

B. Dylan

WE introduce in this chapter the definition of the ζ-calculus. This language is an extension of the internal language of symmetric monoidal closed categories Λ_{SMC} with constructs related to ones presented for observable structures. The intention of this extension is twofold.

Firstly, the main application of the ζ-calculus should be seen as a functional quantum programming language. As such, we rely on the categorical framework introduced by Coecke and Duncan [2], of complementary quantum observables for the same concretisation as the ZX-calculus. This is achieved by making the underlying category concrete in finite-dimensional Hilbert spaces, **FdHilb**, and by employing the observable structures Z and X. Their respective phase groups, rotation about the axes of the Bloch sphere, serve as the basic operations to be applied to quantum states in the language. However, the framework is more general than the specific instances of the Z and X spin observables, and as such, so is the ζ-calculus. We define the language in an abstract way here, referring to categorical and algebraic structures, which we then make concrete for quantum programming in the next chapter.

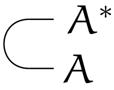
Secondly, in contrast to Λ_{SMC}, we intend the language to be non-linear. This direction is understandably contentious in the quantum programming language community, due to the no-cloning theorem [17]. This theorem states that it is impossible to create independent and identical copies of quantum states. Because of this fact, many quantum programming languages have employed linear type systems to disallow the duplication and discarding of terms that represent quantum states. Examples of these languages include Selinger and Valiron's quantum λ-calculus [18], QWIRE [19], and Quipper [20]. There is, however, another way to interpret duplication in quantum languages called *sharing*. This operation acts on an arbitrary quantum state in some basis $\{|b_0\rangle, |b_1\rangle\}$ as described in (2.1).

$$\alpha |b_0\rangle + \beta |b_1\rangle \mapsto \alpha |b_0 b_0\rangle + \beta |b_1 b_1\rangle \tag{2.1}$$

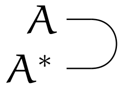
The "copies" produced by this operation are not independent of each other, as they are possibly entangled. We feel that this should not necessarily be seen as a drawback, if allowed. For example, the use of sharing allows the programmer to express multi-qubit gates that use control without a predefined gate set or explicit control structures. It also allows one to define interesting higher-order functions. We will present both of these examples in the next chapter.

Before moving on to the definition of the ζ-calculus we need to define the explicit category we are defining our language in. We start by building upon the symmetric monoidal closed category of Λ_{SMC}. We further restrict the category to be compact closed. This means that we use dual types and the morphisms $\eta_A : I \to A^* \otimes A$ and $\eta^\dagger_A : A \otimes A^* \to I$ (we use the name $\eta^\dagger$ instead of its usual name ϵ to avoid confusion with the counit of an internal comonoid, even though we are not working in a †-SMC). With this, we can define the internal morphism for some morphism $f : A \to B$ as $\ulcorner f \urcorner : I \to A^* \otimes B$. When we present the semantics of the ζ-calculus this will allow us to provide a clear graphical interpretation of abstraction and application as string diagrams, where η and $\eta^\dagger$ are on the form (2.2). From this, we can move from the abstract categorical presentation of the ζ-calculus to the ZX-calculus in a clearer manner.

(2.2) $\eta_A \quad :=$



$\eta^\dagger_A \quad :=$



The ZX-calculus is defined in a †-symmetric monoidal category. The reason for us not presenting an internal language for †-SMC's stems from the difficulty of defining semantics which take the †-functor into account. This is possible though, an example of such a language is the dagger λ-calculus [21]. We will discuss the possibility of extending the ζ-calculus in this direction in the future works section. With these formalities clarified we move on to the definition of the ζ-calculus. This will be presented in the sections: syntax, typing, and semantics.

2.1 Syntax

We extend the internal language of symmetric monoidal categories (Λ_{SMC}) given in section 1.1.4.3 with a modification on the terms which introduce variables. We fix a set of symbols ᛒ whose elements we call *bases* or *observable structures*. The usual λ-abstraction is further decorated with a basis in which the variable is introduced. This is a *basis-abstraction* (or ζ-abstraction) on the form $\zeta x M$, where $\zeta \in \text{ᛒ}$ corresponds to some observable structure. The let-expression concerning tuples is also decorated with a basis. As previously noted, each orthonormal basis in a Hilbert space corresponds to an observable structure, and we will use the terms somewhat interchangeably. The minimal syntax of the ζ-calculus, the set of ζ-terms $\lfloor \text{ⵣ} \rfloor$[1] (*algiz*), is defined inductively over a set of observable structures (or bases) ᛒ in figure 2.1.

$$\frac{}{x \in \lfloor \text{ⵣ} \rfloor}\ \textsc{var} \qquad \frac{}{\star \in \lfloor \text{ⵣ} \rfloor}\ \textsc{uni} \qquad \frac{\zeta \in \text{ᛒ} \qquad M \in \lfloor \text{ⵣ} \rfloor}{\zeta x M \in \lfloor \text{ⵣ} \rfloor}\ \textsc{abs}$$

$$\frac{M \in \lfloor \text{ⵣ} \rfloor \qquad N \in \lfloor \text{ⵣ} \rfloor}{MN \in \lfloor \text{ⵣ} \rfloor}\ \textsc{app} \qquad \frac{M \in \lfloor \text{ⵣ} \rfloor \qquad N \in \lfloor \text{ⵣ} \rfloor}{\langle M, N \rangle \in \lfloor \text{ⵣ} \rfloor}\ \textsc{tup}$$

$$\frac{\zeta \in \text{ᛒ} \qquad M \in \lfloor \text{ⵣ} \rfloor \qquad N \in \lfloor \text{ⵣ} \rfloor}{\text{let } \langle x, y \rangle =_{\zeta} M \text{ in } N \in \lfloor \text{ⵣ} \rfloor}\ \textsc{let} \qquad \frac{M \in \lfloor \text{ⵣ} \rfloor \qquad N \in \lfloor \text{ⵣ} \rfloor}{M \mathbin{⨟} N \in \lfloor \text{ⵣ} \rfloor}\ \textsc{sem}$$

Figure 2.1: The minimal syntax of the ζ-calculus.

This syntax is the minimal extension of Λ_{SMC} for our purposes of making the language non-linear by use of observable structures. In the remainder of this section, we will introduce the full syntax of the ζ-calculus, denoted by the set ⵣ. The extensions we provide further utilise constructs built upon observable structures, phase groups, used for providing a notion of rotation. Moreover, we introduce sized units and counits to act as values in the language. These additions make the ζ-calculus appropriate for use as a programming language, which we will provide examples for later.

[1] A variant on the rune for Z, coincidentally looking like a spider.

2.1.1 Phase groups and rotation

To capture the description of processes presented in the ZX-calculus we wish to add some notion of rotation to the ζ-calculus. Thus, for each $\zeta \in \text{ᛒ}$ we fix a set of symbols $\mathcal{U}_\zeta$, whose elements we call *unbiased points* (see section 1.1.5.2 for an overview). Then we further decorate the ζ-abstraction by some unbiased point α.

$$\frac{\zeta \in \text{ᛒ} \qquad \alpha \in \mathcal{U}_\zeta \qquad M \in \text{ᛡ}}{\zeta^\alpha x M \in \text{ᛡ}}\ \textsc{abs} \tag{2.3}$$

This represents a phase shift relative to the observable structure used in the abstraction. Without concretising the observable structures to some particular structure, Hilbert spaces for example, this notion is rather abstract. We refer to alpha as being a point in the set of unbiased points $\mathcal{U}_\zeta$, which generate the group of phase shifts relative to ζ. Later, when we study the ζ-calculus as a quantum programming language this set is parameterised by a phase $\alpha \in [0, 2\pi)$, generating a group of rotations about the two axes, Z and X, of the Bloch sphere that the observable structures in the 2-dimensional complex Hilbert space represent. Intuitively then, the ζ-abstraction $\zeta^\alpha x M$ should be seen as a function that introduces a variable in the basis ζ, rotating it about ζ with phase α before passing it onto the term M.

We have a special element $u \in \mathcal{U}_\zeta$ for each $\zeta \in \text{ᛒ}$ (in the semantics u will be interpreted as the unit of the internal monoid of the observable structure ζ). For a ζ-abstraction $\zeta^u x M$ we omit the unbiased point and write it as $\zeta x M$.

2.1.2 States and effects

To complete the syntax we add a syntactic construct for states and effects, called a generator. States here represent points $\mathrm{Hom}_{\mathcal{C}}(I, A)$ in the category, generated by some observable structure (A, δ, ϵ). Effects are simply the adjoint of states $(\mathrm{Hom}_{\mathcal{C}}(I, A))^\dagger = \mathrm{Hom}_{\mathcal{C}}(A, I)$, representing a function which simply discards. We give the syntactic introduction rule for generators in (2.4).

$$\frac{\zeta \in \text{ᛒ} \qquad \alpha \in \mathcal{U}_\zeta \qquad n \in \mathbb{Z}}{\zeta^\alpha_n \in \text{ᛡ}}\ \textsc{gen} \tag{2.4}$$

This generator is decorated with the observable structure ζ, an unbiased point α, and a size n. The size decoration represents the number of times the base point is duplicated by the comonoid of ζ, where a positive integer represents a state, a negative integer represents an effect, and zero represents a point $\mathrm{Hom}_{\mathcal{C}}(I, I)$, called a scalar.

2.1.3 The complete language

With the addition of rotations, states, and effects we obtain a language that captures the desired properties of the ZX-calculus for use as a functional language. We present the complete syntax of the ζ-calculus in figure 2.2.

$$\frac{}{x \in ⵣ}\,\textsc{var} \qquad \frac{}{\star \in ⵣ}\,\textsc{uni} \qquad \frac{\zeta \in ᛒ \quad \alpha \in \mathcal{U}_\zeta \quad n \in \mathbb{Z}}{\zeta_n^\alpha \in ⵣ}\,\textsc{gen}$$

$$\frac{\zeta \in ᛒ \quad \alpha \in \mathcal{U}_\zeta \quad M \in ⵣ}{\zeta^\alpha x M \in ⵣ}\,\textsc{abs} \qquad \frac{M \in ⵣ \quad N \in ⵣ}{MN \in ⵣ}\,\textsc{app}$$

$$\frac{M \in ⵣ \quad N \in ⵣ}{\langle M, N \rangle \in ⵣ}\,\textsc{tup} \qquad \frac{\zeta \in ᛒ \quad M \in ⵣ \quad N \in ⵣ}{\text{let } \langle x, y \rangle =_\zeta M \text{ in } N \in ⵣ}\,\textsc{let}$$

$$\frac{M \in ⵣ \quad N \in ⵣ}{M \fatsemi N \in ⵣ}\,\textsc{sem}$$

Figure 2.2: The full syntax of the ζ-calculus.

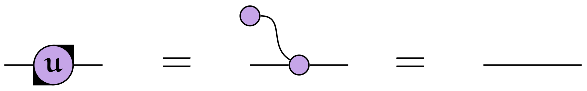
Finally, we define the canonical meta-operations on the syntax. Both free variable calculation and substitution, as described in section 1.1.2, are extended to the syntax of the ζ-calculus in the obvious way. We always identify terms up to α-equivalence, meaning that we consider terms syntactically equivalent when we can make them equal by a change of bound variable names. We denote this syntactic equivalence by $M \equiv N$.

2.1.4 Notational conventions

We define here a set of syntactic conventions which we will use throughout the rest of the thesis. Their definitions differ slightly in their usage and intuition, coming from properties of the categorical constructs, or simply for ease of writing. The notational conventions are defined in figure 2.3.

The first row of definitions rely on two simple facts, which we touched upon briefly when introducing phase groups to the syntax. Since the unit of some observable structure is always an unbiased point, it can always be used in the phase of a ζ-term. Then, by unital elimination of the monoid, and the definition of the phase group, the rotation equals the identity. This is described in the graphical language in (2.5).

(2.5)



$$\begin{aligned}
\zeta x M &:\equiv \zeta^{u} x M & \zeta_n &:\equiv \zeta^{u}_{n} & (*)\\
\lambda x M &:\equiv \zeta^{u} x M & \text{let } \langle x, y\rangle = N \text{ in } M &:\equiv \text{let } \langle x, y\rangle =_{\zeta} N \text{ in } M & (**)\\
M \circ N &:\equiv \lambda x\, M(Nx) & \hat{\zeta}^{\alpha} &:\equiv \zeta^{\alpha} x x &\\
\eta_{\zeta} &:\equiv \zeta_{2} & \eta^{\dagger}_{\zeta} &:\equiv \zeta_{-2} &\\
\langle M, N, L\rangle &:\equiv \langle M, \langle N, L\rangle\rangle & \zeta^{\alpha}\langle x, y\rangle M &:\equiv \zeta^{\alpha} t \text{ let } \langle x, y\rangle =_{\zeta} t \text{ in } M &
\end{aligned}$$

(*) u is the unit of the commutative monoid of ζ
(**) The introduced variables each occur exactly once in M

Figure 2.3: Definition of notational conventions.

The second row describes linear ζ-terms, terms where the variables introduced in them are used exactly once. Since these variables, introduced in some basis, are only used once, we do not need to share or discard them. Then, the observable structure they are introduced in does not matter, since its internal operations are never called upon. This means that it does not matter in which basis the variable is introduced in, and we can refer to any abstraction defined like this, as a λ-abstraction, and the variables as introduced in the λ-basis. This really only means that the basis is arbitrary, since it is not used.

The three final rows describe constructs that are employed frequently when using the ζ-calculus as a programming language. The first of these defines the usual composition function, and the identity function in some basis and rotation. The second one defines the compact structure related to some observable structure, see proposition 6.15 of [2] for a discussion on this. Certain collections of observable structures have compact structures which coincide (meaning that they are all equal), principally this is the case for the set used in the ZX-calculus. In the case that all observable structures in $\mathfrak{B}$ have mutually coinciding compact structures, we omit the basis from the notation (though this depends on the model, rather than the syntax, we wish to bring it up here to collect the notational conventions in one place). Finally, we accept some common conventions regarding the tuple construction. We take n-tuples to be right-nested tuples (by associativity of the tensorial functor), and allow tuples to be accepted as arguments.

2.2 Typing

In this section, we define logical rules of the ζ-calculus which assign a type to every ζ-term. This type system is extended from the one presented in section 1.1.4.3 with the addition of the structural rules of contraction and weakening. Recall that contraction and weakening are the structural rules which are omitted from a linear type system, as they allow variables to be freely duplicated and discarded. We will present in this section the addition of the observable structures that allows for these rules to be defined. The set of types is defined in figure 2.4.

$$\frac{}{I \in \mathsf{Type}} \qquad \frac{}{\mathcal{T} \in \mathsf{Type}} \qquad \frac{A \in \mathsf{Type} \qquad B \in \mathsf{Type}}{A \otimes B \in \mathsf{Type}} \qquad \frac{A \in \mathsf{Type} \qquad B \in \mathsf{Type}}{A \rightarrow B \in \mathsf{Type}}$$

Figure 2.4: Definition of types.

As in section 1.1.4.3 we have the unit type I, the product type $\otimes$, and the function type $\rightarrow$ (the non-linear function type will be used here instead of $\multimap$). The type $\mathcal{T}$ represents the object of any of the observable structures. In an instance of the ζ-calculus, every observable structure will be defined over the same base type $\mathcal{T}$, the canonical example being $\mathbb{C}^2$ in **FdHilb**. For the sake of conciseness when defining the typing rules for generators we define the notion of a *sized type*.

Definition 2.2.1 (Sized types)**.** A sized typed $\underline{n}$ where $n \in \mathbb{Z}$ is defined by:

$$\underline{0} := I \qquad \underline{1} := \mathcal{T} \qquad \underline{n+1} := \underline{n} \otimes \mathcal{T} \qquad \underline{\text{-}n} := \underline{n} \rightarrow I$$

Contexts are defined similarly to all the previously presented type systems, with the addition of keeping track of which observable structure a variable has been introduced in. The grammar of contexts is then defined as $\Gamma ::= \emptyset \mid \Gamma, x :_\zeta A$. Now we move on to the structural rules we wish to define. We keep the exchange rule from Λ_{SMC}, and extend the rules of that language by adding weakening and contraction. In these cases, we use the basis-decorated contexts to keep track of which observable structure duplication and discarding occurs. The full set of structural rules is presented in figure 2.5.

$$\frac{\Gamma \vdash M : B}{\Gamma, x :_\zeta A \vdash M : B}\ \text{w} \qquad \frac{\Gamma, x_1 :_\zeta A, x_2 :_\zeta A \vdash M : B}{\Gamma, x :_\zeta A \vdash M[x_1 := x, x_2 := x] : B}\ \text{c} \qquad \frac{\Gamma, x :_\zeta A, y :_\xi B, \Delta \vdash M : A}{\Gamma, y :_\xi B, x :_\zeta A, \Delta \vdash M : A}\ \text{x}$$

Figure 2.5: Structural rules of the ζ-calculus.

Finally, define the typing rules of the language. From the rules presented in section 1.1.4.3 we add the rule (G) concerning generators. The version presented here, with separate structural rules, still uses the split contexts of Λ_{SMC}, making the contractions explicit in the definition of the semantics in the next section. The syntax-directed set of typing rules can be found in appendix B. The full set of typing rules are now presented in figure 2.6.

$$\frac{}{\vdash \star : \mathrm{I}}\ \textsc{u} \qquad \frac{}{x :_{\zeta} A \vdash x : A}\ \textsc{v} \qquad \frac{}{\vdash \zeta^{\alpha}_{n} : \underline{n}}\ \textsc{g} \qquad \frac{\Gamma, x :_{\zeta} A \vdash M : B}{\Gamma \vdash \zeta^{\alpha} x M : A \multimap B}\ \textsc{b}$$

$$\frac{\Gamma \vdash M : A \multimap B \qquad \Delta \vdash N : A}{\Gamma, \Delta \vdash MN : B}\ \textsc{a} \qquad \frac{\Gamma \vdash M : A \qquad \Delta \vdash N : B}{\Gamma, \Delta \vdash \langle M, N \rangle : A \otimes B}\ \textsc{t}$$

$$\frac{\Gamma \vdash M : A \otimes B \qquad \Delta, x :_{\zeta} A, y :_{\zeta} B \vdash N : C}{\Gamma, \Delta \vdash \text{let } \langle x, y \rangle =_{\zeta} M \text{ in } N : C}\ \textsc{l} \qquad \frac{\Gamma \vdash M : \mathrm{I} \qquad \Delta \vdash N : A}{\Gamma, \Delta \vdash M \fatsemi N : A}\ \textsc{i}$$

Figure 2.6: Typing rules of the ζ-calculus.

From these rules, we see more clearly why the rules which introduce variables (B, L) are decorated with some observable structure ζ. Namely, this is used to extend contexts with the observable structure denoted in the terms.

2.3 Semantics

A model $\mathcal{Z}(\mathcal{C}, \text{฿})$ of the ζ-calculus consists of a symmetric compact monoidal category $\mathcal{C}$ and a set of observable structures ฿. Each observable structure $(\mathcal{T}, \delta_\zeta, \epsilon_\zeta) \in \text{฿}$ is defined over the same base type $\mathcal{T}$. We give the semantics for the ζ-calculus by mapping every derivable judgement to a morphism in the underlying category, described by a labelled string diagram. Every type is translated to an object of the category according to the interpretation presented in (2.6).

$$[\![I]\!] := I \qquad [\![\mathcal{T}]\!] := \mathcal{T} \qquad [\![A \otimes B]\!] := [\![A]\!] \otimes [\![B]\!] \qquad [\![A \to B]\!] := [\![A]\!]^* \otimes [\![B]\!] \tag{2.6}$$

The intention is that a derivable judgement $\Gamma \vdash M : A$ will be mapped to a diagram whose open input wires are labelled by the elements of the labels of $[\![\Gamma]\!]$, and whose open output wire are labelled by the elements of the labels of $[\![A]\!]$. The string diagrams presented in the semantics of the ζ-calculus flow from left to right. We relate each construct presented in the string diagrams to their counterparts in the categorical language in figure 2.7.

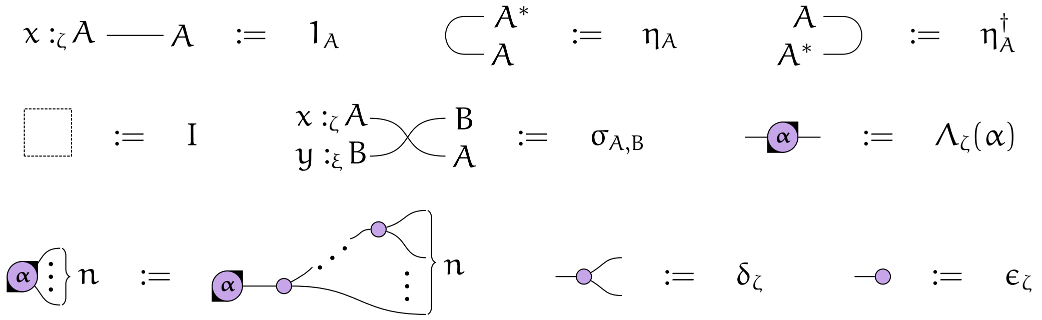


Figure 2.7: The translation between string diagrams and the categorical constructs.

We define the interpretation of a derivable judgement on the induction of its derivation. We begin by giving the interpretation of the structural rules presented in the previous section. Each of the interpretations of the structural rules are presented in figure 2.8.

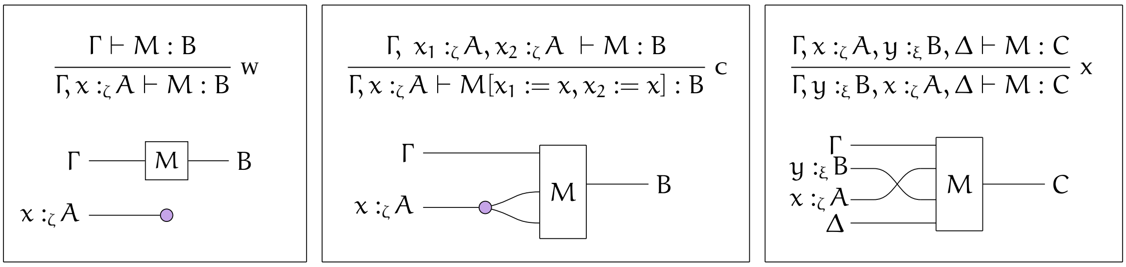
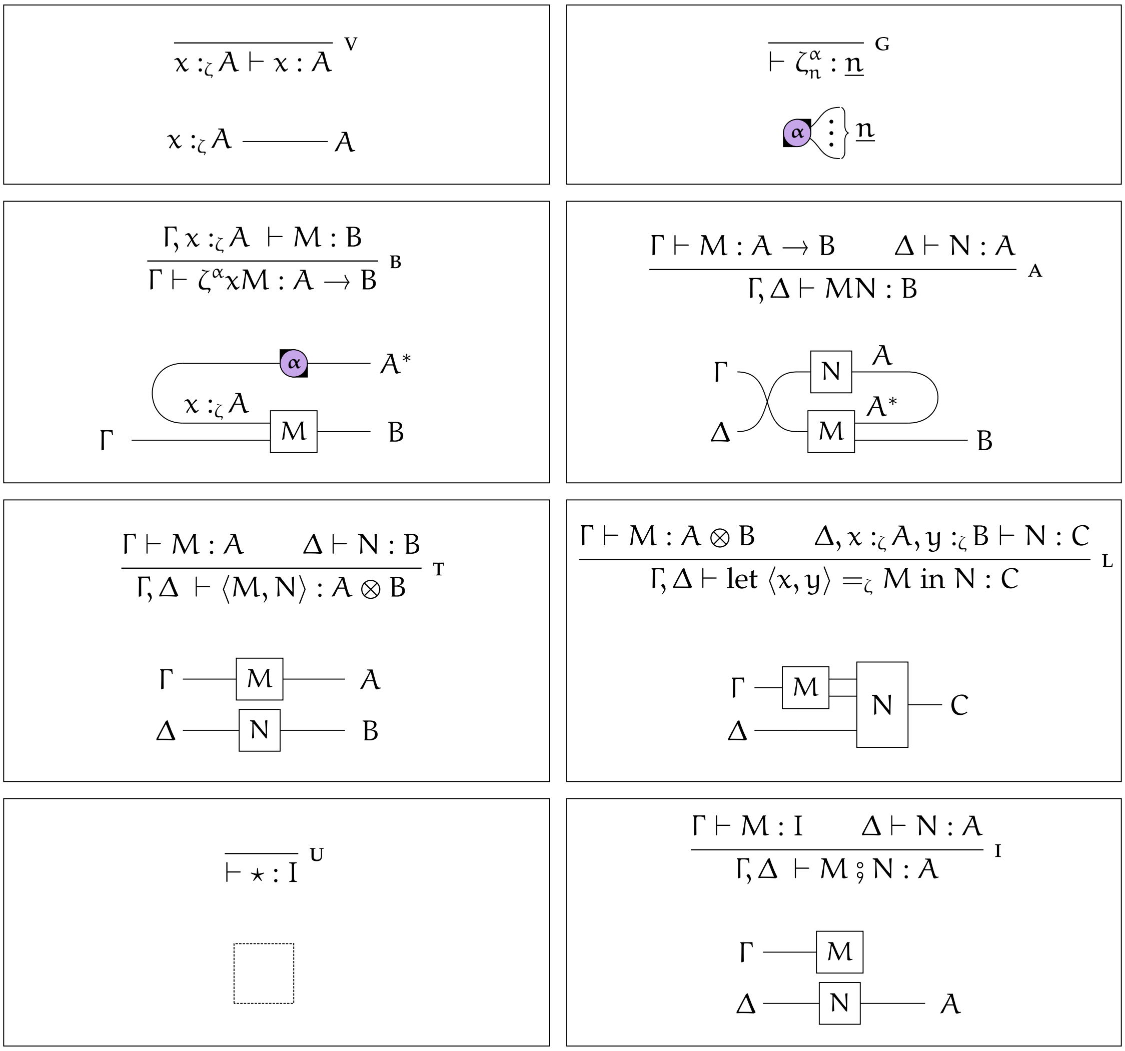

Γ ⊢ M : B
Γ, x :ζ A ⊢ M : B
w
Γ
M
B
x :ζ A
Γ, x₁ :ζ A, x₂ :ζ A ⊢ M : B
Γ, x :ζ A ⊢ M[x₁ := x, x₂ := x] : B
c
Γ
x :ζ A
M
B
Γ, x :ζ A, y :ξ B, Δ ⊢ M : C
Γ, y :ξ B, x :ζ A, Δ ⊢ M : C
x
Γ
y :ξ B
x :ζ A
Δ
M
C

Figure 2.8: Interpretation of structural rules.



$$\frac{}{x :_\zeta A \vdash x : A}\;\textsc{v}$$

$$\frac{}{\vdash \zeta^\alpha_n : \underline{n}}\;\textsc{g}$$

$$\frac{\Gamma, x :_\zeta A \ \vdash M : B}{\Gamma \vdash \zeta^\alpha x M : A \to B}\;\textsc{b}$$

$$\frac{\Gamma \vdash M : A \to B \qquad \Delta \vdash N : A}{\Gamma, \Delta \vdash MN : B}\;\textsc{a}$$

$$\frac{\Gamma \vdash M : A \qquad \Delta \vdash N : B}{\Gamma, \Delta \ \vdash \langle M, N \rangle : A \otimes B}\;\textsc{t}$$

$$\frac{\Gamma \vdash M : A \otimes B \qquad \Delta, x :_\zeta A, y :_\zeta B \vdash N : C}{\Gamma, \Delta \vdash \text{let } \langle x, y \rangle =_\zeta M \text{ in } N : C}\;\textsc{l}$$

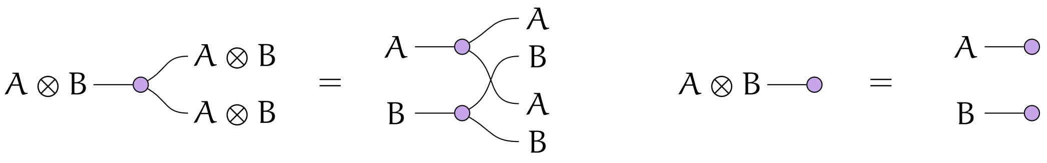
$$\frac{}{\vdash \star : I}\;\textsc{u}$$

$$\frac{\Gamma \vdash M : I \qquad \Delta \vdash N : A}{\Gamma, \Delta \ \vdash M \fatsemi N : A}\;\textsc{i}$$

Figure 2.9: The semantics of the ζ-calculus.

Note that we write the counit and comultiplication with inputs and outputs labelled by general types. We have that the counit and comultiplication scale appropriately over the types, as in proposition 6.28 of [2]. We give this scaling over the tensor type in (2.7). The remaining types remain either unchanged, as for the dual type and base type, or correspond to empty diagrams, as for the unit type.

(2.7)



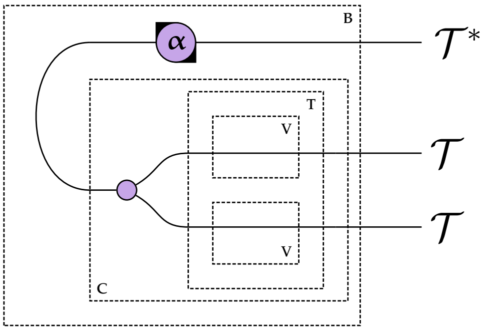
With this, we move on to the definition of the full semantics. We give, for each rule in figure 2.6, an interpretation as a labelled string diagram, as for the structural rules. The interpretations of the typing rules are presented in figure 2.9.

Each box containing a typing rule and a diagram is intended as the statement that the diagram is the interpretation of that rule, something which we will also write later when describing substitution. When writing derivations graphically we denote an instance of a rule by a dashed box with the rule name in its corner. Take for example the type derivation for the term $\zeta^\alpha x\langle x, x\rangle$, given in (2.8).

$$\dfrac{\dfrac{\dfrac{\dfrac{}{x_1 :_\zeta \mathcal{T} \vdash x_1 : \mathcal{T}}\ \text{V} \qquad \dfrac{}{x_2 :_\zeta \mathcal{T} \vdash x_2 : \mathcal{T}}\ \text{V}}{x_1 :_\zeta \mathcal{T}, x_2 :_\zeta \mathcal{T} \vdash \langle x_1, x_2\rangle : \mathcal{T} \otimes \mathcal{T}}\ \text{T}}{x :_\zeta \mathcal{T} \vdash \langle x, x\rangle : \mathcal{T} \otimes \mathcal{T}}\ \text{C}}{\vdash \zeta^\alpha x\langle x, x\rangle : \mathcal{T} \to \mathcal{T} \otimes \mathcal{T}}\ \text{B} \tag{2.8}$$

Note that this derivation uses contraction once, in the (T) rule, to duplicate the variable x. We see the instance of contraction in the interpretation of the term as a diagram in equation 2.9.

(2.9)



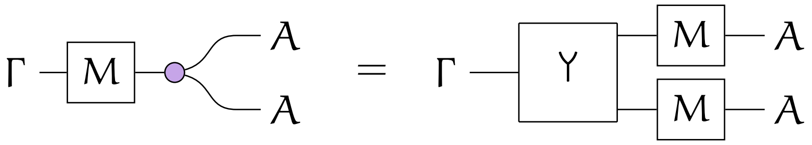
Since this is a closed term, it has no input wires in the outermost box, note however the instance of contraction as the comultiplication and its output being transferred to each of the contexts for the variables. This is a small example of the semantics of the ζ-calculus, intended to help the reader understand the process of translation of ζ-terms into diagrams. In later chapters we will look at various examples of ζ-terms and their graphical interpretations, there we hope that the intuition of these interpretations will become clearer.

2.3.1 Substitution

In the next section, we wish to define some notion of reduction for the calculus. To do this we first need to define what substitution means semantically. This is trivial in the linear case, the details of how this can be done is found in the presentation of Λ_{SMC} by Mackie et al. [1]. When extending this to the non-linear case we have one central problem: *not every term is perfectly duplicated by sharing*. This is expected, of course, as it is a consequence of a widely known fact of quantum states, the no-cloning theorem. For us, this means that we cannot simply substitute every term which is shared, and we have to place some restriction on substitution.

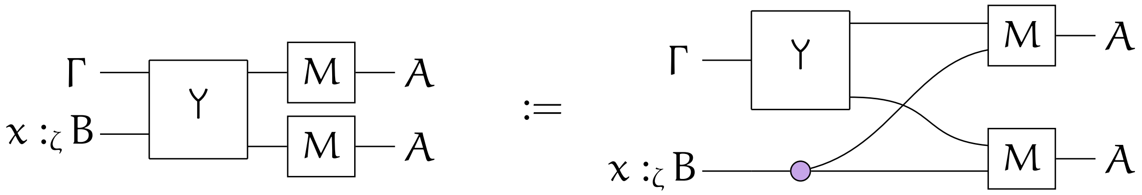
This is common for languages that employ sharing, one example being the linear-algebraic λ-calculus (Lineal) by Arrighi and Dowek [22]. In Lineal, β-reduction is only defined for *basis vector terms*, which are either variables or abstractions. Unlike Lineal, the ζ-calculus does not have a notion of a preferred basis, rather allowing explicit control of bases through observable structures. Thus, to define substitution we also define a condition on which to restrict it, which is dependent on an observable structure. We call this condition *commutation with sharing* (c.w.s.).

Definition 2.3.1. Let $\Gamma \vdash M : A$ be a derivable judgement, and ζ be a basis. Then we say M commutes with sharing over ζ iff



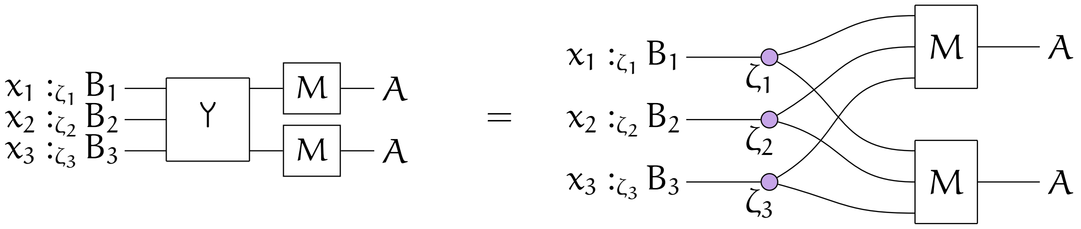
Where ᛉ[2] (*kaun*) is the operation that shares every variable of a context in the basis it is introduced in and sorts them appropriately, as described by (2.10).

(2.10)



For example, for a judgement $x_1 :_{\zeta_1} B_1, x_2 :_{\zeta_2} B_2, x_3 :_{\zeta_3} B_3 \vdash M : A$, we would apply ᛉ on its context to two instances of the interpretation of the judgement as in (2.11).

(2.11)

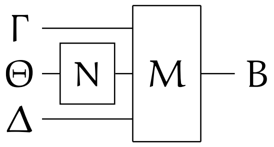


Note that this operation can be performed on any context since every variable in a context is always decorated by the observable structure which was used to introduce it.

[2] A rune, once again, which looks like a wire branching.

Lemma 1 (Substitution)**.** *Let* $\Gamma, x :_{\zeta} A, \Delta \vdash M : B$, *and* $\Theta \vdash N : A$. *If* N *c.w.s over* ζ*:*

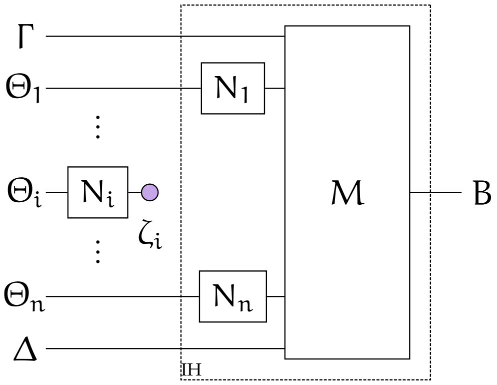
$$\boxed{\Gamma, \Theta, \Delta \vdash M[x := N] : B}$$



Proof. We prove the statement for $\Gamma, x_1 :_{\zeta_1} A_1, \ldots, x_n :_{\zeta_n} A_n, \Delta \vdash M : B$, and judgements $\Phi_i \vdash N_i : A_i$, where N_i c.w.s over ζ_i, which subsumes the above. The proof is by induction on the derivation of $\Gamma, x_1 :_{\zeta_1} A_1, \ldots, x_n :_{\zeta_n} A_n, \Delta \vdash M : B$. We show the cases for which the derivation ends with the application of contraction or weakening on one of the x_i.

Case (w): Let x_i be the variable introduced, then the interpretation is:

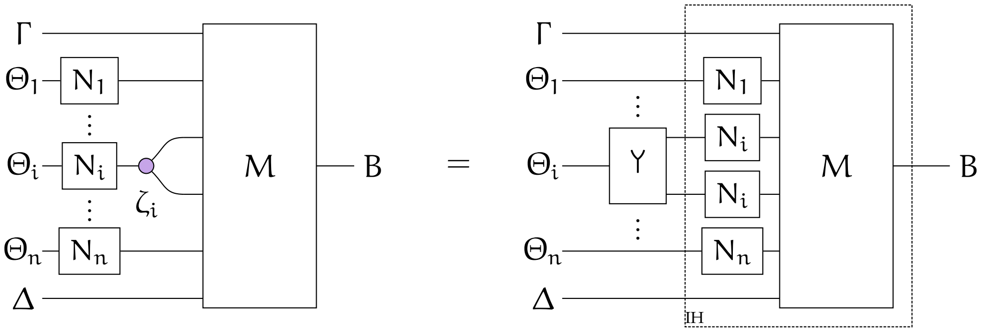
(2.12)



We apply the induction hypothesis on the premise of the weakening to obtain $\Gamma, \Theta_1, \ldots, \Theta_n, \Delta \vdash M[x_1 := N_1, \ldots, x_n := N_n] : B$ without $\Theta_i \vdash N_i : A_i$. Since x_i is introduced by weakening, $x_i \notin \mathbf{fv}(M)$, and it is not substituted.

Case (c): Let x_i be the duplicated variable, then the interpretation is:

(2.13)



Since N_i c.w.s over ζ_i we have the above, on which we apply the induction hypothesis.

□

2.3.2 Reduction

Now we move on to the definition of a notion of reduction. Using the definition of the λ-basis in the notational conventions (figure 2.3), we embed the reductions of Λ_{SMC} for that basis. This is not a basis in the usual sense, one defined by an observable structure, but a notational convention for the observation that if a variable introduced by some basis is only used once, it does not matter what the basis was. This is because neither the counit nor the comultiplication is employed when a variable is used exactly once, and thus no terms need to commute with sharing to be substituted. For the ζ-calculus, this means that we have an embedding of the linear λ-calculus in the notational conventions, with the corresponding reductions. This set of reductions is then extended for the minimal syntax $\lfloor \text{ⵣ} \rfloor$, where we use the c.w.s. condition of lemma 1 to define β-reductions. This is presented in figure 2.10.

$$\begin{array}{rr}
\star \mathbin{⨾} M \to_\beta M & M \mathbin{⨾} \star \to_\eta M \\
\text{let } \langle x, y\rangle = \langle M, N\rangle \text{ in } L \to_\beta L[x := M, y := N] & \text{let } \langle x, y\rangle = M \text{ in } \langle x, y\rangle \to_\eta M \\
\text{let } \langle x, y\rangle =_\zeta \langle M, N\rangle \text{ in } L \overset{(*)}{\to}_\beta L[x := M, y := N] & \text{let } \langle x, y\rangle =_\zeta M \text{ in } \langle x, y\rangle \to_\eta M \\
(\lambda x M)N \to_\beta M[x := N] & \lambda x M x \to_\eta M \\
(\zeta x M)N \overset{(**)}{\to}_\beta M[x := N] & \zeta x M x \overset{(***)}{\to}_\eta M
\end{array}$$

(*) M and N c.w.s. over ζ
(**) M c.w.s. over ζ
(***) $x \notin \mathbf{fv}(M)$

Figure 2.10: Reduction rules of the ζ-calculus.

The rules for η-reduction follow directly from Λ_{SMC} with the addition of the usual requirement (***), that $x \notin \mathbf{fv}(M)$. For the linear cases, where variables introduced are used exactly once and no rotation is applied, we have the usual β-reductions. The other cases for ζ-abstractions have the requirement of commutation with sharing. Because of this requirement being a semantic one, dependent on the model, the reduction relation also depends on the model that is employed. This is also why the reduction relation is defined on the minimal syntax, because the rules associated with the phase group introduced for the full syntax depend on the model. The combined notions of reduction $\beta \cup \eta$ have their reduction relation written as $\to_{\beta\eta}$. We wish to have some notion of soundness for this combined reduction, that it preserves the semantics of the terms involved. For this reason, we prove the following theorem.

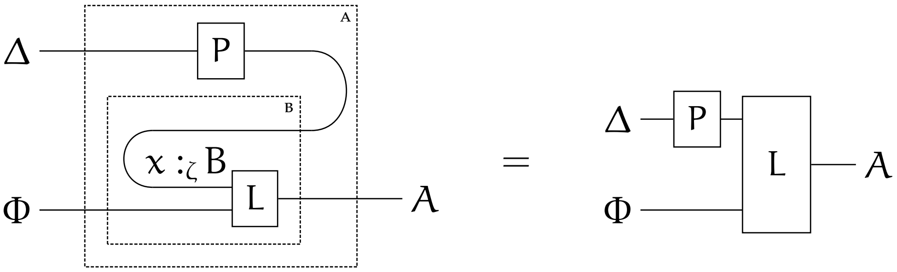
Theorem 1 (Subject reduction). *Let* $\Gamma \vdash M : A$ *and* $M \rightarrow_{\beta\eta} N$*, then* $\Gamma \vdash N : A$ *and*

$$[\![\Gamma \vdash M : A]\!] = [\![\Gamma \vdash N : A]\!]$$

Proof. By induction on the reduction $M \rightarrow_{\beta\eta} N$. We will show two cases, the β and η rule for the ζ-abstraction.

Case 1: $((\zeta x L)P \rightarrow_{\beta} L[x := P])$ This gives a derivation diagram on the form (2.14), where $\Gamma = \Delta \cup \Phi$.

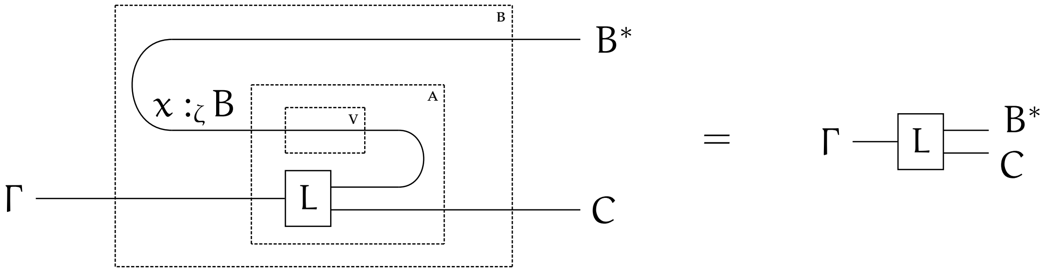
(2.14)



From the condition $(**)$ of this reduction case, that P c.w.s. over ζ, we apply lemma 1, which gives the interpretation of the β-reduct.

Case 2: $(\zeta x L x \rightarrow_{\eta} L)$ From the condition $(*)$, that $x \notin \mathbf{fv}(L)$, we get the derivation:

(2.15)



Where the output type is $A = B \rightarrow C$. Then, by yanking the variable x, we have the interpretation of the η-reduct.

The linear cases are trivial since they do not employ any weakening or contraction on the variables introduced, and can be substituted immediately. The cases for β-reduction on let-expressions use the same method as employed for β-reduction here. Lastly, the cases for $\star$-eliminations employ the unitors of the underlying category.

□

This concludes the definition of the notions of reduction for the calculus. We would like to note that these reduction relations are not complete, and they do not involve the full syntax as we previously described. It is possible to use the properties of observable structures in [2] to define other reductions. Principally, the properties of interest are classical points and unbiased points. Classical points of some observable structure ζ, are closed terms that always commute with sharing over ζ. In the next chapter, we will show examples of these in the complementary observable structures of the ZX-calculus. Unbiased points, which are included in the full definition of the syntax, do not commute, and would as such need another notion of rewriting to be included in the notions of reduction. We will discuss these weaknesses and what needs to be worked on to make this reduction relation strong in later chapters.

2.3.3 Externalisation of closed ζ-terms

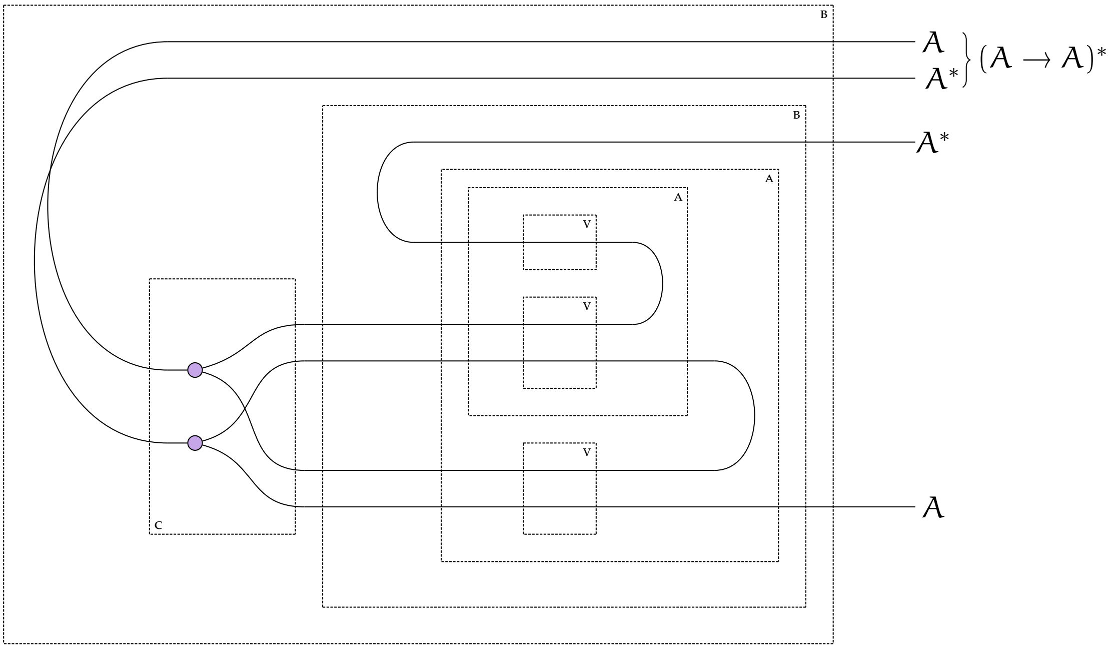
When using the ζ-calculus as a programming language, we will focus heavily on the set of closed ζ-terms $\text{Ж}^{\emptyset}$, that is terms which contain no free variables. These are the terms whose interpretation does not rely on any context, and as such, are directly compilable. Looking at the interpretation of these terms as diagrams, they are precisely the diagrams that have no open input wires. When such a term describes a function, a ζ-abstraction, it represents an internal morphism in the category $\ulcorner \mathsf{f} \urcorner : \mathrm{I} \multimap A^* \otimes \mathrm{B}$, corresponding to some (external) morphism $\mathsf{f} : A \multimap \mathrm{B}$. When ascribing some concrete denotation to the calculus, we would like to have a way of looking at the closed terms in their externalised form. Therefore, we introduce the following operation.

Definition 2.3.2 (Externalisation)**.** Let $M \in \text{Ж}^{\emptyset}$ be a closed ζ-term. The externalised interpretation $M \looparrowright \mathscr{E}$ is the diagrammatic interpretation $[\![\vdash M : A]\!] = \mathscr{D}$ where all higher-order arguments are supplied with boxes, and the remaining open dual output wires reversed to open input wires. The labels and boxes of the diagram interpretation $\mathscr{D}$ are then removed to produce the externalised diagram.

This definition might seem somewhat ad hoc, because it is. It is not a formal notion, but rather a way of looking at how the diagrammatic interpretations of closed ζ-terms act upon both first and higher-order arguments. We shall try to make its intuition clearer by example.

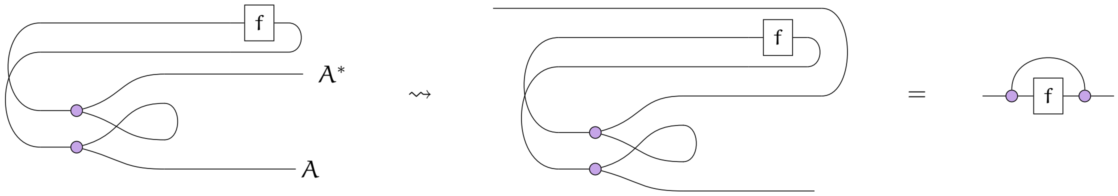
Take the higher-order ζ-abstraction $\zeta \mathsf{f}\ \mathsf{f} \circ \mathsf{f} \in \text{Ж}^{\emptyset}$, which shares some function and composes it with itself. De-sugaring the term (expanding the definition of composition) we have its judgement $\vdash \zeta \mathsf{f} \lambda \mathsf{x}\ \mathsf{f}(\mathsf{f}\mathsf{x}) : (A \multimap A) \multimap A \multimap A$, with interpretation presented in (2.16).

(2.16)



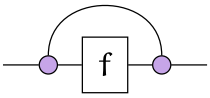
Then, we follow the procedure from definition 2.3.2, placing a function box on the higher-order argument, then reversing the remaining dual output wire, and removing all rule boxes and types. Finally, we deform the diagram according to the string diagram rules.

(2.17)



Thus, we write the externalisation of the aforementioned ζ-term as presented in (2.18).

(2.18) $\zeta f\ f \circ f$ ↦



This is the general procedure for presenting closed ζ-terms as string-diagrams, other than presenting the derivation diagram itself.

2.4 Instances, models, and concretisation

With the particulars of the theory defined we wish to make clear the language we are going to use to describe specific ”versions” of the ζ-calculus. We previously discussed the difference between abstract and concrete categories, ones which are defined simply by their internal structure, and ones which refer to some specific mathematical structure. The theory so far has been defined abstractly in symmetric monoidal compact categories, while the subsequent chapter will instead employ a concrete category, the category of finite-dimensional Hilbert spaces. We call this specification a concretisation of the category.

When defining a specific set of symbols in specifying the syntax of the theory, we write $ᛯ(ᛒ,\mathcal{U}_{\zeta\in ᛒ})$ for an *instance* of the ζ-calculus. This defines the syntax, with the set of basis symbols defined as $ᛒ$, and a set of unbiased point symbols $\mathcal{U}_\zeta$ for each basis $\zeta \in ᛒ$. This notion is purely syntactic.

When specifying a notion of semantics for the calculus, we write $\mathcal{Z}(\mathcal{C}, ᛒ)$ for a *model* of the ζ-calculus. This consists of a concrete category $\mathcal{C}$ together with a set of observable structures $ᛒ$ in that category. The syntax for this model is defined by a symbol for each observable structure in $ᛒ$, a set of symbols for the unbiased points generated by each of the observable structures.

3
Applications

Men stormar man himlen? — Den hvälft sin rund.
Högtidigt i stjärnenätter,
Långt förr än man murade Babels grund
På folkhafvets vida slätter.
Den skickat oss blixtar och stormars brus
Och gifvit oss vårregn och lett i ljus

R. Almén

MOVING on from the abstract categorical and type-theoretic language of the previous chapter we introduce in this chapter an application of the ζ-calculus in concrete models. We will introduce several models of the language, with different sets of observable structures, though all in the concrete category of finite-dimensional Hilbert spaces.

The first model is the quantum ζ-calculus, a functional programming language for quantum computation. We show that the interpretation of terms in this model are ZX-diagrams, producing a denotation of the language suitable for both optimisation and compilation. We demonstrate unique quantum programming techniques that distinguish the ζ-calculus from other quantum programming languages, and give the intuitions behind them through externalisation to the ZX-calculus.

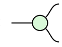
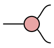
The second instance is the spacetime ζ-calculus, where the set of observable structures are defined by the γ-matrices of fermionic quantum field theory. We then explore the consequences of this construction, specifically the representation of the phase groups as rotations about the spacetime axes of the projective space of the quaternionic Hopf fibration. This 4-sphere serves as an extension of the projective space of the complex Hopf sphere, namely the Bloch sphere.

Finally, we derive a conjecture connecting the previous instances of the ζ-calculus as part of a ladder of instances. We call these *orders of computation*, ranging from the classical λ-calculus to an, as of yet, unexplored order two levels above quantum computation.

3.1 Quantum programming

In the previous chapter, we introduced the theory necessary for a language capable of describing computation in terms of observable structures in any SMC, and while there is much strength in a description at this level of abstraction, it may be that some of the notions introduced will be made more clear when written in a concrete setting. We will thus in this section restrict ourselves to a particular concrete category and give an interpretation of the language in terms of the particulars for that category. In order to obtain a language capable of describing quantum computation, a denotation of the language will be attempted in terms of string diagrams of the ZX-calculus. To do this, the ζ-calculus will be concretised in the category of finite-dimensional Hilbert spaces **FdHilb**.

Definition 3.1.1 (The quantum ζ-calculus)**.** The quantum ζ-calculus is the model $\mathcal{Z}(\mathbf{FdHilb}, \{\zeta, \xi\})$ of the theory concretised in the category of finite-dimensional complex Hilbert spaces. The set of complementary observable structures is defined as $\beta = \{\zeta, \xi\}$, where

(3.1) $\zeta := (\mathbb{C}^2,\ \delta_Z =$ $,\ \epsilon_Z =$ $)$ $\qquad$ $\xi := (\mathbb{C}^2,\ \delta_X =$ $,\ \epsilon_X =$ $)$

The base type $\mathcal{T}$ of the calculus (figure 2.4) denotes $\mathbb{C}^2$, a standard 2-dimensional qubit, which we shall refer to as $\mathcal{Q}$. The comultiplication and counit for each of the observable structures are the copying and erasing maps for the computational and Hadamard basis respectively, as defined by (3.2) and (3.3).

$$\delta_Z := |00\rangle\langle 0| + |11\rangle\langle 1| \qquad \epsilon_Z := \langle 0| + \langle 1| \tag{3.2}$$

$$\delta_X := |{+}{+}\rangle\langle +| + |{-}{-}\rangle\langle -| \qquad \epsilon_X := \langle +| + \langle -| \tag{3.3}$$

Their respective phase groups and unbiased points are defined over a phase $\alpha \in [0, 2\pi)$ in (3.4) and (3.5). These phase shifts are the rotations about the axes Z and X of the Bloch sphere by an angle α.

$$\Lambda_Z(\alpha) := |0\rangle\langle 0| + e^{i\alpha}\,|1\rangle\langle 1| \qquad \alpha_Z := |0\rangle + e^{i\alpha}\,|1\rangle \tag{3.4}$$

$$\Lambda_X(\alpha) := |+\rangle\langle +| + e^{i\alpha}\,|-\rangle\langle -| \qquad \alpha_X := |+\rangle + e^{i\alpha}\,|-\rangle \tag{3.5}$$

The induced compact structures of these observable structures coincide, and thus we shall denote both of the compact structures by η. This means that $[\![\eta_\zeta]\!] = [\![\eta_\xi]\!] = |00\rangle + |11\rangle$ and $[\![\eta^\dagger_\zeta]\!] = [\![\eta^\dagger_\xi]\!] = \langle 00| + \langle 11|$. As such, when there is no ambiguity in the interpretations of terms as diagrams, we shall write the compact structures as cups.

The state space associated with the phase group and its compositions is thus given the topology of the standard unit Bloch-sphere, i.e. S^2. The normalisability condition of the Hilbert space $\mathcal{H}$ (see section 1.1.1.1), ensures that one need not bother with global phases of the form $e^{i\alpha}$ when writing expressions for states in $\mathcal{H}$. We will look at this more closely. A state $\psi \in \mathcal{H} \cong \mathbb{C}^2$ is written $\psi := (\alpha\ \beta)^T$ for $\alpha, \beta \in \mathbb{C}$. The normalisation condition $|\alpha|^2 + |\beta|^2 = \alpha\alpha^* + \beta\beta^* = 1$ which is just the definition of the S^3 as below.

$$S^3 = \{z_1, z_2 \in \mathbb{C} \ : \ z_1 z_1^* + z_2 z_2^* = 1\} \tag{3.6}$$

While states of the form of ψ are usually represented on the Bloch sphere, this is only the projective space of such states with the *total* space of such states being S^3. The projection $S^3 \hookrightarrow S^2$ can be performed as the composition of two conformal maps, the process of which is described further in [23], but we will mention it only shortly here. The projection is composed of two maps.

$$h_1 : S^3 \to \mathbb{R}^2 + \{\infty\} \qquad\qquad h_2 : \mathbb{R}^2 + \{\infty\} \to S^2 \tag{3.7}$$

The first map h_1 is defined as a quotient of the two complex numbers defining a vector ψ in S^3, where the quotient ensures that the global phase term factors out. The second map h_2 is an inverse stereographic map, where we finally reach a representation of the original state ψ on the Bloch sphere.

The complex phase factor that was lost in the normalisation can be written like $e^{i\alpha} \in \mathbb{C}$. For an arbitrary $\alpha \in \mathbb{R}$, this term normalises to unity. From this one can retrieve the definition for the circle, or S^1, given below.

$$S^1 := \{z \in \mathbb{C} \ : \ |z|^2 = 1\} \tag{3.8}$$

The state ψ can always be written like $e^{i\alpha}\psi$ such that in the projection we can identify a family of states in S^3 that map to the same ψ in S^2. This is an interpretation[23] of the statement of the complex *Hopf fibration* written as

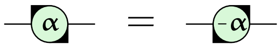
$$S^1 \hookrightarrow S^3 \to S^2 \qquad \text{or} \qquad S^3 \xrightarrow{S^1} S^2 \tag{3.9}$$

which might be read as "S^3 is *fibred* over S^2" with each fibre homeomorphic to S^1 or "S^1 is embedded in S^3 that projects to S^2. With regard to our Hilbert space $\mathcal{H}$, this relation ascribes the non-trivial representation of states $\psi \in \mathcal{H}$ on the Bloch sphere, with this non-triviality being encoded by the global phase freedom. It will be extended further in later sections when we consider a base Hilbert space of $\mathbb{C}^4$. The information presented and statements made with regard to the Hopf fibration are discussed in greater detail in [23, 24].

3.1.1 The ZX-calculus

From the definition of the quantum ζ-calculus we now present the consequences of the concretisation on the interpretations of ζ-terms as string diagrams. The concretisation we presented is precisely the one used by the ZX-calculus, of **FdHilb** and the two complementary observable structures, ζ and ξ. The string diagram language of interpretations is now simplified in the following sense. With the unbiased points parameterised over $\alpha \in [0, 2\pi)$ we have the relationship of conjugation on the phase group as described in (3.10).

(3.10)



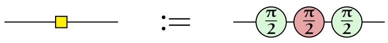
Because of this, which also applies to the generators of the observable structures, we omit the edges of the phase groups $\Lambda_Z(\alpha)$ and $\Lambda_X(\alpha)$, in favour of the negation of the phase. Both in the ζ-terms and their interpretations we will write the unbiased points as phases on this form.

All of the diagrams we present will be the externalised form of closed ζ-terms, which will give a clearer understanding of how the terms act as quantum programs. Since all labels are removed from the externalised diagrams, and only with the addition of arbitrary function boxes, the diagrams presented for ζ-terms will be the usual ZX-diagrams with the addition of these boxes. The diagram derivations of every non-trivial ζ-term presented in this chapter will have its derivation presented in appendix C.

Finally, we introduce a notational convention relating to this specific model of the theory, the Hadamard gate. As presented in [15], this gate can be defined by decomposition to Euler angles, producing the ζ-term shown in (3.11), which we shall call H and denote by a yellow box.

(3.11)

$$H \quad :\equiv \quad \hat{\zeta}^{\frac{\pi}{2}} \circ \hat{\xi}^{\frac{\pi}{2}} \circ \hat{\zeta}^{\frac{\pi}{2}} \qquad \looparrowright$$



To recap, the interpretations of ζ-terms will be presented as ZX-diagrams with function boxes, the externalised form of their derivations. The observable structure ζ is presented as a green spider, while ξ is presented as a red one, both possibly containing some phase in them. Moving on, we will present some examples of quantum programming in the ζ-calculus.

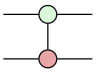
3.1.2 Examples

We will begin by stating an important property of the ζ-calculus, that it is universal for describing quantum computation. This is true by the fact that it is possible to write terms that form a universal gate set for quantum computation. By the Solovay-Kitaev theorem [25], any arbitrary unitary transformation can be approximated by a finite number of unitary transformations from a universal gate set. It is well known that any single-qubit unitary can be modelled by a series of rotations by decomposing it in terms of Euler angles as $U(\alpha, \beta, \gamma) :\equiv Z(\alpha) \circ X(\beta) \circ Z(\gamma)$. Using this construction together with the construction of the CNOT gate presented below, any unitary n-qubit gate can be constructed. [26]

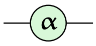
Theorem 2 (Quantum universality). *The ζ-calculus is quantum universal.*

Proof. We present the ζ-terms for the CNOT, X-shift, and Z-shift gates below.

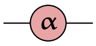
$$\mathsf{cnot} :\equiv \zeta c \xi t\ \eta^\dagger \langle c, t\rangle \fatsemi \langle c, t\rangle$$



$$Z(\alpha) :\equiv \hat{\zeta}^\alpha$$



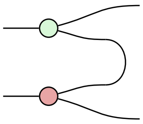
$$X(\alpha) :\equiv \hat{\xi}^\alpha$$



From these gates, and by the argument presented above, we have a universal gate set. These gates, together with permutations and compositions performed in the λ-basis, can then encode any function on a multi-qubit state. □

The presentation of the CNOT gate can look somewhat strange, compared to other quantum programming languages with explicit control structures. Its interpretation, however, comes quite naturally from the style of programming allowed by the ζ-calculus. Reading the ζ-term, it introduces the control variable c in the Z-basis, and the target variable t in the X-basis. It then shares each of them, connecting the first shared variables by a cap $\eta^\dagger$, and returning the rest. The intuition of this term then, is shown in (3.12).

$$\zeta c \xi t\ \eta^\dagger \langle c, t\rangle \fatsemi \langle c, t\rangle \qquad \looparrowright \tag{3.12}$$

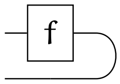


We will now build on the intuition behind the construction of the CNOT gate to construct general functions which link variables introduced by a ζ-abstraction by some other function.

3.1.2.1 Linking functions

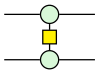
In the previous example, we demonstrated that it is possible to link variables that are introduced in ζ-abstractions. Instead of simply connecting them by a cap, we can place any single qubit function between them. We call the term which connects variables by some function the *linking function*. For some single qubit function f we call its linking function $\ell_f :\equiv \lambda\langle x, y\rangle\ \eta^\dagger\langle fx, y\rangle$. Its interpretation with a function as its argument is presented in (3.13).

(3.13) $$\lambda f \lambda \langle x, y \rangle \; \eta^{\dagger} \langle fx, y \rangle \qquad \looparrowright$$



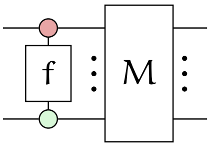
We can then recover the original definition of the CNOT gate as $\zeta c \xi t \; \ell_{I} \langle c, t \rangle \fatsemi \langle c, t \rangle$, where $I :\equiv \lambda xx$ is the identity function. The controlled Z gate is also easily constructed by the same method, but where the variables are both introduced in ζ and linked by a Hadamard gate, shown in (3.14).

(3.14) $$\zeta c \zeta t \; \ell_{H} \langle c, t \rangle \fatsemi \langle c, t \rangle \qquad \looparrowright$$



More generally, we can construct a ζ-abstraction which links its arguments, introduced in arbitrary bases, by some function before passing it onto the body of the abstraction. The general form of the usage of the linking function is shown in (3.15)[1].

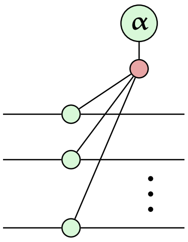
(3.15) $$\xi x \zeta y \; \ell_{f} \langle x, y \rangle \fatsemi M \qquad \looparrowright$$



Where the bases that the variables are introduced in can be chosen freely, of course.

Another class of functions which can be implemented by linking are phase gadgets [15]. Phase gadgets are ZX-diagrams on the form of (3.16), implementing the action of a unitary operator U_f on a string of inputs in the computational basis $|\vec{x}\rangle = |x_0 \ldots x_n\rangle$ as $|\vec{x}\rangle \mapsto e^{if(\vec{x})} |\vec{x}\rangle$. The function f returns some phase α only when the input string folded over XOR is 1, defined as $f(\vec{x}) := \alpha(x_0 \oplus \cdots \oplus x_n)$. For example, for the input state $|010\rangle$, the function f would return $f(0, 1, 0) = \alpha(0 \oplus 1 \oplus 0) = \alpha$, and the action of U_f would produce the output $e^{i\alpha} |010\rangle$. A proper exposition of phase gadgets, and why they are interesting, can be found in [15].

(3.16)

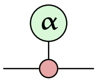


We can define such a gadget as a ζ-abstraction by sharing a variable intro-

[1] Slight abuse of notation here... The term M and the dots are not usually included in an externalised diagram, though we hope the intuition is clear. It is a general term, and thus a general diagram.

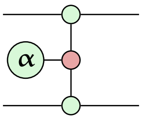
duced in ξ to a phase denoted effect in ζ (3.17).

(3.17) $$G(\alpha) :\equiv \xi x\ \zeta^{\alpha}_{-1} x \fatsemi x \quad \looparrowright$$



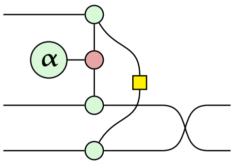
Which we can then give to the linking function to produce a phase gadget diagram, as presented in (3.18).

(3.18) $$\zeta x \zeta y\ \ell_{G(\alpha)}\langle x, y\rangle \fatsemi \langle x, y\rangle \quad \looparrowright$$



This pattern of linking variables can also be extended, connecting more of them by stacking linking functions. An example of this, though on a somewhat crazy form, is shown in (3.19).

(3.19) $$\zeta x \zeta y \zeta z\ \ell_{G(\alpha)}\langle x, y\rangle \fatsemi \ell_{H}\langle x, z\rangle \fatsemi \langle z, y\rangle \quad \looparrowright$$



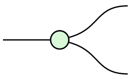
Reading the ζ-term, however, we see that its intention is actually clearer than the diagrammatic form. It introduces three variables in the ζ-basis, sharing them each once, and linking them with a gadget function and a Hadamard, before returning the last two arguments swapped. In general, the intentions behind ζ-terms are quite clear, especially once you are familiar with their diagrammatic interpretations.

Concluding this section, we have shown that having explicit control of which basis variables are shared with, together with the ability to link the variables together, make for a concise method of constructing multi-qubit unitaries. This is somewhat surprising, since many quantum programming languages either come with these unitaries pre-defined [18], or use explicit control structures to define them [27]. Explicit control of the bases of sharing then, is one unique feature of the quantum ζ-calculus.

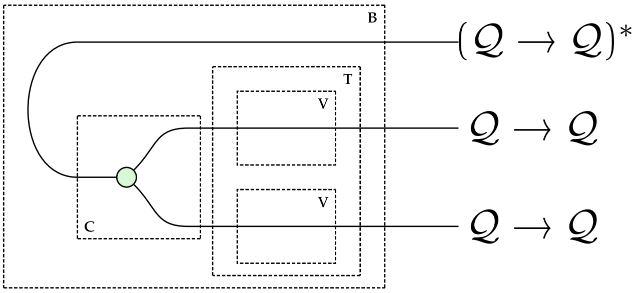
3.1.2.2 Higher-order functions and sharing

In this section, we wish to explore what sharing means when the variable shared is itself a function. This should of course be familiar to computer scientists as *higher-order functions*, which is something that the ζ-calculus explicitly allows. The question is then, what does it look like when sharing and discarding is employed? One should not expect this to look anything like it does for the classical case, that the functions are perfectly copied, but rather investigate the specific properties that arise from this kind of construction. As with first-order sharing we do not think that the fact that entanglement is produced in these terms is a drawback, but rather a cause for exploration. The linear λ-calculus is embedded in the ζ-calculus anyways, and can be enforced syntactically on closed terms (by counting the occurrences of variables). With this, we shall try to showcase some ζ-terms which employ higher-order sharing.

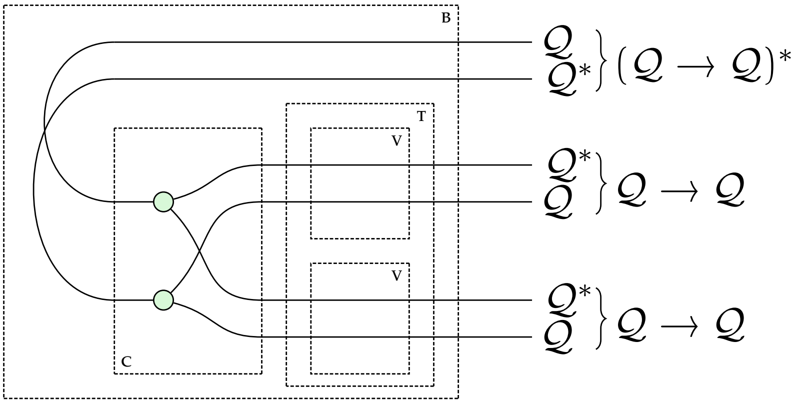
The simplest example comes from the minimal sharing function, that is $\zeta x\langle x, x\rangle$, a function which produces two shared copies of some variable. Its externalisation is shown in (3.20).

(3.20) $\zeta x\langle x, x\rangle \quad \looparrowright$ 

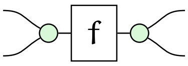
If we supply this ζ-abstraction with a higher-order type instead, we get the diagrammatic derivation shown in (2.8).

(3.21) 
B
T
V
V
C
$(\mathcal{Q} \to \mathcal{Q})^*$
$\mathcal{Q} \to \mathcal{Q}$
$\mathcal{Q} \to \mathcal{Q}$

Then, if we expand its type to the base type, and duplicate the comultiplication according to (2.7), the derivation becomes (3.22).

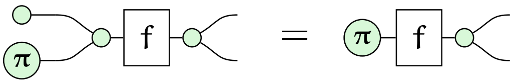
(3.22) 
B
T
V
V
C
$\mathcal{Q}$
$\mathcal{Q}^*$
$(\mathcal{Q} \to \mathcal{Q})^*$
$\mathcal{Q}^*$
$\mathcal{Q}$
$\mathcal{Q} \to \mathcal{Q}$
$\mathcal{Q}^*$
$\mathcal{Q}$
$\mathcal{Q} \to \mathcal{Q}$

The externalisation of the minimal higher-order sharing example then, which we denoted here simply by changing the variable name to f, is shown in (3.23).

(3.23) $\zeta f\langle f, f\rangle \quad \looparrowright$ 

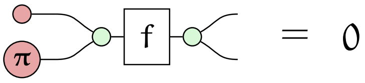
Note here that even though we mentioned several times before that the monoid of observable structures is not really used in this version of the ζ-calculus, it does show up when ζ-terms are externalised, since wires are reversed. Then, we see that the sharing of a variable of a function type simply shares both the input and the output. This has some interesting consequences. For one, it means that arguments to the separated shared instances of the function in the same basis *fuse* together, with their phases added. For example, for two shared copies $\langle g, h\rangle = (Zx\langle x, x\rangle)f$ of some function f, we have (3.24).

(3.24) $\langle g\zeta_1, h\zeta_1^{\pi}\rangle \qquad \looparrowright$



Which is equivalent to applying the original function to the sum of the phases of the arguments, then sharing the result in the same basis as before, that is equivalent to the ζ-term $(\zeta x\langle x, x\rangle)(f\zeta_1^{\pi})$. Moreover, if we do the same thing, but with the arguments in the dual basis ξ, we have (3.25) by the rules of the ZX-calculus.

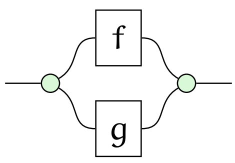
(3.25) $\langle g\xi_1, h\xi_1^{\pi}\rangle \qquad \looparrowright$



In some sense, this means that supplying a function shared in one basis with different arguments in its dual basis is an impossible quantum event! One can interpret this in various ways; we chalk it up to the time-reversing nature of entanglement. If we look at (3.22) again, we can look at the multiplication on inputs as the comultiplication on time-reversed inputs. Since the classical points (points which copy perfectly) of the comultiplication are the points $[\![\xi_1]\!] = |0\rangle$ and $[\![\xi_1^{\pi}]\!] = |1\rangle$, it only produces perfect copies of such points, not ones on the form we supplied as arguments in (3.25). Thus, making the event impossible.

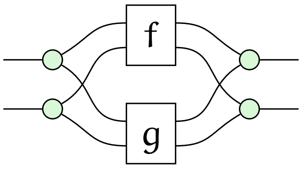
This demonstrates the inability to use shared functions as perfect independent copies of each other, not unexpected. So what can else can we do with higher-order sharing? Well, it is possible to use this construction to modify the behaviour of a function, thus creating a higher-order ζ-abstraction which modifies a function, instead of duplicating it. We looked at such a term when defining externalisation, the self-composition function (2.18). We can extend this ζ-abstraction to inject another function in the composition (3.26).

(3.26) $\lambda f\zeta g\ g \circ f \circ g \qquad \looparrowright$

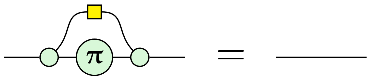


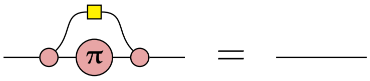
That is, it runs two functions in parallel with respect to some basis. It also scales suitably when defined for multi-qubit functions (3.27).

(3.27)

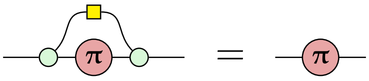


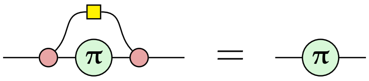
An example of such a function would be $\text{switch}_\beta :\equiv \beta f\ f \circ H \circ f$ for some basis $\beta \in \{\zeta, \xi\}$, which we call the *Pauli switching function*. So called because of its behaviour when applied to a Pauli gate ($\sigma_x :\equiv \hat{\xi}^\pi$, $\sigma_z :\equiv \hat{\zeta}^\pi$). When applied to a Pauli gate of the same basis it switches it off, that is, it becomes the identity gate instead. Its behaviour is shown in (3.28) and (3.29).

(3.28) $\text{switch}_\zeta\ \sigma_z \quad \looparrowright$ 

(3.29) $\text{switch}_\xi\ \sigma_x \quad \looparrowright$ 

When applied to a Pauli gate of the dual basis, it leaves the gate unchanged, shown in (3.30) and (3.31).

(3.30) $\text{switch}_\zeta\ \sigma_x \quad \looparrowright$ 

(3.31) $\text{switch}_\xi\ \sigma_z \quad \looparrowright$ 

The proofs of this behaviour in the ZX-calculus are left as an exercise to the reader (the solutions can be found in [28]). It is also easy to show that the Pauli switching function is self-inverse, and as such, it switches on Pauli gates when applied to the identity function [28].

3.2 The spacetime ζ-calculus

In the previous section, we defined a model of the ζ-calculus for two complementary observable structures over a base type of $\mathbb{C}^2$. Using an underlying representation of the observable structures $\mathbf{Z}, \mathbf{X}$ in the Pauli-matrices σ_Z, σ_X, we managed to define a model of the ζ-calculus for describing quantum computation. In this section, we will extend the set of observable structures to a set of four and define a model of the ζ-calculus thereupon.

The set of four observable structures to be defined here will, like in the previous case, have an underlying matrix representation and a corresponding base type. For the underlying matrix representation, we choose a set of four matrices that are widely used in fermionic quantum field theory, the γ-matrices. This is a set of four four-dimensional complex matrices $\gamma^\mu \in \mathbb{C}^{4\times4}$ whose eigenvectors must thus also be four-dimensional, meaning that we get a base type of $\mathbb{C}^4$. We will proceed with the definition of this model but for the unacquainted, we will first provide a short introduction to these matrices and motivate this particular choice of representation.

We have previously related the Pauli-matrices to the three axes of particle spin in section 1.1.1.2 and we can likewise geometrically interpret the γ-matrices. Specifically, they are related to the geometry of flat four-dimensional spacetime, derived from the spacetime metric[2] $\eta^{\mu\nu}$, in the anti-commutation relation (3.32).

$$\{\gamma^\mu, \gamma^\nu\} = \gamma^\mu\gamma^\nu + \gamma^\nu\gamma^\mu = 2\eta^{\mu\nu} \tag{3.32}$$

Put simply, (3.32) says that if we combine two sets of γ-matrices in the right way, we get back information about the geometry of spacetime. As such, we can associate each of the four γ-matrices to one axis of spacetime, γ^0 to time and γ^k for $k = 1, 2, 3$ to the three spatial axes. The above relation (3.32) is the defining relation of a particular type of geometric algebra, called a Clifford algebra, which in this case is denoted $\mathcal{C}\ell_{1,3}$.

There is a set of particular Clifford algebras with special relation to various symmetries present in quantum mechanics and relativity theory. A hierarchy of these algebras was presented by Hiley in [29]. This hierarchy includes Clifford algebras generated by the matrices that we have used so far. The algebra generated by the Pauli matrices, the Pauli algebra $\mathcal{C}\ell_{0,3}$ and that generated by the γ-matrices, the Dirac algebra $\mathcal{C}\ell_{1,3}$. It is this step in the hierarchy going from $\mathcal{C}\ell_{0,3}$ to $\mathcal{C}\ell_{1,3}$ that we consider in this section, ending up with the *spacetime* ζ-calculus.

Moving on to the definition of the γ-matrices. There are a number of different representations of the γ-matrices for us to choose from. Among them, the so-called Dirac, Weyl and Majorana representations are perhaps most used. We will

[2] Put simply, the metric is used to measure distances between points in space. From this measurement, however, one obtains further information about the geometry of that particular space.

use the Dirac representation, shown in 3.1, where σ^k denotes a Pauli-matrix for $k = \{1, 2, 3\}$.[3]

$$\gamma^0 = \begin{pmatrix} I & 0 \\ 0 & -I \end{pmatrix} \qquad \gamma^k = \begin{pmatrix} 0 & \sigma^k \\ -\sigma^k & 0 \end{pmatrix} \qquad \gamma^5 = \begin{pmatrix} 0 & I \\ I & 0 \end{pmatrix}$$

Figure 3.1: Dirac representation of γ-matrices

Here we have introduced another matrix, defined as $\gamma^5 := i\gamma^0\gamma^1\gamma^2\gamma^3$. It can also be shown to generate an observable structure. However, because it is definable from the original set, it can either be seen either simply as an abstraction representing this product. Using the matrices listed above, we move on to the following definition.

Definition 3.2.1. (*The spacetime ζ-calculus*) We define a model of the ζ-calculus $\mathcal{Z}(\mathbf{FdHilb}, \{\tau, \xi, \upsilon, \zeta\})$ with the set of observable structures $\flat := \{\tau, \xi, \upsilon, \zeta\}$ shown in figure 3.2.

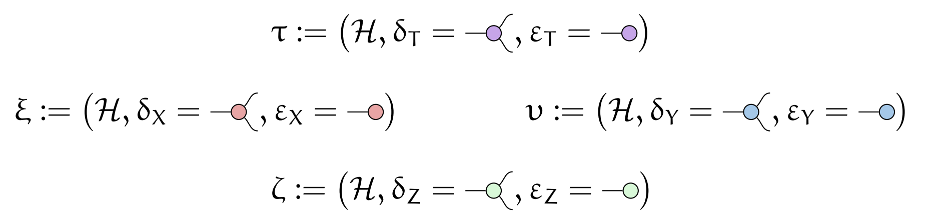

$\tau := (\mathcal{H}, \delta_T =$, $\varepsilon_T =$)
$\xi := (\mathcal{H}, \delta_X =$, $\varepsilon_X =$)
$\upsilon := (\mathcal{H}, \delta_Y =$, $\varepsilon_Y =$)
$\zeta := (\mathcal{H}, \delta_Z =$, $\varepsilon_Z =$)

Figure 3.2: Observable structures of the spacetime ζ-calculus.

Four basis abstractions are introduced by these observable structures, adhering to the usual relations of 1.1.9. When referring to specific bases, the names above will be used and when referring to a general observable structure, we will use ζ^μ with index $\mu \in \{T, X, Y, Z\}$.

We will use the set of four eigenvectors of the γ-matrices to construct the comonoid and counits of the corresponding observable structures shown in equations 3.33 and 3.34.

$$\delta^\mu := |\mu_0\mu_0\rangle\langle\mu_0| + |\mu_1\mu_1\rangle\langle\mu_1| + |\mu_2\mu_2\rangle\langle\mu_2| + |\mu_3\mu_3\rangle\langle\mu_3| \tag{3.33}$$

$$\varepsilon^\mu := \langle\mu_0| + \langle\mu_1| + \langle\mu_2| + \langle\mu_3| \tag{3.34}$$

where $|\mu_i\rangle$ denotes the i-th eigenvector of γ^μ.

[3] The block-matrix representation used above hints at some manner of redundancy for the $\mathbb{C}^{4\times4}$ representation. This can also be seen in the degeneracy of the eigenvalues of the γ-matrices and this is something which will be explored below in discussion about a possible quaternionic representation of these observable structures

From this, we have seen that it is possible to define four observable structures using the eigenvectors of the γ-matrices. As opposed to the observable structures of the quantum ζ-calculus, these are not complementary.

We move on to defining phase groups Λ^{μ} and unbiased points α^{μ}, shown in equations (3.35) and (3.36).

$$\Lambda^{\mu}(\alpha, \beta, \theta) := |\mu_0\rangle\langle\mu_0| + e^{i\alpha}\,|\mu_1\rangle\langle\mu_1| + e^{i\beta}\,|\mu_2\rangle\langle\mu_2| + e^{i\theta}\,|\mu_3\rangle\langle\mu_3| \tag{3.35}$$

$$\alpha^{\mu} := |\mu_0\rangle + e^{i\alpha}\,|\mu_1\rangle + e^{i\beta}\,|\mu_2\rangle + e^{i\theta}\,|\mu_3\rangle \tag{3.36}$$

Each observable structure thus has a phase group parameterised by three angles, generating in the end the full state space. The topology of this space is yet to be understood, although we can recover from the three relative phases that the space must be considerably more complex than that of $\mathbb{C}^2$. A particular subgroup of the total phase space can be constructed similarly to the one made in the two-dimensional case. This is done in a manner appropriate for the structure of the eigenvalues. Specifically, each of the four γ-matrices has two sets of two eigenvectors with degenerate eigenvalues (of ± 1). We interpret this by assigning to each set a common phase, relative to the other set. Following this, the definitions of (3.35) and (3.36) are re-written in equations (3.37) and (3.38).

$$\Lambda^{\mu}(\alpha) := |\mu_0\rangle\langle\mu_0| + |\mu_1\rangle\langle\mu_1| + e^{i\alpha}(|\mu_2\rangle\langle\mu_2| + |\mu_3\rangle\langle\mu_3|) \tag{3.37}$$

$$\alpha := |\mu_0\rangle + |\mu_1\rangle + e^{i\alpha}(|\mu_2\rangle + |\mu_3\rangle) \tag{3.38}$$

This defines a subgroup of the total phase group, which we will take as the canonical way to represent these phase groups..

As in the quantum ζ-calculus, we can derive the generators of the underlying representation of the spacetime ζ-calculus using ζ-terms parameterised by a phase of π (3.39).

$$\gamma^0 := \hat{\tau}^{\pi} \qquad i\gamma^1 := \hat{\xi}^{\pi} \qquad i\gamma^2 := \hat{\upsilon}^{\pi} \qquad i\gamma^3 := \hat{\zeta}^{\pi} \qquad \gamma^5 := \hat{\omega}^{\pi} \tag{3.39}$$

The extra factors in the spatial (i.e. for $\{\xi, \upsilon, \zeta\}$) occur because the eigenvalues of these matrices are imaginary[4], and as such we need an extra global scalar of i.

Before moving on, we must stress the rather strong speculative nature of some of the claims made in the next few paragraphs. We make extrapolations from correspondences found in the quantum ζ-calculus and apply them here, and they should be seen as such naive extrapolations that may fail were they to be projected to further scrutiny. However, as the authors find the interpretations to be of interest, we carry on and explore what can be found from these interpretations.

[4] The spatial γ-matrices are anti-Hermitian.

Previously in the definition of the quantum ζ-calculus, we provided a geometric interpretation of the phase group. We associated the two observable structures ζ, ξ, with the two axes of the Bloch sphere S^2 and interpreted the phase group as denoting rotations around these axes. In the case of four observable structures, we will take the same stance. To each observable structure in ᛒ, we associate an axis of the four-dimensional Bloch sphere S^4 around which the respective phase groups rotate. The Bloch sphere of S^4, as in the quantum case, is also the projective space of a Hopf fibration, in particular the quaternionic Hopf fibration (3.40) that we will look at for the moment.

$$S^3 \hookrightarrow S^7 \rightarrow S^4 \tag{3.40}$$

This can be read "S^7 is fibred over S^4 with fibres homeomorphic to S^3", and interpreted as there being a family of points 'with the topology'[5] of S^3 in S^7 that can be identified with a single point on S^4. With S^4 as our base space, we will attempt to give similar interpretations as in the quantum case. For a general vector $\Psi = (\alpha\ \beta\ \rho\ \sigma)^T \in \mathbb{C}^4$ the normalisability condition reads (3.41).

$$|\alpha|^2 + |\beta|^2 + |\rho|^2 + |\sigma|^2 = 1 \tag{3.41}$$

Which clearly yields the definition for S^7, meaning this is the total space of our base type $\mathbb{C}^4$. Now onto the fibre space. The only possible configuration of complex elements whose normalisation condition yields an equation for S^3 is that of vectors in $\mathbb{C}^2$, however, these cannot multiply vectors in $\mathbb{C}^4$, so we are at an impasse. Let us instead follow the lead of the Hopf fibration and shortly consider the quaternionic case.

For quaternionic γ-matrices in $\mathbb{H}^{2\times 2}$, we get a base type of $\mathbb{H}^2$. The same logic as above follows here, the base space corresponds to S^4 and the total space to S^7. The fibre space S^3, in this case, can be shown to geometrically be a unit quaternion [23], corresponding to the global phase freedom. Thus, for the quaternionic spacetime ζ-calculus, we have a stronger correspondence with the quaternionic Hopf fibration (3.40) than in the complex case.

Quaternions do however have the property of non-commutativity. The effect of this in the setting of **FdHilb** would be that quaternionic scalars affect states or processes differently depending on which side of the state or process they act, meaning that there is some loss in interpretability of string diagrams in an SMC where the objects are quaternionic [11]. We leave it at that for the discussion of quaternionic observable structures for now and explore it further in section 5.1.

As described in Section 1.1.5, we can define an operation that transforms between different observable structures, referred to in the colour-change rule. We

[5] This is somewhat vague. What is meant here by 'with the topology of S^3' is any number or vector whose normalisation condition yields the equation for S^3.

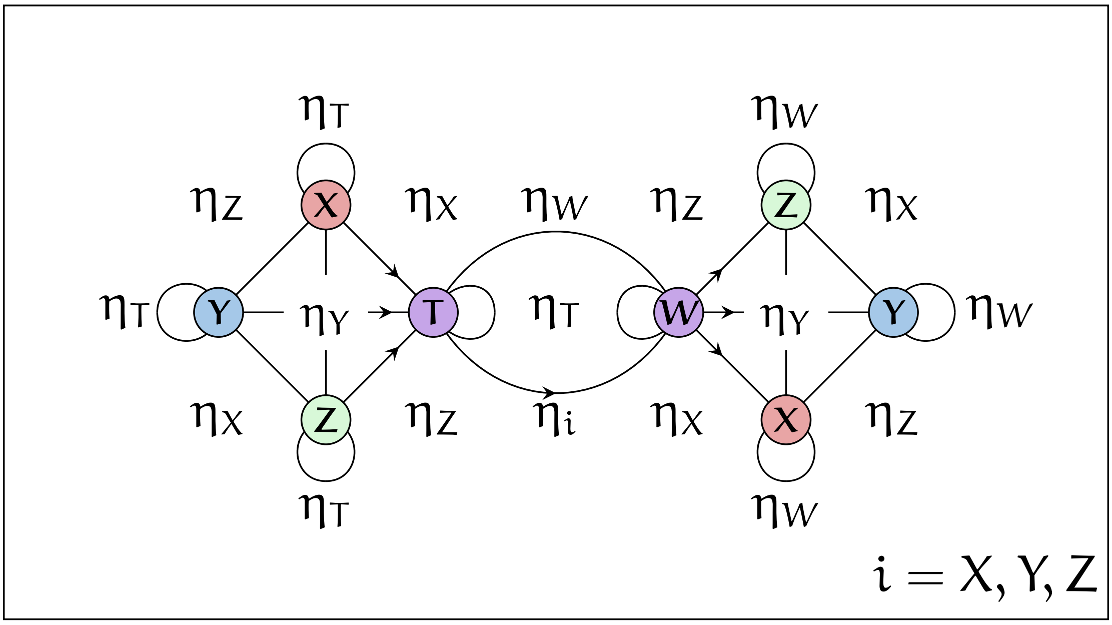


Figure 3.3: Color change crystal for the spacetime calculus.

can define a similar operation in the spacetime calculus to switch between the four (five) observable structures. We call these operators η^μ, for $\mu = 0, 1, 2, 3, 5$[6], with η^0 the identity matrix. The colour-change transformations for the observable structures in the spacetime calculus are visualised in figure 3.3. In this diagram, we interpret the nodes and edges as follows. A node represents the phase group of an observable structure, the edges its η-matrix. They are directed by the order of conjugation of the η-matrices[7]. The colour-change transformations are defined as follows.

$$\eta_\nu^\dagger \circ \Lambda_\mu(\alpha) \circ \eta_\nu = \Lambda_\rho(\alpha) \qquad \rightsquigarrow \tag{3.42}$$

$$\begin{tikzcd}\mu & \rho \arrow[l, "\nu"']\end{tikzcd}$$

$$\eta_\nu \circ \Lambda_\mu(\alpha) \circ \eta_\nu^\dagger = \Lambda_\rho(\alpha) \qquad \rightsquigarrow \tag{3.43}$$

$$\begin{tikzcd}\mu \arrow[r, "\nu"] & \rho\end{tikzcd}$$

$$\eta_\nu^\dagger \circ \Lambda_\mu(\alpha) \circ \eta_\nu = \Lambda_\rho(\alpha) = \eta_\nu \circ \Lambda_\mu(\alpha) \circ \eta_\nu^\dagger \qquad \rightsquigarrow \tag{3.44}$$

$$\begin{tikzcd}\mu \arrow[r, dash, "\nu"] & \rho\end{tikzcd}$$

The inclusion of ω can now be more readily explained. As mentioned there are multiple representations of the γ-matrices, among them there are the Dirac and Weyl representations. These differ only in the choice of the *temporal* basis, choosing γ^0 and γ^5 respectively, leaving the *spatial* bases the same. Including ω enables one to choose freely between these two representations, given by the two lobes in the above figure, by simply switching between the two temporal bases by applying this higher-dimensional version of the colour-change rule. Further, the inclusion of ω is necessitated for the closure of the colour change rule as it has been constructed.

[6] Henceforth, we take this index to run over the five observable structures, including ω.

[7] Note that η^μ is non-Hermitian, meaning $(\eta^\mu)^\dagger \neq \eta^\mu$, why we make the distinction in the order of conjugation.

3.3 Orders of computation

Moving on from the two previous models of the ζ-calculus we will in this section try to complete the pattern outlined by them. The justification for the connection between the quantum and spacetime ζ-calculus is their phase group construction relating to rotations around axes of the projective spaces of the complex and quaternionic Hopf fibrations. Here we will try to complete this pattern. By Adams' theorem [30], the only Hopf fibrations are the real, complex, quaternionic, and octonionic cases, presented in (3.45).

$$\begin{array}{lclclcl} \text{ᚺ}_1 & := & S^0 \hookrightarrow S^1 \to S^1 & \qquad & \text{ᚺ}_2 & := & S^1 \hookrightarrow S^3 \to S^2 \\ \text{ᚺ}_3 & := & S^3 \hookrightarrow S^7 \to S^4 & \qquad & \text{ᚺ}_4 & := & S^7 \hookrightarrow S^{15} \to S^8 \end{array} \tag{3.45}$$

For each Hopf fibration ᚺ_n (*haglaz*[8]) we construct an *order of computation* ᛟ_n (*odal*), a model of the ζ-calculus with observable structures generated as axes for their respective projective spaces. The two orders presented earlier used the generators of the Pauli and Dirac algebras in the Clifford hierarchy as presented by Basil Hiley [29] to construct bases for the respective Hilbert spaces as the set of eigenvectors for the generators. In the quantum ζ-calculus these are the Pauli matrices σ_z and σ_x which generate the observable structures ζ and ξ. For the spacetime ζ-calculus it is the same construction, but for the γ-matrices. Generally, the connection between bases of Hilbert spaces and observable structures is employed as in [16]. For these orders then, we obtain a set of observable structures, together with a corresponding generalised Bloch sphere.

The phase group of each observable structure in the orders is restricted to have one phase, the angle of rotation about the axis defined by the observable structure. We presented the argument for this restriction in the spacetime ζ-calculus in the previous section. We identify a notion of colour change between these phase groups by generalising the action of the Hadamard gate. In quantum computation, the matrix of this gate is defined as $H := |+\rangle\langle 0| + |-\rangle\langle 1|$, mapping the basis vectors from the computational basis to the Hadamard basis. We generalise this construction by mapping from the *standard basis* (the basis $\{|i\rangle\}_i$, as ζ in ᛟ_2 and τ in ᛟ_3) to each of the other bases in the order. The construction of these is presented in (3.46), where β_i is the i-th basis vector.

$$\eta_{\beta\in\mathbb{B}} \quad := \quad \sum_i |\beta_i\rangle\langle i| \tag{3.46}$$

We call these matrices η_β-matrices for each basis β. For the standard basis, these are always the identity matrix.

[8] Hail is the coldest of grain; it is whirled from the vault of heaven and is tossed about by gusts of wind and then it melts into water. (A runic poem for ᚺ)

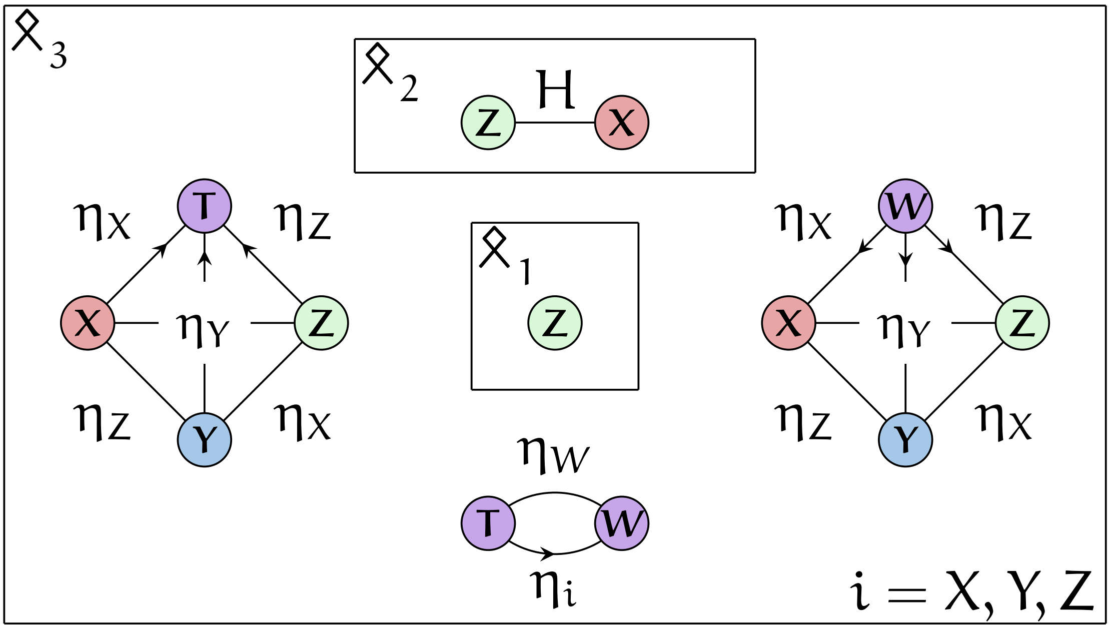


Figure 3.4: The basis crystal for orders up to ᛟ$_3$.

In each of the orders we have explored and presented the relation between the bases by the η-matrices form a completely connected graph between them. We call this graph the basis crystal of the order, where the nodes of the graph correspond to the phase groups of bases, and the edges to the η-matrices. For an edge η connecting the nodes β_1 and β_2, this denotes that $\eta^\dagger \beta_1 \eta = \beta_2$, and vice-versa. If the η-matrix in question is not self-conjugate it might act differently depending on which side of the equation is the adjoint of the matrix. In this case we have marked these edges by a direction (described in detail in the previous section). The basis crystals for orders up to ᛟ$_3$ are presented together in figure 3.4.

As discussed in the previous section the *W*-basis in ᛟ$_3$ is similar to the Y-basis in ᛟ$_2$, in that it is the product of the other bases. With the connection to fermionic quantum field theory, one can choose a representation to work within, where both the T and *W*-bases are time-like such that you pick one of the representations. Because of this we have presented them as separate crystals in the diagram, as they should be interpreted. The η-transforms between the representations are presented at the bottom of the figure. Note the symmetries present in the basis crystal of order ᛟ$_3$, that the edges opposite each other are always the same η-matrix. Note also that the identity edges have been removed from this diagram (compared to figure 3.3), for sake of decluttering.

Notice how the two representations of ᛟ$_3$ are complete graphs with the number of nodes equal to the dimension of the projective space of ᚺ$_3$. Notice also that the shift between the representations (at the bottom of figure 3.4) is the same as the basis crystal of ᛟ$_2$. Looking at the basis crystals like this we see that they come in pairs of basis graphs and representation graphs, where the representation graph

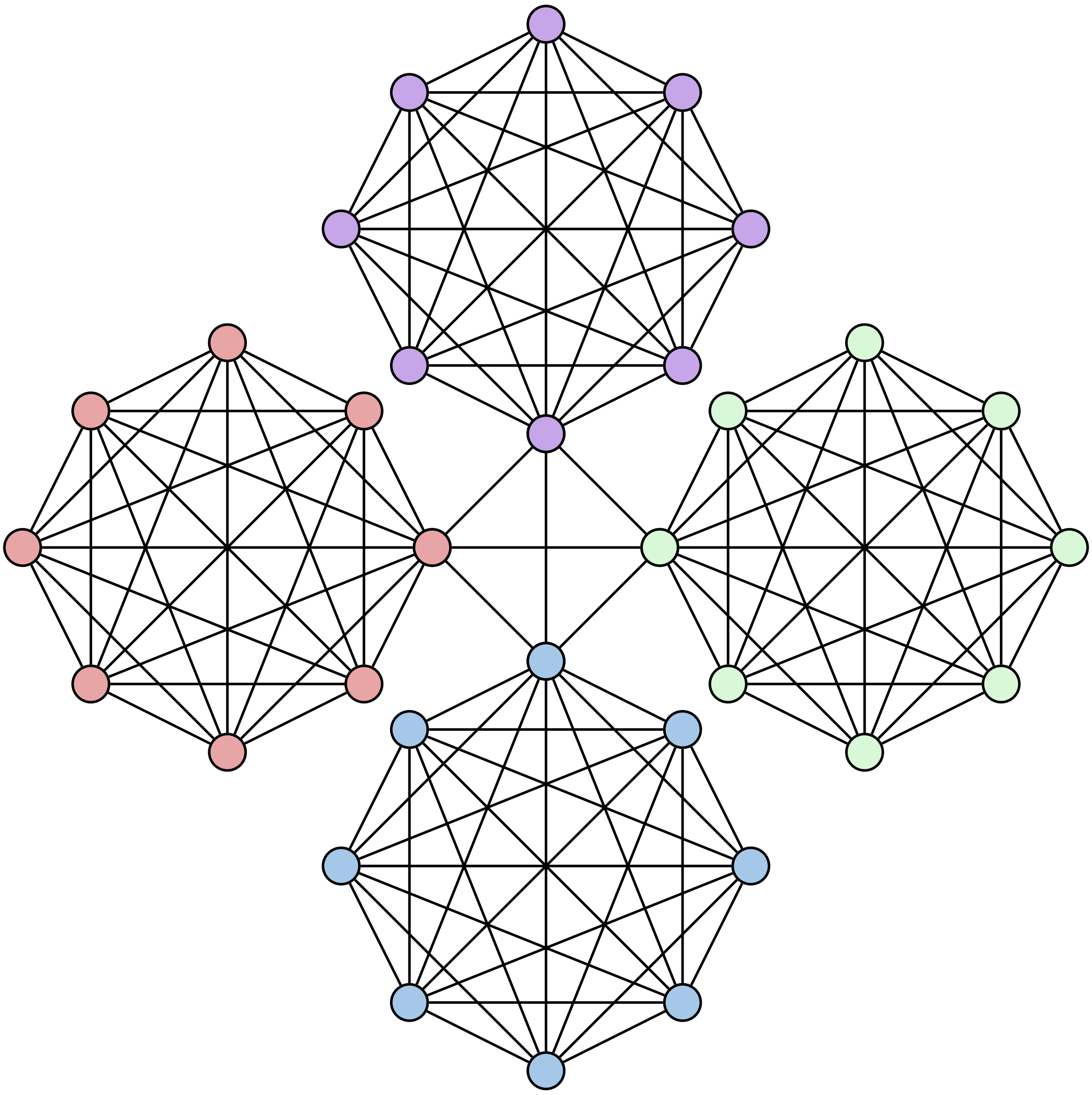


Figure 3.5: Conjectured basis crystal for ᛟ_4.

is the basis graph of the previous order. This also holds for ᛟ_2 where the basis graph is a K_2[9] graph, and the representation graph is the one for ᛟ_1 of one representation. If we continue this pattern to ᛟ_4 we should see a K_8 basis graph, and a K_4 representation graph, producing a full basis crystal on the form in figure 3.5.

We have not investigated a set of observable structures for ᛟ_4 that fulfil the connectivity of this basis crystal, and the argument for its presentation is purely inductive on the structure of the previous ones, wherefore we have not given the nodes any names. The structure of this pattern is interesting, however. It produces a set of 8 observable structures for each representation, where one of the bases is of a different nature than the others. The full set then is 7 constant observable structures (the four lobes of the basis crystal being the same bases), and 4 variable ones that determine the representation (the centre of the basis crystal), totalling

[9] K_n is the complete graph of n nodes.

11 bases. The centre of the basis crystal is the spacetime bases of the previous order, and the 7 others are of some different nature. This setup is intriguing, with regards to the 11 dimensions of some string theories [31, 32], four of them being the 4-dimensional anti-de Sitter spacetime, and 7 of them being extra compact dimensions. We have not investigated this order any more than this, but thought this was worth mentioning here anyway.

With the construction of a model of the ζ-calculus for each of the Hopf fibrations we present the general conjecture on the orders of computation.

Conjecture 1 (Orders of computation)**.** *For each Hopf fibration ᚺ_n there exists a corresponding order of computation ᛟ_n. Each order is a model of ζ-calculus of 2^{n-1} observable structures, each relating to each other by a completely connected graph with the construction of η-transformations. The phase groups of each observable structure in ᛟ_n are rotation operators for the projective space of ᚺ_n.*

We also include the classical order of computation ᛟ_0, as the model of the ζ-calculus defied on the canonical cartesian duplication and discarding. This would be a non-linear λ-basis, which recovers the classical λ-calculus. With this we conclude the discussions on concrete models of the ζ-calculus. The sections other than quantum programming are meant as an exploration of the generality of the definition of the theory. In parts speculative, though mostly meant to showcase the fact that it is possible to apply the theory to topics broader than only quantum computation. Whether or not these further applications have any use we have not been able to definitively answer.

3.4 An analysis of sharing

We will now make a short digression which will be useful later when we discuss the features of the ζ-calculus with regards to sharing. We seek to give an algebraic analysis of the "quantity" of entanglement that is produced by a ζ-term which uses sharing. The main idea here is to convert a ζ-abstraction to a function mapping quantum states to the entanglement entropy of the bipartite state that results from the application of the abstraction on that state.

For a given function using sharing $f : \mathcal{H}_A \to \mathcal{H}_A \otimes \mathcal{H}_A$, the function describing its entropy is $s_E(f) := |\psi\rangle \mapsto S_E[f\,|\psi\rangle]$. Where $S_E : \mathcal{H}_{AB} \to [0,1]$ is the entropy of entanglement for quantum states, defined on the Schmidt decomposition [33] in (3.47).

$$S_E\left[\sum_j \kappa_j\,|u_j\rangle \otimes |\nu_j\rangle\right] := -\sum_j |\kappa_j|^2 \log|\kappa_j|^2 \qquad |u_j\rangle \in \mathcal{H}_A\,,\ |\nu_j\rangle \in \mathcal{H}_B \tag{3.47}$$

The vectors $|u_j\rangle$ and $|\nu_j\rangle$ are the orthonormal bases for the subspaces $\mathcal{H}_A$ and $\mathcal{H}_B$ respectively. We begin by examining the simplest instance of sharing, the ζ-term $\beta^\alpha x\langle x, x\rangle$ for some basis $\beta \in \{\zeta, \xi\}$ in the model $\mathcal{Z}(\mathbf{FdHilb}, \{\zeta, \xi\})$. We write Ʒ^α_β for the Hilbert space interpretation of this term (3.48).

$$\text{Ʒ}^\alpha_\beta := [\![\vdash \beta^\alpha x\langle x, x\rangle : \mathcal{Q} \to \mathcal{Q} \otimes \mathcal{Q}]\!] = |\beta_0\beta_0\rangle\,\langle\beta_0| + e^{i\alpha}\,|\beta_1\beta_1\rangle\,\langle\beta_1| \tag{3.48}$$

We supply a general qubit state presented in Bloch sphere coordinates, written in the same basis as the ζ-abstraction, and calculate the resulting entropy of entanglement in (3.49).

$$\begin{aligned} S_E\left[\text{Ʒ}^\alpha_\beta\left(\cos\tfrac{\theta}{2}\,|\beta_0\rangle + e^{i\phi}\sin\tfrac{\theta}{2}\,|\beta_1\rangle\right)\right] &= S_E\left[\cos\tfrac{\theta}{2}\,|\beta_0\beta_0\rangle + e^{i\alpha}e^{i\phi}\sin\tfrac{\theta}{2}\,|\beta_1\beta_1\rangle\right] \\ &\overset{(3.47)}{=} -\left|\cos\tfrac{\theta}{2}\right|^2\log\left|\cos\tfrac{\theta}{2}\right|^2 - \ \left|e^{i\alpha}e^{i\phi}\sin\tfrac{\theta}{2}\right|^2\log\left|e^{i\alpha}e^{i\phi}\sin\tfrac{\theta}{2}\right|^2 \\ &= -\left|\cos\tfrac{\theta}{2}\right|^2\log\left|\cos\tfrac{\theta}{2}\right|^2 - \ \left(\left|e^{i(\phi+\alpha)}\right|\left|\sin\tfrac{\theta}{2}\right|\right)^2\log\left(\left|e^{i(\phi+\alpha)}\right|\left|\sin\tfrac{\theta}{2}\right|\right)^2 \\ &= -\left|\cos\tfrac{\theta}{2}\right|^2\log\left|\cos\tfrac{\theta}{2}\right|^2 - \ \left|\sin\tfrac{\theta}{2}\right|^2\log\left|\sin\tfrac{\theta}{2}\right|^2 \\ &= -\cos^2\left(\tfrac{\theta}{2}\right)\log\left(\cos^2\left(\tfrac{\theta}{2}\right)\right) - \ \sin^2\left(\tfrac{\theta}{2}\right)\log\left(\sin^2\left(\tfrac{\theta}{2}\right)\right) \end{aligned} \tag{3.49}$$

From this, we obtain the function describing the entanglement entropy of like-basis sharing (3.50).

$$s_E\text{Ʒ}^\alpha_\beta = \cos\tfrac{\theta}{2}\,|\beta_0\rangle + e^{i\phi}\sin\tfrac{\theta}{2}\,|\beta_1\rangle \mapsto -\cos^2\tfrac{\theta}{2}\log\left(\cos^2\tfrac{\theta}{2}\right) - \ \sin^2\tfrac{\theta}{2}\log\left(\sin^2\tfrac{\theta}{2}\right) \tag{3.50}$$

Since the phases α and ϕ have no effect on this entropy measure, we can write a simpler function $s_E(\theta) := -\cos^2\left(\frac{\theta}{2}\right)\log\left(\cos^2\left(\frac{\theta}{2}\right)\right) - \sin^2\left(\frac{\theta}{2}\right)\log\left(\sin^2\left(\frac{\theta}{2}\right)\right)$.

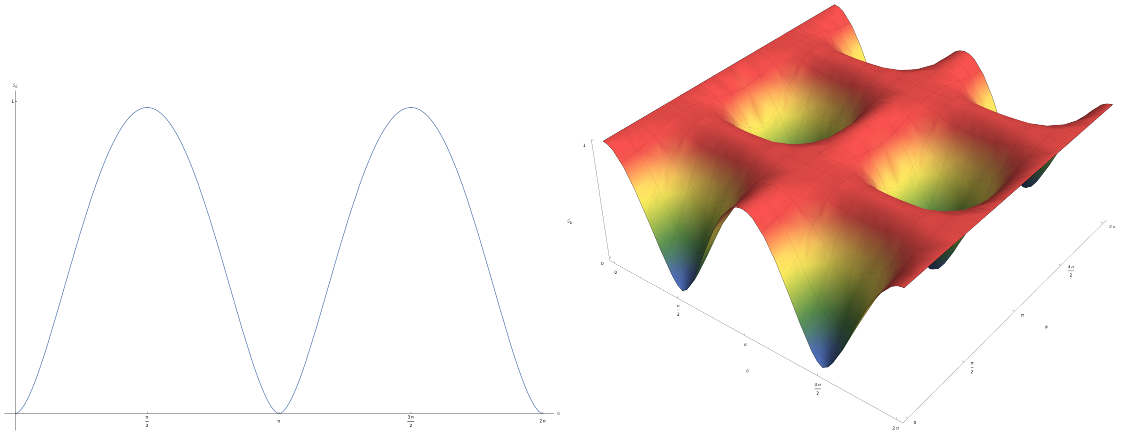
If we instead write the general qubit state in some basis χ dual to the basis of sharing δ, we obtain the function for the entropy of entanglement in (3.51).

$$S_E\left[\mathfrak{Z}_\delta^\alpha\left(\cos\tfrac{\theta}{2}|\chi_0\rangle + e^{i\phi}\sin\tfrac{\theta}{2}|\chi_1\rangle\right)\right] = -\sum_{\eta\in\Phi^\pm}|\eta|^2\log|\eta|^2 \qquad \Phi^\pm = \cos\tfrac{\theta}{2} \pm e^{i\phi}\sin\tfrac{\theta}{2} \tag{3.51}$$

Where, once again, the measure of entanglement is independent of the phase of the sharing ζ-abstraction. The relative phase of the general quantum state does have an effect on the entropy produced in this case of dual-basis sharing. We describe the function of the entropy of entanglement produced by dual-basis sharing in (3.52).

$$s_E^*(\theta,\phi) := -\sum_{\eta\in\Phi^\pm}|\eta|^2\log|\eta|^2 \qquad \Phi^\pm = \cos\tfrac{\theta}{2} \pm e^{i\phi}\sin\tfrac{\theta}{2} \tag{3.52}$$

Thus, we have produced two functions $s_E(\theta)$ and $s_E^*(\theta,\phi)$ for the entropy of entanglement produced by like-basis and dual-basis sharing. We give the graphs of these functions in figure 3.6, depending on the amplitude θ and, in the dual-basis case, phase ϕ of a general quantum state.



(a) Like-basis sharing.

(b) Dual-basis sharing.

Figure 3.6: Sharing analysis.

This small digression is intended to be a small result which may be useful when discussing the effects of sharing in different bases, instead of simply sharing in some preferred basis. We will discuss this result in the conclusion of the thesis, for now it is simply an investigation of the effects of sharing.

4

Philosophical Investigations

> Mit jeder epochemachenden Entdeckung
> schon auf naturwissenschaftlichem Gebiet
> mußte [der Materialismus] seine Form ändern.
>
> *F. Engels*

In *Ludwig Feuerbach and the end of classical German philosophy* [34], from which this chapter's epigraph is quoted, Engels lays out the way in which the naive materialism of 18th century scientists grew to its mature philosophical form out of the philosophy of Hegel. He specifically criticises the critique of Christianity by Feuerbach, one of the young Hegelians. He accuses Feuerbach of confusing the emerging materialist worldview with purely mechanical science. The limitation of which is its inability to describe a world in ever-changing historical development, something which is not surprising considering the «metaphysical and antidialectical» philosophical frameworks of 18th century science, according to Engels. Examples of this are the, now superseded, theories of Newtonian mechanics, phlogiston[1] chemistry, and pre-Darwinian biology. Though the world was described as being in constant motion, this motion was mechanical and deterministic, stuck in an eternal cycle. Compared to the dialectical view of constant motion in context of its historical development.

Contrasting the epistemological theories of knowledge being driven by pure reason alone, Engels posits that our knowledge is intimately tied to developments of experimental science and industry; hand-in-hand with the developments in the means of production, that is. This is evident, of course, in that the validity of science lies not in its ingenuity or beauty, but rather in experimental verification. The ability of experimental verification then, is limited by the larger process of developments of production, materials, and technology. The limits of knowledge then, scientific or otherwise, are limited by this larger process. From this, Engels draws the conclusion that materialist philosophy, the philosophy of the external reality being prior to our consciousness in it, must change its form in pace with revolutionary developments in science, our verified knowledge of the material. In this chapter, we will present a short exploration of what the scientific underpinnings

[1] The theory that all combustible substances contained some hypothetical element called *phlogiston*.

of our thesis implies, and in this light, try to venerate a materialist philosophy in the wake of ”real” matter.

We will begin by making our position on the foundations and philosophy of quantum physics clear, outlining the *relational interpretation* as proposed by Carlo Rovelli. Then, we will highlight the ontological process of this interpretation and show that it is already established in the works of other philosophers. Principally, we will explore the philosophy of language of the late Ludwig Wittgenstein, as well as the theory of value of Karl Marx. The goal is not to argue for these concepts on their own, but rather explore the larger process of establishing properties of our shared experience by way of analysis of context and relation. This stands in opposition to philosophy which regards objects having intrinsic properties of their own.

Before embarking on this exploration we wish to make our philosophical language clear, and specifically the materialism-idealism distinction we will discuss here. As described in [34], the historical roots of this distinction (of great importance to medieval scholasticism) relate to the relationship of *thinking* and *being*, and of the prerequisite of a creator. This creator need not refer to a creator god, per se, and could also refer to more convoluted explanations of the substance prior to matter, this could be reason, consciousness, or idea, for example. Engels explains the distinction in the following way.

> Frågan om förhållandet mellan tänkandet och varandet, frågan: vilket är det primära, anden eller naturen, denna fråga tillspetsades gentemot kyrkan till att lyda: har gud skapat världen eller har världen evigt existerat? Allteftersom denna fråga besvarades, delade sig filosoferna i två stora läger. De som hävdade, att anden existerade före naturen och som således sist och slutgligen förutsatte en världskapare av ett eller annat slag bildade idealismens läger. De andra, som betraktade naturen som det primära, tillhör materialismens olika skolor.[2]

This is the distinction we will also employ, whether or not some phenomenon requires something external and prior to observable and verifiable nature. This we shall call *idealism*, especially when one appeals to the human consciousness as an integral part of their explanation of reality. Note that we do not put any moral weight on either of the camps, even though we do claim that materialism is the only framework capable of producing any knowledge compatible with a scientific worldview.

[2] The question of the position of thinking in relation to being, the question: which is primary, spirit or nature, that question, in relation to the church, was sharpened into this: Did God create the world or has the world been in existence eternally? The answers which the philosophers gave to this question split them into two great camps. Those who asserted the primacy of spirit to nature and, therefore, in the last instance, assumed world creation in some form or other comprised the camp of idealism. The others, who regarded nature as primary, belong to the various schools of materialism.

4.1 The relational interpretation of quantum physics

The relational interpretation of quantum mechanics [35], or RQM, is one where the ontological weight is put solely on what is observable, disregarding notions relating to what cannot be ultimately observed. With regard to the relativism of observers and observer-dependant values, Rovelli states that (i) all observers are equivalent, that (ii) the result of a measurement of a system is dependent on what other system is doing the measurement where (iii) the value of an observable of a system is determined only at measurement relative to another system.

With regard to (i), the statement covers the supposition that there is no observer whose observation is privileged from that of any other observer. This view of measurement stands in opposition to that taken in the Copenhagen interpretation, explained by John Wheeler in [36].

> It associates a state function with the system under study — as for example a particle — but not with the ultimate observing equipment. [..] The ultimate observing equipment still lies outside the system that is treated by a wave equation.

While the interaction of observer-observed is naturally to be taken as a relation between these two objects, the observer carrying out the measurement is taken as being external to the system under observation. The relational aspect of this type of measurement is extended in RQM to the interactions of any systems. To this, Rovelli states the following. [35]

> Any physical system can play the role of the "Copenhagen observer", but only for the facts defined with respect to itself. From this perspective, RQM is nothing else than a minimal extension of the textbook Copenhagen interpretation, based on the realisation that any physical system can play the role of the "observer" and any interaction can play the role of a "measurement"

Thus, in RQM, there is no observer-observed distinction, in this sense. This brings us to the second point which is a statement on the dependence of the value of a measurement on which system is doing the measurement. Consider for example the interaction of a measurement apparatus $S'(\phi)$ angled at some degree ϕ with relation to the stream of particles, with an as-yet undetermined spin-component A of a system S of a set of spin-$\frac{1}{2}$ particles. The value of A depends on which particular instance of the system S' determined by ϕ it is that S interacts with.

This relational aspect is clearly inherent to quantum theory, realised in RQM in terms of *facts*, that is statements about the value of an observable relative to the interacting systems. Rovelli describes quantum mechanics as a theory about facts, that is statements about the value that observables take relative to the systems of an interaction. Written as follows.

> Facts are relative to the systems that interact. That is, they are labelled by the interacting systems. This is the core idea of RQM. It gives a general and precise formulation to the central feature of quantum theory, on which Bohr correctly long insisted: contextuality.

The value of an observable taken in an interaction is thus in RQM taken to be inherently dependent on the context in which it appears, that is in relation to which system the value is actualised. About facts, he further notes the following. [35]

> Facts are sparse: they are realised only at the interactions between (any) two physical systems. This is the key physical insight in Heisenberg's seminal paper and a basic assumption of RQM.

This brings us to the third note about RQM. As noted above, RQM posits that a fact about the value of an observable is determined only at an interaction between two systems and furthermore what value is taken depends on which systems are interacting. One conclusion to be drawn from this is that any possible values taken are a property of the interactions and not of any particular quantum states by themselves. This begs the question of what occurs in between the interactions that define the determinate values taken in those interactions. In [37], Villars considers a question which brings to light the fully relational aspect of RQM. It concerns the position of a particle in between measurements. He asks the following.

> What is the value of a microphysical object's position-defining interaction between position-defining interactions?', the answer, 'In this case it does not have a value', makes sense; where no interaction of the appropriate kind is occurring, the corresponding observable cannot have a value.

This rings close to the statement of RQM with regard to the well-definedness of values of observables in between measurements. Such a question is meaningless in RQM. A stance is taken against the usual descriptions of quantum mechanics wherein a state in between measurements in the form of a wave function ψ is considered. As noted by Rovelli, ψ is to be seen merely as a bookkeeping device of known facts about a system from interactions with another system, not something that is inherent to the system itself.

Of note in our exploration on the philosophical implications of RQM is the Bogdanov-Lenin polemic, as pointed out by Rovelli in his book *Helgoland* [38]. This is a polemic between the two Bolsheviks, Alexander Bogdanov and Vladimir Lenin. In his book *Empiriomonism* [39], Bogdanov proposes a materialist philosophy drawing inspiration from the works of the Austrian physicist Ernst Mach. This is of great interest here since, as Rovelli writes in Helgoland, Mach had served as inspiration to both Einstein and Heisenberg in the foundations of both quantum mechanics and the theory of relativity. Partly in response to Bogdanov's work, Lenin published *Materialism and Empirio-Criticism* [40], where he passionately criticises Bogdanov and the "Machists" in his characteristic polemical style.

We highlight here a quote from Lenin's work, from the discussion on the "disappearance of matter".

> Ty den *enda* "egenskap" hos materien, vid vars erkännande den filosofiska materialismen är bunden, är dess egenskap att *vara en objektiv realitet*, att existera utanför vårt medvetande.[3]

This brings up a contentious point with regard to relational quantum mechanics, that of objective reality. With regards to realism, Rovelli distinguishes between different forms of the term, the weak and the strong sense [41]. Regarding the weak sense of the term, Rovelli states the following.

> 'Realism' is a term used with different meanings. Its weak meaning is the assumption that there is a world outside our mind, which exists independently from our perceptions, beliefs or thoughts. Relational QM is compatible with realism in this weak sense. "Out there" there are plenty of physical systems interacting among themselves and about which we can get reliable knowledge by interacting with them; there are plenty of variables taking values, and so on. There is nothing similar to 'mind' required to make sense of the theory.

Thus, relational quantum mechanics is indeed compatible with the criterion of matter exposed by Lenin, as far as the least sense of which, weak realism, is concerned. This is, however, not enough since this definition of materialism is rather loose. What is needed is to examine the conclusions drawn from materialist philosophy and review them in light of the description of reality that RQM provides. With this in mind we now look at the strong sense of realism, with which RQM is incompatible.

> Relational QM is anti-realist about the wave function, but is realist about quantum events, systems, interactions... It maintains that "space is blue and birds fly through it" and space and birds can be constituted by molecules, particles, fields, or whatever. What it denies is the utility –even the coherence– of thinking that all this is made up by some underlying ψ entity. But there is a stronger meaning of 'realism': to assume that it is in principle possible to list all the features of the world, all the values of all variables describing it at some fundamental level, at each moment of continuous time, as is the case in classical mechanics. This is not possible in relational QM.

What this stance of the relational interpretation implies for materialism requires substantial investigation, much more than we are able to expound in these

[3] For the only property of matter, with which philosophical materialism is bound to the proclamation thereof, is its property of being an objective reality, to exist outside of our consciousness.

short pages. A close examination of the works cited here, and the relation of the philosophical frameworks they outline, is work for the future. Here we wish to simply state the surface level connections between the frameworks.

The final quote from Lenin's work that we wish to bring up with regard to matter and objects is as follows.

> Tingens "väsen" eller "substans" är också *relativa*; de uttrycker endast djupet av den mänskliga kunskapen om objekten, och om denna kunskap igår inte sträckte sig längre än till atomen och idag inte når längre än till elektronen och etern, så hävdar den dialektiska materialismen att alla dessa *milstolpar* för den framåtskridande mänskliga vetenskapens kunskap om naturen är temporära, relativa, ungefärliga.[4]

We will forgo the confusion of the words used by Lenin here in the context of modern science, Wittgenstein will have something to say about this whole endeavour of mixing language games anyway... This quote is interesting in the context of the anti-monist (monism being the view that all existing things can be explained by a unified substance) stance that has been ascribed to the relational interpretation: there is no meaning to 'the wave-function of the universe'. If the view of substance is that it is of an explanatory nature, a collection of facts with which we explain events, rather than being ascribed as the constituent of objects, then one could indeed describe Lenin's stance as anti-monist. This is not what is usually ascribed to materialism, however. The naive view of materialism is that it claims precisely that reality is materially monist, that nothing exists other than material objects moving through space and time. In light of the nonexistence of substance inherent in objects themselves, a *midenist* (Greek. *μηδέν*, nothing) position, we would like to reframe this naive materialism. The term *śūnyatā* is also fitting here, referring to the Buddhist philosopher Nāgārjuna whom Rovelli also mentions in Helgoland [38]. In this view, the claim of materialism is rather on the causal relationship between the natural world (matter) and the position of our consciousness in it. This reframing of materialism, examining what the conclusions of the framework rely on, and stating this with nature in relation to itself, is precisely what is needed for materialism to be in accordance with the revelations of quantum theory in the description of the relational interpretation. Further, this midenist materialism has implications for the existence of individual consciousness, and the assumptions of individualism. These aspects of our experience would have to have a solely social-relational character, mediated by something, a language perhaps.

[4] The nature or substance of objects is also relative; they only express the depth of the human knowledge of the objects, and if this knowledge as of yesterday was limited to the atom and today not any further than the electron and the æther, then dialectical materialism only claims that all of these milestones of the progressing human scientific knowledge of nature are temporary, relative, and approximate.

4.2 Wittgenstein: language games and meaning

In *Philosophical Investigations* [42] Wittgenstein introduces a multitude of concepts relevant to our discussion here. Principally, we are concerned with the change of perspective in the philosophy of language presented in his work. This is the shift from the view that the meaning of language is described by words being representative of perceived objects, either external to or internal sensations of a subject, to the view that meaning is best described by the social use of words in the context of where they are used. Wittgenstein also has a specific argument against the language used for sensations internal to the subject, what he calls *private language* [43], which we will not cover here. Wittgenstein explains this perspective, often called *meaning-as-use* (though not by him), in the following way.

> För en *stor* klass av fall i vilka ordet ”betydelse” används -om än inte för *alla* fall där det används- kan man förklara detta ord så: Ett ords betydelse är dess bruk i språket. Och ett namns *betydelse* förklarar man ofta genom att man pekar på dess *bärare*.[5] [6] (§ 43)

Then, to give a description of the function of language in use, Wittgenstein oftentimes brings up the concept of a *language game*. This concept, as well as the other multitude of concepts used by Wittgenstein are never precisely defined anywhere in the book, but rather repeatedly revisited among the thousand statements of the work. This however, might be a symptom of trying to state the ineffable, and a reflection of the view of language he is trying to portray. It does make defining these concepts in a concise way here, quite difficult though.

In § 23, Wittgenstein lists a number of things to be considered as language games, including «Giving orders, and acting on orders - Describing an object by appearance - Recalling a sequence of events - Guessing riddles - Writing and telling a story». What is the common factor here, the unifying definition of language as language games? Nothing! At least nothing precise which we could call the defining feature which is the *essence* of what a game is (or language at that). Rather they are «related to each other in many different ways» (§ 65), they are defined by the web of relations in which they appear in use. This is precisely the *relational* aspect we wish to highlight, that the meaning of language is inherent in neither the speaker nor the objects to which they supposedly refer to. It is a game of relations, social, between people and their shared environment. Our language, and the advanced social systems we build in it, is material in nature. It is a collective navigation of our shared material reality, it loses all meaning without the material. This is where we hope that the connections to both Rovelli, Lenin, and

[5] *Philosophical Investigations* is written as a collection of numbered statements, here we will refer to statement n as § n.

[6] For a large class of cases where the word ”meaning” is employed -if not for all cases- it can be explained as such: The meaning of a word is its use in language. And the meaning of a name is often explained by pointing to the bearer of that name.

Engels are clear. In the ontology of properties, the criterion of materiality, and materialist epistemology, all stating this simple fact of our relational-material reality.

We are not the only ones who have made this connection, as we are inspired by the brilliant philosophical simplicity of Rovelli, and the connections he has already made. Other examples of the connection between Rovelli and Wittgenstein being made are [44], which also includes the similarity of both Wittgenstein and relational quantum mechanics to early Buddhist philosophers (something which Rovelli also mentions in Helgoland).

With regards to the inclusion of Wittgenstein as a materialist philosopher, we turn to Louis Althusser and *The Underground Current of the Materialism of the Encounter* [45]. Though his inclusion of Wittgenstein in this underground current of materialism is underexplored (in the opinion of the authors), we wish to highlight the similarity of Althusser's focus on the *encounter* and Rovelli's ontology of *events*. Althusser also calls this current *the materialism of the rain*, referring to Epicurus *clinamen* (infinitesimal swerve) of the parallel rain of atoms in the genesis. This clinamen causes the rain of atoms to collide, creating our world in the first *encounter*, the first event. As Althusser puts it.

> The clinamen is an infinitesimal *swerve*, 'as small as possible'; 'no one knows where, or when, or how' it occurs, or what causes an atom to 'swerve' from its vertical fall in the void, and, breaking the parallelism in an almost negligible way at one point, induce *an encounter* with the atom next to it, and, from encounter to encounter, a pile-up and the birth of a world - that is to say, of the agglomeration of atoms induced, in a chain reaction, by the initial swerve and encounter.

Althusser includes a great number of other philosophers in this current, including Democritus, Machiavelli, Spinoza, Hobbes, Rousseau, Montesquieu, and Heidegger. Whether or not these philosophers can rightly be called materialists, in the sense we outlined earlier, is up to further exploration. Moreover, the conclusion that Althusser draws from his description of this current is *anti-dialectial*, instead being something he coins as *aleatory* (dependant on chance). This is not a stance we give credence to. Even if the origin of, or our knowledge of, the world is in some sense probabilistic, why would that entail that the historical progress of it should be indescribable by us? The movement of history, and of nature, determined by the contradictions which give rise to it, is still observable and describable. When Althusser sees the inability of naive materialism to describe certain events, overdetermined or not, shouldn't he then, as a materialist, question his very knowledge of matter, instead of its historically determined motion? Materialism is indeed pliable, as we outline in the beginning of this chapter, and contingent on the resolution of nature we have access to. We humbly put forth that an investigation into the implications of modern science on materialist philosophy is needed, instead of discrediting dialectics by means of reference to the unknowable genesis, or the works of great philosophers.

4.3 Marx: commodity exchange and value

Here we will provide yet another example of the relational thinking we have built upon so far, the analysis of the commodity by Karl Marx. We will cover here specifically the theory of value as proposed by Marx in the first chapter of *Capital* [46]. The focus will be on value as something relational, not inherent to commodities themselves. To begin, we have to define what Marx calls «the dual nature of commodities». Firstly, commodities have value on the *natural form*, the natural properties of commodities making them useful in some sense (in relation to human need, that is), this is called *use-value*. This form of value is not quantitative in the sense we usually mean by value; answering the question ″what is an apple worth″ with ″my hunger″ is indeed unexpected... Instead, the *value-form* of a commodity is quantitative in this sense, answering the previous question in a proportion of exchange, for example ″an apple is worth two pears″. This is the *exchange-value*, a relation of exchanging use-values. As put by Marx.

> Exchange-value appears first of all as the quantitative relation, the proportion, in which use-values of one kind exchange for use-values of another kind. This relation changes constantly with time and place. Hence exchange-value appears to be something accidental and purely relative, and consequently an intrinsic value, i.e. an exchange-value that is inseparably connected with the commodity, inherent in it, seems a contradiction in terms.

Again, the point of the discussion of these seemingly separate ideas is not a statement of the concepts being the same, but rather an exposition of the *ontology of relations*, the works of philosophers prior to the interpretations of quantum mechanics making use of this mode of analysis that is present in the relational interpretation. As we put forth in the previous section, when the properties we are investigating are concerned with humans, as in language, we do not need to refer to individual consciousness in any way, but focus instead on the social activity of humans. This leads us to a collective, sociological analysis of the concepts, not an idealist or metaphysical one. This is also true here, in the property of value in commodities. Marx explains the social nature of value as follows.

> However, let us remember that commodities possess an objective character as values only in so far as they are all expressions of identical social substance, human labour, that their objective character as values is therefore purely social. From this it follows self-evidently that it can only appear in the social relation between commodity and commodity

Thus, what we see as central to the process of analysis we are pointing out is: that whatever the object of analysis, we have to move the properties we are ascribing it from the thing in itself to all the objects it manifests itself to, that it is in relation to.

4.4 Concluding remarks

In this chapter, we have made clear the relational interpretation of quantum mechanics, and connected its underlying ontology of relations to the works of Wittgenstein and Marx. With the form and intention of materialism having been made evident, we have proposed a direction of investigation for this philosophical current, that as proposed by this chapter's epigraph, materialism has to change and evolve with the developments of science. We put forth a unifying trend in this direction, as described by the relational interpretation, a mode of analysis that moves the properties of objects to their relations in the material world. In this sense, we intend to defend this analysis as a purely material one. As mentioned by Rovelli, we also see that the long tradition of ascribing any ontological weight to the wave function of quantum mechanics as a metaphysical (and idealist) stance. In general, this relational analysis, when applied to concepts related to humans, suggests a social character, as we explored in the philosophy of language and theory of value. With the works of Bogdanov not being explored here we wish to suggest what directions this could take.

For Bogdanov, the question of how the material world interacts with our cognition is central. The relational mode of analysis would place the home of discussion of things such as consciousness in the social realm, instead of the neurophysiological one. This direction of cognition could also include the critique of Bogdanov by Ilyenkov[7] [47], and the social theory of consciousness of the Vygotsky circle. Though not of any relevance to the scientific works of the authors of this thesis, we wish to leave it as a point of further exploration, especially with the current developments in cognitive science (predictive processing, for example).

To conclude, we hope that this philosophical investigation can serve as a natural part of the thesis, and not purely as a digression. The point in the flow of the thesis is precisely to make our understanding of the subject we are writing about clear, including its implications and larger mode of analysis. It is the conclusive view of the authors that philosophy should and needs to play a larger role in the everyday work of scientists, if we are to develop our tools any further. The confusion surrounding quantum mechanics as a whole could at least in some part be mediated by the therapeutic use of philosophy.

And yes, the authors are aware of the seeming contradiction of proposing a general mode of analysis while also including the philosophy of the late Wittgenstein in it. The mixing of language games is abundant, especially regarding the word *relation*. We leave it up to our future selves, and the readers, to sort this out. For now, it stands only as a humble suggestion.

[7] We thank Johannes of AlltÅtAlla for making this thinker known to us.

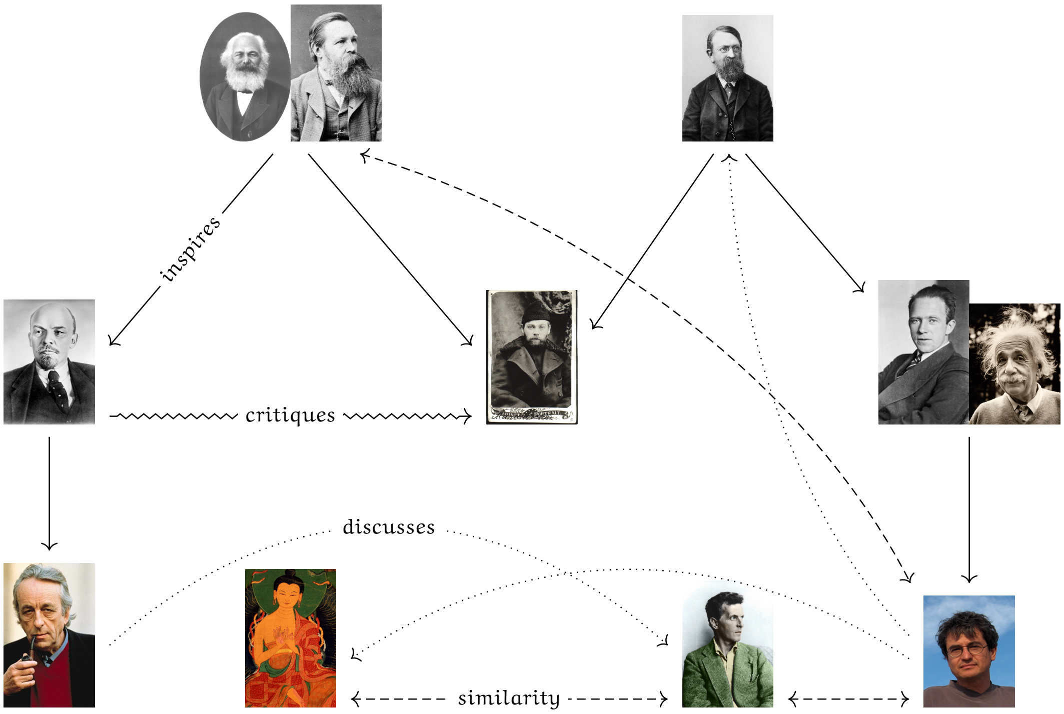


Figure 4.1: The philosophy chapter, graphically.

5
Conclusion

Bien sûr, il n'est rien besoin de dire
À l'horizontale
Mais on ne trouve plus rien à se dire
À la verticale
Alors pour tuer le temps
Entre l'amour et l'amour
J'prends l'journal et mon stylo
Et je remplis et les A et les O

S. Gainsbourg

FINALLY, the rather abstract theory that is the ζ-calculus is defined and applied. The authors hope that the short excursion into philosophy can serve as a palate cleanser of sorts. In this chapter, we will discuss the novel features of the theory in more detail. We will begin with a general discussion of the contributions of the theory of the ζ-calculus, before moving on to specific points from the different models of the theory which were presented in the applications. We will leave the chapter on philosophy where it stands. The philosophical investigations serve mainly as a collection of the philosophical roots of the intersecting fields of the thesis, a side step in congruence with, though not part of the theoretical contributions. Thus, we will leave it out of the discussion here.

We will try to justify, and present the weaknesses, of using the ζ-calculus as a quantum programming language. This will also include comparisons with other contemporary quantum programming languages, contrasting the rather unique paradigm of the quantum ζ-calculus with the more common approaches. Then we continue the discussion of the spacetime ζ-calculus, where we try to reason about what exactly a spacetime programming language is, since no spacetime computers exist (yet?). The discussion of these models as programming languages, versus viewing them as computational physical theories, is an important point here.

And at last, we will try to draw a conclusion from the varied directions that the thesis has taken, trying to present the theory, its applications, and the philosophical justifications as a unified whole. The direction we wish to take here diverges from the view of the theory as one of programming. Then, we suggest some directions for the future. Trying to expose the flaws with the work as we see them, and what can be worked on to mend these.

5.1 Discussion

Now onto discussing, how interesting... We will try to stay focused on what matters here, what have we done, what are the strengths, and most importantly the flaws. To begin, we walk through the models of the ζ-calculus presented in the chapter on applications.

5.1.1 Quantum programming

In section 3.1 we introduced the quantum ζ-calculus $\mathscr{Z}(\mathbf{FdHilb}, \{\zeta, \xi\})$, a model of the theory intended –in some sense– to be a quantum programming language. Though this term, programming language, might not be entirely fitting for the version we presented in this thesis. In the sense that it allows for writing programs that are able to be executed by a quantum computer, it does fit the label. It does not, however, include many of the features one might expect for this use. Explicit control structures are one such feature, present in various other quantum programming languages. We usually distinguish two forms of this construction, *classical* and *quantum* control. A classical conditional usually depends either on some classical bit or the measurement of a quantum one. A prominent example featuring classical control from the early days of quantum programming is the quantum λ-calculus of Selinger and Valiron [48]. This language, featuring a linear type system with exponentials (types which are allowed to be duplicated) for classical data, has such a classical conditional. Its type rule is presented in (5.1).

$$\frac{\Gamma_1, !\Delta \vdash P : \mathrm{bit} \qquad \Gamma_2, !\Delta \vdash M : A \qquad \Gamma_2, !\Delta \vdash N : A}{\Gamma_1, \Gamma_2, !\Delta \vdash \text{if } P \text{ then } M \text{ else } N : A} \tag{5.1}$$

The reason for including classical conditionals is to have some control flow dependent on the measurement of qubits, which is useful though not for reasons of pure quantum programming. Quantum conditionals, however, are harder to implement. An example of a language which features this construct is QML [27], which depending on the different iterations of the language, features either a quantum if-expression or case-expression. The typing rule for the quantum if-expression, where $\mathcal{Q}$ is the qubit type, is presented in (5.2).

$$\frac{\Gamma \vdash^{a} P : \mathcal{Q} \qquad \Delta \vdash^{\circ} M : A \qquad \Delta \vdash^{\circ} N : A \qquad M \perp N}{\Gamma, \Delta \vdash^{a} \text{if}^{\circ} \text{ } P \text{ then } M \text{ else } N : A} \tag{5.2}$$

Two things are of note here, the notation $\vdash^{a}$ where $a \in \{\circ, -\}$ denotes whether or not a judgement is *strict* (containing no discarding) or not, respectively. The other condition being the *orthogonality judgement* on terms $M \perp N$. Both of these conditions require a large formal machinery (and are therefore implementation heavy, see [49] for implementation of these constructs) to verify. The reason for including this construct is more related to quantum programming than the classical one, for this construct allows for programs to be run in superposition. If the control term (P in the case of (5.2)) is in superposition, then the output would be too. This allows, for example, for many quantum gates to be implemented directly in the language instead of relying on a predefined gate set (as in Selinger and Valiron's quantum λ-calculus). An example implementation of the quantum not (σ_x) gate in QML compared to the ζ-calculus is presented in (5.3).

$$\begin{aligned} &\mathrm{qnot} : \mathcal{Q} \multimap \mathcal{Q} \\ &\mathrm{qnot}\ x \ = \ \mathrm{if}^{\circ}\ x \text{ then } |0\rangle \text{ else } |1\rangle \qquad\qquad \mathrm{qnot} :\equiv \xi^{\pi}xx \end{aligned} \tag{5.3}$$

It is the opinion of the authors that this is an important feature of any quantum programming language. Relying on a predefined gate set often leads to a style of quantum programming that looks more like a specification language for quantum circuits. In a functional language, this tends to look like an approximation of imperative programming with stacks of let-expressions. An example of a language which suffers from this weakness is Quipper [20]. As mentioned briefly before, the authors feel that the quantum circuit paradigm should be considered harmful... Languages which are modelled by it seem bound to be very low-level, focusing largely on the "correct placement of gates in sequence" rather than the more interesting features of quantum computing which may lead to more fruitful paradigms of programming. For the authors, these features seem to be entanglement and complementarity, which we will focus on in the coming sections.

We return to the example in (5.3). The quantum if-expression of QML allows for a more intuitive representation of what the σ_x gate does, that it negates. Comparing this with the ζ-term, this action is less clear, at least from the perspective of a classical programmer. This is one reason why the version of the ζ-calculus presented in this thesis is less of a programming language and more of an exploration into certain features which might prove interesting for the further development of quantum programming languages. The addition of programming-specific constructs to the ζ-calculus is entirely possible, though not the focus of the thesis.

We note, however, that the ζ-calculus, with ζ-abstractions, exposes another possible paradigm of programming opposed to the one of QML, a *rotational* one. This should be somewhat clear to the reader, with the focus we have put on Hopf fibrations, spheres, axes, and phases. Together with the focus the ζ-calculus puts on observable structures, representing the axes of rotation, this makes for an interesting way to reason about computation with higher-dimensional data, which the authors feel should be explored. Though for purposes of familiarity and expressiveness, programming constructs from classical programming could be implemented on top of the rotational aspect.

Quantum conditionals can also be used to implement multi-qubit gates, the controlled not gate for example. In section 3.1.2.1 we presented *linking functions*, functions that can connect the shared instances of variables by some other functions. We showed that this concept enables the definition of several important multi-qubit functions in the ζ-calculus. We show the comparison of our implementation of the CNOT gate, by way of linking functions ($\ell :\equiv \eta^{\dagger}$), to the use of quantum conditionals in QML in (5.4).

$$\begin{aligned}
&\mathtt{cnot} : \mathcal{Q} \multimap \mathcal{Q} \multimap \mathcal{Q} \otimes \mathcal{Q} \\
&\mathtt{cnot}\ c\ t \ = \ \text{if}^{\circ}\ c \text{ then } \langle |1\rangle\,, \mathtt{qnot}\ t\rangle \text{ else } \langle |0\rangle\,, t\rangle \qquad \mathtt{cnot} :\equiv \zeta c \xi t\ \ell\langle c, t\rangle \fatsemi \langle c, t\rangle
\end{aligned} \tag{5.4}$$

Once again, the implementation in QML is more intuitive from the perspective of classical programming. The ζ-term does however indicate one very surprising feature of the ζ-calculus, that it is still possible to implement multi-qubit gates, and even controlled gates, without quantum conditionals. And the reasons for this are, to the authors at least, even more interesting. It comes mainly from the fact that the ζ-calculus allows for explicit control of which basis a variable is shared in, together of course with the inclusion of the $\eta^{\dagger}$ term. This explicit control of bases also allows for the definition of a whole class of interesting gates, those which entangle and modify input variables of a ζ-abstraction before the execution of the body of the abstraction. In the next sections, we will cover these features in more detail, of sharing (linearity), and of explicit control of bases (complementarity).

One last comparison is worth mentioning here, Borgna and Romero's encoding of a subset of Proto-Quipper-D [50], as scalable ZX-diagrams [51]. The scalable ZX-calculus [52] (the SZX-calculus) is an extension of the ZX-calculus which allows one to define a family of diagrams that scale dependent on a natural number. Proto-Quipper-D has a similar feature, where types can be dependent on a natural number, allowing the programmer to write higher-level programs that scale appropriately. This encoding is the only other language, to the knowledge of the authors, that translates a functional language to ZX-diagrams, being developed independently of the ζ-calculus. We feel, however, that their translation does not really use any of the unique features of the ZX-calculus, rather just using it as a denotation. Much of this denotation could also be made in a scalable quantum circuit notation as well. The gate primitives of the relevant subset of Proto-Quipper-D are translated to gate primitives defined by unary spiders in the ZX-calculus (and the CNOT gate), something which is also possible to do with quantum circuits. Though it is nice to see other languages using the ZX-calculus instead of quantum circuits, we feel that this type of translation is still essentially equivalent to the quantum circuit paradigm. It is very elegant in its way of encoding dependant types as scalable diagrams, however, and this is also something that has been discussed with regard to our language.

5.1.1.1 On linearity

One of the early directions of this thesis work was to create a language which is non-linear, that is a type system which does not limit the number of times variables can be used. The reasons for this are twofold. Firstly, the authors feel that linear languages are restrictive, especially with regard to expressiveness. Writing linear code requires a certain caution when programming. This might be good in one sense, if you view the no-cloning theorem as putting a *hard* restriction on how variables are used. However, when one views duplication of variables as sharing instead one opens up a whole other area of largely underexplored programming techniques, while still respecting the no-cloning theorem with regard to physical realisability.

This brings us to the second point, of entanglement. As we covered in the weird digression of section 3.4, the principle effect of sharing, other than creating "copies" of states, is that it produces entanglement. The point of the section was exactly to quantify how much entanglement is produced, which may or may not be useful for further discussion of sharing. The point here, however, is that the restriction of linear type systems denies entanglement from being produced implicitly. Instead, entanglement can be produced by explicitly constructing it with the constructs of the language. For example, in the quantum λ-calculus of Selinger and Valiron, we can produce a Bell-state (two maximally entangled qubits) by $\lambda x.\mathrm{CNOT}\langle \mathrm{H}(new\ 0), new\ 0\rangle$, which in the quantum ζ-calculus would be produced by the application of $\zeta x\langle x, x\rangle$ on ζ_1 (or equivalently just η). The reason for using a linear type system is usually not justified beyond the reference to the no-cloning theorem (for example [19, 20, 18, 53]), though one could argue against sharing that it is proper to forbid the implicit production of entanglement. This argument would be that entanglement produced in this way is an "unintended" side-effect. We do feel, however, that this restriction is too harsh, and that it limits the expressiveness of the language to a great extent.

Allowing sharing is not unique to the ζ-calculus. Examples of languages that include this feature are QML [27], Qunity [54], and the linear-algebraic λ-calculus [22] (Lineal). Note that the word "linear" in Lineal does not refer to a linear type system but to linear algebra; Lineal does allow sharing (the authors of this thesis made this mistake in [28]). In this regard, the distinguishing feature of the ζ-calculus is the control of which basis a variable is shared in, as the aforementioned languages instead rely on a preferred basis (taken to be the computational basis).

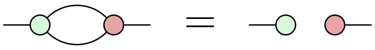
Whether or not the language allows for higher-order sharing is another point of discussion here. Lineal does allow for functions to be shared, λ-abstractions even act as basis terms that can be copied, while QML and Qunity do not. The authors of Qunity give a reason for this restriction in that, according to them, existing quantum algorithms (and apparently quantum computing in general), do not and need not employ higher-order functions. We feel that, as with sharing in general, this viewpoint is somewhat short-sighted. The paradigm of quantum programming is nascent, and as such we do not yet know what programming techniques will be useful. We feel that quantum programming languages should be more allowing in this sense, such that novel programming techniques can be discovered, rather than relying on what algorithms we have and basing our design on them. It is of course still of great importance that one develops the formal machinery to properly verify programs while allowing this expressiveness. At times, this freedom does come at a significant implementation cost and as such, one has to tread the expressiveness-implementation dialectic with care, something we feel that the ζ-calculus has managed to do to some extent. We will discuss this in terms of reduction later.

To summarise, the intention of the thesis was to create a non-linear language, one that allows for the duplication of variables. The reason for this was to afford more expressiveness and to create a framework for further exploration of sharing, both first and higher-order. Now we move on to a discussion of explicit basis control, something that, together with sharing, makes the type of quantum programming we discussed earlier possible.

5.1.1.2 On complementarity

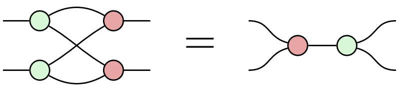
Though the term ”complementarity” might be misleading here, we will try to clear up any confusion. The ζ-calculus does not fully utilise the equational rules which come from the complementarity of the observable structures ζ and ξ. Principally this would be the Hopf and bialgebra rules. As outlined in [2, 55] these rules are equivalent to the notions of complementarity and strong complementarity. Recall that the internal cocommutative comonoid $(A, \delta_\beta, \epsilon_\beta)$ of some observable structure β represents, in the ζ-calculus, the ability for a ζ-abstraction in the basis β to utilise contraction (by the comultiplication δ_β) and weakening (by the counit ϵ_β). The internal commutative monoid (A, m_β, e_β) then, where in the †-setting $m_\beta = \delta_\beta^\dagger$ and $e_\beta = \epsilon_\beta^\dagger$, could represent a time reversal of contraction and weakening, a cocontraction and coweakening. This would really only make sense if the underlying abstract category of the ζ-calculus is extended to a †-symmetric monoidal category, an extension we will outline in the future work section. Then the cocommutative comonoid enables the structural rules, and the commutative monoid enables the time-reversed costructural rules. In this setting, the Hopf rule, presented in (5.5), would be a rule concerning the relationship between complementary (co-)contraction and (co-)weakening.

(5.5) $$\delta_\xi \circ \delta_\zeta^\dagger = \epsilon_\xi^\dagger \circ \epsilon_\zeta$$



And the bialgebra rule, presented in (5.6), would be a rule concerning the relationship between strongly complementary contraction and cocontraction.

(5.6) $$(\delta_\xi^\dagger \otimes \delta_\xi^\dagger) \circ (1 \otimes \sigma \otimes 1) \circ (\delta_\zeta \otimes \delta_\zeta) = \delta_\zeta \circ \delta_\xi^\dagger$$



The point here being that the ζ-abstraction syntax and observable structure semantics of the ζ-calculus is a distinguishing feature of the language, revealing, in the quantum model, the complementary observables for explicit control by the programmer. We believe that complementarity is an essential feature of quantum theory. To the knowledge of the authors, the syntactic construct we presented in ζ-abstractions is novel, and a useful tool for further exploration in making complementarity explicit in quantum programming. Together with the fact that this syntactic construction allows for a non-linear type system to be defined, we feel that the ζ-calculus as a quantum programming language is an important contribution of the thesis.

This explicit control of the complementary observables could also be useful with specific extensions of the ζ-calculus, with regards to programming, namely data structures. To further utilise this feature, one could provide the language with an extension of custom data types which can be instantiated in a basis. Say for example that one defines a boolean data type $\mathtt{Bool} ::= \mathtt{True}_\beta \mid \mathtt{False}_\beta$, then its interpretation would be a basis vector in the basis the constructor is instantiated in. This construction would of course need to be expanded further, possibly formalising the interpretations of the constructors as orthonormal vectors in the instantiated basis with dimension depending on the size (number of constructors) of the data type. An example interpretation of the boolean data type is presented in (5.7).

$$\llbracket \mathtt{True}_\beta \rrbracket = \beta_1 : \mathtt{Bool} \qquad \text{and} \qquad \llbracket \mathtt{False}_\beta \rrbracket = \beta_1^\pi : \mathtt{Bool} \tag{5.7}$$

In summary, we believe that the explicit basis control of the ζ-calculus allows for a paradigm of quantum programming that opens up the possibility of utilising complementary observables in a novel way. We speculate that further development of high-level programming constructs which also use this basis denotation can be used in an interesting way together with the quantum ζ-calculus. This concludes our discussion on quantum programming, now we move on to the more speculative models of the language.

5.1.2 Orders of computation and the spacetime calculus

In this section, we will provide some motivation and discuss the further applications of the ζ-calculus made regarding the spacetime calculus and orders of computation. In all previous instances where we have specifically discussed the concrete category **FdHilb**, its definition has been the monoidal category with the tensor product $\otimes$ as the tensorial functor, and the field $\mathbb{C}$ of complex numbers as the tensorial unit. As will be made clear later in this section, we now wish to distinguish between two types of monoidal categories where the tensorial unit is either of the fields $\mathbb{R}$ and $\mathbb{C}$. Since both of these scalar fields are commutative, there is no difference here with regard to coherence. We shall write $\mathbf{FdHilb}_{\mathbb{K}}$ for these concrete categories, where $\mathbb{K} \in \{\mathbb{R}, \mathbb{C}\}$. Further, we will speculate on the possibility of defining a concrete category of Hilbert spaces where its tensorial unit is a *division ring* instead, thus allowing $\mathbb{K}$ in the previous definition to range over the reals $\mathbb{R}$, the complex numbers $\mathbb{C}$, and the quaternions $\mathbb{H}$. To distinguish these, very speculative cases, we shall denote these concrete categories by $\mathbf{FdHilb}_{\mathbb{K}}^{\text{🚲}}$[1], where $\mathbb{K} \in \{\mathbb{R}, \mathbb{C}, \mathbb{H}\}$.

5.1.2.1 Motivation

We introduced the spacetime ζ-calculus as the model $\mathcal{Z}(\mathbf{FdHilb}_{\mathbb{C}}, \{\tau, \xi, \upsilon, \zeta\})$ and referred to this in the hierarchy of orders of computation as ᛟ$_3$. Intended as a stepping stone in our connection with the various Hopf fibrations, it functions as a neat example of an application of this language to a specific algebra that is known to most physicists. A somewhat primitive digression into an application of the ζ-calculus to a physical theory beyond quantum mechanics. To this, we wish to make two points of motivation, as well as give a discussion of the four orders of computation in which the quantum and spacetime ζ-calculi are included.

Firstly, the spacetime calculus is to be seen as an indication towards the generality of the construction of the ζ-calculus that we presented above. While originally intended to function as a calculus in the style of a programming language deeply inspired by the ZX-calculus, it became further apparent that a description of a general language would be possible. Thus, we presented the general ζ-calculus, not dependent upon any underlying representation. For the purpose of displaying this generality, another representation in terms of the γ-matrices from fermionic quantum field theory was constructed.

While this choice of representation might seem somewhat arbitrary, as discussed in chapter 3.2, going from a representation in terms of the Pauli-matrices to one in terms of the γ-matrices is not completely unfounded, at least from a geometrical point of view. The geometrical motivation for this connection is made by Hiley in [29].

This step in the general hierarchy of orders concerns our second point, regarding the conjecture presented in section 3.3. We introduced the orders of computation, positing that each order corresponds to a model of the ζ-calculus describing rotational computation over the constituent axes of the projective spaces for each level of Hopf fibration. We took that there was a correspondence between the number of observable structures and the dimensionality of the projective spaces proposing that a construction similar to that of the quantum ζ-calculus ought to exist for each order. This consideration was further strengthened upon learning of the Clifford algebra hierarchy, where we posited to find the necessary vector spaces associated with each algebra for the different orders.

In order two, that is ᛟ$_2$, we constructed the quantum ζ-calculus in the model $\mathcal{Z}(\mathbf{FdHilb}_{\mathbb{C}}, \{\zeta, \xi\})$, describing rotation of the spin-components of quantum states

[1] `https://en.wiktionary.org/wiki/vara_ute_och_cykla`

in the base space of $\mathbb{C}^2$ as a basis for computation. A correspondence to the complex Hopf fibration ᚻ$_2$ was made, with the projective space S^2 as the state space and the fibre space S^1 as the global phase freedom.

The correspondences made thus far have been at least somewhat sober, in the opinion of the authors. However, in ᛟ$_3$ we considered the spacetime calculus in the model $\mathcal{Z}(\mathbf{FdHilb}_{\mathbb{C}}, \{\tau, \xi, \upsilon, \zeta\})$. The observable structures of this order correspond to four sets of orthonormal bases for the space $\mathbb{C}^4$ whose matrix representations generate the Clifford algebra $\mathcal{C}\ell_{1,3}$. We considered, again, the observable structures generated by the γ-matrices as axes[2] of the generalised higher-dimensional Bloch sphere S^4 around which rotation is generated by the phase group. We saw that the fibre space S^3 can not be as easily interpreted as a global phase freedom in the complex spacetime ζ-calculus. From this we were led to consider the quaternionic spacetime ζ-calculus instead, which we will cover shortly. However, we wish to cover the remaining orders first. As a side-note, it is known to the authors that their language with regard to the many intersecting subjects involved is at this time insufficient to give a detailed description of the connections made. We must thus leave any further descriptions of this topic for the future although it was in our view that the connections made were necessary to point out.

This brings us to the orders as of yet not described. Extending the more well-known language of ᛟ$_2$ we can consider the other orders in a similar manner. This would force us to consider the first order ᛟ$_1$ as a model $\mathcal{Z}(\mathbf{FdHilb}_{\mathbb{R}}, \{\zeta\})$ consisting of a single abstraction basis describing rotational computation over a single axis. We are also forced to consider a real Hilbert space $\mathbb{R}^2$ as we still want to consider two-level systems.[3] The scalar monoid in this instance of **FdHilb** should be taken as the real numbers $\mathbb{R}$ which is commutative. We will follow the interpretation for each ᚻ$_n$ of the fibre space as the family of points identifying the same state in the base space, corresponding to the global phase freedom in this order. In this case, for ᚻ$_1$ the family of points consists of points in S^0, that is the set of points $\{-1, 1\}$ in the total space S^1 that map to the same state in the base space S^1.

Next we consider ᛟ$_4$, to which we assign the model $\mathcal{Z}(\mathbf{FdHilb}_{\mathbb{C}}, \{\zeta^{\omega}\})$ for $\omega = \{0 \ldots 7\}$. As this is the least explored of the proposed orders of computation, we do not wish to say much about this other than including a number of perhaps baseless speculations. We assume the base space of the corresponding Hopf fibration ᚻ$_4$ which would suggest a base vector space of $\mathbb{C}^8$, which we note can be identified with the space of two-dimensional octonionic vectors $\mathbb{O}^2$. Going with the pattern, we assume the axes of S^8 to correspond to our eight basis

[2] The interpretation of the observable structures generated by the γ-matrices as axes in a four-dimensional space is not unique to this thesis. A similar interpretation to this has in fact been studied before in the *spacetime algebra*. See the introduction by Hestenes[56]. To the seeming confusion of many physics students, the set γ^{μ} contains a single index which is usually reserved for vectors, meaning that this set transforms as a vector. Hestenes provides an interpretation of this, regarding γ^{μ} as an orthonormal basis for a vector space, as opposed to a set of matrices.

[3] We note that $\mathbb{R}^2$ may be identified with $\mathbb{C}$.

abstractions with similar arguments being made regarding the global phase freedom here as above. We make another interpretation of this order from the basis crystal for this order in section 3.3, however, as previously mentioned, this is the least explored of the orders, thus really up to any interpretation. This is, however, where the connection between the orders to the Clifford hierarchy appears less strong. The fourth entry in the Clifford hierarchy is the conformal or twistor Clifford $\mathcal{C}\ell_{2,4}$ consisting of six generators over a six-dimensional hypersphere. [29] This is somewhat unfortunate. However, until the authors have made themselves more comfortable with twistor algebra, we do not wish to make further comments on this order.

5.1.2.2 Quaternionic spacetime calculus

Thus, we arrive at our short discussion on the possibility of defining the spacetime calculus over the division ring of the quaternions $\mathbb{H}$. Quaternionic interpretations of quantum mechanics as well as quantum field theory have been developed as an alternative to the usual description over the complex numbers. [57, 58] We will here also make an attempt at such a quaternionic interpretation of the language of observable structures of ζ, in the spacetime ζ-calculus.

What is pertinent to our discussion is what effects this type of description has on the possibility of a categorical description of quaternionic quantum mechanics. Specifically, we wish to construct a model of the ζ-calculus similarly defined as the complex spacetime calculus, only over $\mathbb{H}$[4]. This is the model we denote by $\mathcal{Z}(\mathbf{FdHilb}_{\mathbb{H}}^{🚲}, \{\tau, \xi, \upsilon, \zeta\})$, with the base Hilbert space consisting of vectors in $\mathbb{H}^2$.

The basis abstractions are defined with respect to one particular representation of the quaternionic γ-matrices, given in figure 5.1 where for each k, $\mathbb{I}$ ranges over the quaternionic units $\{i, j, k\}$.

$$\gamma^0 := \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \qquad \gamma^k := \begin{pmatrix} 0 & -\mathbb{I} \\ \mathbb{I} & 0 \end{pmatrix} \qquad \gamma^5 := \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$

Figure 5.1: Definition of quaternionic γ-matrices.

Here we can find a clear semblance of γ^0 and γ^5 to the Pauli-Z and X matrices used in the quantum ζ-calculus, and likewise the spatial γ-matrices to the Pauli-Y matrix. Note further here that, as opposed to the complex case, these matrices are all Hermitian. For the spatial γ-matrices, the eigenvalues are not well-defined as in the temporal γ-matrices. It can be shown, following the method presented in [59], that the spectrum of these matrices is infinite. Specifically, it allows a spectrum of points covering S^2, meaning there is no single set of two eigenvectors that would function as the canonical way to generate these matrices. Regardless of this, we move on and make a choice of two eigenvectors corresponding to the real

[4] More generally, this pattern extends as ᛟ$_1 \sim \mathbb{R}^2$, ᛟ$_2 \sim \mathbb{C}^2$, ᛟ$_3 \sim \mathbb{H}^2$, and ᛟ$_4 \sim \mathbb{O}^2$.

eigenvalues ± 1, meanwhile running the risk of future criticism on this. Using these we can construct the corresponding quaternionic observable structures.

And now we come to defining what scalars correspond to the quaternionic **FdHilb**, which is where we run into problems. It was shown in [60][5] that in any monoidal category, relying on a particular coherence axiom of the left and right-unitors[6], that scalars in a monoidal category must always be commutative. Given that we wish to consider a scalar constituted of quaternions, this is seemingly not possible. Hopefully critical, yet mostly despondent, this seems to be a warrant for the cancelling of our endeavours in a quaternionic ζ-calculus. Beyond considering any adjustments of the fundamental aspects of the categorical description, such as considering whether a further categorified 2-Hilbert space [61] description would be of any help in this regard, the authors do for the moment possess insufficient knowledge to further tackle this problem. Thus it is left for future endeavours.

5.1.3 Reduction

The final point of discussion will be focused on what we introduced in section 2.3.2, the reduction relation. We started out from the reduction relation of Λ_{SMC}, the internal type theory of symmetric monoidal categories as presented in [1]. The principal difficulty in extending this reduction relation to the ζ-calculus relates to non-linearity. Since we cannot perfectly copy every term over every basis it is difficult to define substitution in the same way as for the linear case. This is because for a β-redex on the form $(\zeta x M)N$, if x appears multiple times in the body of the abstraction, the argument N has to perfectly copy over ζ for the interpretation of the redex to be equal to the interpretation of the reduct $M[x := N]$. Note that we also lose the information of which basis the variable was introduced in. To aid in the definition of substitution then, we introduced the condition of commutation with sharing over a basis. This essentially states that a term N commutes with sharing over ζ if it is perfectly copied, while possibly containing some rest where its context needs to be duplicated. Copying a context is always possible since every variable in it is decorated with the basis in which it was introduced.

With this condition, we can extend the reduction relation of Λ_{SMC} where most of the cases follow directly. All of the cases of η-reduction are the same in the ζ-calculus as in Λ_{SMC}, with the exception of the η-reduction of $\zeta x M x$ which is simply conditioned by the usual requirement of non-linear η-reduction, that $x \notin \mathbf{fv}(M)$. For β-reduction every linear case (where variables are introduced in the λ-basis) the reductions are the same as in Λ_{SMC}. For the non-linear cases we add the requirement of commutation with sharing. To have some notion of soundness for the combined reduction relation $\rightarrow_{\beta\eta}$ we prove subject reduction. This is the statement that, given that there exists a valid judgement $\Gamma \vdash M : A$ and that $M \rightarrow_{\beta\eta} N$, it holds that N is of the same type as M, and that their interpretations

[5] See also [11] for a simple review, specifically regarding the quaternionic quantum mechanics.

[6] Which reads $\lambda_I = \rho_I$.

are equal.

This is the presentation of the reduction relation of the ζ-calculus as it currently stands. The main problem of which is that it is not complete in the sense that we cannot reduce every ζ-term to some desired normal form. If we were to construct an equational theory instead, it would not be able to show that two terms whose interpretations are equal are equal in the equational theory. In this sense, the name ζ-calculus is semi-justified, since this form of the language cannot be used for fully defined reduction on its own, rather relying on some notion of convertibility in the interpretations of each model. Though we feel the name is still fitting as the language is in every sense an extension of the classical λ-calculus. For us to be completely satisfied with our reduction relation we would have to prove that it is Church-Rosser, as is the case for the reduction relation of Λ_{SMC}. Though because of it being proven for Λ_{SMC} the same holds for the linear subset of the ζ-calculus. The Church-Rosser theorem is the statement that for a term $M \in \text{Ж}$, if $M \twoheadrightarrow_{\beta\eta} M'$ and $M \twoheadrightarrow_{\beta\eta} N$, then there exists a term $N' \in \text{Ж}$ such that $M' \twoheadrightarrow_{\beta\eta} N'$ and $N \twoheadrightarrow_{\beta\eta} N'$, where $\twoheadrightarrow_{\beta\eta}$ is the reflexive transitive closure of $\rightarrow_{\beta\eta}$ (the reduction relation on $\beta\eta$). This is illustrated by the diagram in (5.8).

(5.8)
$$\begin{tikzcd}
& M \arrow[dl, two heads, "\beta\eta"'] \arrow[dr, two heads, "\beta\eta"] & \\
M' \arrow[dr, two heads, "\beta\eta"'] & & N \arrow[dl, two heads, "\beta\eta"] \\
& N' &
\end{tikzcd}$$

Onto damage control. What exactly are the implications of these problems? It mainly concerns the ζ-calculus as a general physical theory. It would of course be very nice to be able to reason computationally about the physical phenomena that the ζ-calculus is able to capture within the theory itself. This is the main motivation for further examination and work on the reduction relation. With regards to the ζ-calculus as a programming language, in the quantum model for example, we do not think that this problem affects it to any significant extent. It is still possible to get an interpretation of every ζ-term as a ZX-diagram, and thus, it is possible to optimise and extract a program to be run on a quantum computer utilising the various tools for that language. We will discuss options for the implementation of this quantum programming language in the future works section. For this reason, and because of the allure of other interesting directions of the work, together with the time restrictions of the thesis, we have not been able to investigate reduction further. We leave it up to our future selves to mend these problems if the ζ-calculus is to become a rigorous physical theory by itself.

5.2 Future work

Here we will discuss more concretely the directions of future work that was touched upon during the discussion. The points of discussion here will be fairly short, since we already brought up these points in the context of where they are relevant. This section then will serve mainly as a collection of suggestions for anyone that wishes to develop any parts of the project further.

5.2.1 The †-functor

With the definition of the ζ-calculus being housed in symmetric monoidal compact categories, we do not fully utilise the algebraic properties of observable structures. This mainly concerns the interaction between the internal cocommutative comonoid and commutative monoid, with the requirements of the Frobenius condition and speciality. Then, to make use of the full definition of observable structures we would want to move the underlying category to †-symmetric monoidal categories. The additional structure added here is the functor $\dagger : \mathcal{C}^{\mathrm{op}} \to \mathcal{C}$ called the †-functor, which is identity-on-objects (i.e. $\forall A \in \mathrm{Obj}(\mathcal{C}) \,.\, A^\dagger = A$), and which reverses morphisms, for which $\forall f \in \mathrm{Hom}(\mathcal{C}) \,.\, f^{\dagger\dagger} = f$ and $(f \otimes g)^\dagger = f^\dagger \otimes g^\dagger$. In our case, where objects are the types of the language, the reversal of morphisms would require us to be able to reverse a judgement $\Gamma \vdash M : A$ to $A \vdash M : \Gamma$. This is easy with regards to the interpretations as diagrams, it is simply the horizontal reflection. However, it would also require us to extend the terms and typing rules of the language to be able to derive this reversal with regards to the †-functor, which is substantially more difficult. A language which does take the †-functor into account is the $\dagger\lambda$-calculus [21], which would be a valuable resource in further work in this regard.

5.2.2 Reduction

We discussed earlier the problems the ζ-calculus with regards to its reduction relation. As we pointed out there our main issue with it is that it is limited to what kinds of terms can reduce. The only property of observable structures we use in reduction is the ones of the internal cocommutative comonoid, and possibly the set of classical points, by the condition of commutation with sharing. A more extensive reduction relation would be possible, utilising more of the concepts introduced in [2]. This reduction relation would then be more similar to the equational theory of the ZX-calculus, which is built by these concepts. Many of these concepts rely on the properties specific to certain observable structures, and thus, the reduction relation has to depend in some sense on the model used. An attempt at such an equational theory in the model $\mathcal{Z}(\mathbf{FdHilb}, \{\zeta, \xi\})$ was made in [28], though also incomplete.

5.2.3 Dependent type theory

The encoding of a subset λ_D of Proto-Quipper-D in the SZX-calculus was introduced in [51]. This encoding utilises scalable ZX-diagrams, where a diagram can be scaled by some natural number, to encode dependent types that depend on natural number. This is useful for describing quantum algorithms at a higher level, a quantum Fourier transform on n qubits for example. This type of encoding could also be employed in the ζ-calculus. This would first require a size operator on each type, and on contexts. From this, a SZX-diagram dependent on some type can be scaled by the size of that type. This would also require an extension of the type system of the ζ-calculus to a dependent one, as in λ_D.

5.2.4 Implementation

The simplest way to produce an implementation of the ζ-calculus would be a compiler from the syntax of the language to some graph file format. The one fitting our purposes best would be the diagrammatic proof assistant Quantomatic [62]. From an instance $ᚼ(ᛒ,\mathcal{U}_{\zeta\in ᛒ})$ one would construct a diagrammatic language in Quantomatic, one node type for each symbol in $ᛒ$, each having the set $\mathcal{U}_{\zeta}$ as possible labels. The general compilation chain would then be as follows.

Firstly, the front-end of the compiler would consist of a parser for the syntax of the chosen instance, and then a type-checker. Then, through an intermediate graph representation of the string diagram interpretations, each typed term would be assigned a diagram by the semantics of the ζ-calculus. This intermediate representation could then be translated to the Quantomatic `qgraph` format. Here, the coordinates of the elements of the diagram would have to be scaled in relation to the data structure. This compiler then would be a way of generating a Quantomatic diagram for programs in the ζ-calculus.

In the case of the quantum ζ-calculus, with the instance $ᚼ(\{\zeta, \xi\}, [0, 2\pi))$ the diagrammatic language in Quantomatic would be the ZX-calculus, where ζ corresponds to the green nodes and ξ corresponds to the red ones. For this to be translated into a language that is runnable by a quantum computer one could use the PyZX tool[7]. This tool can then load `qgraph` files to be interpreted as ZX-diagrams. PyZX then allows for translation into many other formats for quantum computer code, principally QASM, but also Quipper and QC. One could also use PyZX to optimise the ZX-diagrams that the compiler generates. With this implementation of a ζ-compiler, and the quantum instance, the ζ-calculus would be usable as a functional quantum programming language, with features for circuit optimisation.

This concludes our short discussion on future work. We again refer back to the discussion of the different aspects of the ζ-calculus for more nuance as to where possible improvements on the theory could be made.

[7] `https://github.com/Quantomatic/pyzx`

5.3 Conclusion

At last, we shall try to draw some sort of conclusion from the work that we have done. This is not an easy task seeing as the work has taken on a myriad of directions. Whether or not the absentmindedness of the authors has played any part in this is up for future discussion. The results of our work should be viewed through the perspective of the intersection of the chosen academic fields of the authors, of theoretical computer science and theoretical physics. The combined expression of which hopefully highlights the unifying trend we wish to place this thesis in, the creeping intrusion of the methods of computer science in physics. We do not refer to computational physics here, the application of insane computing power to model physical systems. What we wish to demonstrate is the fruitfulness of applying the rigorous and formal methods of theoretical computer science as an alternative to the conventional mathematics of physical theories.

In our case this trend is greatly inspired by the field of categorical quantum mechanics, having relied heavily on its methods throughout the thesis. The shining elegance of the ZX-calculus is what inspired us to attempt a thesis in this field from the beginning. Our shared love for the λ-calculus and functional programming was the other source of inspiration.

Regarding philosophy, its inclusion in this thesis is largely an expression of our opinion that philosophy should play an integral part of the work of scientists. Having this stance then, compelled us to write a short chapter on the developments of our personal philosophical discussions that played out during the writing of this thesis. We of course have our philosophically and politically motivated friends to thank for the many nights of discussion. We also thank our supervisor Robin for the philosophical input, and Ilyas for the kind words of encouragement, finally making us believe that the ideas could be taken seriously. The work Rovelli has done to encourage philosophy in physics has also been a great inspiration in this endeavour. On this point, it would be nice to see more theses include some mention of the philosophical roots of their technical work.

To summarise the entirety of the thesis, we have defined an extension of the λ-calculus in symmetric monoidal categories. We have equipped this language with an algebraic structure capturing the properties of quantum observables. With this extension we have showed that it is precisely what is needed to make the language non-linear, and thus creating a novel way to look at quantum theory through the lens of computer science. The language has been applied for use as a quantum programming language, giving rise to interesting functional programming techniques. Further, the language has been investigated for use in other physical theories, principally quantum field theory, showing that its use extends beyond the programming of quantum computers. Then, a short exploration of philosophy ties together the form and content of the work to a larger mode of analysis. With this, we conclude the ζ-calculus, a λ-calculus for quantum theories.

Bibliography

[1] Ian Mackie, Leopoldo Román, and Samson Abramsky. "An internal language for autonomous categories". In: *Applied Categorical Structures* 1.3 (1993), pp. 311–343. ISSN: 0927-2852. DOI: 10.1007/BF00873993.

[2] Bob Coecke and Ross Duncan. "Interacting quantum observables: categorical algebra and diagrammatics". In: *New Journal of Physics* 13.4 (2011), p. 043016. ISSN: 1367-2630. DOI: 10.1088/1367-2630/13/4/043016. URL: https://dx.doi.org/10.1088/1367-2630/13/4/043016.

[3] P. A. M. Dirac. *Spinors in Hilbert Space*. Boston, MA: Springer US, 1974. ISBN: 978-1-4757-0036-7. DOI: 10.1007/978-1-4757-0034-3.

[4] John C. Baez. "Division Algebras and Quantum Theory". In: (Jan. 2011). DOI: 10.1007/s10701-011-9566-z.

[5] J. J. Sakurai and Jim Napolitano. *Modern Quantum Mechanics*. Cambridge University Press, Sept. 2017. ISBN: 9781108499996. DOI: 10.1017/9781108499996. URL: https://www.cambridge.org/core/product/identifier/9781108499996/type/book.

[6] Abhay Ashtekar and Troy A. Schilling. "Geometrical Formulation of Quantum Mechanics". In: (June 1997).

[7] P. A. M. Dirac. "A new notation for quantum mechanics". In: *Mathematical Proceedings of the Cambridge Philosophical Society* 35.3 (July 1939), pp. 416–418. ISSN: 0305-0041. DOI: 10.1017/S0305004100021162. URL: https://www.cambridge.org/core/product/identifier/S0305004100021162/type/journal_article.

[8] Henk (Hendrik) Barendregt and E Barendsen. "Introduction to lambda calculus". In: *Nieuw archief voor wisenkunde* 4 (Apr. 1984), pp. 337–372.

[9] Henk Barendregt. *The Lambda Calculus - Its Syntax and Semantics*. Vol. 103. Elsevier, 1984. ISBN: 9780444875082. DOI: 10.1016/C2009-0-14341-6.

[10] Philip Wadler. "Linear Types can Change the World!" In: *Programming Concepts and Methods*. 1990.

[11] Bob Coecke and Eric Oliver Paquette. "Categories for the practising physicist". In: (May 2009).

[12] John C Baez and Michael Stay. "Physics, Topology, Logic and Computation: A Rosetta Stone". In: *Lecture Notes in Physics* 813 (2009), pp. 95–172.

[13] Roger Penrose. "Applications of negative dimensional tensors". In: *Combinatorial mathematics and its applications* 1 (1971), pp. 221–244.

[14] André Joyal and Ross Street. "The geometry of tensor calculus, I". In: *Advances in Mathematics* 88.1 (1991), pp. 55–112. ISSN: 0001-8708. DOI: `https://doi.org/10.1016/0001-8708(91)90003-P`. URL: `https://www.sciencedirect.com/science/article/pii/000187089190003P`.

[15] John van de Wetering. "ZX-calculus for the working quantum computer scientist". In: (Dec. 2020).

[16] BOB COECKE, DUSKO PAVLOVIC, and JAMIE VICARY. "A new description of orthogonal bases". In: *Mathematical Structures in Computer Science* 23.3 (June 2013), pp. 555–567. ISSN: 0960-1295. DOI: `10.1017/S0960129512000047`.

[17] W K Wootters and W H Zurek. "A single quantum cannot be cloned". en. In: *Nature* 299.5886 (Oct. 1982), pp. 802–803. ISSN: 1476-4687. DOI: `10.1038/299802a0`. URL: `https://www.nature.com/articles/299802a0`.

[18] P Selinger and B Valiron. "A lambda calculus for quantum computation with classical control". English. In: *TYPED LAMBDA CALCULI AND APPLICATIONS, PROCEEDINGS*. Ed. by P Urzyczyn. Vol. 3461. LECTURE NOTES IN COMPUTER SCIENCE. HEIDELBERGER PLATZ 3, D-14197 BERLIN, GERMANY: SPRINGER-VERLAG BERLIN, 2005, pp. 354–368. ISBN: 3-540-25593-1.

[19] Jennifer Paykin, Robert Rand, and Steve Zdancewic. "QWIRE: a core language for quantum circuits". In: *Proceedings of the 44th ACM SIGPLAN Symposium on Principles of Programming Languages*. POPL '17. New York, NY, USA: Association for Computing Machinery, Jan. 2017, pp. 846–858. ISBN: 978-1-4503-4660-3. DOI: `10.1145/3009837.3009894`. URL: `https://doi.org/10.1145/3009837.3009894`.

[20] Alexander S. Green et al. "Quipper". In: *ACM SIGPLAN Notices* 48.6 (June 2013), pp. 333–342. ISSN: 0362-1340. DOI: `10.1145/2499370.2462177`.

[21] Philip Atzemoglou. "The dagger lambda calculus". In: (June 2014). DOI: `10.4204/EPTCS.172.15`.

[22] Pablo Arrighi and Gilles Dowek. "Lineal: A Linear-Algebraic Lambda-Calculus". In: *Logical Methods in Computer Science* 13.1 (2017), p. 8.

[23] Rémy Mosseri and Rossen Dandoloff. "Geometry of entangled states, Bloch spheres and Hopf fibrations". In: *Journal of Physics A: Mathematical and General* 34.47 (Nov. 2001), pp. 10243–10252. ISSN: 0305-4470. DOI: `10.1088/0305-4470/34/47/324`.

[24] Hongwan Liu. "HOPF FIBRATION". In: 2009.

[25] A Yu Kitaev. "Quantum computations: algorithms and error correction". In: *Russian Mathematical Surveys* 52.6 (Dec. 1997), pp. 1191–1249. ISSN: 0036-0279. DOI: 10.1070/RM1997v052n06ABEH002155.

[26] Michael A Nielsen and Isaac L Chuang. *Quantum Computation and Quantum Information: 10th Anniversary Edition*. en. Dec. 2010. DOI: 10.1017/CBO9780511976667. URL: https://www.cambridge.org/highereducation/books/quantum-computation-and-quantum-information/01E10196D0A682A6AEFFEA52D53BE9AE.

[27] T Altenkirch and J Grattage. "A functional quantum programming language". In: *20th Annual IEEE Symposium on Logic in Computer Science (LICS' 05)*. June 2005, pp. 249–258. DOI: 10.1109/LICS.2005.1.

[28] Nicklas Botö and Fabian Forslund. "The Zeta Calculus". In: (Mar. 2023).

[29] B J Hiley. "Process, distinction, groupoids and Clifford algebras: An alternative view of the quantum formalism". In: *Lect. Notes Phys.* 813 (2011), pp. 705–752. DOI: 10.1007/978-3-642-12821-9{_}12.

[30] J Frank Adams. "On the Non-Existence of Elements of Hopf Invariant One". In: *Annals of Mathematics* 72 (1960), p. 20.

[31] Barton Zwiebach. *A First Course in String Theory*. Cambridge: Cambridge University Press, 2004. DOI: DOI:10.1017/CBO9780511841682. URL: https://www.cambridge.org/core/books/first-course-in-string-theory/74E0A2533A3CA80D1EE266D2E27F74F2.

[32] Horatiu Nastase. "Introduction to supergravity". In: (Dec. 2011).

[33] Marcus Huber and Julio I de Vicente. "Structure of Multidimensional Entanglement in Multipartite Systems". In: *Physical Review Letters* 110.3 (Jan. 2013), p. 30501. DOI: 10.1103/PhysRevLett.110.030501. URL: https://link.aps.org/doi/10.1103/PhysRevLett.110.030501.

[34] Friedrich Engels. *Ludwig Feuerbach och den klassiska tyska filosofins slut*. Göteborg: Proletärkultur, 1981.

[35] Carlo Rovelli. "The Relational Interpretation of Quantum Physics". In: (Sept. 2021).

[36] John A Wheeler. "Assessment of Everett's "Relative State" Formulation of Quantum Theory". In: *Reviews of Modern Physics* 29.3 (July 1957), pp. 463–465. DOI: 10.1103/RevModPhys.29.463. URL: https://link.aps.org/doi/10.1103/RevModPhys.29.463.

[37] C N Villars. "Observables, states and measurements in quantum physics". In: *European Journal of Physics* 5.3 (July 1984), pp. 177–183. ISSN: 0143-0807. DOI: 10.1088/0143-0807/5/3/010.

[38] Carlo Rovelli. *Helgoland*. New York: Riverhead Books, 2020.

[39] Alexander Bogdanov. *Empiriomonism*. Chicago: Haymarket Books, 2019.

[40] Vladimir Iljitj Lenin. *Valda verk i tio band*. Vol. 4. Moscow: Progress, Moskva, 1985.

[41] Carlo Rovelli. “”Space is blue and birds fly through it””. In: (Dec. 2017). DOI: 10.1098/rsta.2017.0312.

[42] Ludwig Wittgenstein. *Filosofiska Undersökningar*. Stockholm: Thales, 2020.

[43] Saul A. Kripke. *Wittgenstein on Rules and Private Language*. Cambridge: Harvard University Press, 1982.

[44] Michael A. Peters. “Wittgenstein, Nāgārjuna and relational quantum mechanics”. In: *Educational Philosophy and Theory* 54.12 (Oct. 2022), pp. 1942–1951. ISSN: 0013-1857. DOI: 10.1080/00131857.2022.2034620.

[45] Louis Althusser. *Philosophy of the Encounter*. London: Verso, 2006, pp. 163–207.

[46] Karl Marx. *Capital*. Vol. 1. London: Penguin Classics, 1990.

[47] Evald Ilyenkov. *The Metaphysics of Positivism*. New Park Publications, 1982.

[48] Peter Selinger and Benoît Valiron. “A Lambda Calculus for Quantum Computation with Classical Control”. In: 2005, pp. 354–368. DOI: 10.1007/11417170{_}26.

[49] Nicklas Botö and Fabian Forslund. *Compiling Quantum Conditionals in a Functional Language*. Tech. rep. Göteborg: Chalmers University of Technology, Jan. 2021. URL: https://hdl.handle.net/20.500.12380/304420.

[50] Peng Fu et al. “A tutorial introduction to quantum circuit programming in dependently typed Proto-Quipper”. In: (May 2020).

[51] Agustín Borgna and Rafael Romero. “Encoding High-level Quantum Programs as SZX-diagrams”. In: (June 2022).

[52] Titouan Carette, Dominic C Horsman, and Simon Perdrix. “SZX-Calculus: Scalable Graphical Quantum Reasoning”. In: *International Symposium on Mathematical Foundations of Computer Science*. 2019.

[53] Peter Selinger. “Towards a quantum programming language”. en. In: *Mathematical Structures in Computer Science* 14.4 (Aug. 2004), pp. 527–586. ISSN: 1469-8072, 0960-1295. DOI: 10.1017/S0960129504004256. URL: https://www.cambridge.org/core/journals/mathematical-structures-in-computer-science/article/abs/towards-a-quantum-programming-language/54D5BCF28724CA6BE38F98DC4B6803DF.

[54] Finn Voichick et al. “Qunity: A Unified Language for Quantum and Classical Computing (Extended Version)”. In: (Apr. 2022). DOI: 10.1145/3571225.

[55] B Coecke et al. “Strong Complementarity and Non-locality in Categorical Quantum Mechanics”. In: *2012 27th Annual IEEE Symposium on Logic in Computer Science*. 2012, pp. 245–254. ISBN: 1043-6871. DOI: 10.1109/LICS.2012.35.

[56] David Hestenes. "Spacetime Physics with Geometric Algebra1". In: *American Journal of Physics* 71 (July 2003). DOI: `10.1119/1.1571836`.

[57] Stephen L. Adler. *Quaternion Quantum Mechanics and Quantum Fields*. 1995.

[58] David Finkelstein et al. "Foundations of Quaternion Quantum Mechanics". In: *Journal of Mathematical Physics* 3.2 (Mar. 1962), pp. 207–220. ISSN: 0022-2488. DOI: `10.1063/1.1703794`.

[59] Li-Ping Huang and Wasin So. "On left eigenvalues of a quaternionic matrix". In: *Linear Algebra and Its Applications - LINEAR ALGEBRA APPL* 323 (Jan. 2001), pp. 105–116. DOI: `10.1016/S0024-3795(00)00246-9`.

[60] G M Kelly and M L Laplaza. "Coherence for compact closed categories". In: *Journal of Pure and Applied Algebra* 19 (1980), pp. 193–213. ISSN: 0022-4049. DOI: `https://doi.org/10.1016/0022-4049(80)90101-2`. URL: `https://www.sciencedirect.com/science/article/pii/0022404980901012`.

[61] Chris Heunen and Jamie Vicary. *Categories for Quantum Theory*. Oxford University PressOxford, Nov. 2019. ISBN: 0198739621. DOI: `10.1093/oso/9780198739623.001.0001`.

[62] Aleks Kissinger and Vladimir Zamdzhiev. "Quantomatic: A Proof Assistant for Diagrammatic Reasoning". In: (Mar. 2015). DOI: `10.1007/978-3-319-21401-6{\_}22`.

A

Color change rule in $ᛟ_3$

A table entry β/η represents the matrix product $\overline{\eta\beta\eta^{\dagger}}$.

Table A.1: Colour change table for $ᛟ_3$.

	β_T	β_X	β_Y	β_Z	β_W
η_T	T	X	Y	Z	W
η_X	$-W$	$-T$	Z	$-Y$	X
η_Y	$-W$	Z	T	$-X$	$-Y$
η_Z	$-W$	Y	$-X$	$-T$	Z
η_W	$-W$	X	Y	Z	T

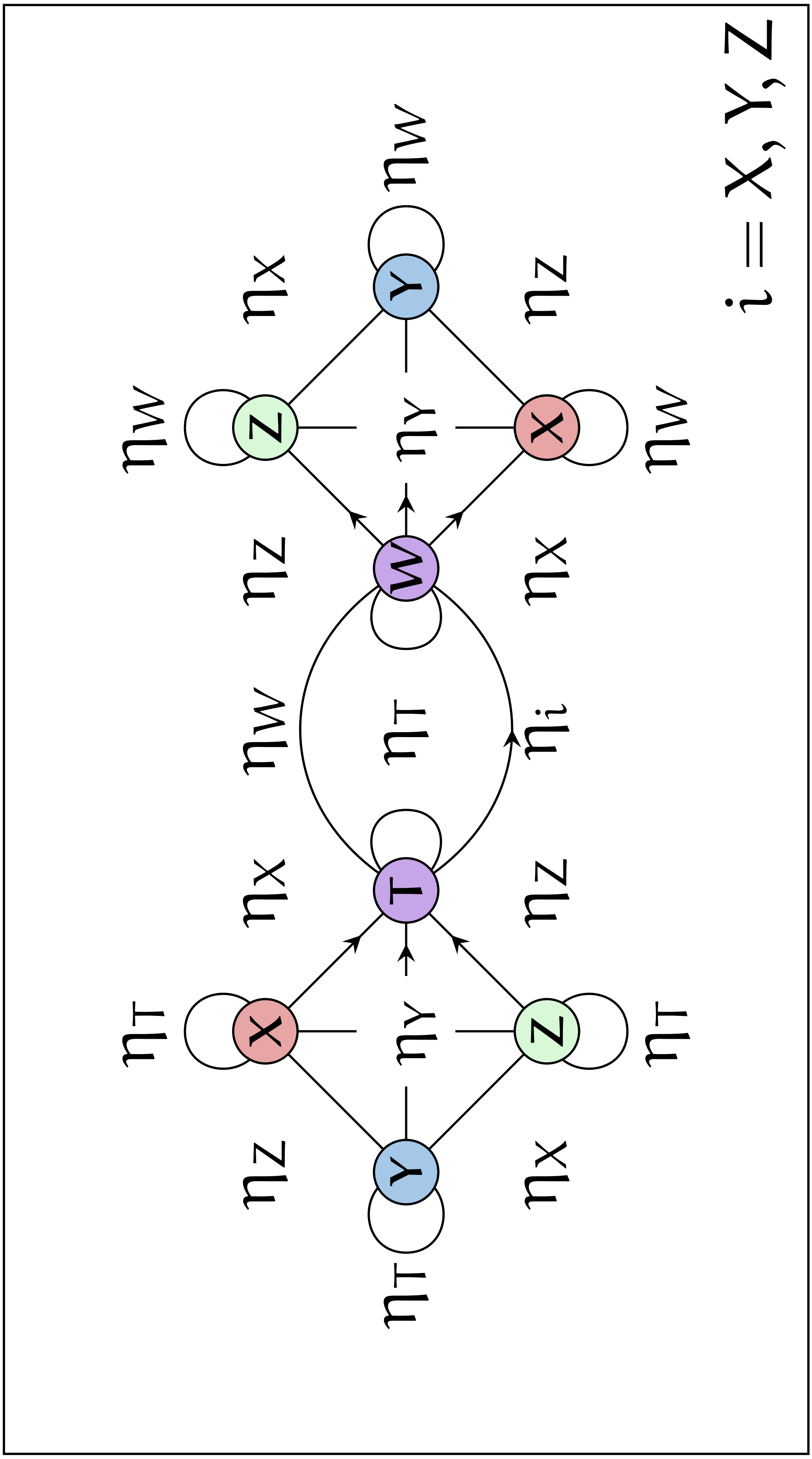


Figure A.1: Fully specified basis crystal of ᛟ$_3$.

Table A.2: Colour change table for $ᛟ_3$ in complex form.

η		β_T/γ_0	β_X/γ_1	β_Y/γ_2	β_Z/γ_3	β_W/γ_5
$\begin{pmatrix}1&0&0&0\\0&1&0&0\\0&0&1&0\\0&0&0&1\end{pmatrix}$	η_T	$\begin{pmatrix}1&0&0&0\\0&1&0&0\\0&0&-1&0\\0&0&0&-1\end{pmatrix}$	$\begin{pmatrix}0&0&0&1\\0&0&1&0\\0&-1&0&0\\-1&0&0&0\end{pmatrix}$	$\begin{pmatrix}0&0&0&-i\\0&0&i&0\\0&i&0&0\\-i&0&0&0\end{pmatrix}$	$\begin{pmatrix}0&0&1&0\\0&0&0&-1\\-1&0&0&0\\0&1&0&0\end{pmatrix}$	$\begin{pmatrix}0&0&1&0\\0&0&0&1\\1&0&0&0\\0&1&0&0\end{pmatrix}$
$\begin{pmatrix}0&i&1&0\\i&0&0&1\\0&-i&1&0\\-i&0&0&1\end{pmatrix}$	η_X	$\begin{pmatrix}0&0&-1&0\\0&0&0&-1\\-1&0&0&0\\0&-1&0&0\end{pmatrix}$	$\begin{pmatrix}i&0&0&0\\0&i&0&0\\0&0&-i&0\\0&0&0&-i\end{pmatrix}$	$\begin{pmatrix}0&0&-1&0\\0&0&0&1\\1&0&0&0\\0&-1&0&0\end{pmatrix}$	$\begin{pmatrix}0&0&0&-i\\0&0&i&0\\0&i&0&0\\-i&0&0&0\end{pmatrix}$	$\begin{pmatrix}0&0&0&i\\0&0&i&0\\0&-i&0&0\\-i&0&0&0\end{pmatrix}$
$\begin{pmatrix}0&-1&1&0\\1&0&0&1\\0&1&1&0\\-1&0&0&1\end{pmatrix}$	η_Y	$\begin{pmatrix}0&0&-1&0\\0&0&0&-1\\-1&0&0&0\\0&-1&0&0\end{pmatrix}$	$\begin{pmatrix}0&0&-1&0\\0&0&0&1\\1&0&0&0\\0&-1&0&0\end{pmatrix}$	$\begin{pmatrix}-i&0&0&0\\0&-i&0&0\\0&0&i&0\\0&0&0&i\end{pmatrix}$	$\begin{pmatrix}0&0&0&1\\0&0&1&0\\0&-1&0&0\\-1&0&0&0\end{pmatrix}$	$\begin{pmatrix}0&0&0&-1\\0&0&1&0\\0&1&0&0\\-1&0&0&0\end{pmatrix}$
$\begin{pmatrix}i&0&1&0\\0&-i&0&1\\-i&0&1&0\\0&i&0&1\end{pmatrix}$	η_Z	$\begin{pmatrix}0&0&-1&0\\0&0&0&-1\\-1&0&0&0\\0&-1&0&0\end{pmatrix}$	$\begin{pmatrix}0&0&0&i\\0&0&-i&0\\0&-i&0&0\\i&0&0&0\end{pmatrix}$	$\begin{pmatrix}0&0&0&1\\0&0&1&0\\0&-1&0&0\\-1&0&0&0\end{pmatrix}$	$\begin{pmatrix}i&0&0&0\\0&i&0&0\\0&0&-i&0\\0&0&0&-i\end{pmatrix}$	$\begin{pmatrix}0&0&i&0\\0&0&0&-i\\-i&0&0&0\\0&i&0&0\end{pmatrix}$
$\begin{pmatrix}1&0&1&0\\0&1&0&1\\-1&0&1&0\\0&-1&0&1\end{pmatrix}$	η_W	$\begin{pmatrix}0&0&-1&0\\0&0&0&-1\\-1&0&0&0\\0&-1&0&0\end{pmatrix}$	$\begin{pmatrix}0&0&0&1\\0&0&1&0\\0&-1&0&0\\-1&0&0&0\end{pmatrix}$	$\begin{pmatrix}0&0&0&-i\\0&0&i&0\\0&i&0&0\\-i&0&0&0\end{pmatrix}$	$\begin{pmatrix}0&0&1&0\\0&0&0&-1\\-1&0&0&0\\0&1&0&0\end{pmatrix}$	$\begin{pmatrix}1&0&0&0\\0&1&0&0\\0&0&-1&0\\0&0&0&-1\end{pmatrix}$

B

Syntax-directed typing rules

$$\frac{}{\vdash \star : I}\ \textsc{u} \qquad \frac{x :_{\zeta} A \in \Gamma}{\Gamma \vdash x : A}\ \textsc{v} \qquad \frac{}{\Gamma \vdash \zeta^{\alpha}_{n} : \underline{n}}\ \textsc{g} \qquad \frac{\Gamma, x :_{\zeta} A \vdash M : B}{\Gamma \vdash \zeta^{\alpha} x M : A \to B}\ \textsc{b}$$

$$\frac{\Gamma \vdash M : A \to B \qquad \Gamma \vdash N : A}{\Gamma \vdash MN : B}\ \textsc{a} \qquad \frac{\Gamma \vdash M : A \qquad \Gamma \vdash N : B}{\Gamma \vdash \langle M, N \rangle : A \otimes B}\ \textsc{t}$$

$$\frac{\Gamma \vdash M : A \otimes B \qquad \Gamma, x :_{\zeta} A, y :_{\zeta} B \vdash N : C}{\Gamma \vdash \text{let } \langle x, y \rangle =_{\zeta} M \text{ in } N : C}\ \textsc{l} \qquad \frac{\Gamma \vdash M : I \qquad \Gamma \vdash N : A}{\Gamma \vdash \text{let } \star = M \text{ in } N : A}\ \textsc{i}$$

Figure B.1: Syntax-directed typing rules.

C
Derivations of ζ-terms

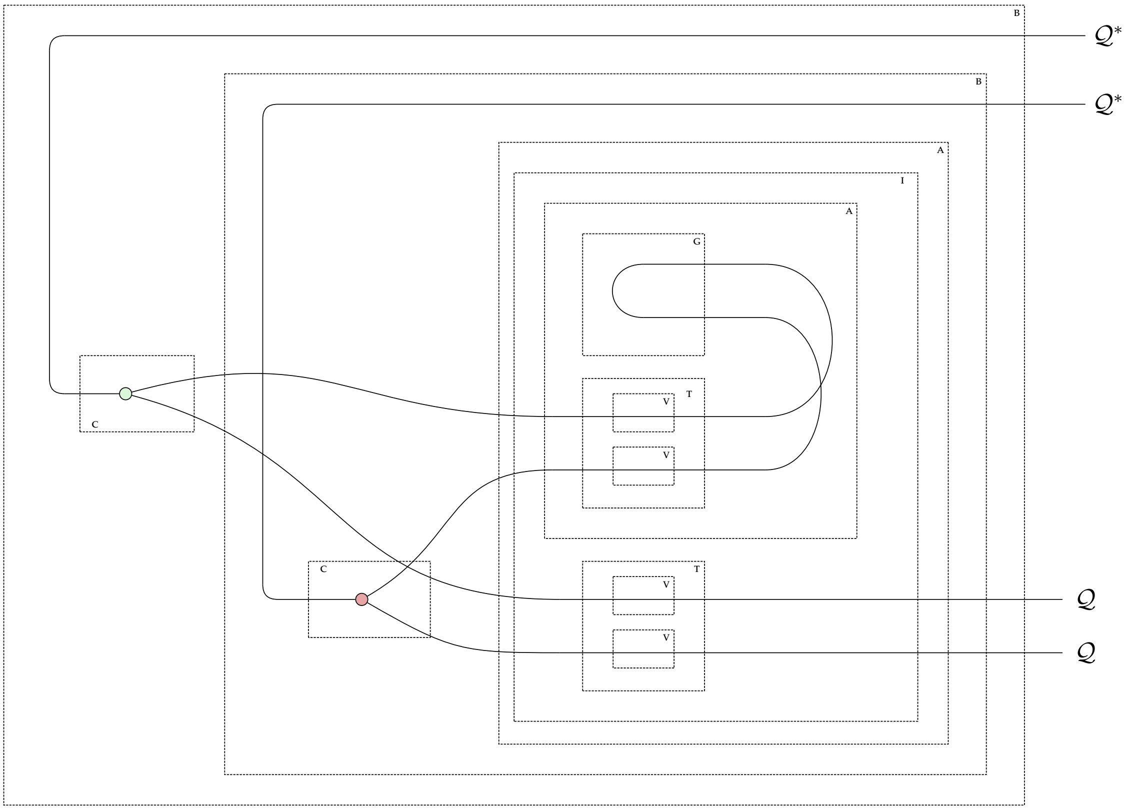


Figure C.1: The CNOT-gate.

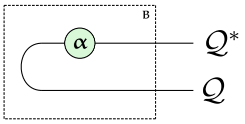


Figure C.2: The Z rotation gate.

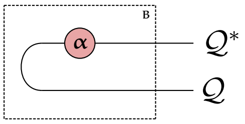


Figure C.3: The X rotation gate.

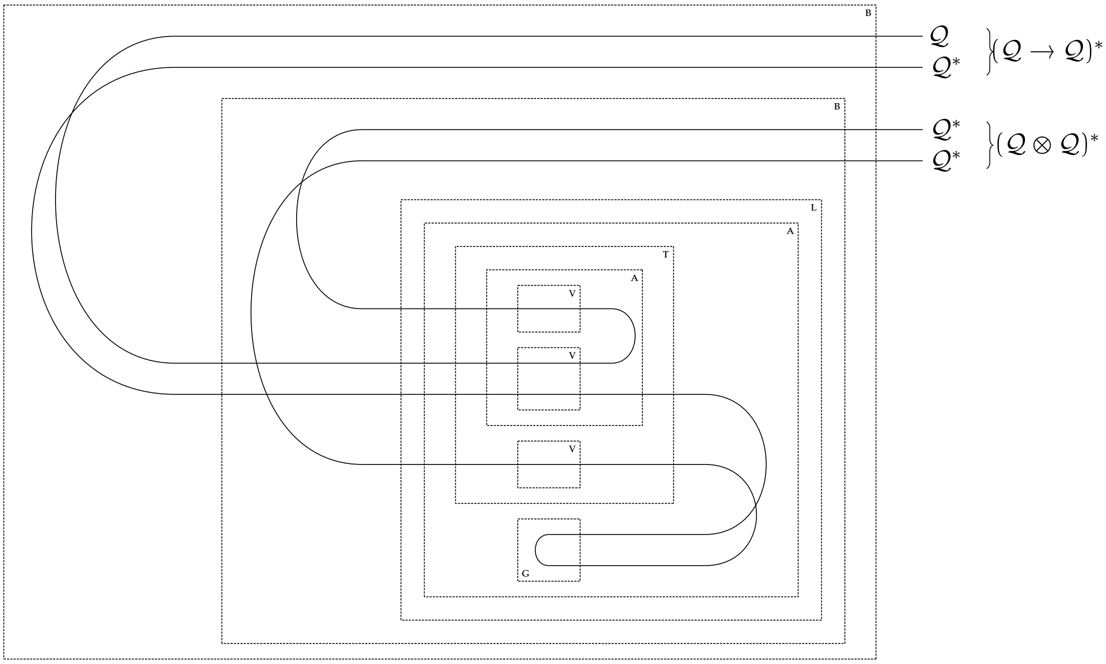

B
B
L
A
T
A
V
V
V
G
$\mathcal{Q}$
$\mathcal{Q}^*$
$(\mathcal{Q} \to \mathcal{Q})^*$
$\mathcal{Q}^*$
$\mathcal{Q}^*$
$(\mathcal{Q} \otimes \mathcal{Q})^*$

Figure C.4: The linking function.

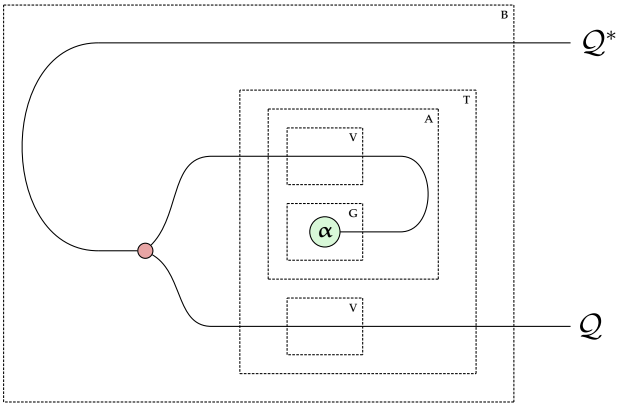


Figure C.5: The gadget function.

D

Formatting of runic characters

The formatting of runic characters used in this thesis is achieved by the excellent *allrunes* package by Carl-Gustav Werner. This package can be found at `https://www.ctan.org/tex-archive/fonts/allrunes`. The specific runes we use, and their respective commands in the allrunes package are presented below.

- Odal (ᛟ): `\textarc{o}`
- Algiz (ᛉ): `\textarc{z}`
- Berkano (ᛒ): `\textarc{b}`
- Haglaz (ᚻ): `\textarc{\h}`
- Kaun (ᚴ): `\textarc{K}`

We do recognise the use of runic symbols by certain far-right groups. If it is not already clear by a certain chapter of this thesis that we are disgusted by these vile ”people” we want to make it abundantly clear here: Fuck Nazis!